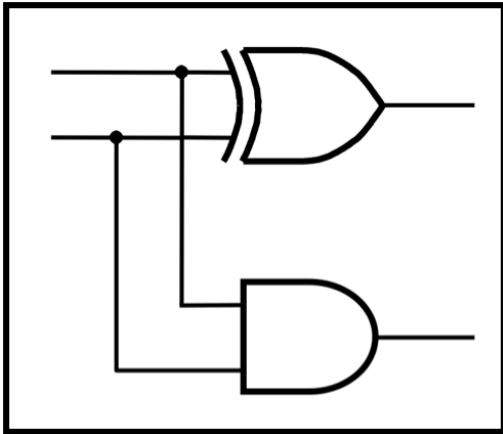




CprE 2810: Digital Logic

Instructor: Alexander Stoytchev

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

Multiplication

CprE 2810: Digital Logic
Iowa State University, Ames, IA
Copyright © Alexander Stoytchev

Administrative Stuff

- **No HW is due today**
- **HW 6 is due on Monday Oct. 13 @ 10 pm.**
- **Posted on the class web page.**

Administrative Stuff

- **Labs this week**
- **Mini-Project**
- **This is worth 3% of your grade (x2 labs)**
- **https://www.ece.iastate.edu/~alexs/classes/2025_Fall_2810/labs/Project-Mini/**

Multiplication and division by 10 in the decimal system

Decimal Multiplication by 10

What happens when we multiply a number by 10?

$$4 \times 10 = ?$$

$$542 \times 10 = ?$$

$$1245 \times 10 = ?$$

Decimal Multiplication by 10

What happens when we multiply a number by 10?

$$4 \times 10 = 40$$

$$542 \times 10 = 5420$$

$$1245 \times 10 = 12450$$

Decimal Multiplication by 10

What happens when we multiply a number by 10?

$$4 \times 10 = 40$$

$$542 \times 10 = 5420$$

$$1245 \times 10 = 12450$$

You simply add a zero as the rightmost number

Decimal Division by 10

What happens when we divide a number by 10?

$$14 / 10 = ?$$

$$540 / 10 = ?$$

$$1240 / 10 = ?$$

Decimal Division by 10

What happens when we divide a number by 10?

$$14 / 10 = 1 \quad \text{//integer division}$$

$$540 / 10 = 54$$

$$1240 / 10 = 124$$

You simply delete the rightmost number

Multiplication and division by 2 in the binary system

Binary Multiplication by 2

What happens when we multiply a number by 2?

011 times 2 = ?

101 times 2 = ?

110011 times 2 = ?

Binary Multiplication by 2

What happens when we multiply a number by 2?

$$011 \text{ times } 2 = 011\mathbf{0}$$

$$101 \text{ times } 2 = 101\mathbf{0}$$

$$110011 \text{ times } 2 = 110011\mathbf{0}$$

You simply add a zero as the rightmost number

Binary Multiplication by 4

What happens when we multiply a number by 4?

011 times 4 = ?

101 times 4 = ?

110011 times 4 = ?

Binary Multiplication by 4

What happens when we multiply a number by 4?

$$011 \text{ times } 4 = 011\mathbf{00}$$

$$101 \text{ times } 4 = 101\mathbf{00}$$

$$110011 \text{ times } 4 = 110011\mathbf{00}$$

add two zeros in the last two bits and shift everything else to the left

Binary Multiplication by 2^N

What happens when we multiply a number by 2^N ?

011 times 2^N = 011**00...0** // add N zeros

101 times 4 = 101**00...0** // add N zeros

110011 times 4 = 110011**00...0** // add N zeros

Binary Division by 2

What happens when we divide a number by 2?

0110 divided by 2 = ?

1010 divides by 2 = ?

110011 divides by 2 = ?

Binary Division by 2

What happens when we divide a number by 2?

0110 divided by 2 = 011

1010 divides by 2 = 101

110011 divides by 2 = 11001

You simply delete the rightmost number

Multiplication of two unsigned binary numbers

Decimal Multiplication By Hand

$$\begin{array}{r} 5127 \\ \times 4265 \\ \hline 25635 \\ 307620 \\ 1025400 \\ 20508000 \\ \hline 21866655 \end{array}$$

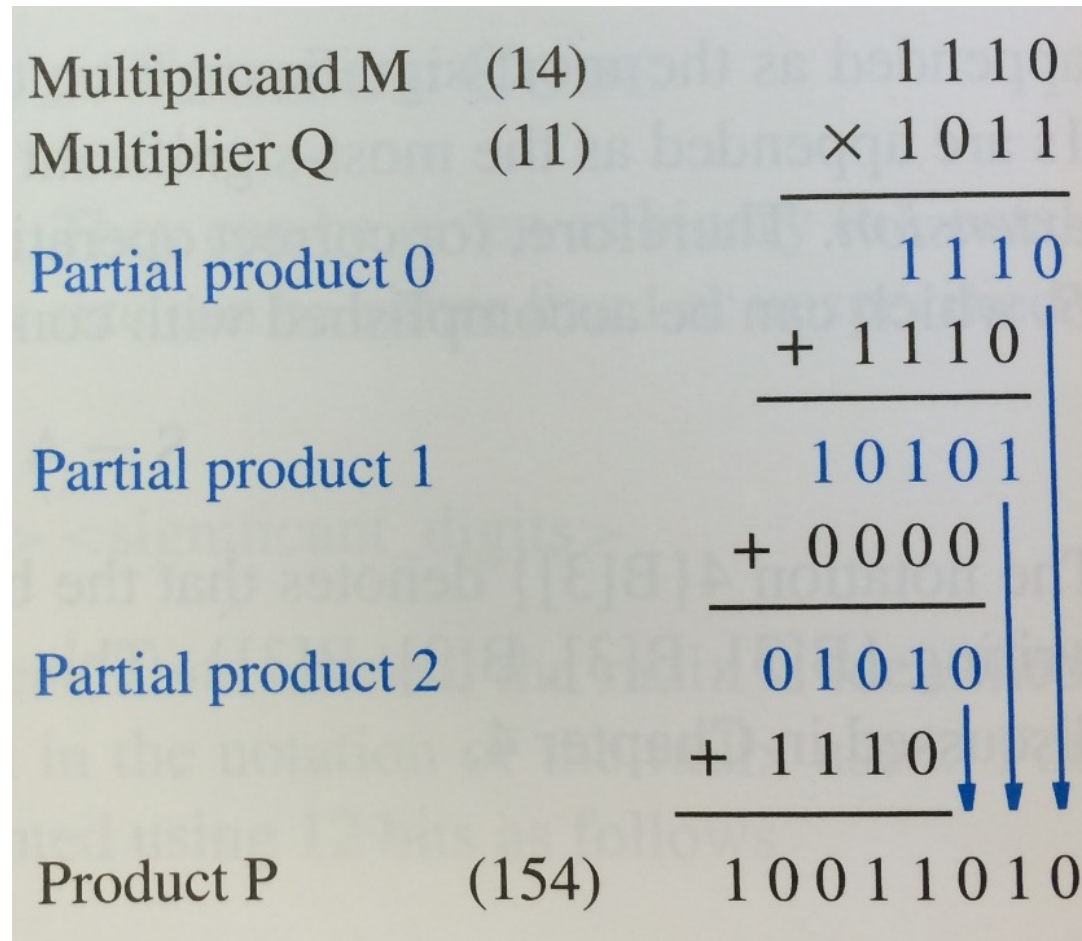
Binary Multiplication By Hand

Multiplicand M	(14)	1 1 1 0
Multiplier Q	(11)	x 1 0 1 1
		1 1 1 0
		1 1 1 0
		0 0 0 0
		1 1 1 0
Product P	(154)	1 0 0 1 1 0 1 0

[Figure 3.34a from the textbook]

Binary Multiplication By Hand

Multiplicand M	(14)	1 1 1 0
Multiplier Q	(11)	× 1 0 1 1
Partial product 0		1 1 1 0
		+ 1 1 1 0
Partial product 1		1 0 1 0 1
		+ 0 0 0 0
Partial product 2		0 1 0 1 0
		+ 1 1 1 0
Product P	(154)	1 0 0 1 1 0 1 0



[Figure 3.34b from the textbook]

Binary Multiplication By Hand

					m_3	m_2	m_1	m_0
				$\times$	q_3	q_2	q_1	q_0
Partial product 0					m_3q_0	m_2q_0	m_1q_0	m_0q_0
				+	m_3q_1	m_2q_1	m_1q_1	m_0q_1
Partial product 1					$PP1_5$	$PP1_4$	$PP1_3$	$PP1_2$
				+	m_3q_2	m_2q_2	m_1q_2	m_0q_2
Partial product 2					$PP2_6$	$PP2_5$	$PP2_4$	$PP2_3$
				+	m_3q_3	m_2q_3	m_1q_3	m_0q_3
Product P	p_7	p_6	p_5	p_4	p_3	p_2	p_1	p_0

[Figure 3.34c from the textbook]

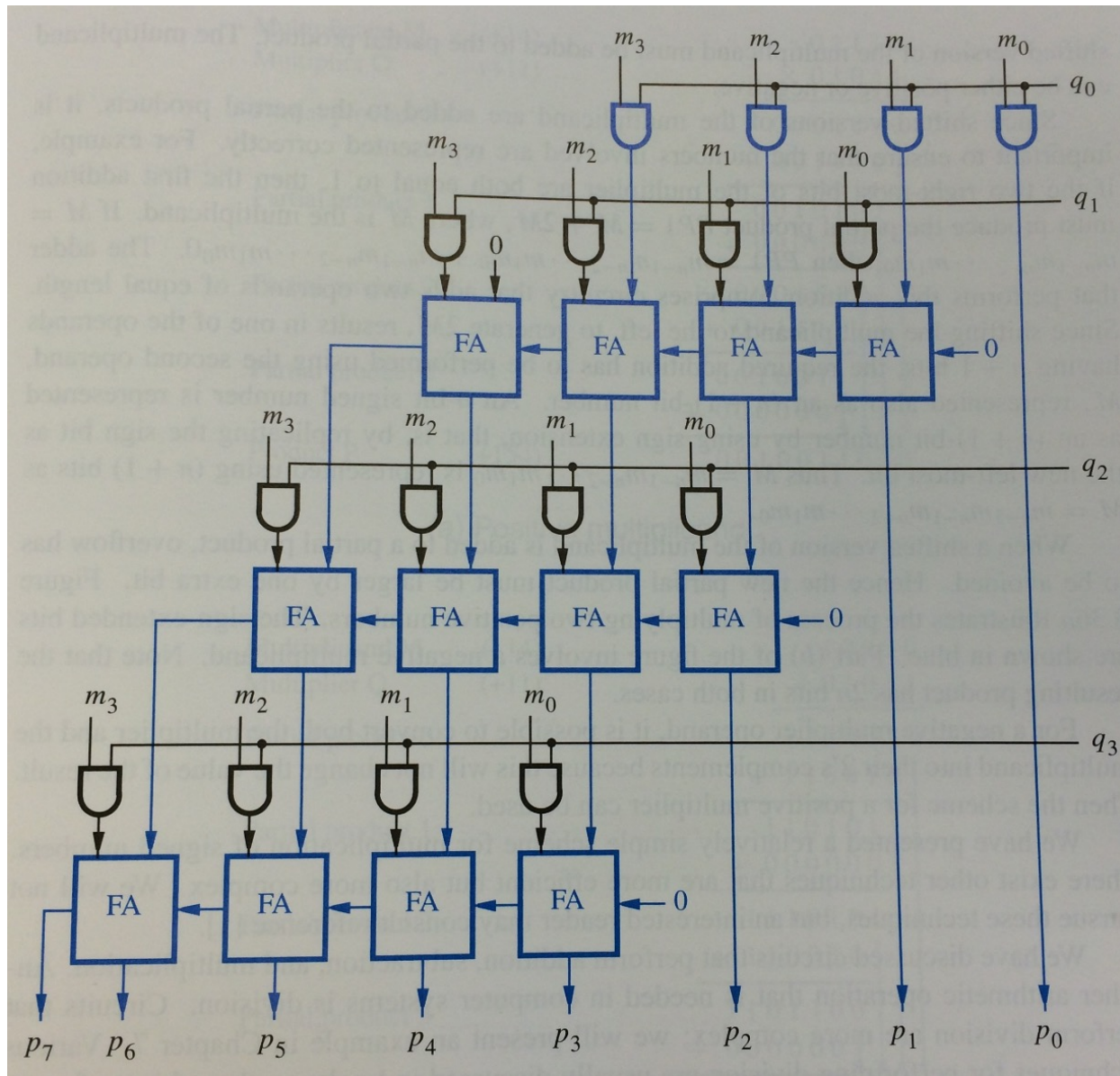


Figure 3.35. A 4x4 multiplier circuit.

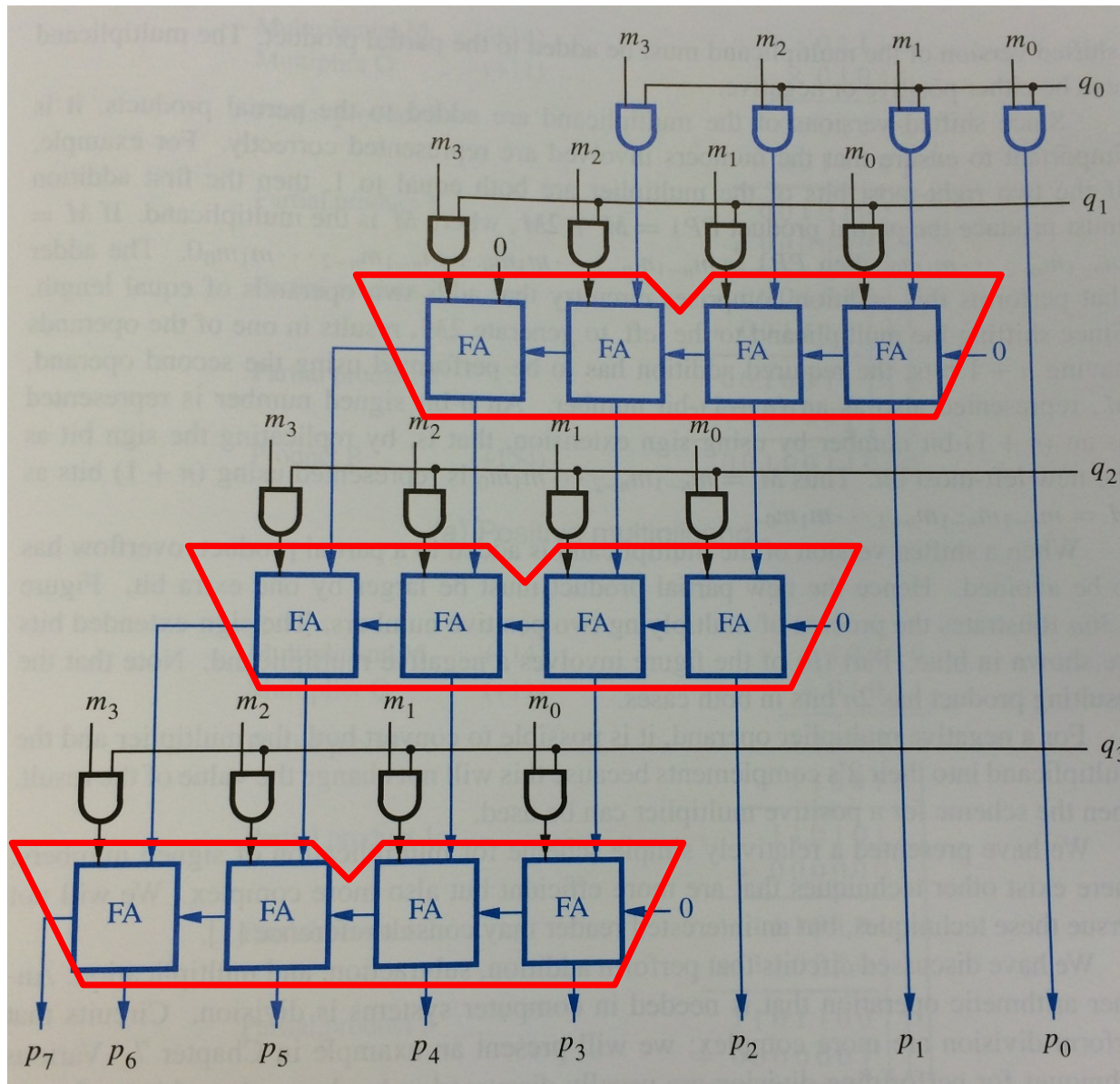


Figure 3.35. A 4x4 multiplier circuit.

Sign Extension

Sign extension for positive numbers

- If we want to represent the same positive number with more bits, we simply pad it on the left with zeros.

- For example:

0110	(+6 with 4-bits)
00110	(+6 with 5-bits)
000110	(+6 with 6-bits)

Sign extension for negative numbers

- If we want to represent the same negative number with more bits, we simply pad it on the left with ones.
- For example:

1011	(-5 with 4-bits)
11011	(-5 with 5-bits)
111011	(-5 with 6-bits)

Multiplication of two signed binary numbers

Positive Multiplicand Example

Multiplicand M (+14)

Multiplier Q (+11)

Partial product 0

Partial product 1

Partial product 2

Partial product 3

Product P (+154)

$$\begin{array}{r}
 \begin{array}{cccccc}
 & & & 0 & 1 & 1 & 1 & 0 \\
 & & & \times & 0 & 1 & 0 & 1 & 1 \\
 \hline
 & & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
 & + & 0 & 0 & 1 & 1 & 1 & 0 & \\
 \hline
 & & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\
 & + & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 \hline
 & & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
 & + & 0 & 0 & 1 & 1 & 1 & 0 & \\
 \hline
 & & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
 & + & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 \hline
 & & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0
 \end{array}
 \end{array}$$

[Figure 3.36a in the textbook]

Positive Multiplicand Example

Multiplicand M	(+14)	0 1 1 1 0
Multiplier Q	(+11)	x 0 1 0 1 1
Partial product 0	add an extra bit to avoid overflow	0 0 0 1 1 1 0 + 0 0 1 1 1 0
Partial product 1		0 0 1 0 1 0 1 + 0 0 0 0 0 0
Partial product 2		0 0 0 1 0 1 0 + 0 0 1 1 1 0
Partial product 3		0 0 1 0 0 1 1 + 0 0 0 0 0 0
Product P	(+154)	0 0 1 0 0 1 1 0 1 0

[Figure 3.36a in the textbook]

Negative Multiplicand Example

Multiplicand M (−14)

Multiplier Q (+11)

Partial product 0

Partial product 1

Partial product 2

Partial product 3

Product P (−154)

$$\begin{array}{r}
 10010 \\
 \times 01011 \\
 \hline
 1110010 \\
 + 110010 \\
 \hline
 1101011 \\
 + 000000 \\
 \hline
 1110101 \\
 + 110010 \\
 \hline
 1101100 \\
 + 000000 \\
 \hline
 1101100110
 \end{array}$$

[Figure 3.36b in the textbook]

Negative Multiplicand Example

Multiplicand M	(−14)	1 0 0 1 0
Multiplier Q	(+11)	× 0 1 0 1 1
Partial product 0	add an extra bit to avoid overflow but now it is 1	1 1 1 0 0 1 0 + 1 1 0 0 1 0
Partial product 1		1 1 0 1 0 1 1 + 0 0 0 0 0 0
Partial product 2		1 1 1 0 1 0 1 + 1 1 0 0 1 0
Partial product 3		1 1 0 1 1 0 0 + 0 0 0 0 0 0
Product P	(−154)	1 1 0 1 1 0 0 1 1 0

[Figure 3.36b in the textbook]

What if the Multiplier is Negative?

- Negate both numbers.
- This will make the multiplier positive.
- Then proceed as normal.
- This will not affect the result.
- Example: $5 * (-4) = (-5) * (4) = -20$

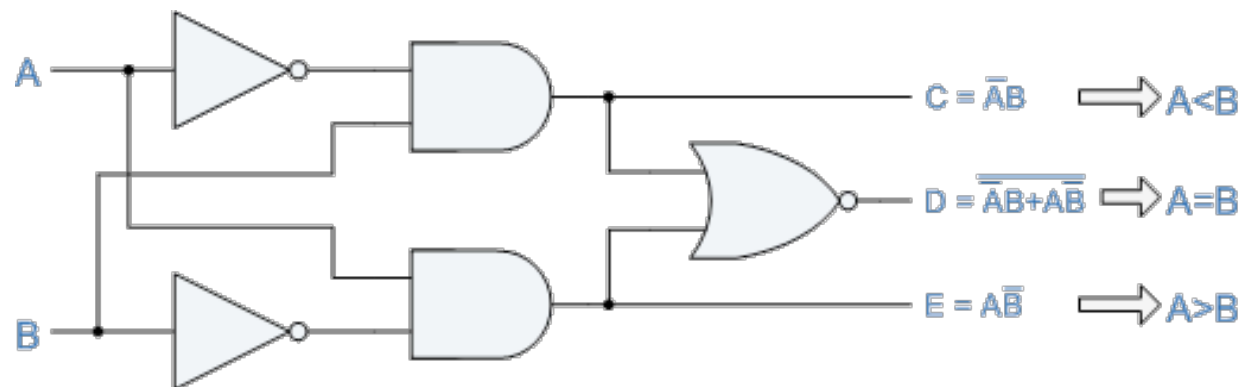
Arithmetic Comparison Circuits

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

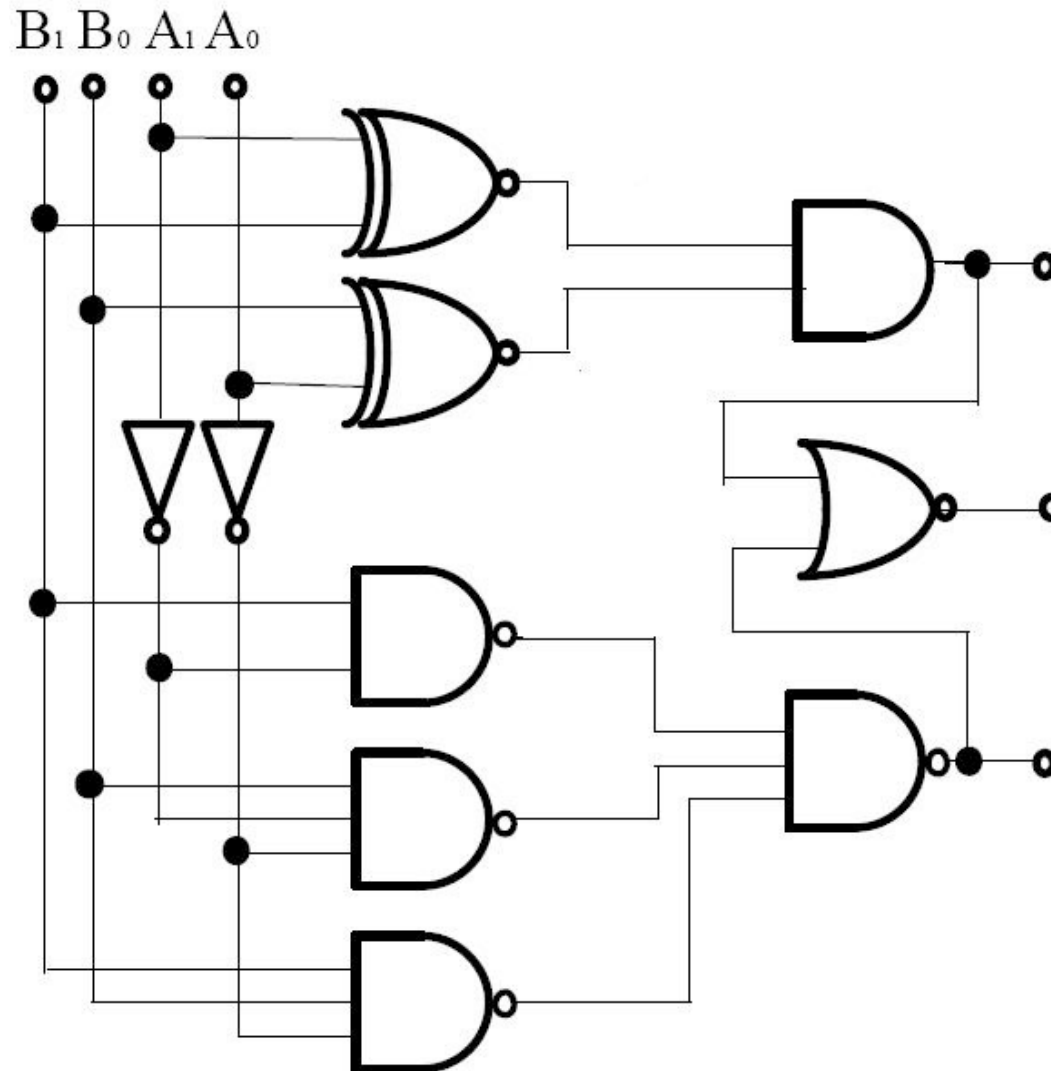


Truth table for a two-bit digital comparator

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

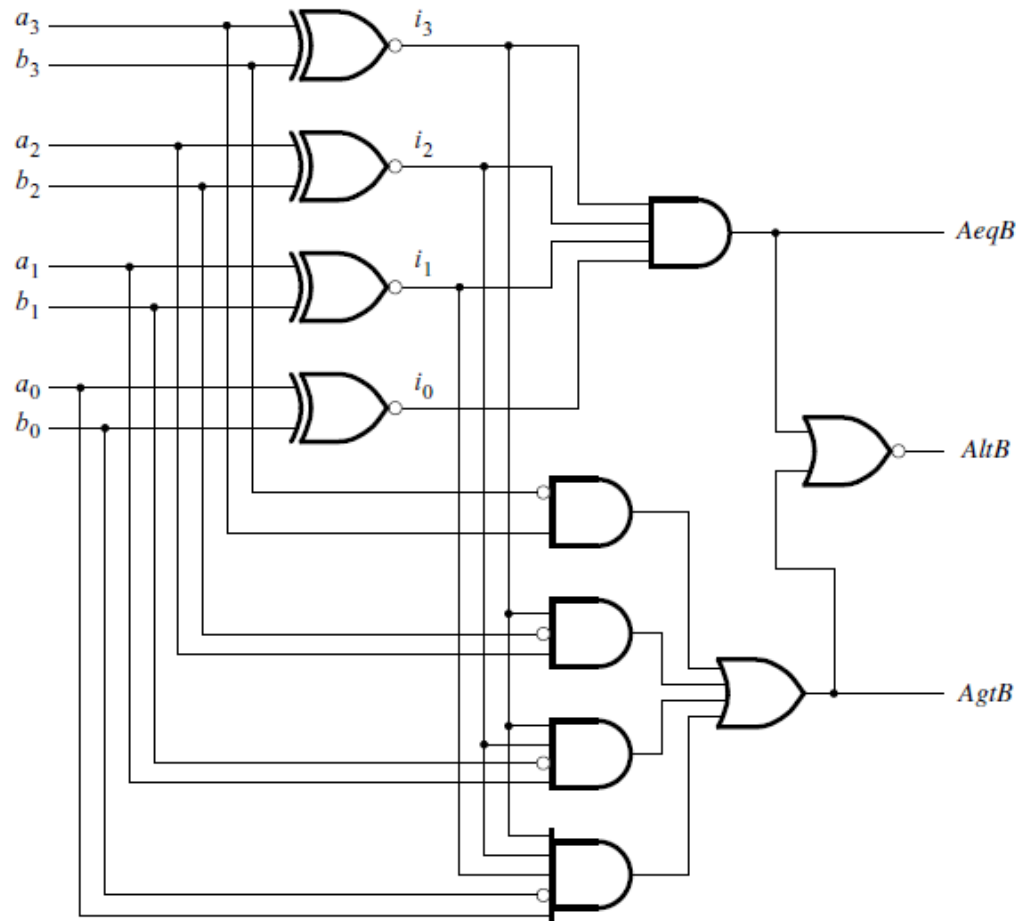
[http://en.wikipedia.org/wiki/Digital_comparator]

A two-bit digital comparator circuit



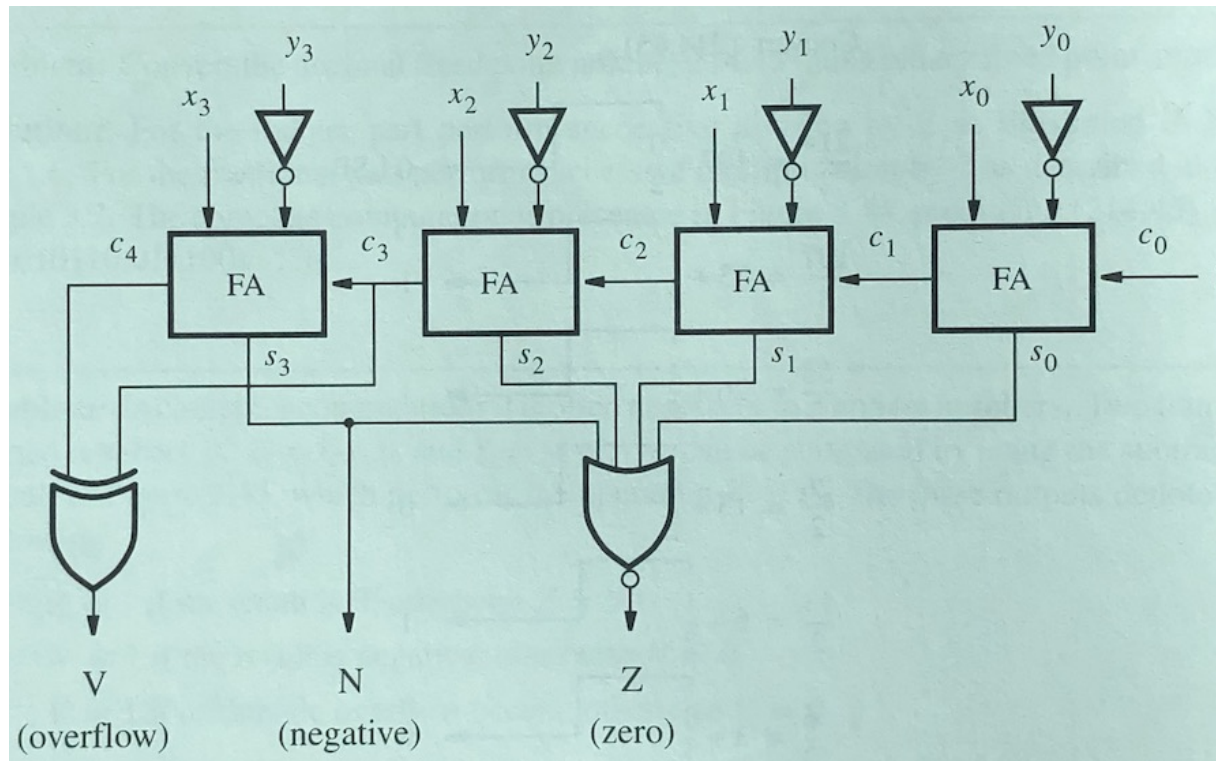
[<http://forum.allaboutcircuits.com/showthread.php?t=10561>]

A four-bit comparator circuit



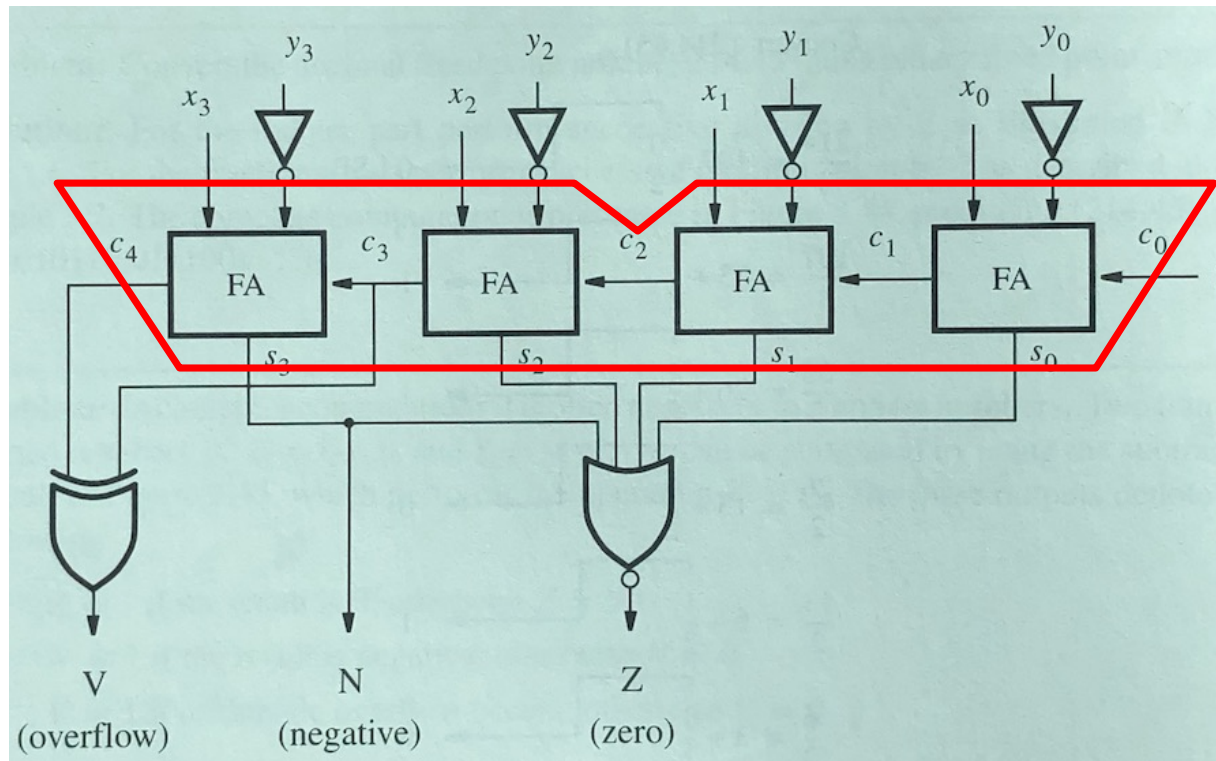
[Figure 4.22 from the textbook]

Another four-bit comparator circuit



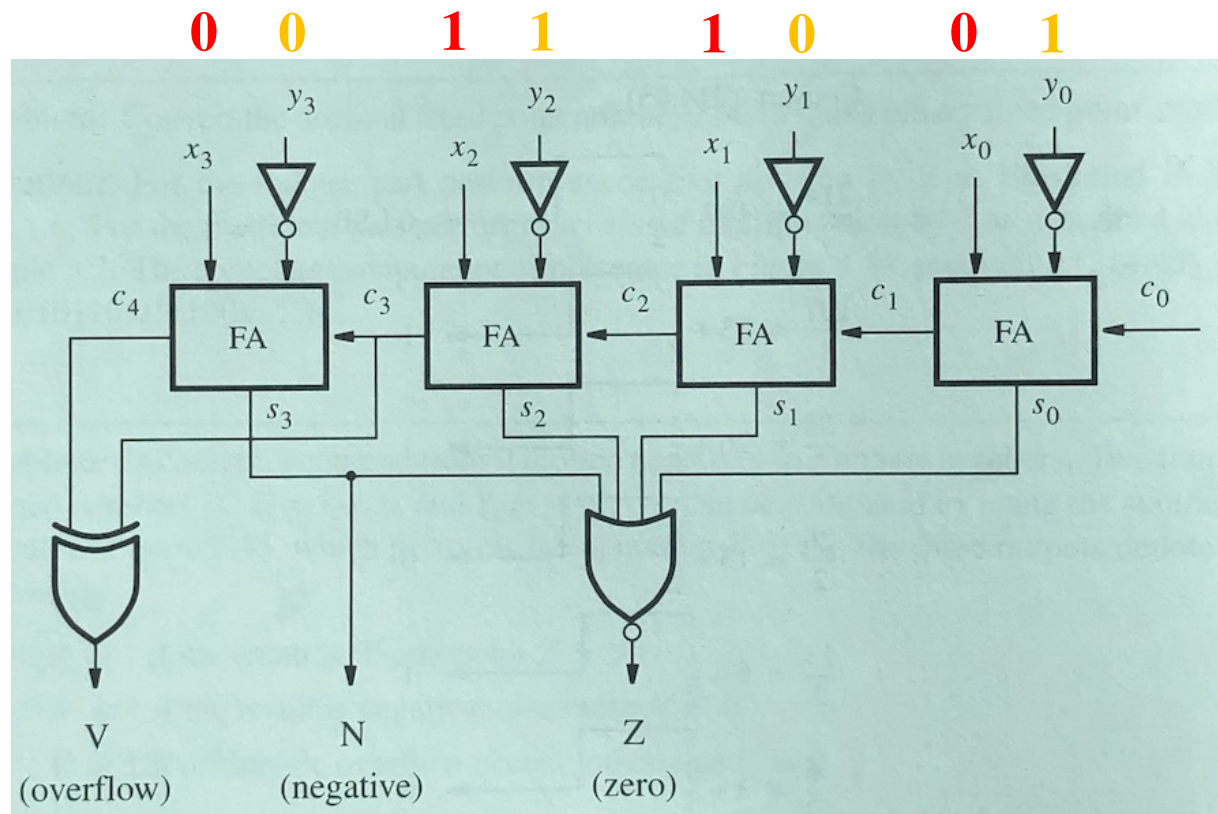
[Figure 3.45 from the textbook]

Another four-bit comparator circuit



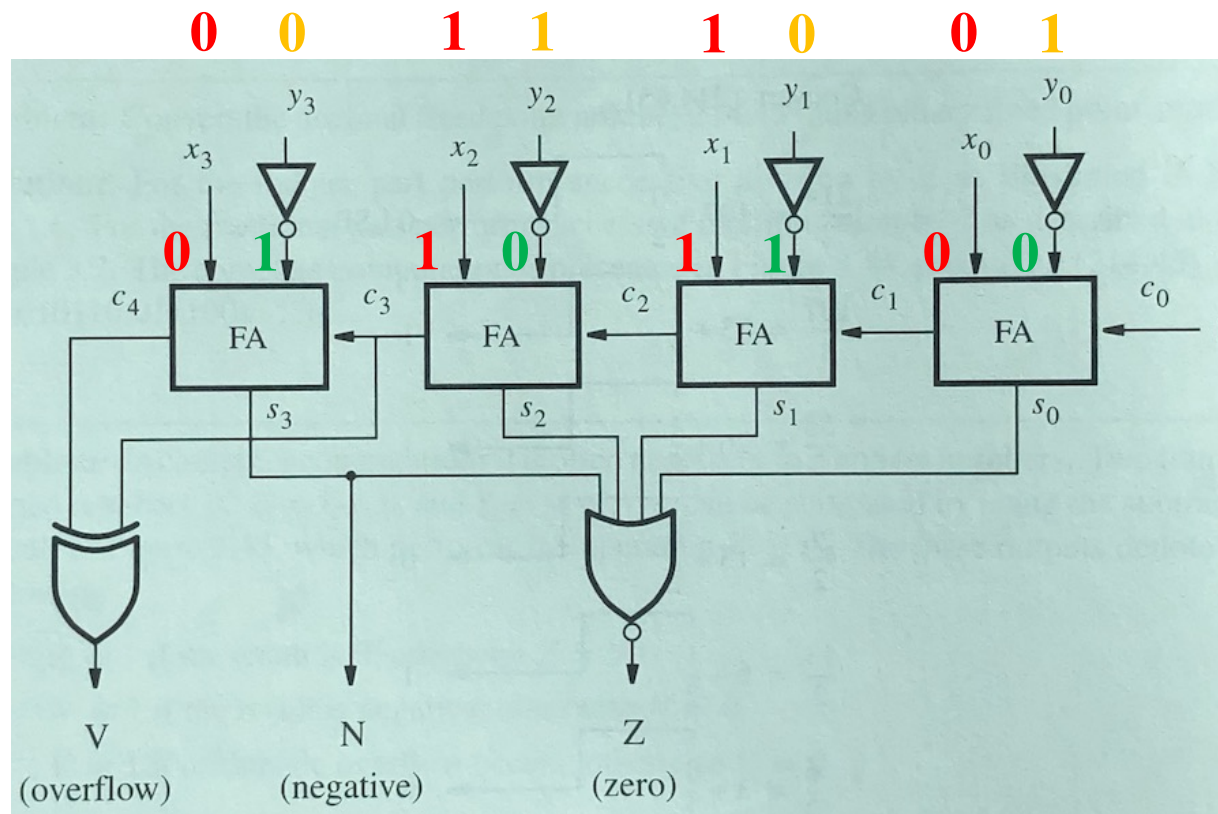
[Figure 3.45 from the textbook]

Another four-bit comparator circuit

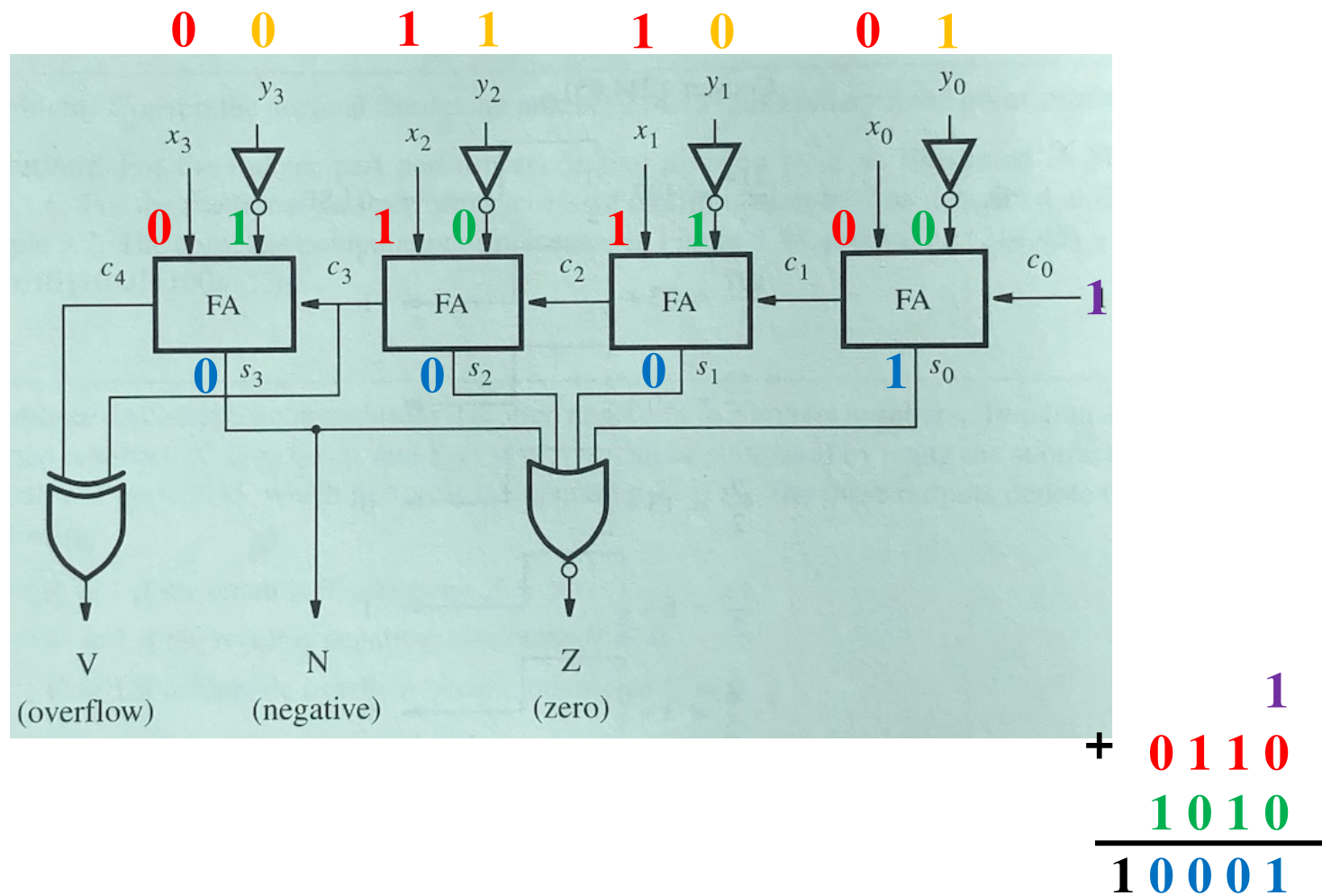


Compare 6 with 5 by subtraction (6-5).

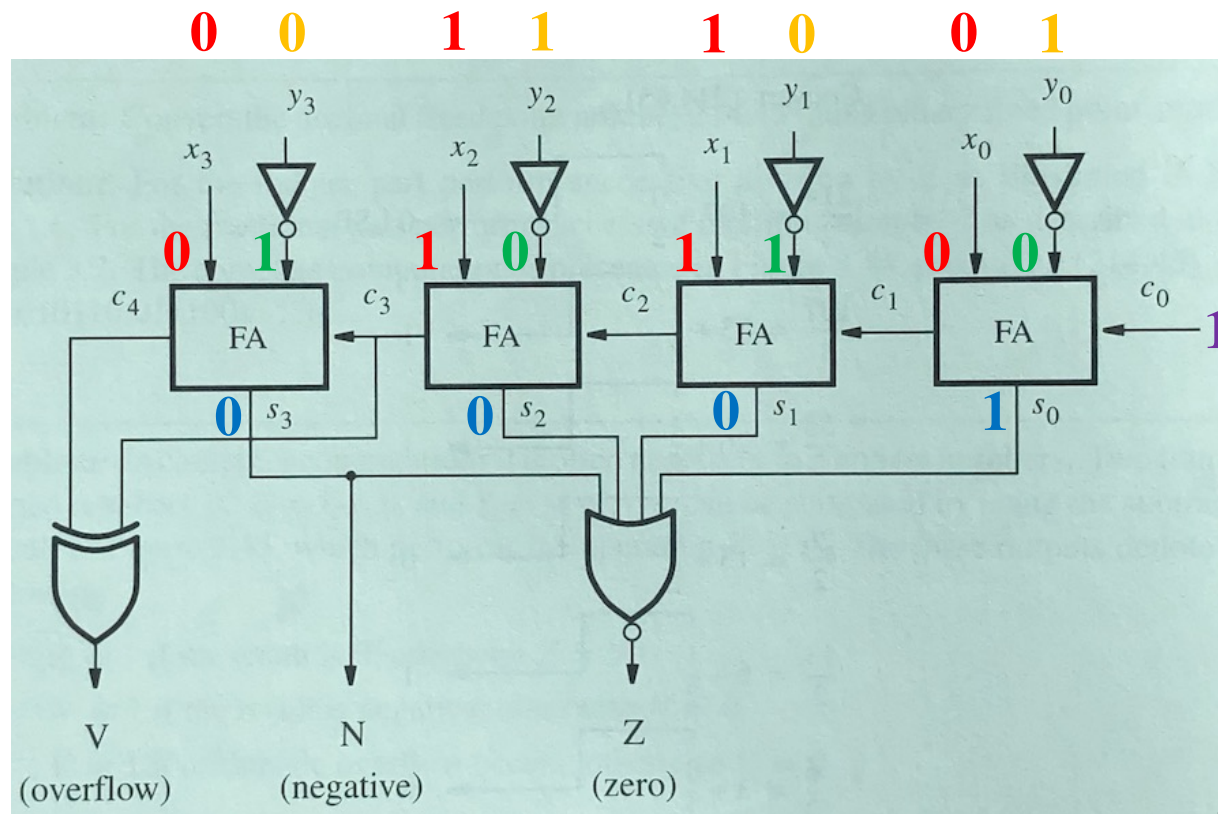
Another four-bit comparator circuit



Another four-bit comparator circuit

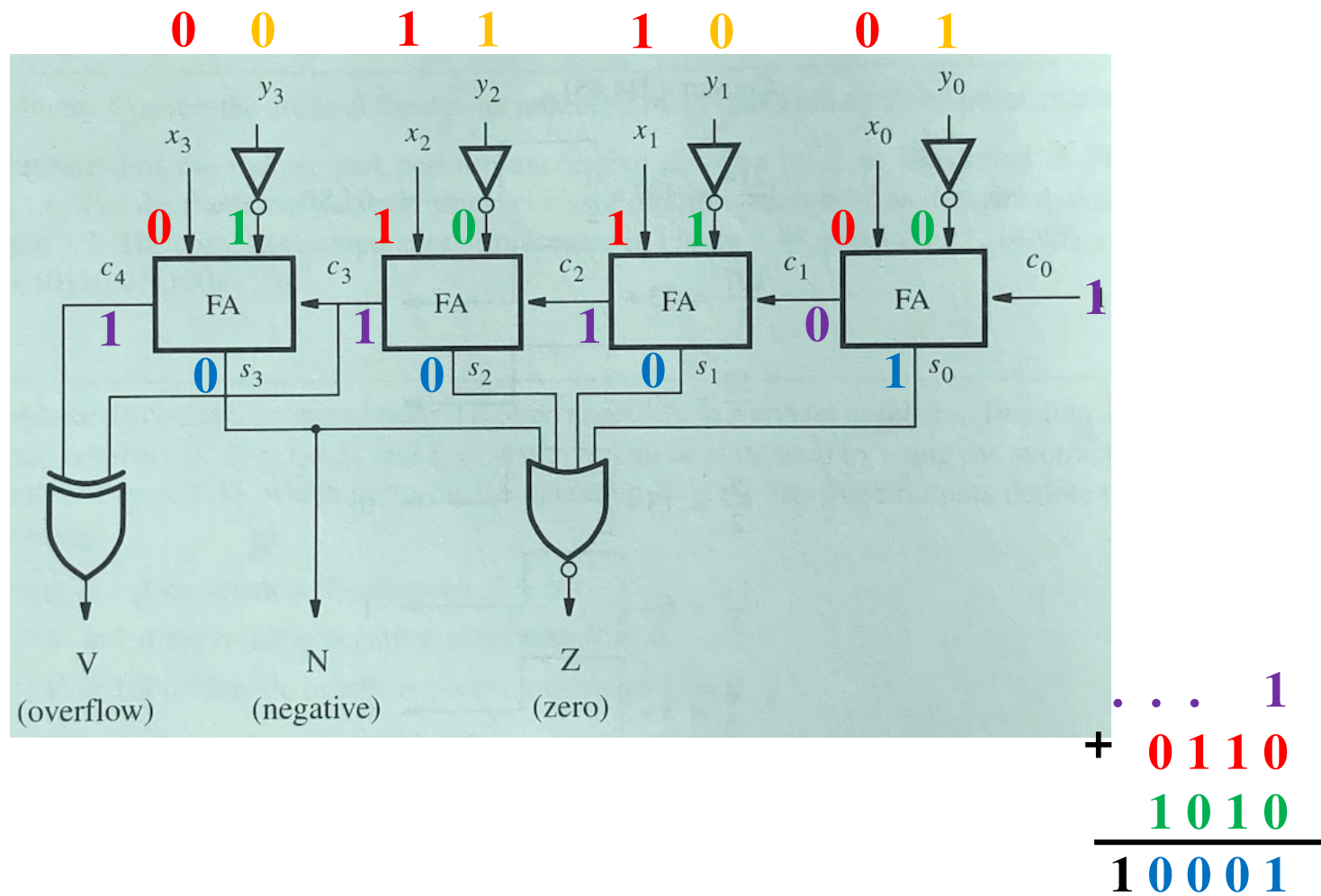


Another four-bit comparator circuit

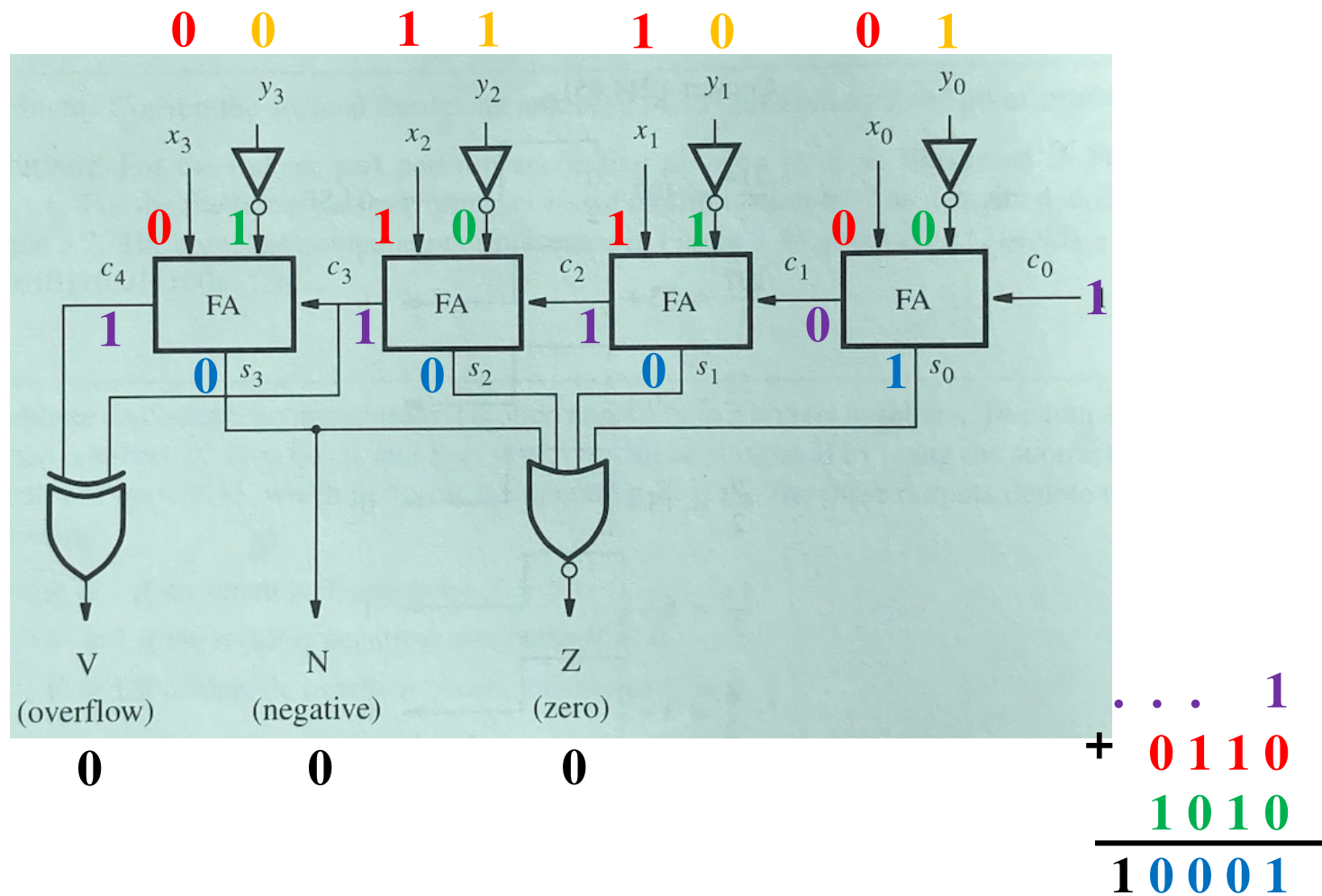


$$\begin{array}{r}
 1 \\
 + 0110 \\
 1010 \\
 \hline
 \text{Ignore } \textcircled{1} 0001
 \end{array}$$

Another four-bit comparator circuit



Another four-bit comparator circuit



Binary Coded Decimal (BCD)

Table of Binary-Coded Decimal Digits

Decimal digit	BCD code
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Addition of BCD digits

X	0 1 1 1	7
+ Y	+ 0 1 0 1	+ 5
Z	1 1 0 0	12

[Figure 3.38a in the textbook]

Addition of BCD digits

X	0 1 1 1	7
+ Y	+ 0 1 0 1	+ 5
Z	1 1 0 0	12

The result is greater than 9, which is not a valid BCD number

[Figure 3.38a in the textbook]

Addition of BCD digits

The diagram illustrates the addition of two BCD digits, 7 and 5, and the correction process:

X	0 1 1 1	7
+ Y	+ 0 1 0 1	+ 5
Z	1 1 0 0	12
	+ 0 1 1 0	
carry →	1 0 0 1 0	
	⏟	
	S = 2	

A red arrow points from the text "add 6" to the blue correction value "0 1 1 0".

[Figure 3.38a in the textbook]

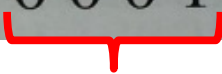
Addition of BCD digits

X	1 0 0 0	8
+ Y	+ 1 0 0 1	+ 9
Z	1 0 0 0 1	17

[Figure 3.38b in the textbook]

Addition of BCD digits

X	1 0 0 0	8
+ Y	+ 1 0 0 1	+ 9
Z	1 0 0 0 1	17



The result is 1, but it should be 7

[Figure 3.38b in the textbook]

Addition of BCD digits

The diagram illustrates the addition of BCD digits 8 and 9. It consists of a table and a detailed bit-level calculation.

X	1 0 0 0	8
+ Y	+ 1 0 0 1	+ 9
Z	1 0 0 0 1	17

Below the table, the bit-level calculation is shown:

1 0 0 0 1
+ 0 1 1 0

A red arrow points from the text "add 6" to the blue "0 1 1 0" value. A black arrow points from the text "carry" to the blue "1 0 1 1 1" value.

1 0 1 1 1
└───┘
S = 7

[Figure 3.38b in the textbook]

Why add 6?

- Think of BCD addition as a mod 16 operation
- Decimal addition is mod 10 operation

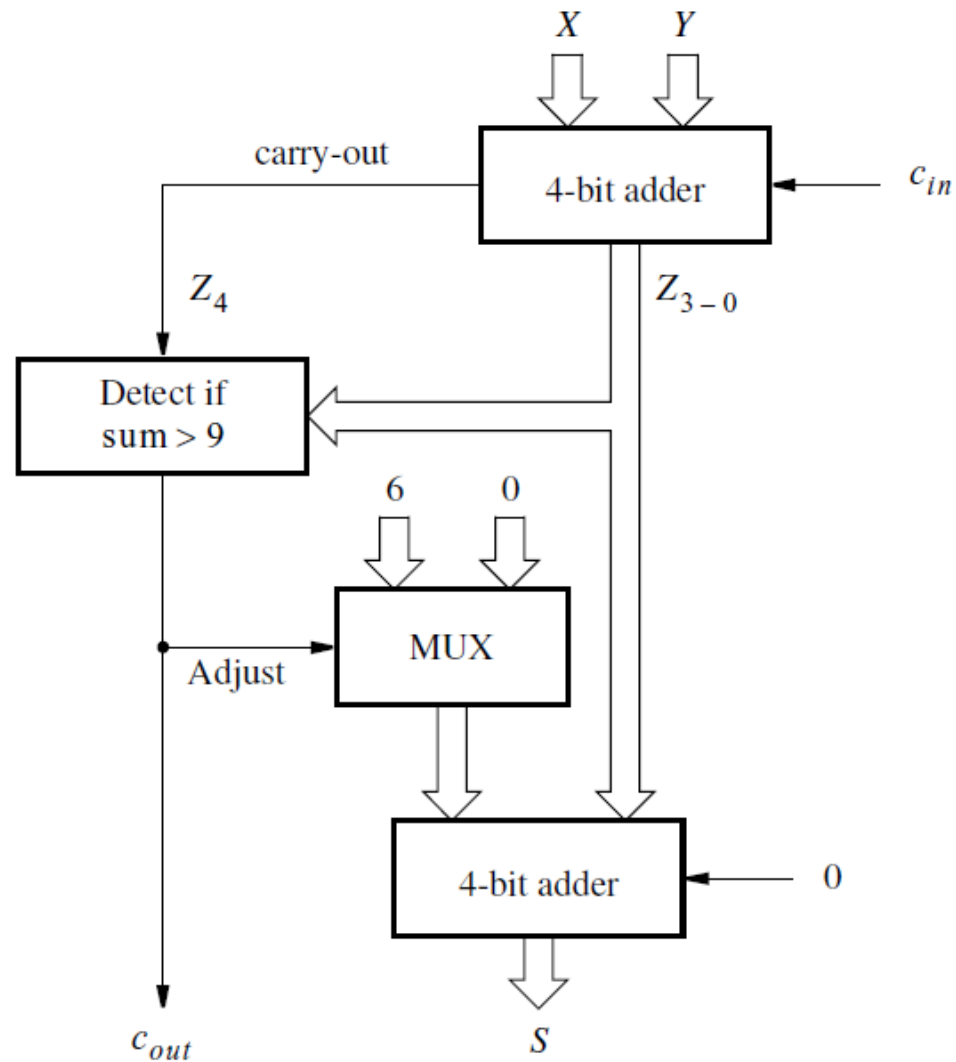
BCD Arithmetic Rules

$$Z = X + Y$$

If $Z \leq 9$, then $S=Z$ and carry-out = 0

If $Z > 9$, then $S=Z+6$ and carry-out =1

Block diagram for a one-digit BCD adder



[Figure 3.39 in the textbook]

How to check if the number is > 9?

7 - 0111

8 - 1000

9 - 1001

10 - 1010

11 - 1011

12 - 1100

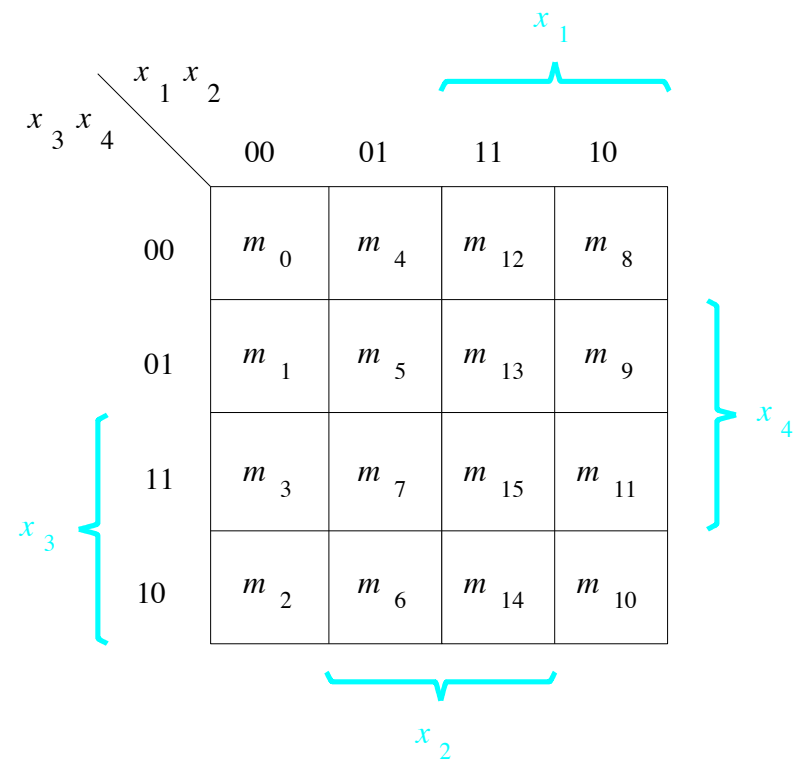
13 - 1101

14 - 1110

15 - 1111

A four-variable Karnaugh map

x1	x2	x3	x4		
0	0	0	0	m0	0
0	0	0	1	m1	0
0	0	1	0	m2	0
0	0	1	1	m3	0
0	1	0	0	m4	0
0	1	0	1	m5	0
0	1	1	0	m6	0
0	1	1	1	m7	0
1	0	0	0	m8	0
1	0	0	1	m9	0
1	0	1	0	m10	1
1	0	1	1	m11	1
1	1	0	0	m12	1
1	1	0	1	m13	1
1	1	1	0	m14	1
1	1	1	1	m15	1



How to check if the number is > 9?

z3	z2	z1	z0		
0	0	0	0	m0	0
0	0	0	1	m1	0
0	0	1	0	m2	0
0	0	1	1	m3	0
0	1	0	0	m4	0
0	1	0	1	m5	0
0	1	1	0	m6	0
0	1	1	1	m7	0
1	0	0	0	m8	0
1	0	0	1	m9	0
1	0	1	0	m10	1
1	0	1	1	m11	1
1	1	0	0	m12	1
1	1	0	1	m13	1
1	1	1	0	m14	1
1	1	1	1	m15	1

		$z_3 z_2$			
		00	01	11	10
$z_1 z_0$	00	0	0	1	0
	01	0	0	1	0
	11	0	0	1	1
	10	0	0	1	1

How to check if the number is > 9?

z3	z2	z1	z0		
0	0	0	0	m0	0
0	0	0	1	m1	0
0	0	1	0	m2	0
0	0	1	1	m3	0
0	1	0	0	m4	0
0	1	0	1	m5	0
0	1	1	0	m6	0
0	1	1	1	m7	0
1	0	0	0	m8	0
1	0	0	1	m9	0
1	0	1	0	m10	1
1	0	1	1	m11	1
1	1	0	0	m12	1
1	1	0	1	m13	1
1	1	1	0	m14	1
1	1	1	1	m15	1

		$z_3 z_2$			
		00	01	11	10
$z_1 z_0$	00	0	0	1	0
	01	0	0	1	0
	11	0	0	1	1
	10	0	0	1	1

$$f = z_3 z_2 + z_3 z_1$$

How to check if the number is > 9?

z3	z2	z1	z0		
0	0	0	0	m0	0
0	0	0	1	m1	0
0	0	1	0	m2	0
0	0	1	1	m3	0
0	1	0	0	m4	0
0	1	0	1	m5	0
0	1	1	0	m6	0
0	1	1	1	m7	0
1	0	0	0	m8	0
1	0	0	1	m9	0
1	0	1	0	m10	1
1	0	1	1	m11	1
1	1	0	0	m12	1
1	1	0	1	m13	1
1	1	1	0	m14	1
1	1	1	1	m15	1

		z ₃ z ₂			
		00	01	11	10
z ₁ z ₀	00	0	0	1	0
	01	0	0	1	0
	11	0	0	1	1
	10	0	0	1	1

$$f = z_3z_2 + z_3z_1$$

In addition, also check if there was a carry

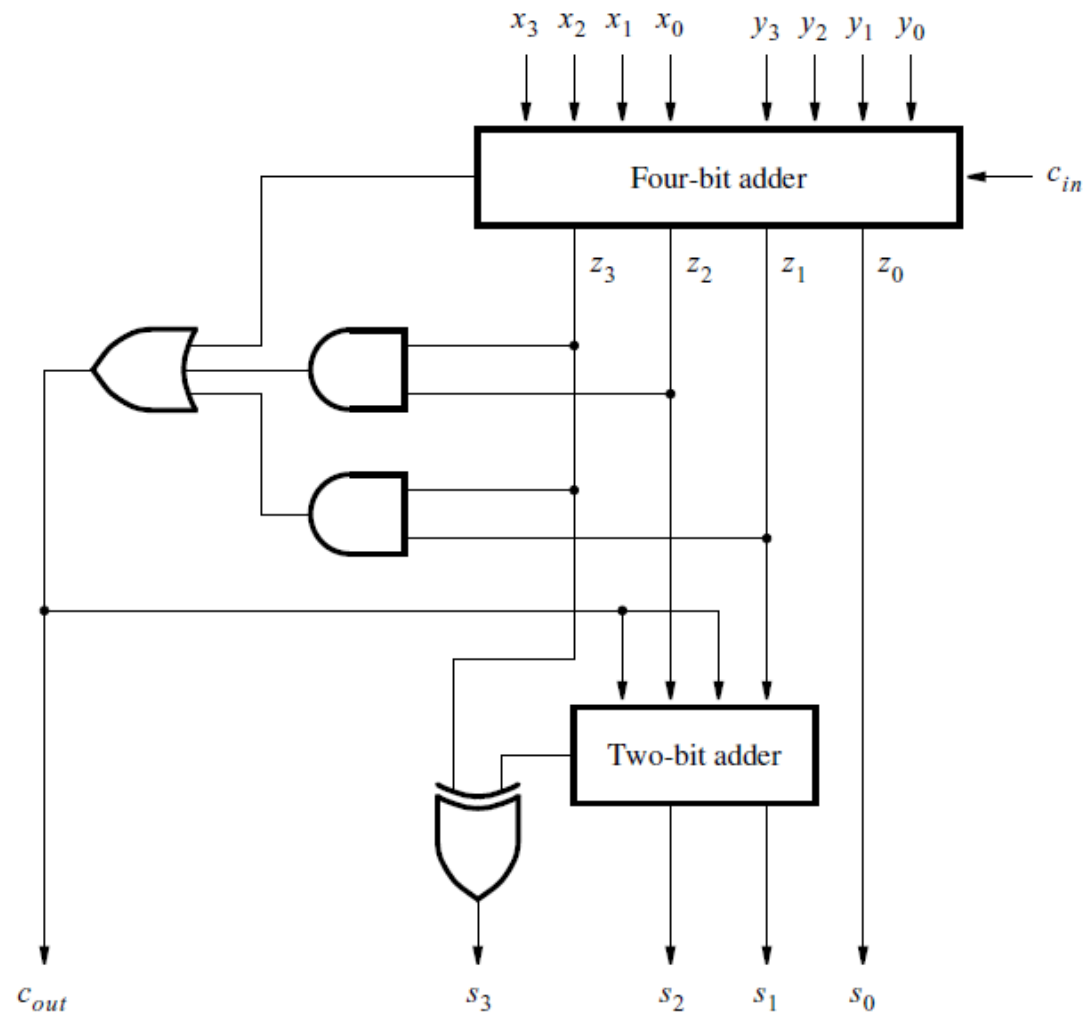
$$f = \text{carry-out} + z_3z_2 + z_3z_1$$

Verilog code for a one-digit BCD adder

```
module bcdadd(Cin, X, Y, S, Cout);  
    input Cin;  
    input [3:0] X,Y;  
    output reg [3:0] S;  
    output reg Cout;  
    reg [4:0] Z;  
  
    always@ (X, Y, Cin)  
    begin  
        Z = X + Y + Cin;  
        if (Z < 10)  
            {Cout,S} = Z;  
        else  
            {Cout, S} = Z + 6;  
    end  
  
endmodule
```

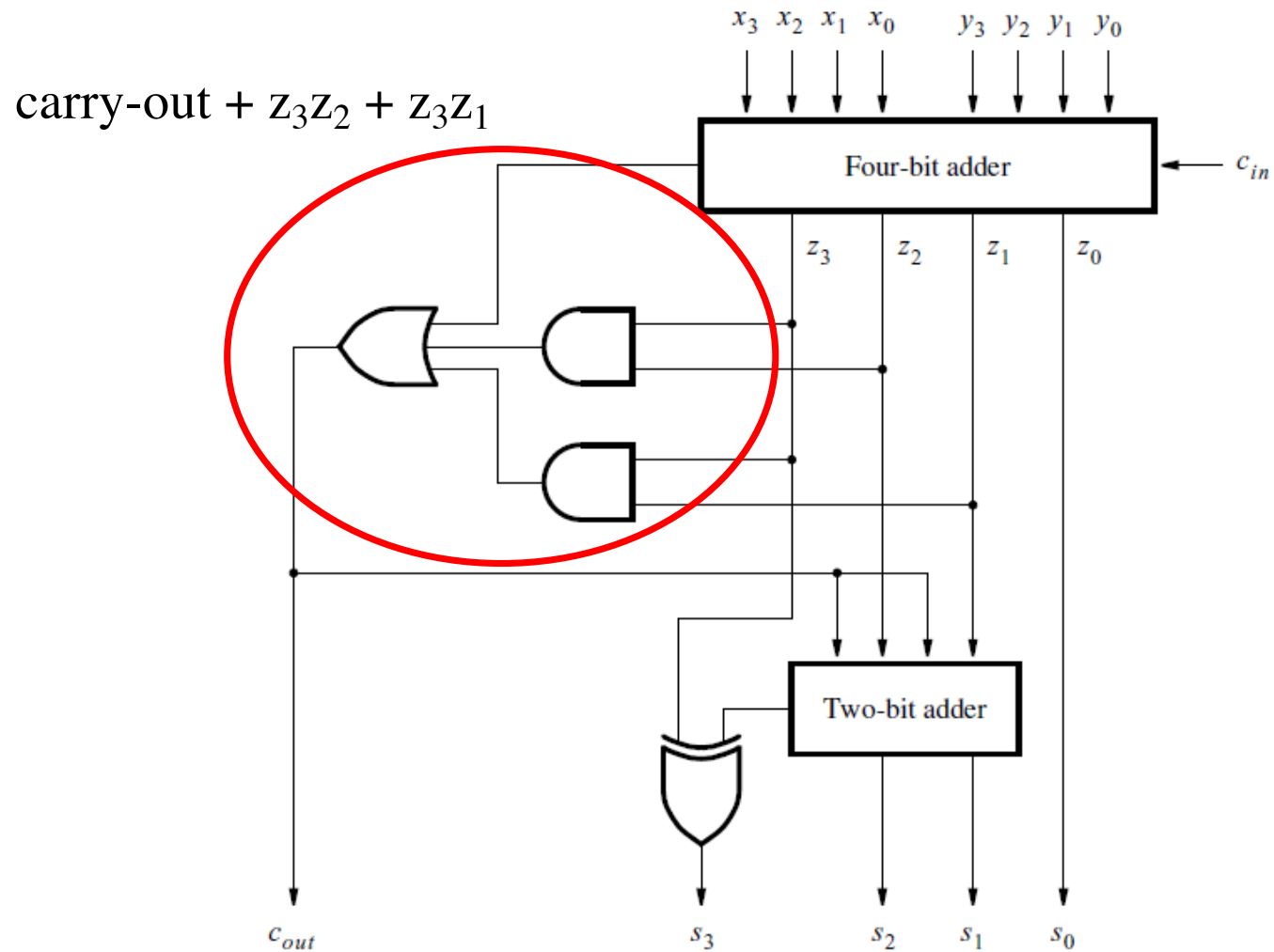
[Figure 3.40 in the textbook]

Circuit for a one-digit BCD adder



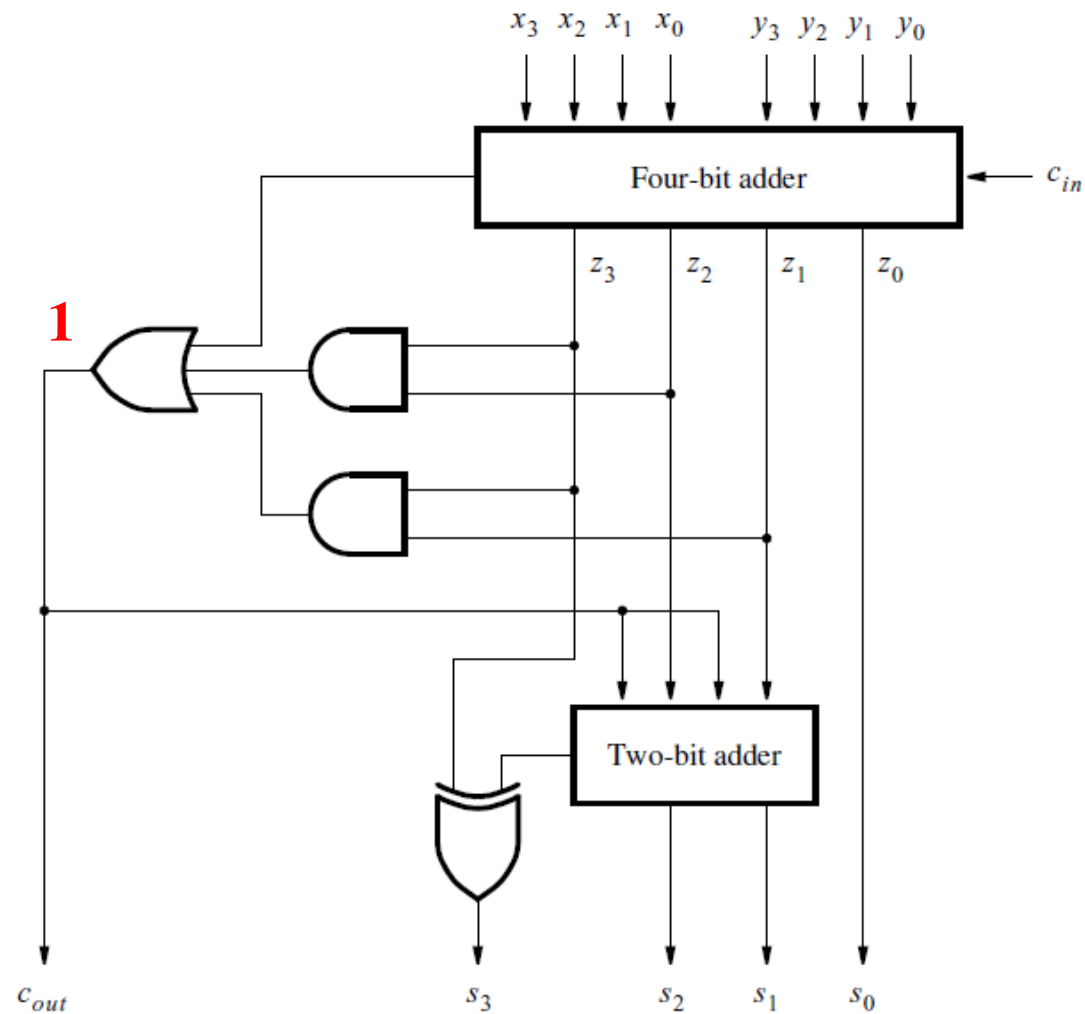
[Figure 3.41 in the textbook]

Circuit for a one-digit BCD adder



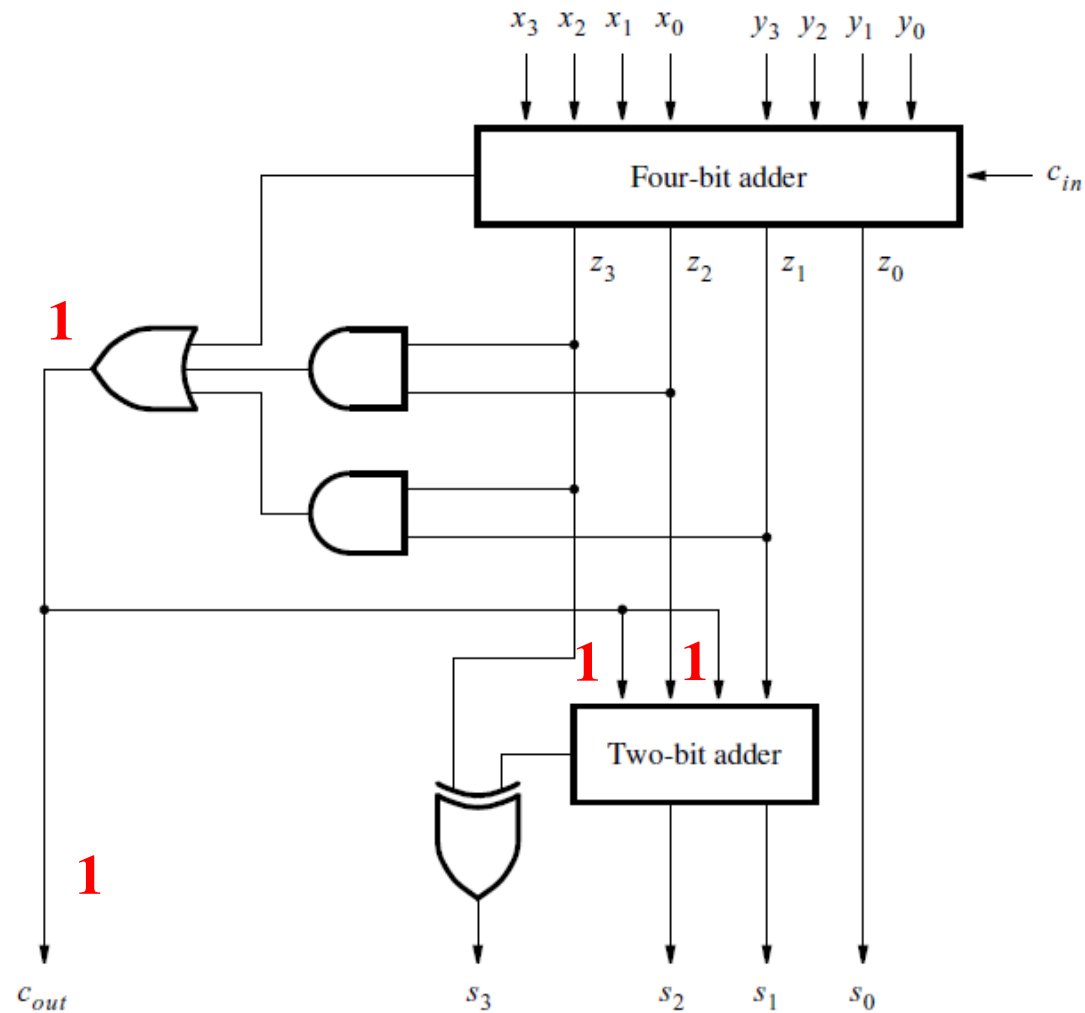
[Figure 3.41 in the textbook]

Circuit for a one-digit BCD adder



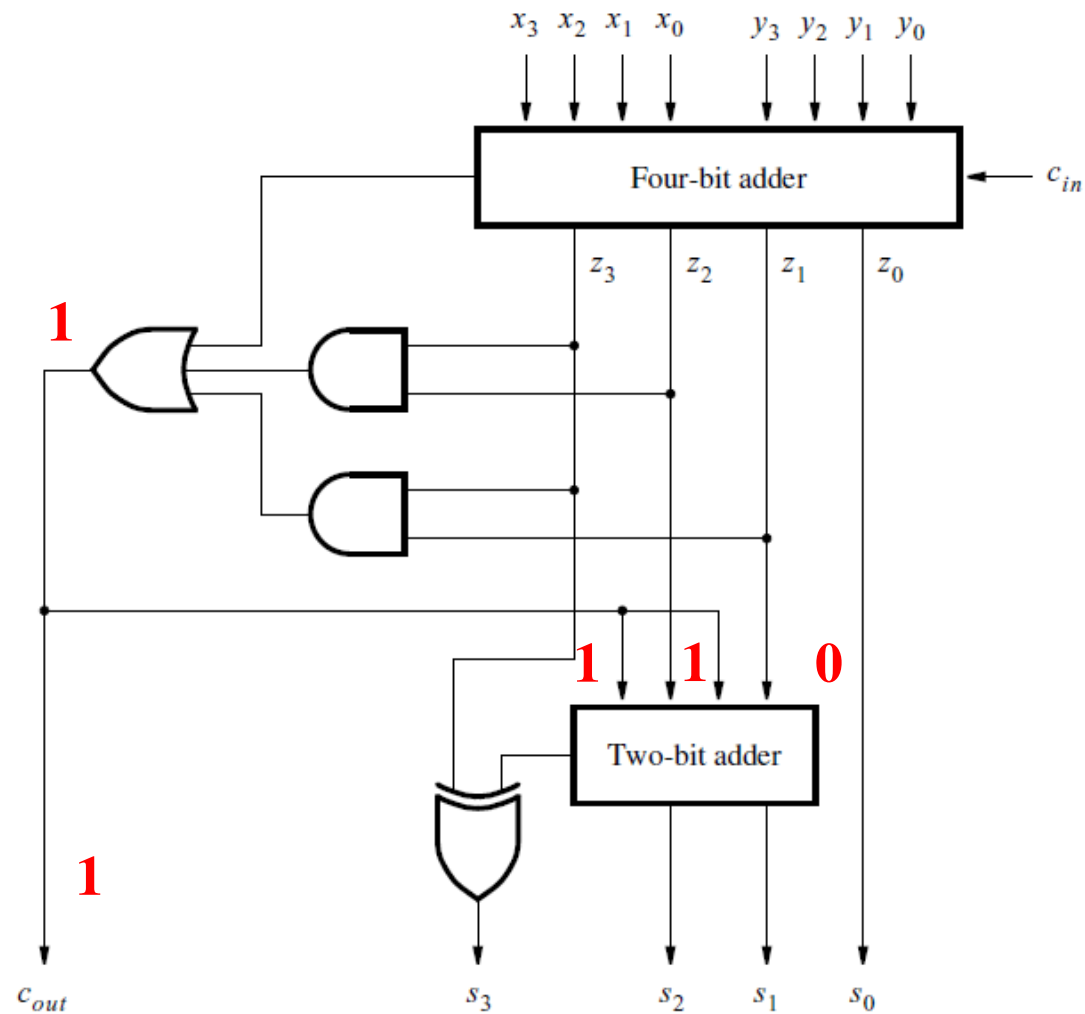
[Figure 3.41 in the textbook]

Circuit for a one-digit BCD adder



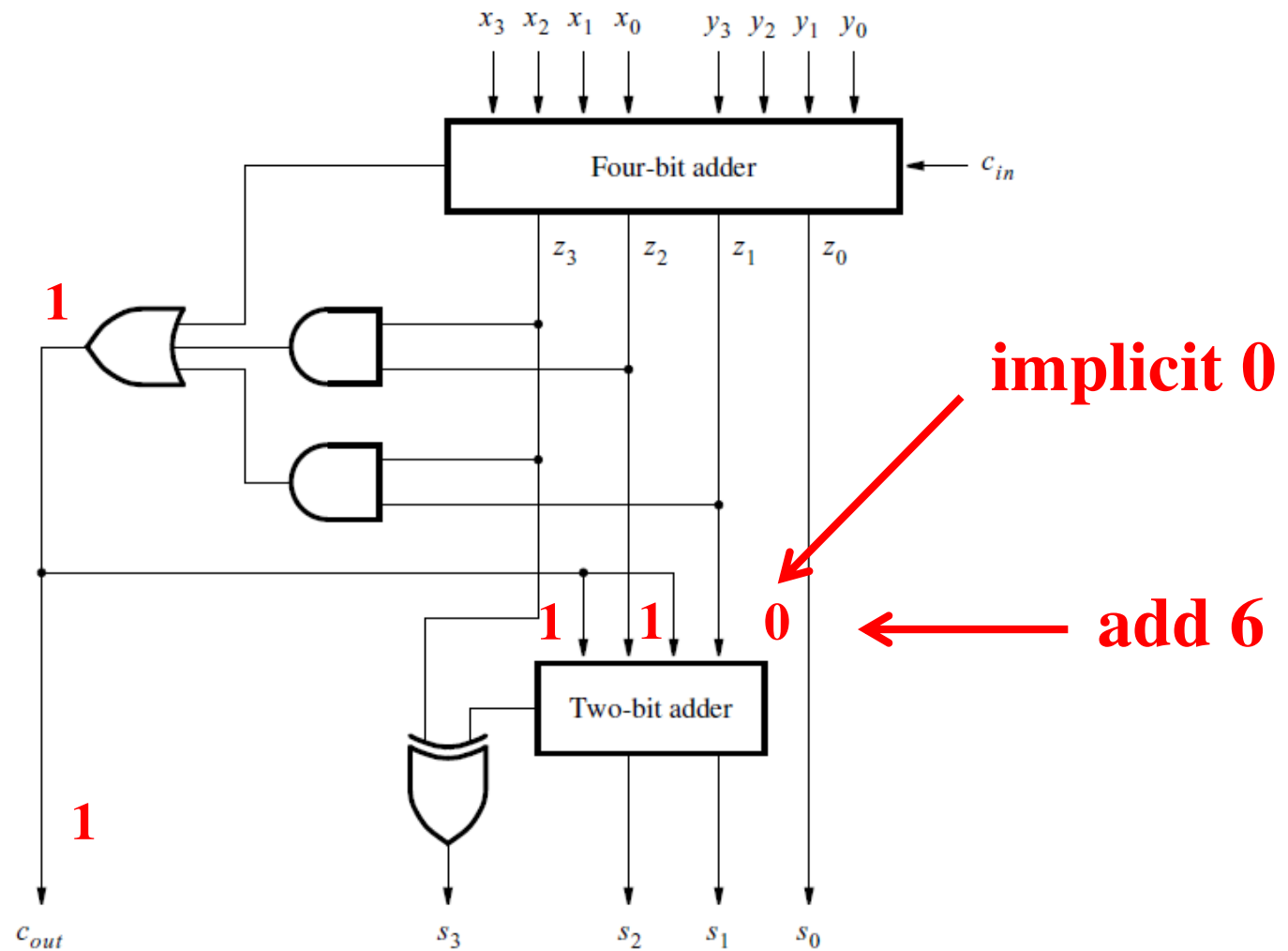
[Figure 3.41 in the textbook]

Circuit for a one-digit BCD adder



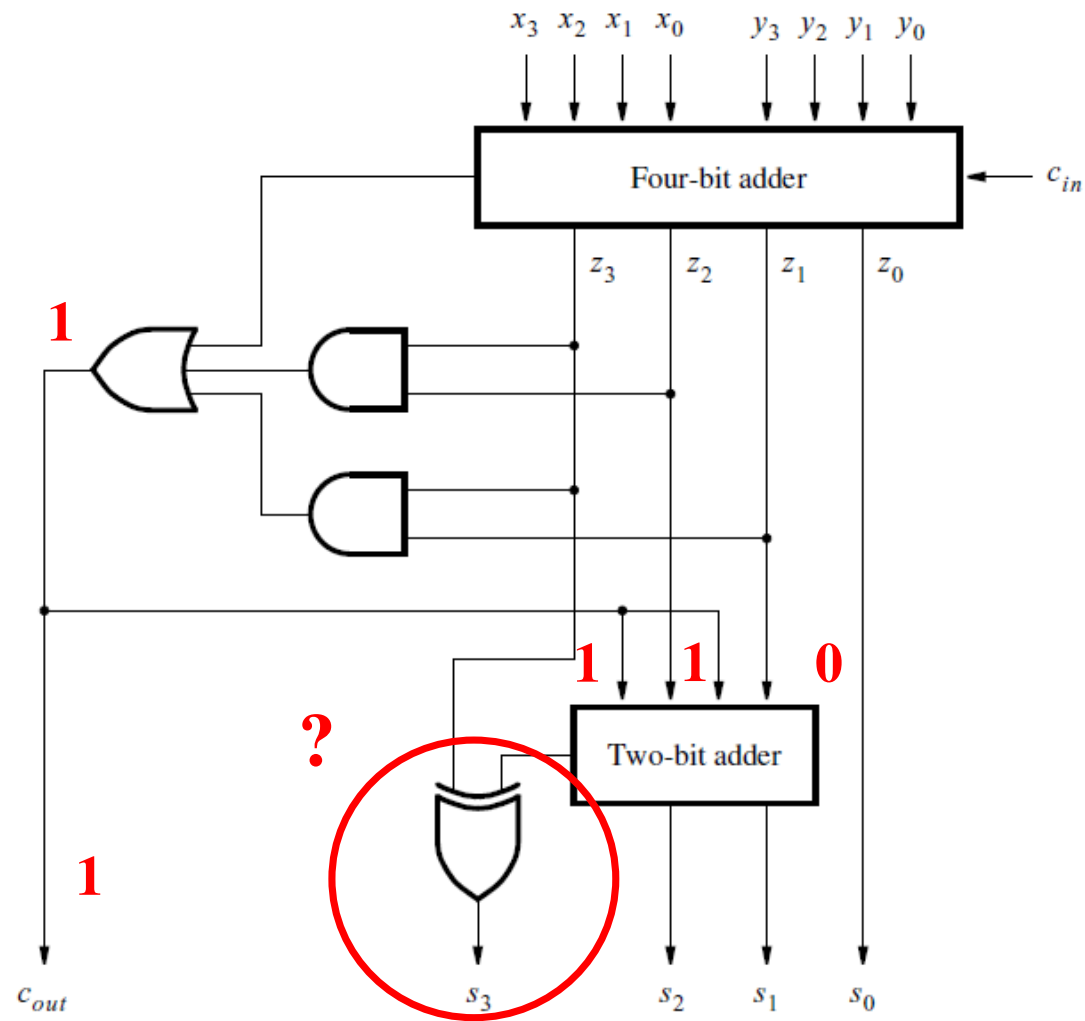
[Figure 3.41 in the textbook]

Circuit for a one-digit BCD adder



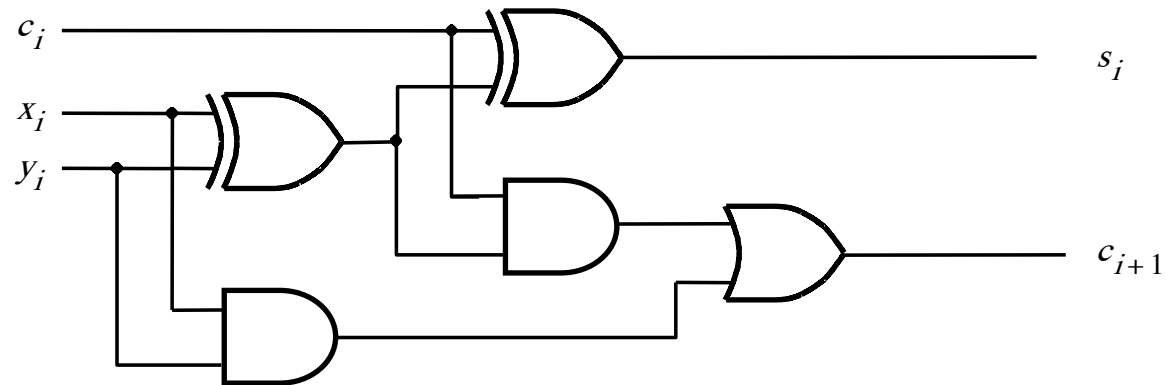
[Figure 3.41 in the textbook]

Circuit for a one-digit BCD adder



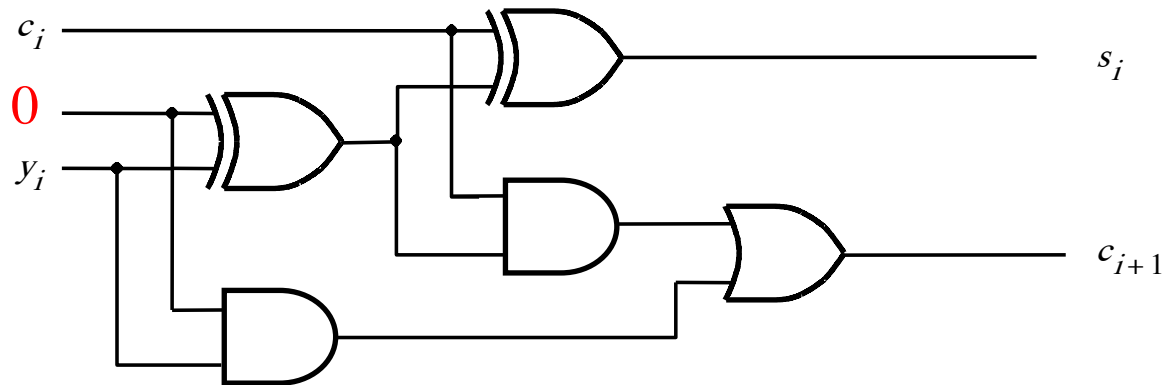
[Figure 3.41 in the textbook]

Simplification of the Full-Adder circuit when $x_i=0$

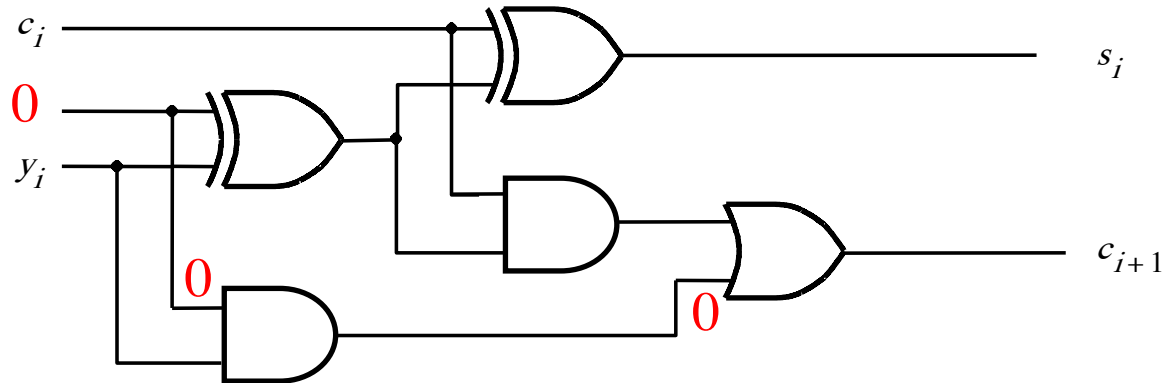


[Figure 3.4b from the textbook]

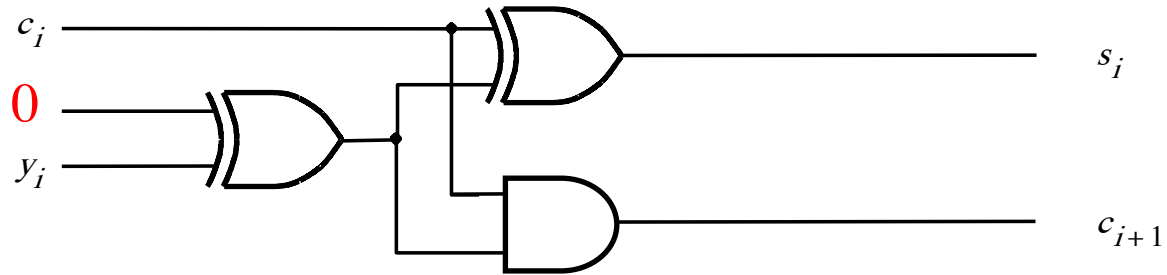
Simplification of the Full-Adder circuit when $x_i=0$



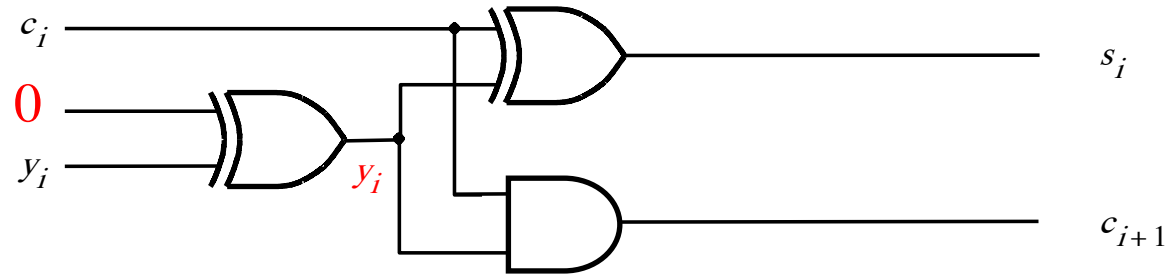
Simplification of the Full-Adder circuit when $x_i=0$



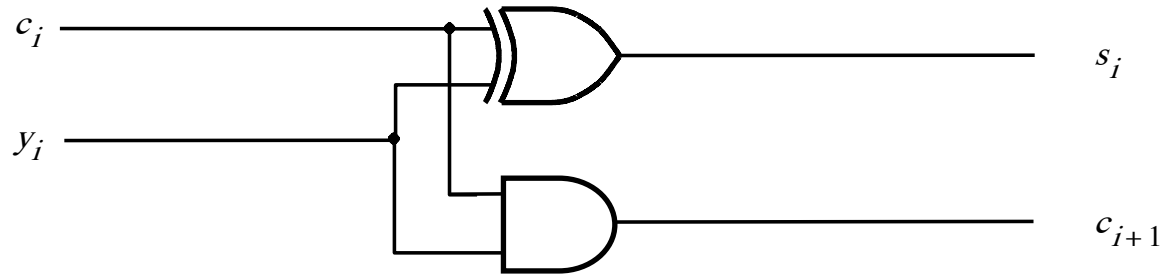
Simplification of the Full-Adder circuit when $x_i=0$



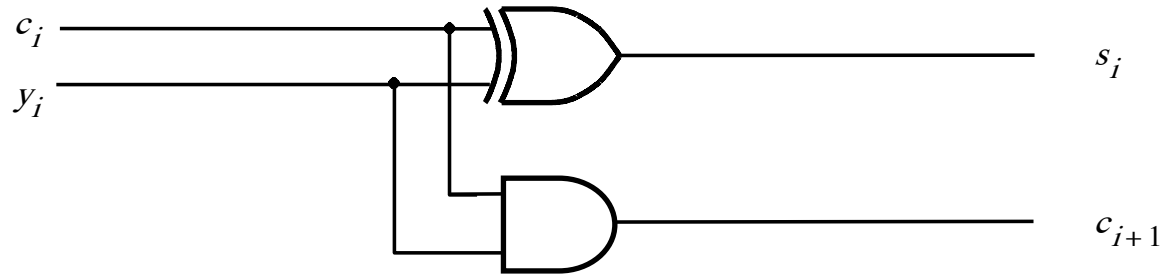
Simplification of the Full-Adder circuit when $x_i=0$



Simplification of the Full-Adder circuit when $x_i=0$

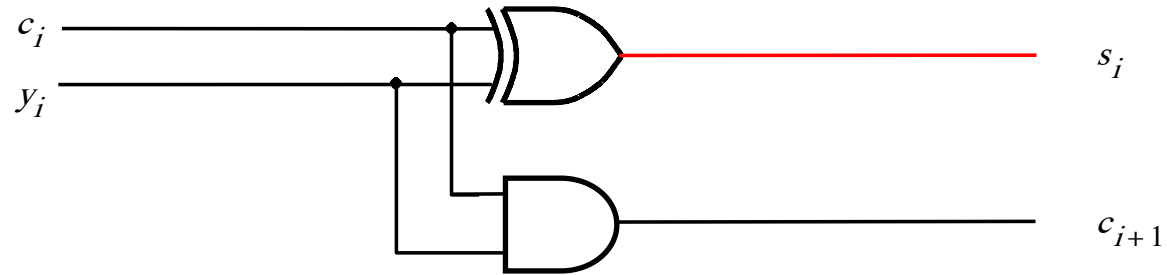


Simplification of the Full-Adder circuit when $x_i=0$



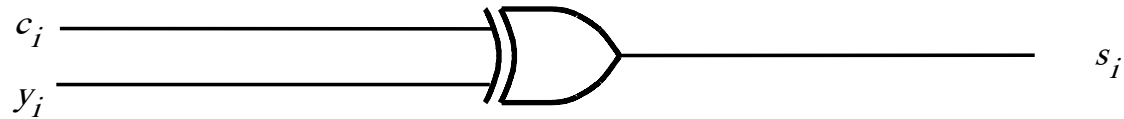
It reduces to a half-adder.

Simplification of the Full-Adder circuit when $x_i=0$



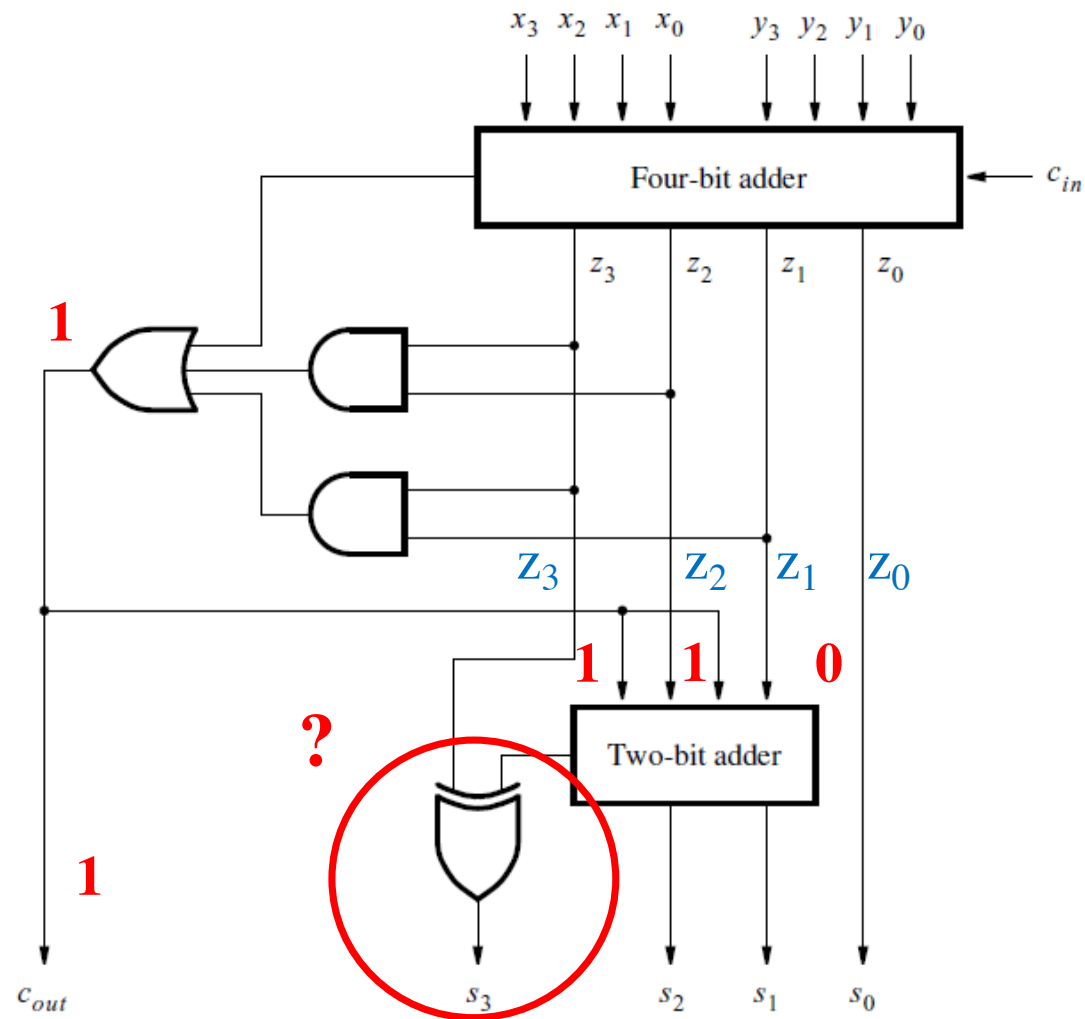
But if we only need the sum bit ...

Simplification of the Full-Adder circuit when $x_i=0$



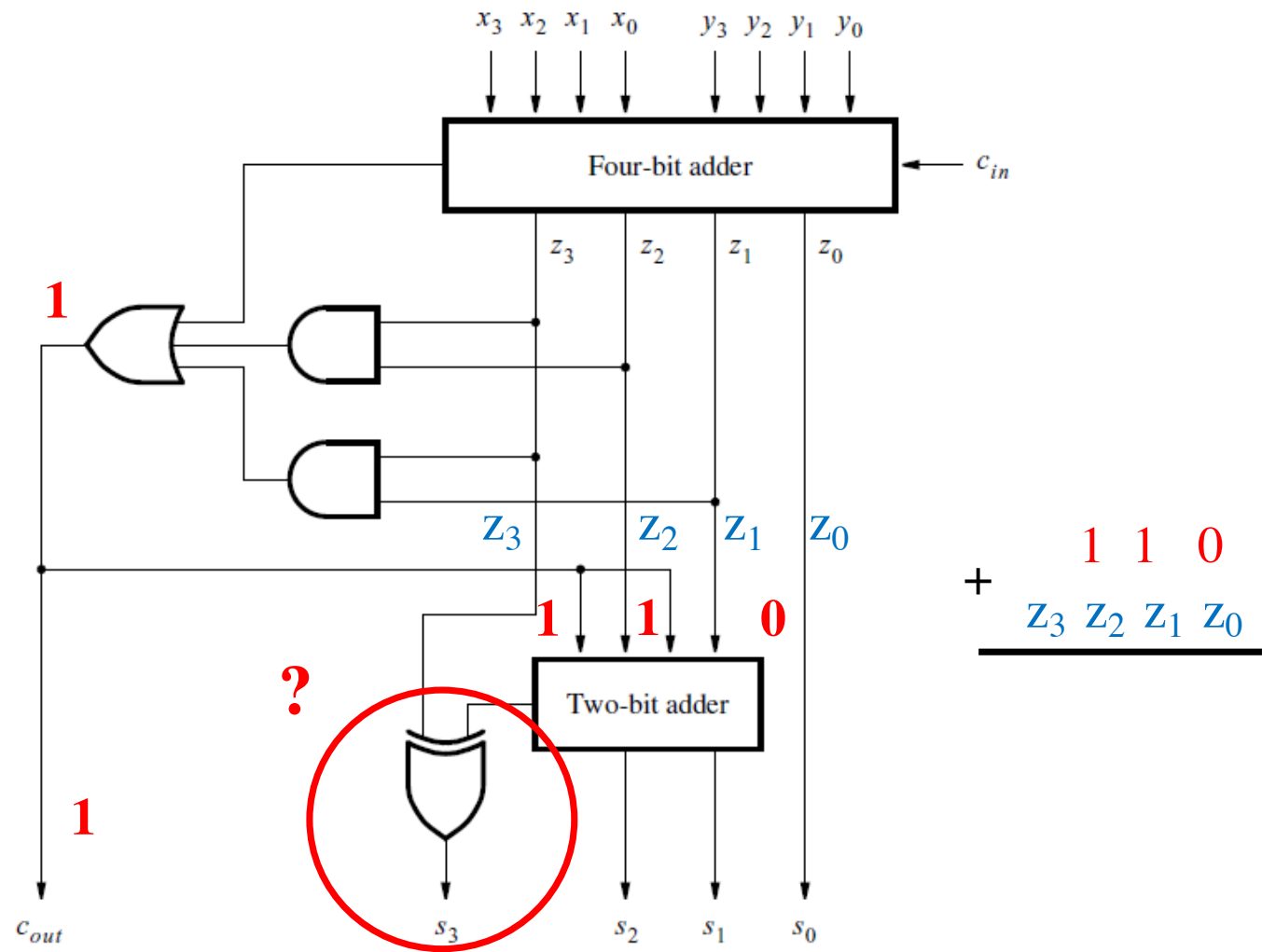
... it reduces to an XOR.

Circuit for a one-digit BCD adder



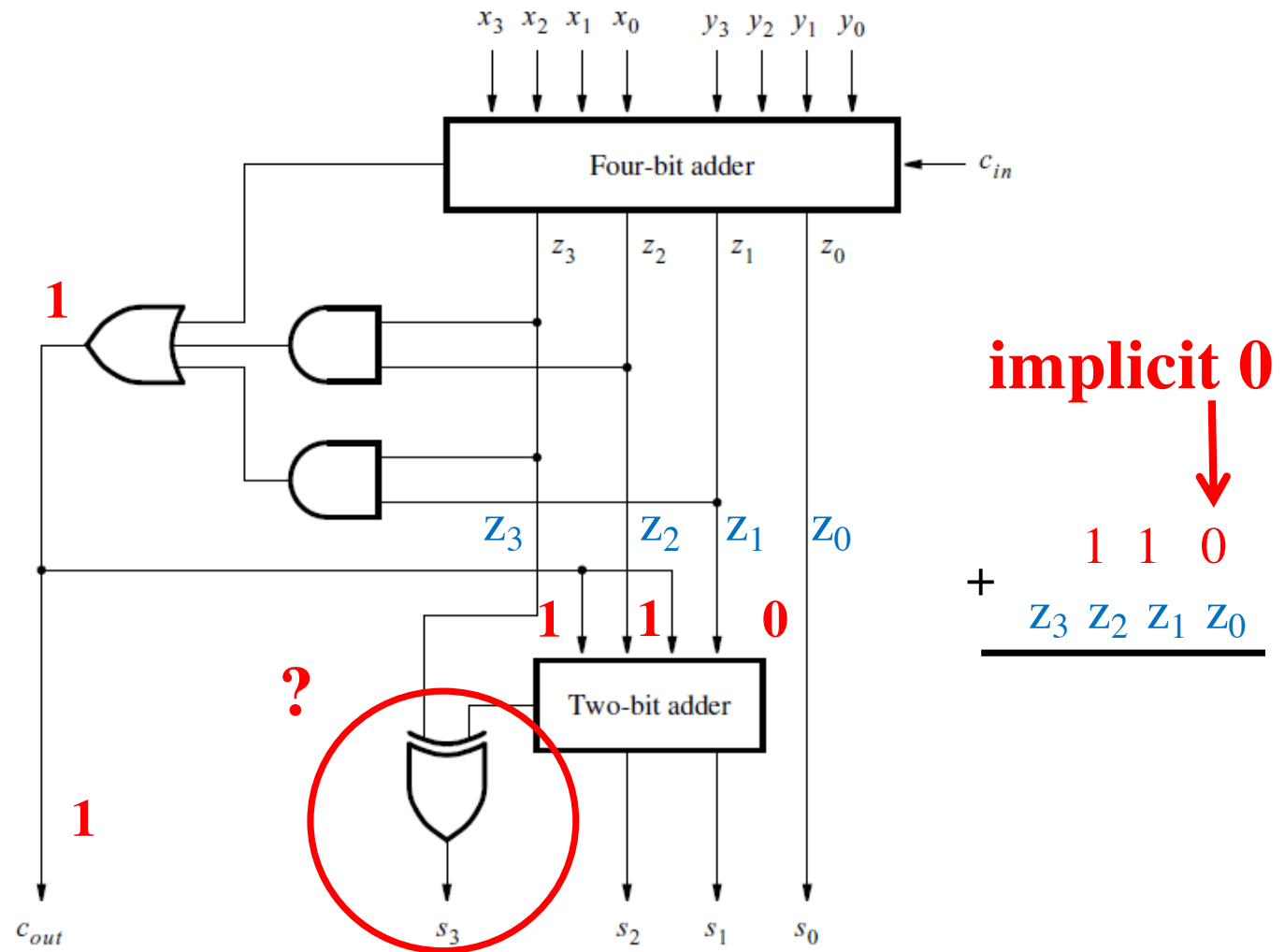
[Figure 3.41 in the textbook]

Circuit for a one-digit BCD adder



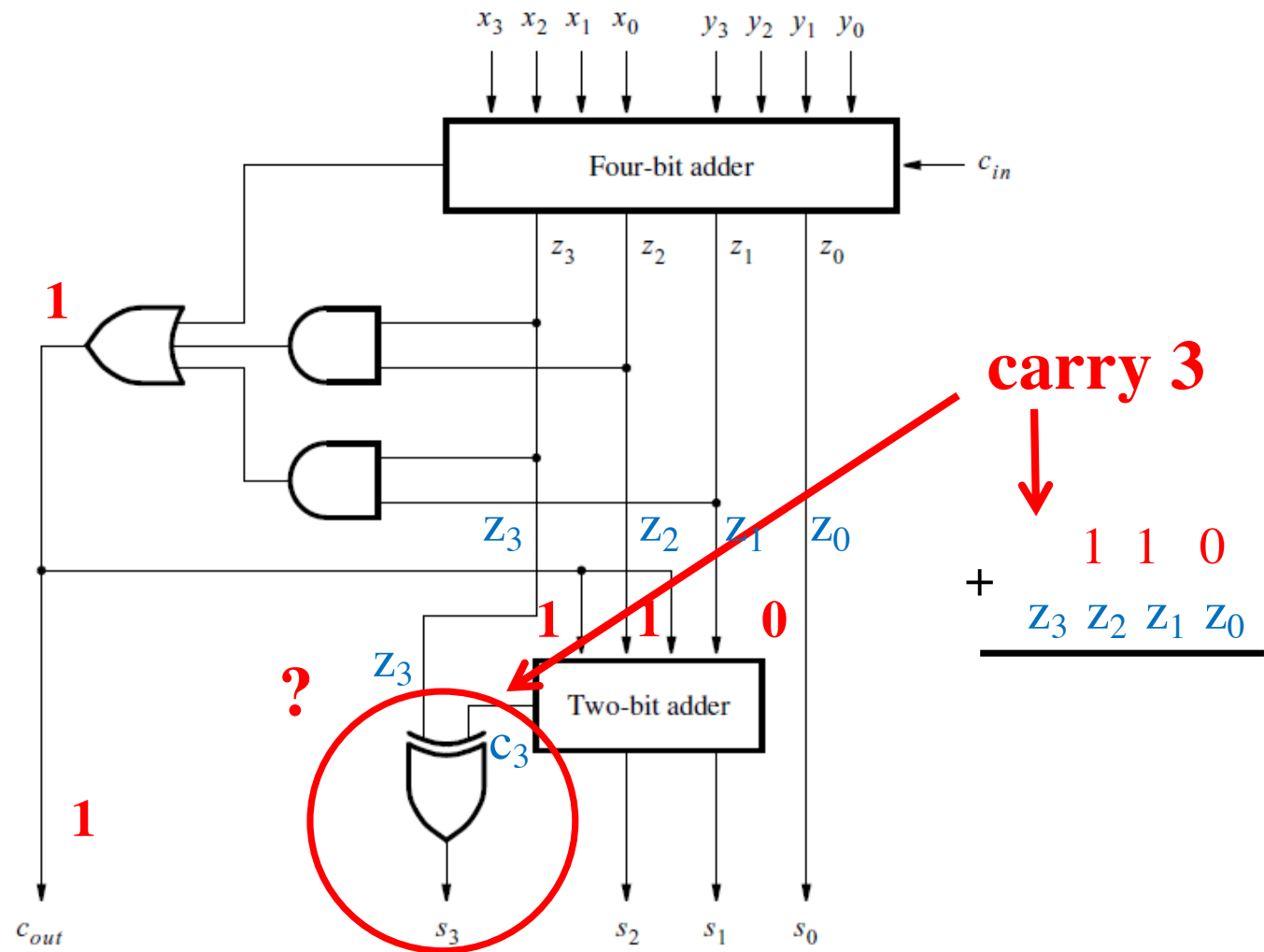
[Figure 3.41 in the textbook]

Circuit for a one-digit BCD adder



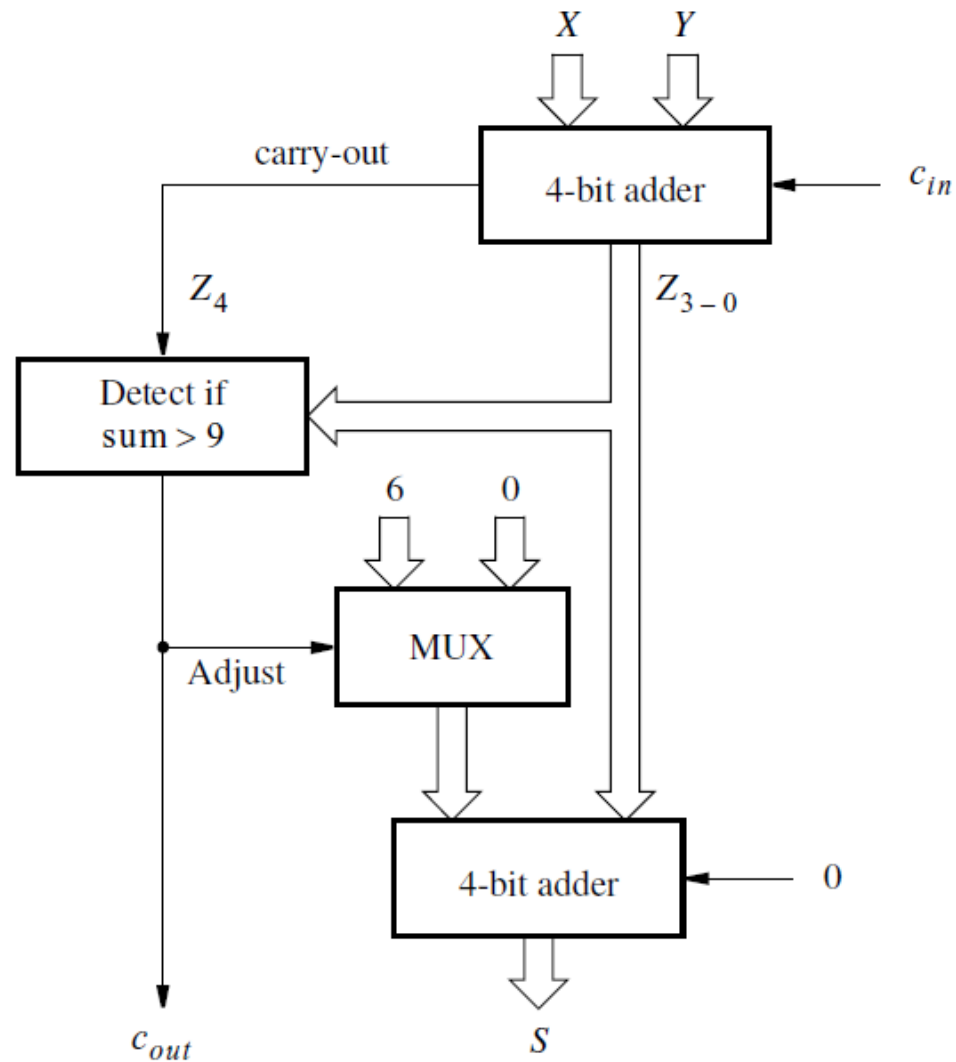
[Figure 3.41 in the textbook]

Circuit for a one-digit BCD adder



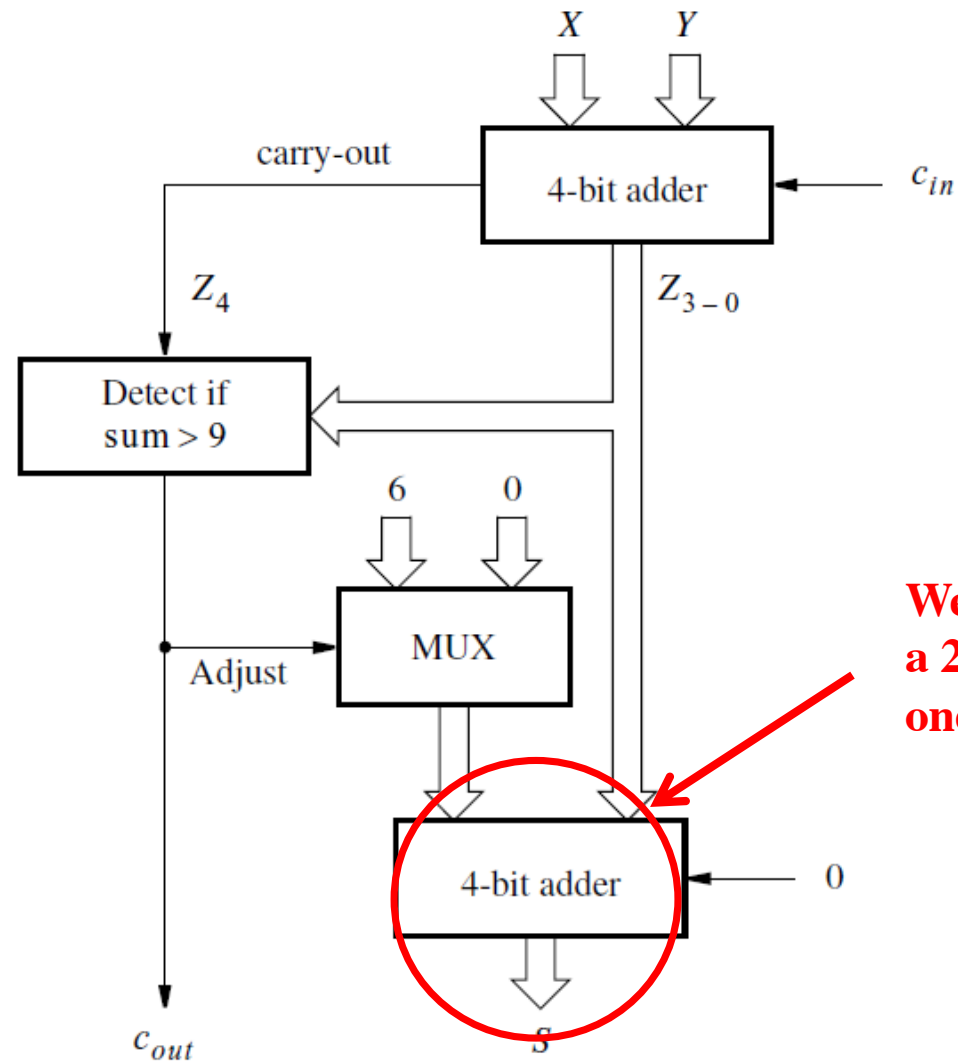
[Figure 3.41 in the textbook]

Block diagram for a one-digit BCD adder



[Figure 3.39 in the textbook]

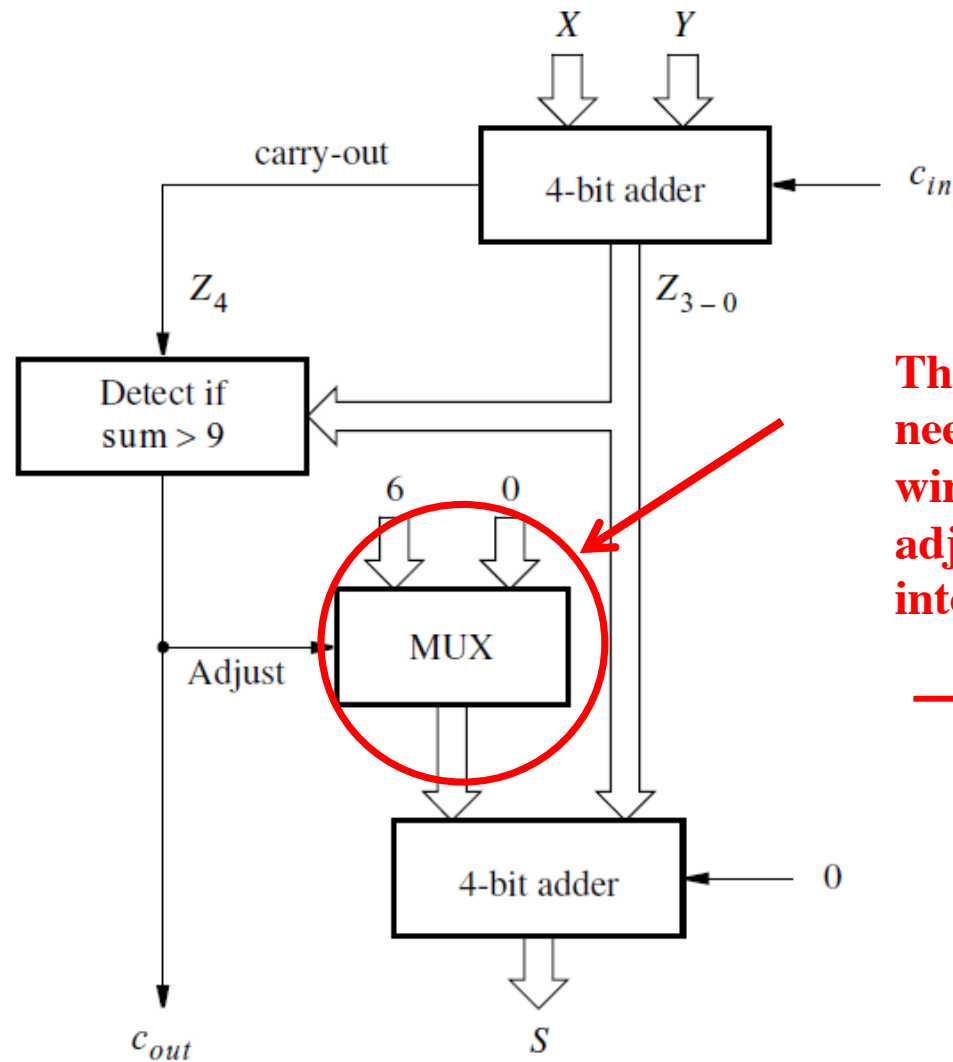
Block diagram for a one-digit BCD adder



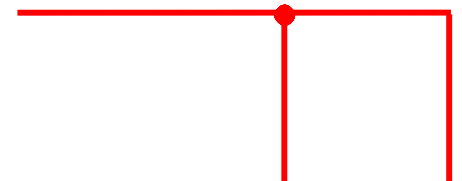
We can get by with a 2-bit adder and one XOR gate.

[Figure 3.39 in the textbook]

Block diagram for a one-digit BCD adder

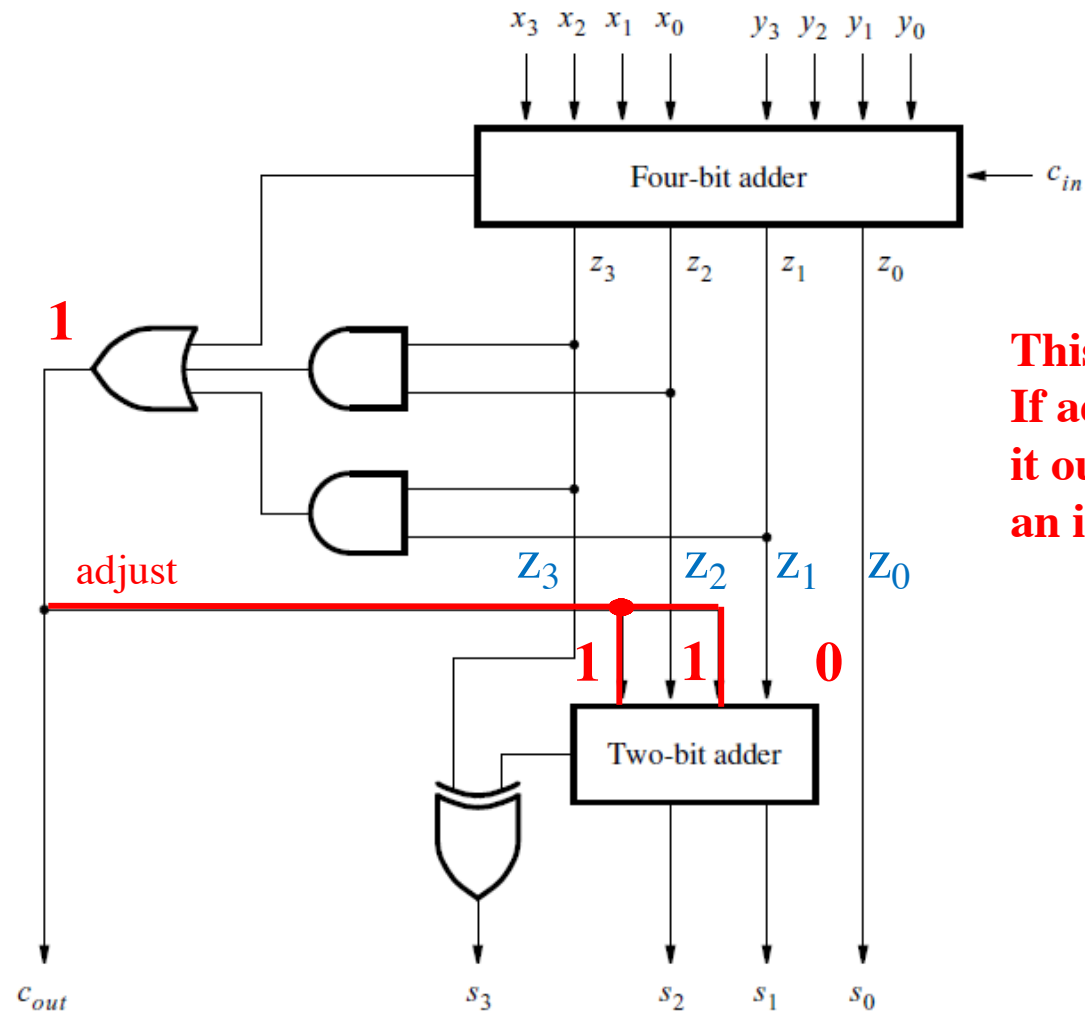


This bus mux is not needed. It reduces to 2 wires that carry the adjust signal and feed into the 2-bit adder.



[Figure 3.39 in the textbook]

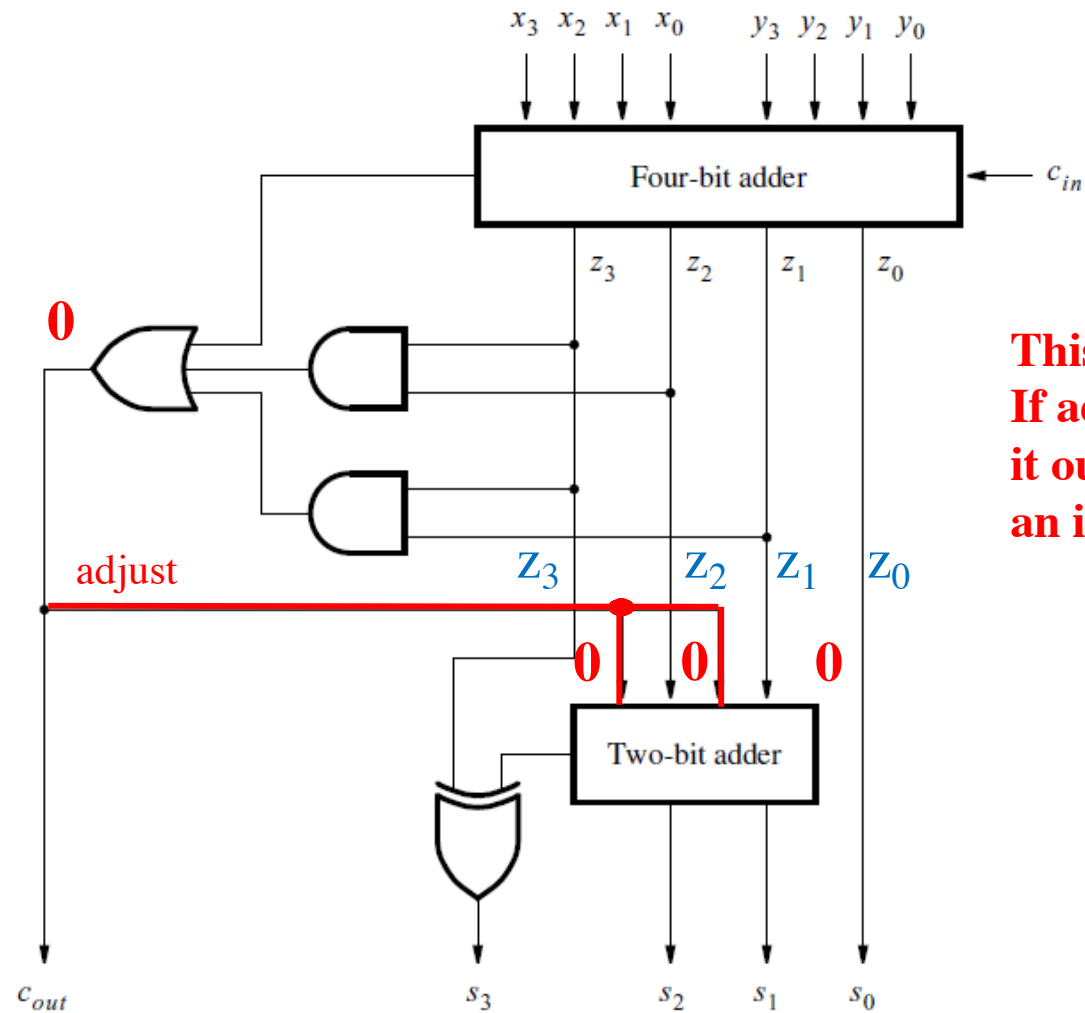
Circuit for a one-digit BCD adder



This is the bus mux!
If adjust is equal to one
it outputs +6, i.e., 11 (with
an implicit zero).

[Figure 3.41 in the textbook]

Circuit for a one-digit BCD adder



This is the bus mux!
If adjust is equal to zero
it outputs 0, i.e., 00 (with
an implicit zero).

[Figure 3.41 in the textbook]

Questions?

THE END