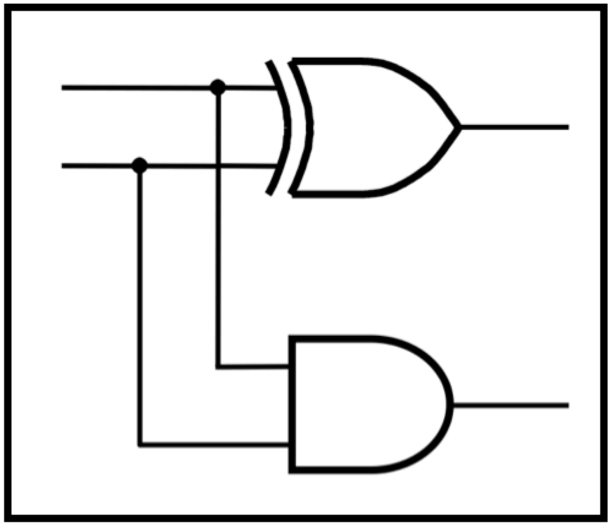


CprE 281: Digital Logic

Instructor: Alexander Stoytchev

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

Addition of Unsigned Numbers

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

Administrative Stuff

- **HW5 is due today**

Administrative Stuff

- **No homework due next week**

- **HW6 will be due on Monday, Oct 10**

Quick Review

Number Systems

$$N = d_n B^n + d_{n-1} B^{n-1} + \cdots + d_1 B^1 + d_0 B^0$$

Number Systems

$$N = d_n B^n + d_{n-1} B^{n-1} + \cdots + d_1 B^1 + d_0 B^0$$

n-th digit
(most significant)

0-th digit
(least significant)

Number Systems

base

power

$$N = d_n B^n + d_{n-1} B^{n-1} + \cdots + d_1 B^1 + d_0 B^0$$

n-th digit
(most significant)

0-th digit
(least significant)

The Decimal System

$$524_{10} = 5 \times 10^2 + 2 \times 10^1 + 4 \times 10^0$$

The Decimal System

$$524_{10} = 5\times10^2+2\times10^1+4\times10^0$$

$$= 5\times100+2\times10+4\times1$$

$$= 500+20+4$$

$$= 524_{10}$$

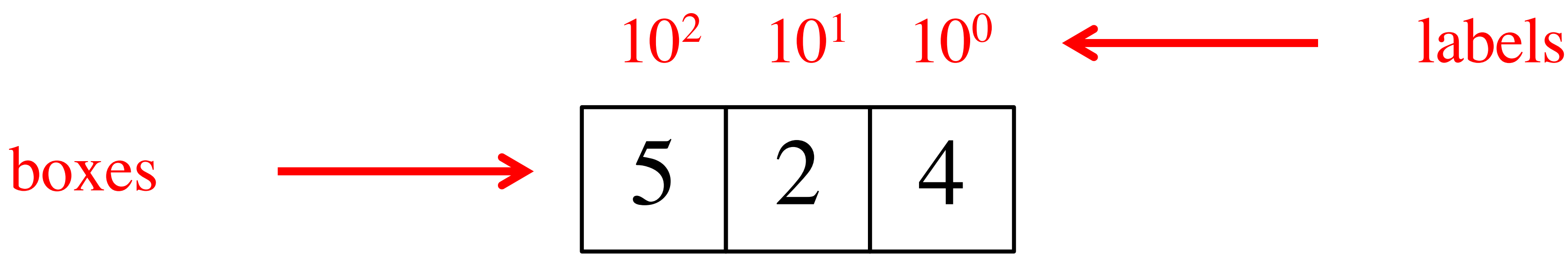
Another Way to Look at This

5	2	4

Another Way to Look at This

10^2	10^1	10^0
5	2	4

Another Way to Look at This



Each box can contain only one digit and has only one label. From right to left, the labels are increasing powers of the base, starting from 0.

Base 7

$$524_7 = 5 \times 7^2 + 2 \times 7^1 + 4 \times 7^0$$

Base 7

base power

$$524_7 = 5 \times 7^2 + 2 \times 7^1 + 4 \times 7^0$$

Base 7

base

power

$$524_7 = 5 \times 7^2 + 2 \times 7^1 + 4 \times 7^0$$

most significant digit

least significant digit

Base 7

$$\begin{aligned} 524_7 &= 5 \times 7^2 + 2 \times 7^1 + 4 \times 7^0 \\ &= 5 \times 49 + 2 \times 7 + 4 \times 1 \\ &= 245 + 14 + 4 \\ &= 263_{10} \end{aligned}$$

Another Way to Look at This

7^2	7^1	7^0		10^2	10^1	10^0
5	2	4	=	2	6	3

Binary Numbers (Base 2)

$$1001_2 = \quad 1 \times 2^3 \quad + \quad 0 \times 2^2 \quad + \quad 0 \times 2^1 \quad + \quad 1 \times 2^0$$

Binary Numbers (Base 2)

base

power

$$1001_2 = 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

most significant bit

least significant bit

Binary Numbers (Base 2)

$$
\begin{aligned}
1001_2 = \;& 1 \times 2^3 && + \; 0 \times 2^2 && + \; 0 \times 2^1 && + \; 1 \times 2^0 && = \\
=\;& 1 \times 8 && + \; 0 \times 4 && + \; 0 \times 2 && + \; 1 \times 1 && = \\
=\;& 8 && + \; 0 && + \; 0 && + \; 1 && = \\
=\;& 9_{10}
\end{aligned}
$$

Another Example

$$\begin{aligned}
11101_2 &= 1\times 2^4 + 1\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0 = \\
&= 1\times 16 + 1\times 8 + 1\times 4 + 0\times 2 + 1\times 1 = \\
&= 16 + 8 + 4 + 0 + 1 = 29_{10}
\end{aligned}$$

Powers of 2

$$
\begin{aligned}
2^{10} &= 1024 \\
2^{9} &= 512 \\
2^{8} &= 256 \\
2^{7} &= 128 \\
2^{6} &= 64 \\
2^{5} &= 32 \\
2^{4} &= 16 \\
2^{3} &= 8 \\
2^{2} &= 4 \\
2^{1} &= 2 \\
2^{0} &= 1
\end{aligned}
$$

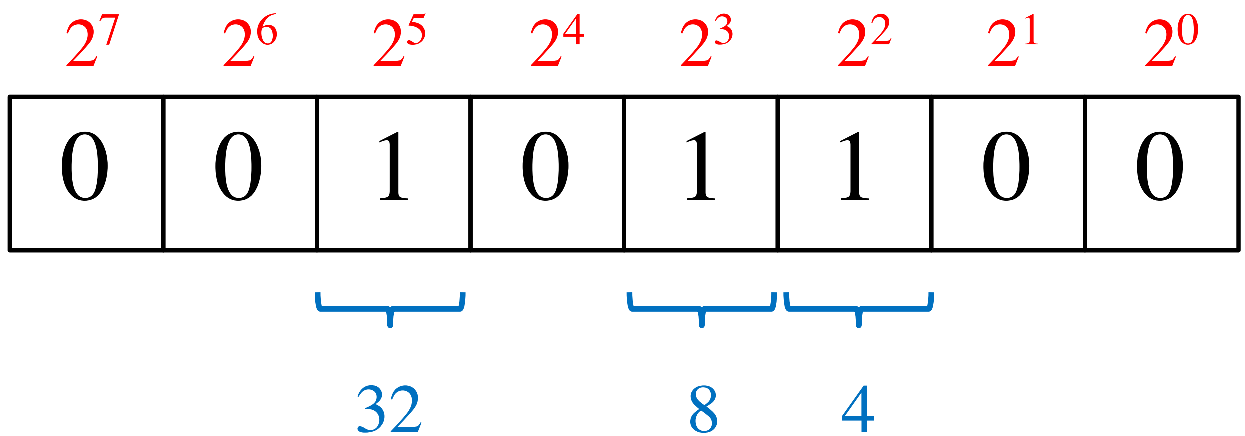
What is the value of this binary number?

- 0 0 1 0 1 1 0 0
- 0 0 1 0 1 1 0 0
- $0*2^7 + 0*2^6 + 1*2^5 + 0*2^4 + 1*2^3 + 1*2^2 + 0*2^1 + 0*2^0$
- 0*128 + 0*64 + 1*32 + 0*16 + 1*8 + 1*4 + 0*2 + 0*1
- 0*128 + 0*64 + **1*32** + 0*16 **+ 1*8 + 1*4** + 0*2 + 0*1
- 32+ 8 + 4 = 44 (in decimal)

Another Way to Look at This

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	1	0	1	1	0	0

Another Way to Look at This



Signed v.s. Unsigned Numbers

Two Different Types of Binary Numbers

Unsigned numbers

- **All bits jointly represent a positive integer.**
- **Negative numbers cannot be represented this way.**

Signed numbers

- **The left-most bit represents the sign of the number.**
- **If that bit is 0, then the number is positive.**
- **If that bit is 1, then the number is negative.**
- **The magnitude of the largest number that can be represented in this way is twice smaller than the largest number in the unsigned representation.**

Unsigned Representation

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	1	0	1	1	0	0

This represents + 44.

Unsigned Representation

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	0	1	0	1	1	0	0

This represents + 172.

Signed Representation (using the left-most bit as the sign)

sign	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	1	0	1	1	0	0

This represents + 44.

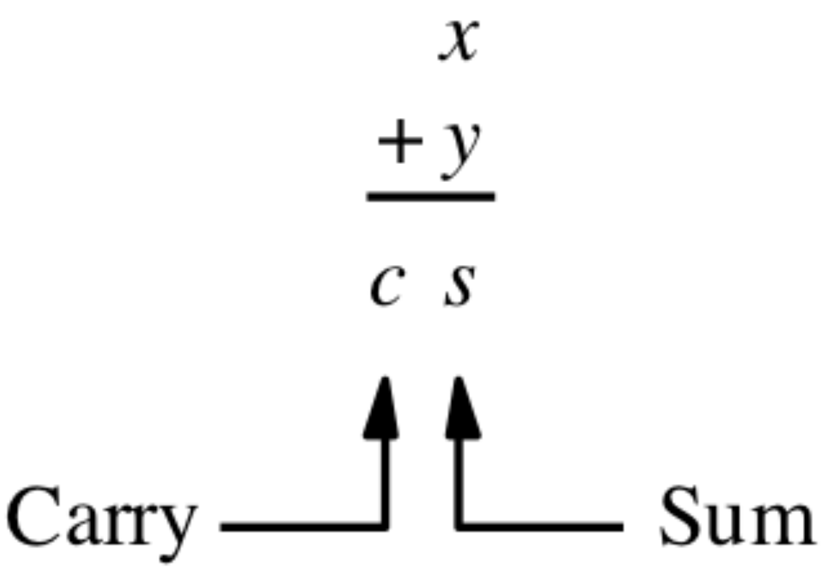
Signed Representation (using the left-most bit as the sign)

sign	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	0	1	0	1	1	0	0

This represents – 44.

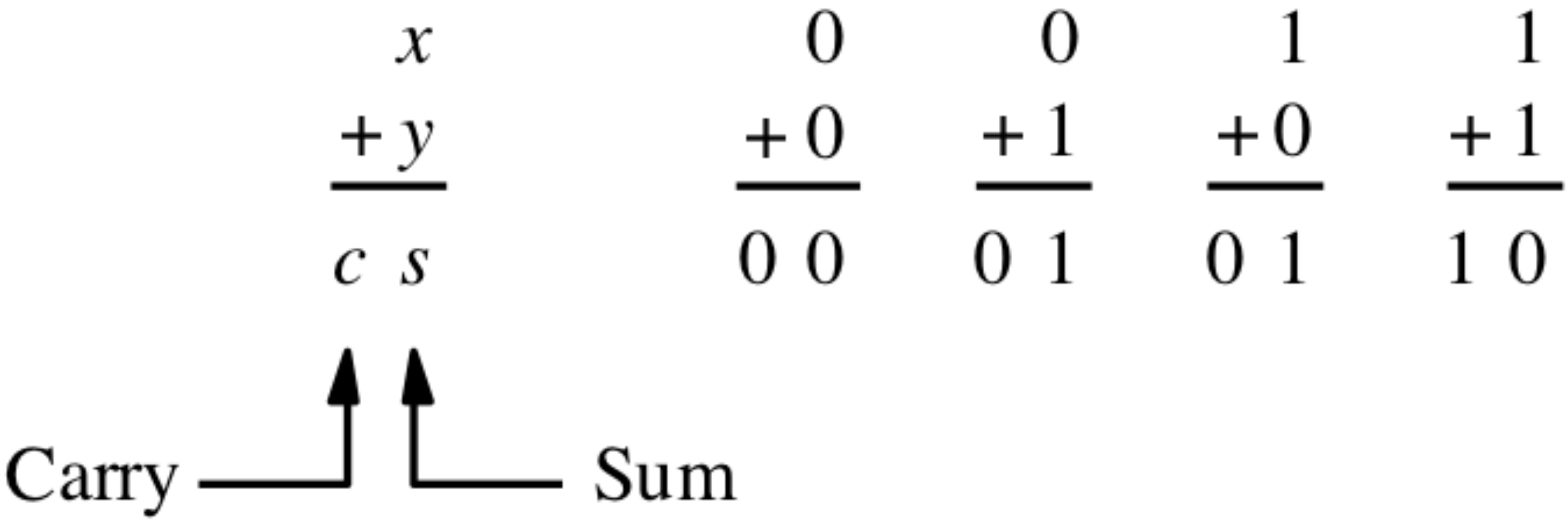
Today’s Lecture is About Addition of Unsigned Numbers

Addition of two 1-bit numbers



[Figure 3.1a from the textbook]

Addition of two 1-bit numbers
(there are four possible cases)



[Figure 3.1a from the textbook]

Addition of two 1-bit numbers (the truth table)

x	y	Carry c	Sum s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

[Figure 3.1b from the textbook]

Addition of two 1-bit numbers

x	0	0	1	1
$+\,y$	$+\,0$	$+\,1$	$+\,0$	$+\,1$
$c\ s$	0 0	0 1	0 1	1 0

[Figure 2.12 from the textbook]

Addition of two 1-bit numbers

$$\begin{array}{r} x \\ +\,y \\ \hline c\;s \end{array} \qquad \begin{array}{r} 0 \\ +\,0 \\ \hline 0\;0 \end{array} \qquad \begin{array}{r} 0 \\ +\,1 \\ \hline 0\;1 \end{array} \qquad \begin{array}{r} 1 \\ +\,0 \\ \hline 0\;1 \end{array} \qquad \begin{array}{r} 1 \\ +\,1 \\ \hline 1\;0 \end{array}$$

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

x	0	0	1	1
$+y$	+0	+1	+0	+1
$c\ s$	0 0	0 1	0 1	1 0

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

x	0	0	1	1
$+\,y$	$+\,0$	$+\,1$	$+\,0$	$+\,1$
$c\ s$	0 0	0 1	0 1	1 0

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

$$\begin{array}{r} x \\ +\,y \\ \hline c\ \ s \end{array} \qquad \begin{array}{r} 0 \\ +\,0 \\ \hline 0\ 0 \end{array} \qquad \begin{array}{r} 0 \\ +\,1 \\ \hline 0\ 1 \end{array} \qquad \begin{array}{r} 1 \\ +\,0 \\ \hline 0\ 1 \end{array} \qquad \begin{array}{r} 1 \\ +\,1 \\ \hline 1\ 0 \end{array}$$

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

x	0	0	1	1
$+\,y$	$+\,0$	$+\,1$	$+\,0$	$+\,1$
$c\ \ s$	0 0	0 1	0 1	1 0

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

x	0	0	1	1
$+\,y$	$+\,0$	$+\,1$	$+\,0$	$+\,1$
$c\ s$	0 0	0 1	0 1	1 0

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

$$\begin{array}{r} x \\ +\,y \\ \hline c\ \ s \end{array} \qquad \begin{array}{r} 0 \\ +\,0 \\ \hline 0\ 0 \end{array} \qquad \begin{array}{r} 0 \\ +\,1 \\ \hline 0\ 1 \end{array} \qquad \begin{array}{r} 1 \\ +\,0 \\ \hline 0\ 1 \end{array} \qquad \begin{array}{r} 1 \\ +\,1 \\ \hline 1\ 0 \end{array}$$

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

$$
\begin{array}{r}
x \\
+\,y \\
\hline
c\;s
\end{array}
\qquad
\begin{array}{r}
0 \\
+\,0 \\
\hline
0\;0
\end{array}
\qquad
\begin{array}{r}
0 \\
+\,1 \\
\hline
0\;1
\end{array}
\qquad
\begin{array}{r}
1 \\
+\,0 \\
\hline
0\;1
\end{array}
\qquad
\begin{array}{r}
1 \\
+\,1 \\
\hline
1\;0
\end{array}
$$

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

$$\begin{array}{r} x \\ +\,y \\ \hline c\;s \end{array} \qquad \begin{array}{r} 0 \\ +\,0 \\ \hline 0\;0 \end{array} \qquad \begin{array}{r} 0 \\ +\,1 \\ \hline 0\;1 \end{array} \qquad \begin{array}{r} 1 \\ +\,0 \\ \hline 0\;1 \end{array} \qquad \begin{array}{r} 1 \\ +\,1 \\ \hline 1\;0 \end{array}$$

x	*y*	*c*	*s*
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

$$\begin{array}{r} x \\ +\,y \\ \hline c\ s \end{array} \qquad \begin{array}{r} 0 \\ +\,0 \\ \hline 0\ 0 \end{array} \qquad \begin{array}{r} 0 \\ +\,1 \\ \hline 0\ 1 \end{array} \qquad \begin{array}{r} 1 \\ +\,0 \\ \hline 0\ 1 \end{array} \qquad \begin{array}{r} 1 \\ +\,1 \\ \hline 1\ 0 \end{array}$$

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

$$\begin{array}{r} x \\ +\,y \\ \hline c\ \boxed{s} \end{array} \qquad \begin{array}{r} 0 \\ +\,0 \\ \hline 0\ \boxed{0} \end{array} \qquad \begin{array}{r} 0 \\ +\,1 \\ \hline 0\ \boxed{1} \end{array} \qquad \begin{array}{r} 1 \\ +\,0 \\ \hline 0\ \boxed{1} \end{array} \qquad \begin{array}{r} 1 \\ +\,1 \\ \hline 1\ \boxed{0} \end{array}$$

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

$$\begin{array}{r} x \\ +\,y \\ \hline c\ s \end{array} \qquad \begin{array}{r} 0 \\ +\,0 \\ \hline 0\ 0 \end{array} \qquad \begin{array}{r} 0 \\ +\,1 \\ \hline 0\ 1 \end{array} \qquad \begin{array}{r} 1 \\ +\,0 \\ \hline 0\ 1 \end{array} \qquad \begin{array}{r} 1 \\ +\,1 \\ \hline 1\ 0 \end{array}$$

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

$$\begin{array}{r} x \\ +\,y \\ \hline c\ s \end{array} \qquad \begin{array}{r} 0 \\ +\,0 \\ \hline 0\ 0 \end{array} \qquad \begin{array}{r} 0 \\ +\,1 \\ \hline 0\ 1 \end{array} \qquad \begin{array}{r} 1 \\ +\,0 \\ \hline 0\ 1 \end{array} \qquad \begin{array}{r} 1 \\ +\,1 \\ \hline 1\ 0 \end{array}$$

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

?

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

AND

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

x	y	c	? s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

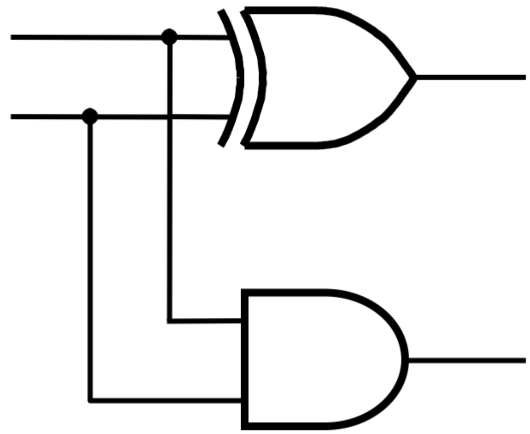
Addition of two 1-bit numbers

x	y	c	s (XOR)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers

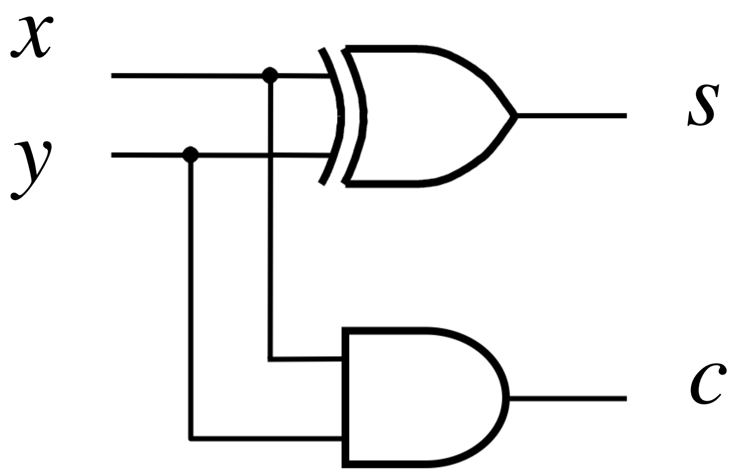
x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers



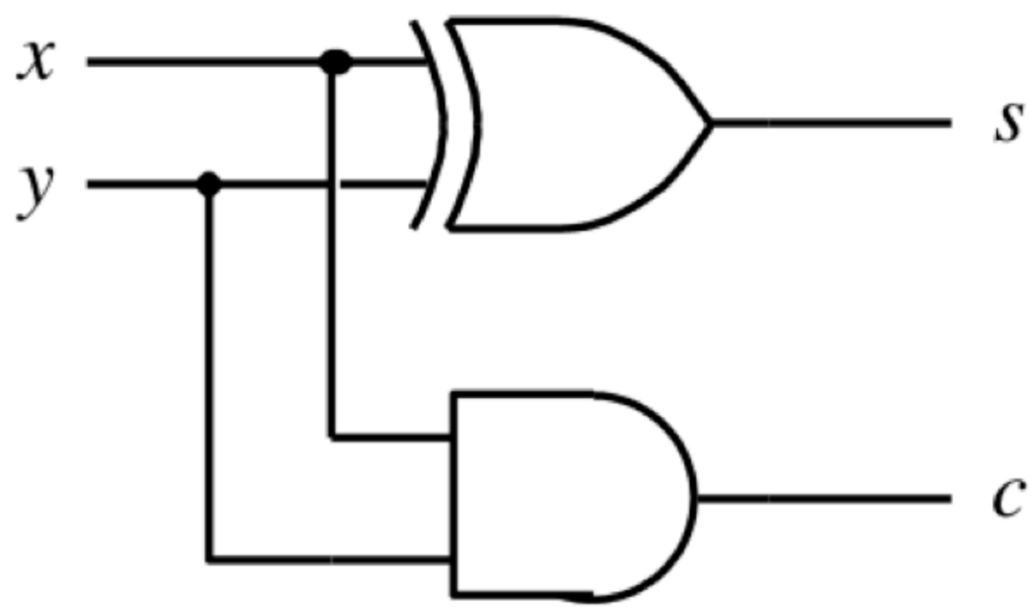
x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers



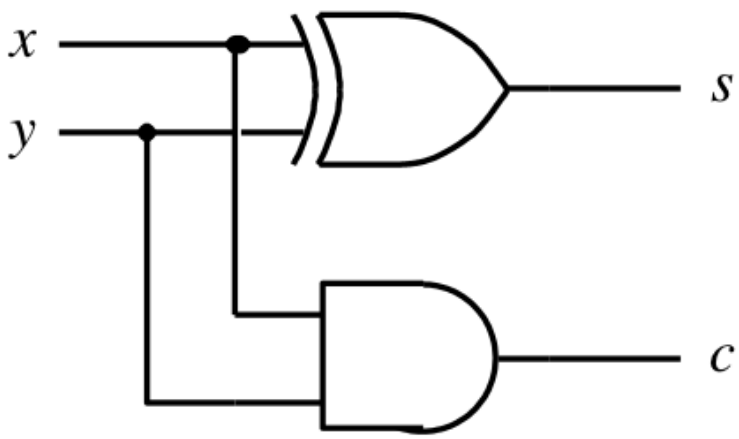
x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Addition of two 1-bit numbers (the logic circuit)



[Figure 3.1c from the textbook]

The Half-Adder



(c) Circuit

(d) Graphical symbol

[Figure 3.1c-d from the textbook]

Addition of Multibit Unsigned Numbers

Analogy with addition in base 10

$$
\begin{array}{r}
x_2\ x_1\ x_0 \\
+\quad y_2\ y_1\ y_0 \\
\hline
s_2\ s_1\ s_0
\end{array}
$$

Analogy with addition in base 10

$$\begin{array}{r} 3\ 8\ 9 \\ +\ 1\ 5\ 7 \\ \hline 5\ 4\ 6 \end{array}$$

Analogy with addition in base 10

carry	0	1	1	0
		3	8	9
+		1	5	7
		5	4	6

Analogy with addition in base 10

$$
\begin{array}{rcccc}
 & c_3 & c_2 & c_1 & c_0 \\
 & & x_2 & x_1 & x_0 \\
+ & & y_2 & y_1 & y_0 \\
\hline
 & & s_2 & s_1 & s_0
\end{array}
$$

Analogy with addition in base 10

given these
3 inputs

$$
\begin{array}{rcccc}
 & c_3 & c_2 & c_1 & c_0 \\
 & & x_2 & x_1 & x_0 \\
+ & & y_2 & y_1 & y_0 \\
\hline
 & & s_2 & s_1 & s_0
\end{array}
$$

Analogy with addition in base 10

$$
\begin{array}{cccccc}
 & c_3 & c_2 & c_1 & c_0 \\
 & & x_2 & x_1 & x_0 \\
+ & & y_2 & y_1 & y_0 \\
\hline
 & & s_2 & s_1 & s_0
\end{array}
$$

given these
3 inputs

compute these
2 outputs

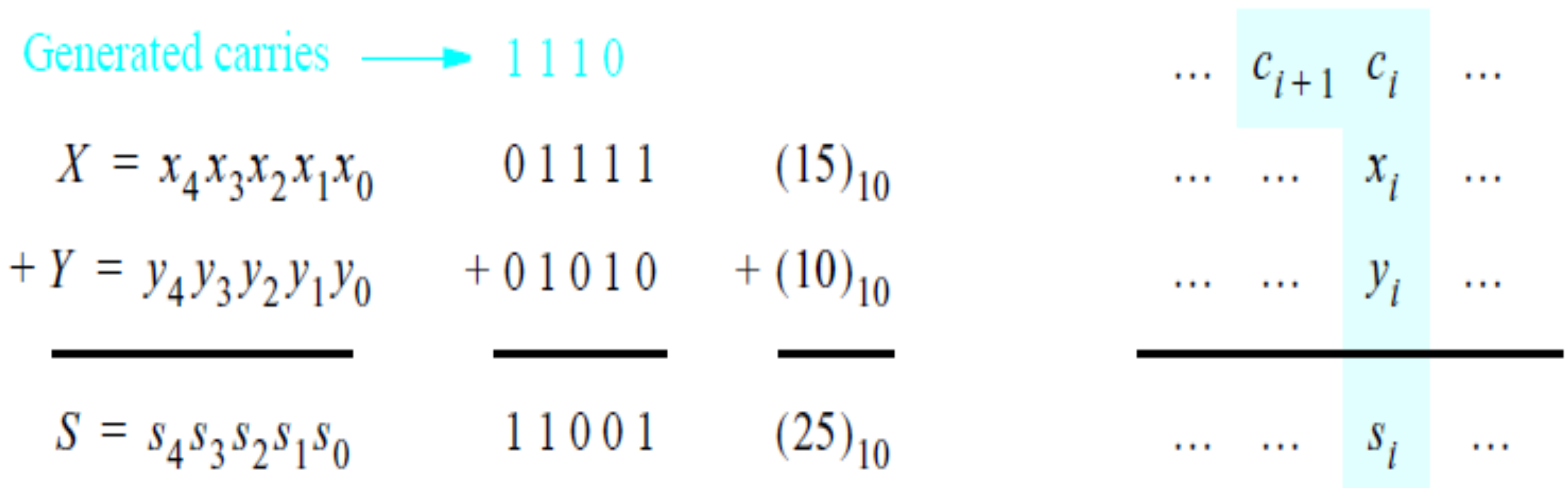
Analogy with addition in base 10

$$
\begin{array}{ccccc}
 & c_3 & c_2 & c_1 & c_0 \\
 & & x_2 & x_1 & x_0 \\
+ & & y_2 & y_1 & y_0 \\
\hline
 & & s_2 & s_1 & s_0
\end{array}
$$

Analogy with addition in base 10

$$
\begin{array}{rcccc}
 & c_3 & c_2 & c_1 & c_0 \\
 & & x_2 & x_1 & x_0 \\
+ & & y_2 & y_1 & y_0 \\
\hline
 & & s_2 & s_1 & s_0
\end{array}
$$

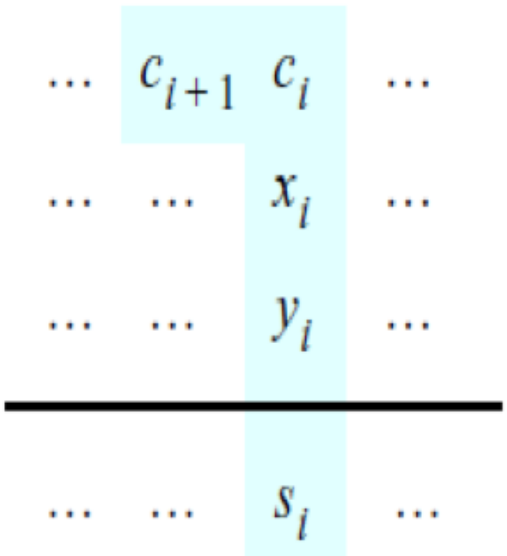
Addition of multibit numbers



Bit position i

[Figure 3.2 from the textbook]

Problem Statement and Truth Table



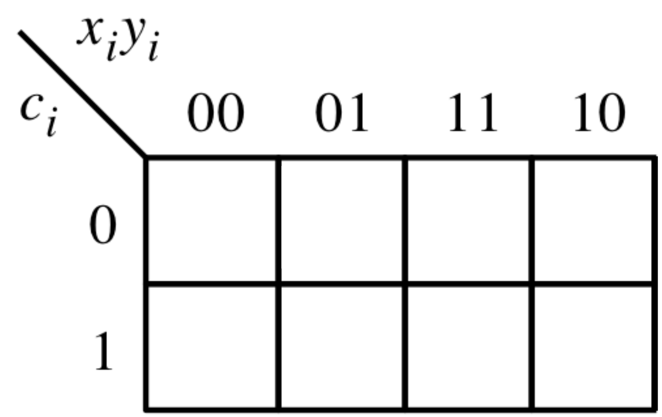
[Figure 3.2b from the textbook]

c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

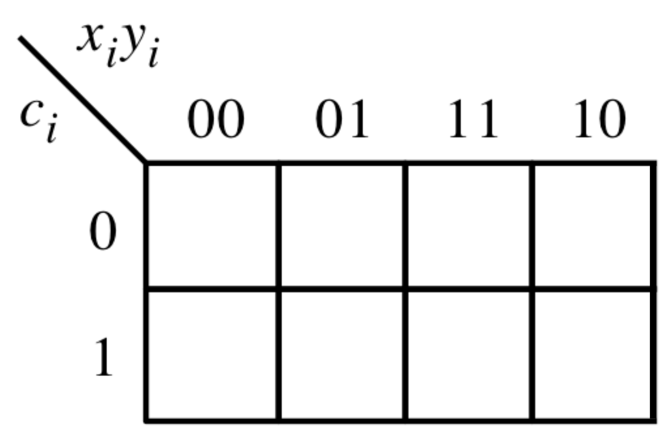
[Figure 3.3a from the textbook]

Let’s fill-in the two K-maps

c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



s_i =



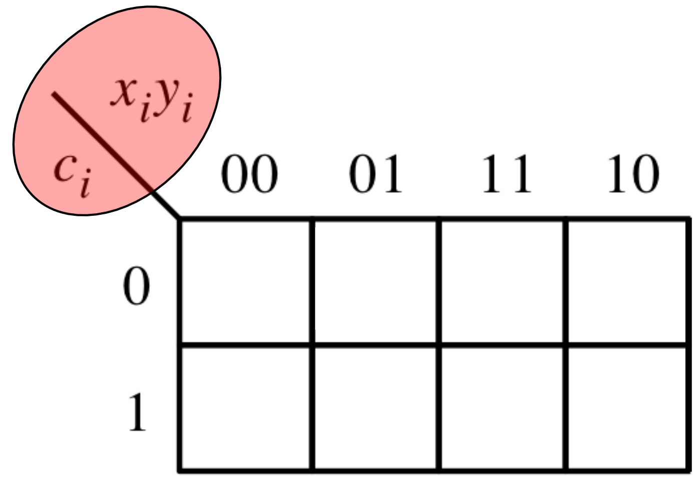
c_{i+1} =

[Figure 3.3a-b from the textbook]

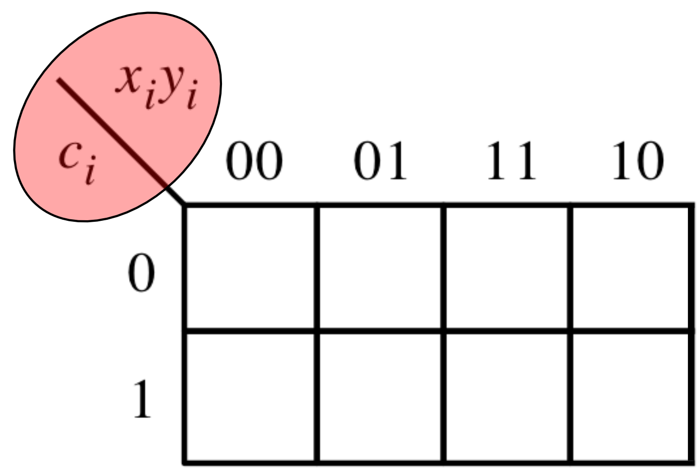
Let's fill-in the two K-maps

Note that the textbook switched to the other way to draw a K-Map

c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



$s_i =$



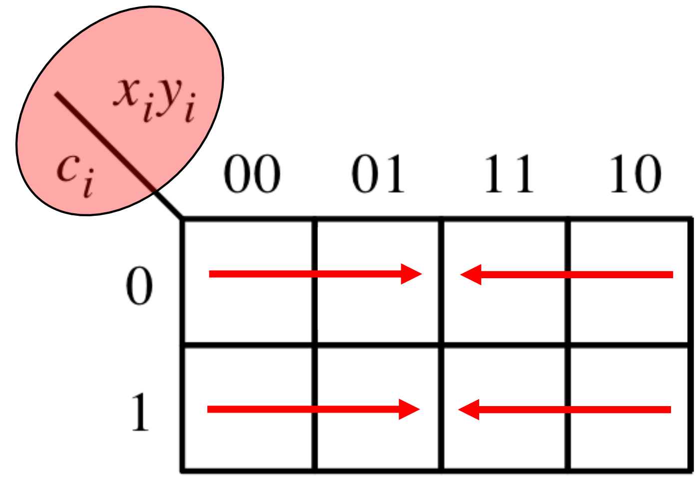
$c_{i+1} =$

[Figure 3.3a-b from the textbook]

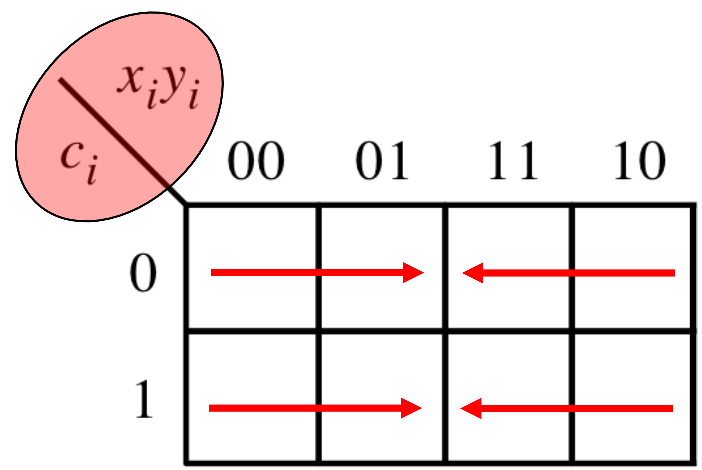
Let’s fill-in the two K-maps

Note that the textbook switched to the other way to draw a K-Map

c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



$s_i =$



$c_{i+1} =$

[Figure 3.3a-b from the textbook]

Let's fill-in the two K-maps

c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

c_i \ $x_i y_i$	00	01	11	10
0		1		1
1	1		1	

$$s_i = x_i \oplus y_i \oplus c_i$$

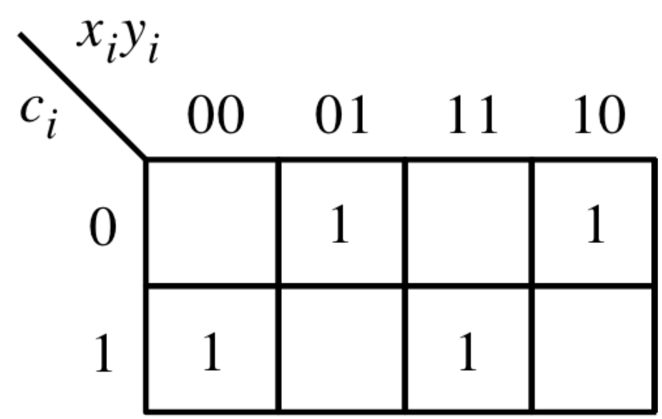
c_i \ $x_i y_i$	00	01	11	10
0			1	
1		1	1	1

$$c_{i+1} = x_i y_i + x_i c_i + y_i c_i$$

[Figure 3.3a-b from the textbook]

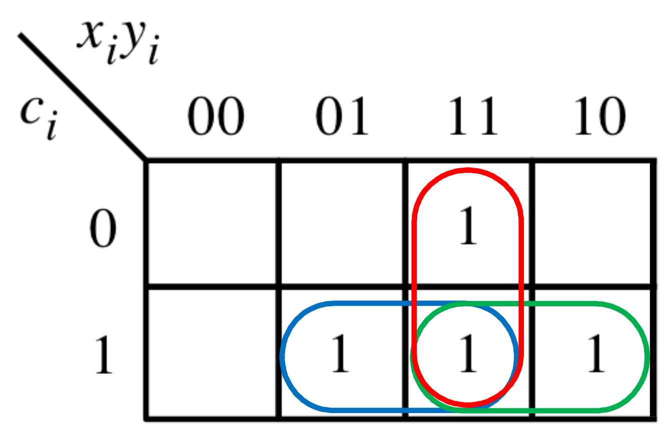
Let’s fill-in the two K-maps

c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



3-input XOR

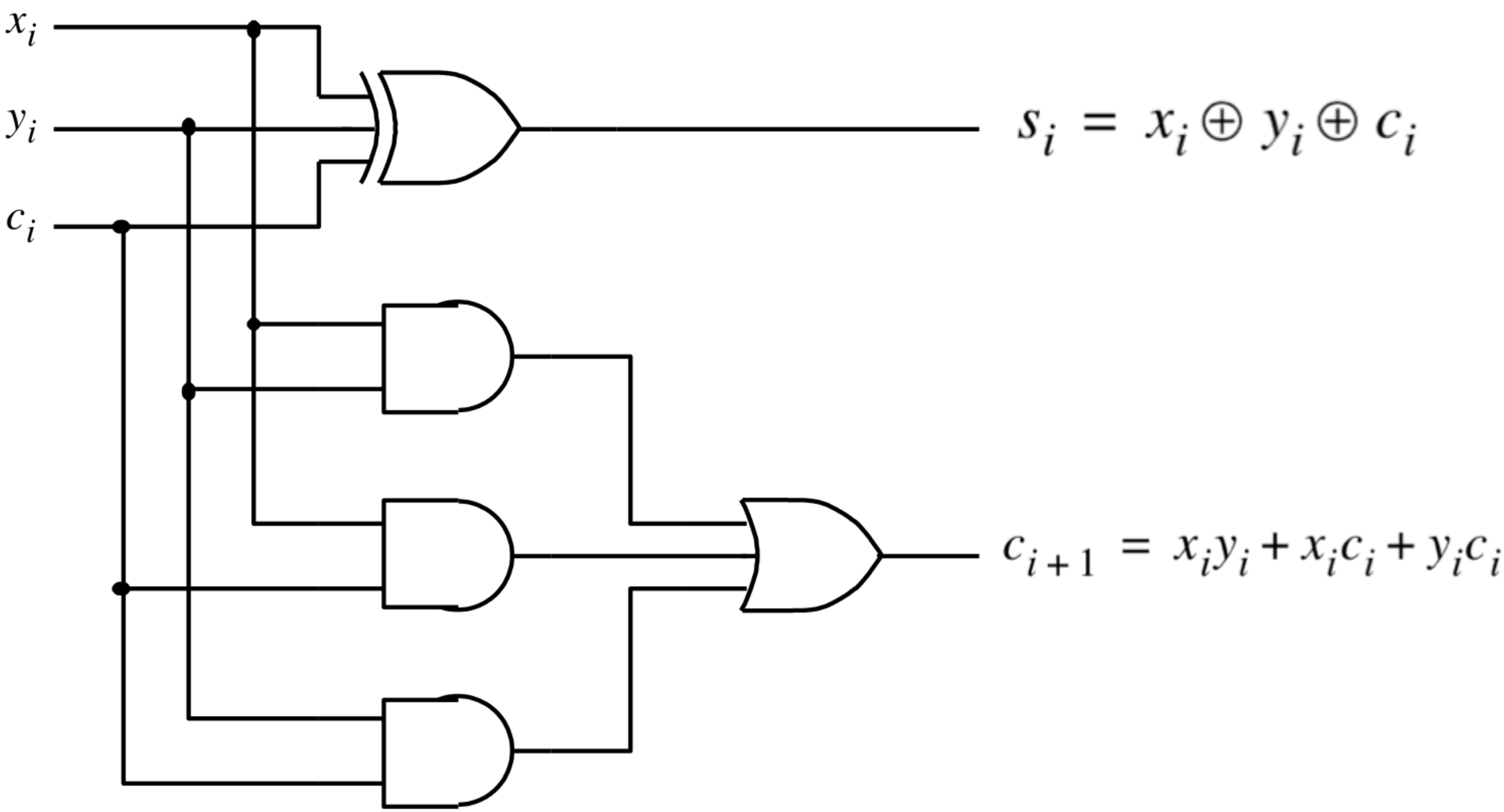
$$s_i = x_i \oplus y_i \oplus c_i$$



$$c_{i+1} = x_iy_i + x_ic_i + y_ic_i$$

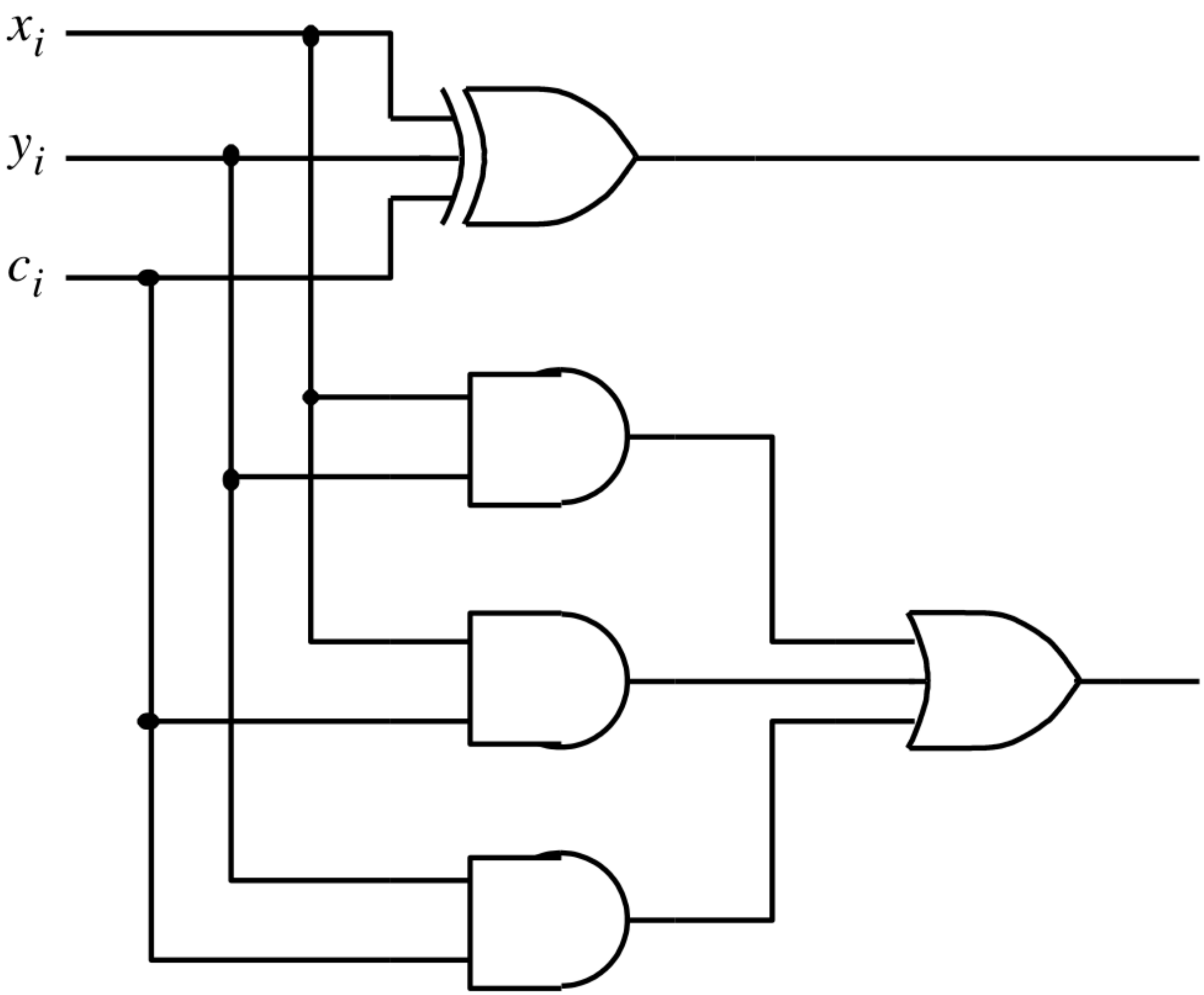
[Figure 3.3a-b from the textbook]

The circuit for the two expressions



[Figure 3.3c from the textbook]

This is called the Full-Adder

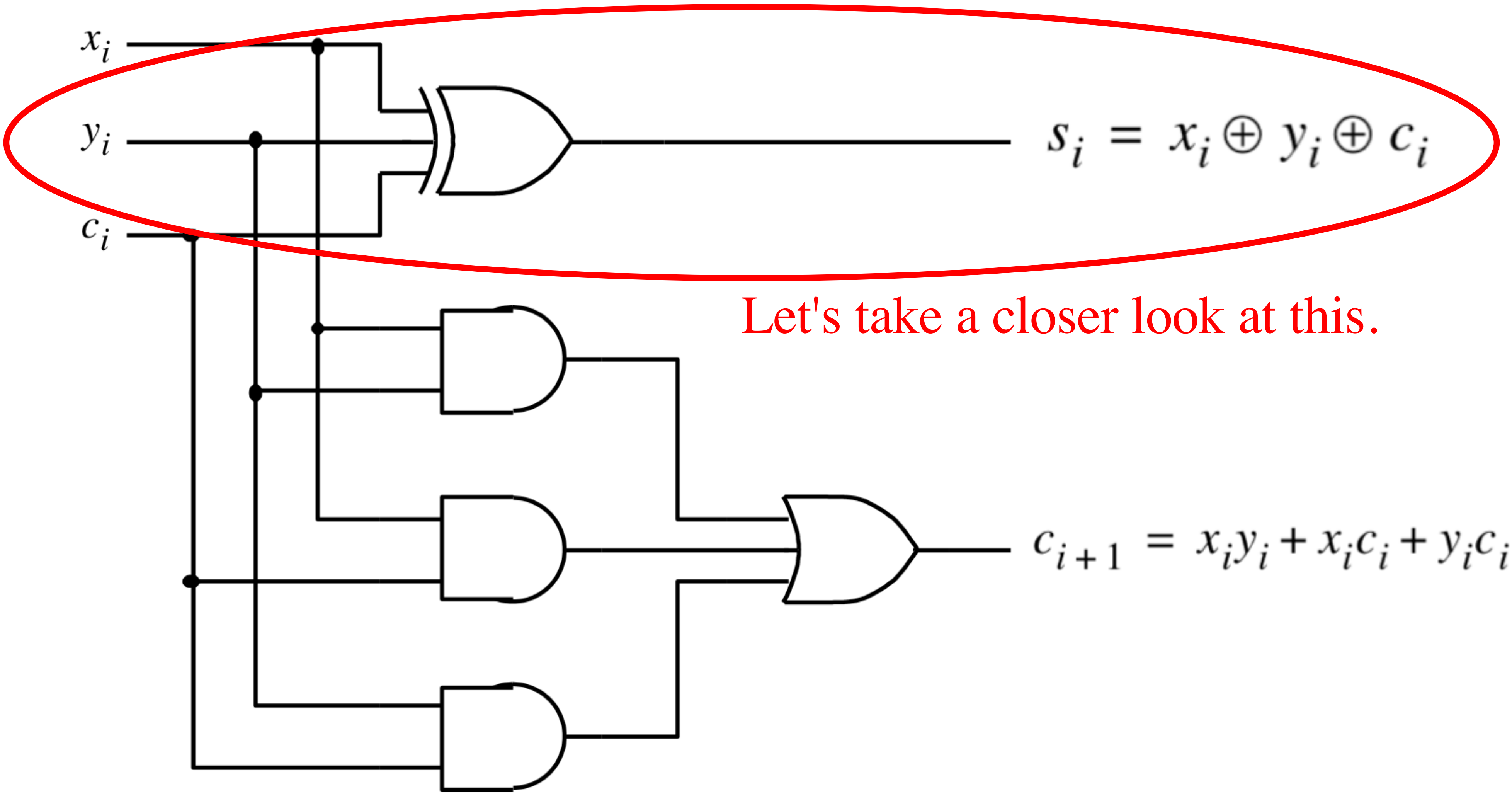


$$s_i = x_i \oplus y_i \oplus c_i$$

$$c_{i+1} = x_i y_i + x_i c_i + y_i c_i$$

[Figure 3.3c from the textbook]

This is called the Full-Adder



[Figure 3.3c from the textbook]

XOR Magic

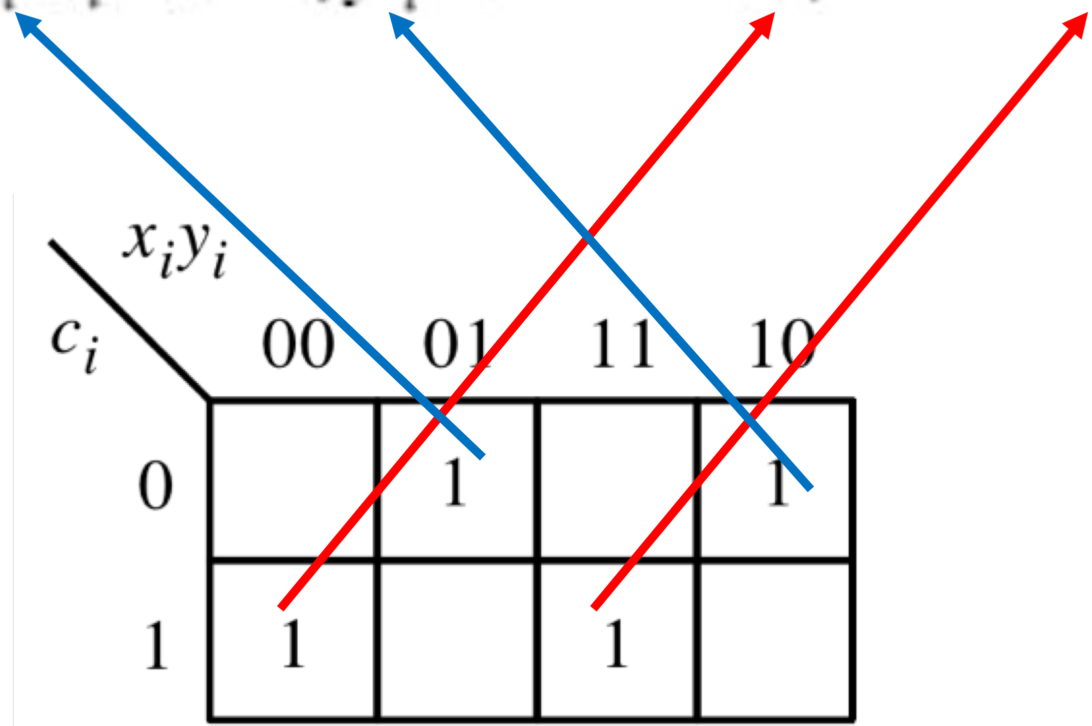
$$s_i = \bar{x}_i y_i \bar{c}_i + x_i \bar{y}_i \bar{c}_i + \bar{x}_i \bar{y}_i c_i + x_i y_i c_i$$

c_i \ $x_i y_i$	00	01	11	10
0		1		1
1	1		1	

$$s_i = x_i \oplus y_i \oplus c_i$$

XOR Magic

$$s_i = \bar{x}_i y_i \bar{c}_i + x_i \bar{y}_i \bar{c}_i + \bar{x}_i \bar{y}_i c_i + x_i y_i c_i$$



$$s_i = x_i \oplus y_i \oplus c_i$$

XOR Magic

$$s_i = \bar{x}_i y_i \bar{c}_i + x_i \bar{y}_i \bar{c}_i + \bar{x}_i \bar{y}_i c_i + x_i y_i c_i$$

$$\begin{aligned} s_i &= (\bar{x}_i y_i + x_i \bar{y}_i)\bar{c}_i + (\bar{x}_i \bar{y}_i + x_i y_i) c_i \\ &= (x_i \oplus y_i)\bar{c}_i + \overline{(x_i \oplus y_i)} c_i \\ &= (x_i \oplus y_i) \oplus c_i \end{aligned}$$

XOR Magic

$$s_i = \bar{x}_i y_i \bar{c}_i + x_i \bar{y}_i \bar{c}_i + \bar{x}_i \bar{y}_i c_i + x_i y_i c_i$$

Can you prove this?

$$s_i = (\bar{x}_i y_i + x_i \bar{y}_i)\bar{c}_i + (\bar{x}_i \bar{y}_i + x_i y_i) c_i$$

$$= (x_i \oplus y_i)\bar{c}_i + \overline{(x_i \oplus y_i)} c_i$$

$$= (x_i \oplus y_i) \oplus c_i$$

XOR Magic

$$\overline{x_i}\,\overline{y_i} + x_i\, y_i = \overline{x_i \oplus y_i}$$

XOR Magic

$$\overline{x_i}\,\overline{y_i} + x_i y_i = \overline{x_i \oplus y_i}$$

XNOR

XOR Magic

$$\overline{x_i}\,\overline{y_i} + x_i\, y_i = \overline{x_i \oplus y_i}$$

XOR

XOR Magic

$$\overline{x_i}\,\overline{y_i} + x_i y_i = \overline{x_i \oplus y_i}$$

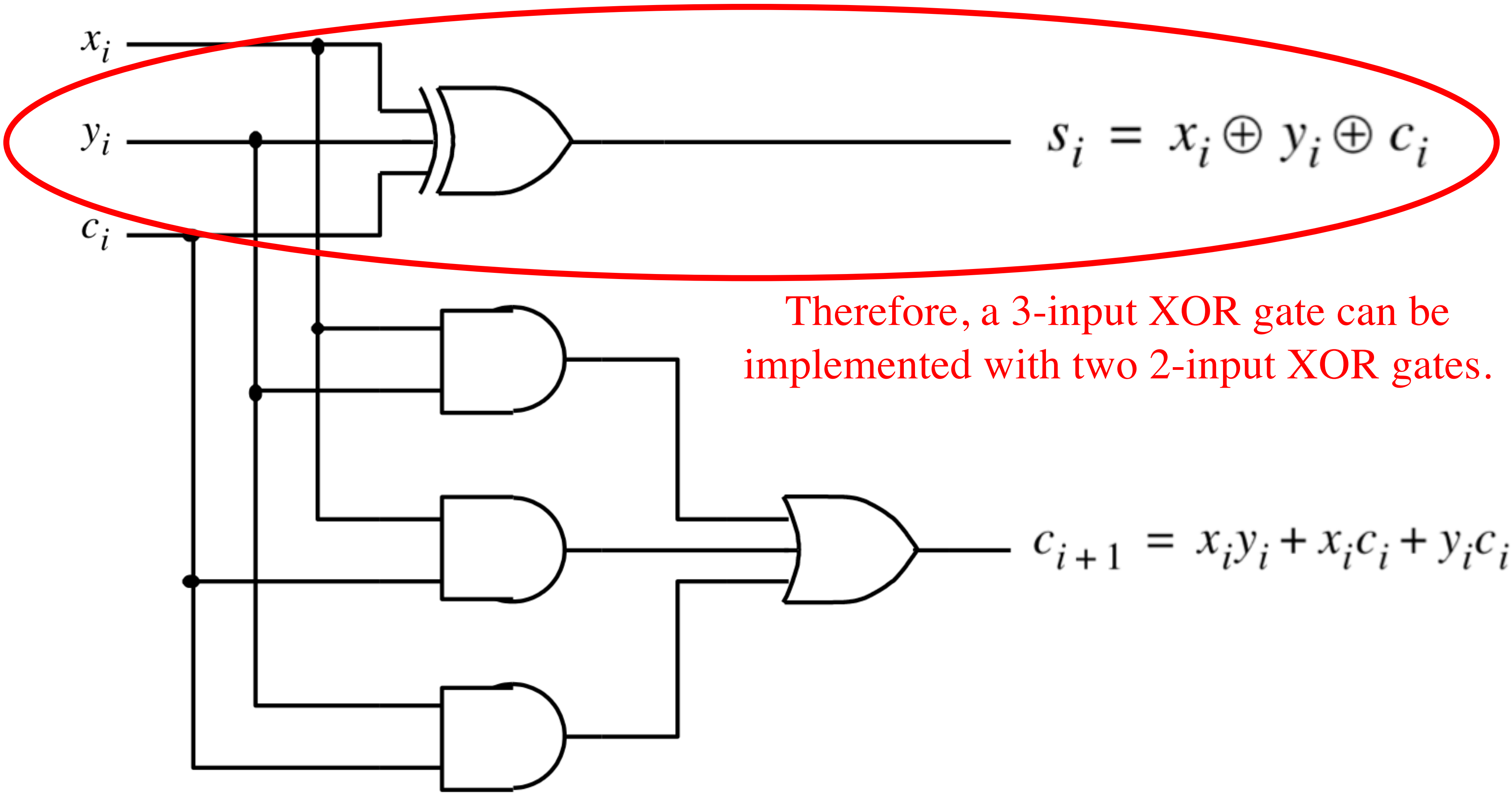
XNOR

XOR Magic

$$\overline{x_i}\,\overline{y_i} + x_i\, y_i = \overline{x_i \oplus y_i}$$

You can also prove this using the theorems of Boolean algebra.
Try that at home.

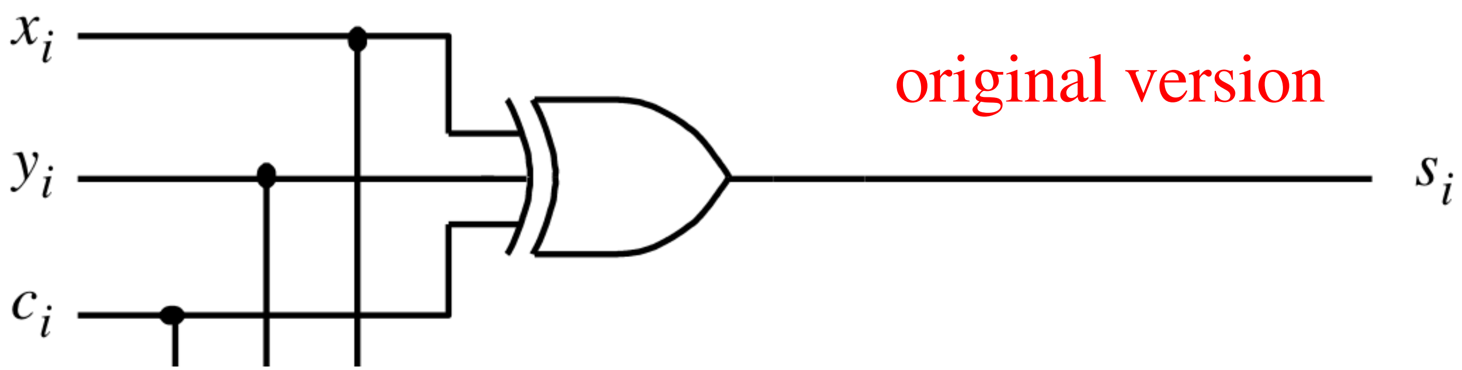
The Full-Adder Circuit



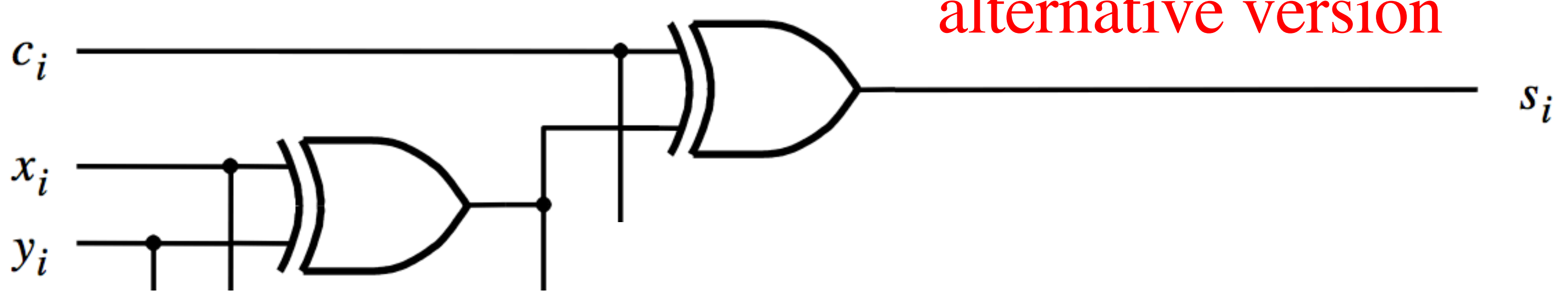
[Figure 3.3c from the textbook]

s_i can be implemented in two different ways

$$s_i = x_i \oplus y_i \oplus c_i$$

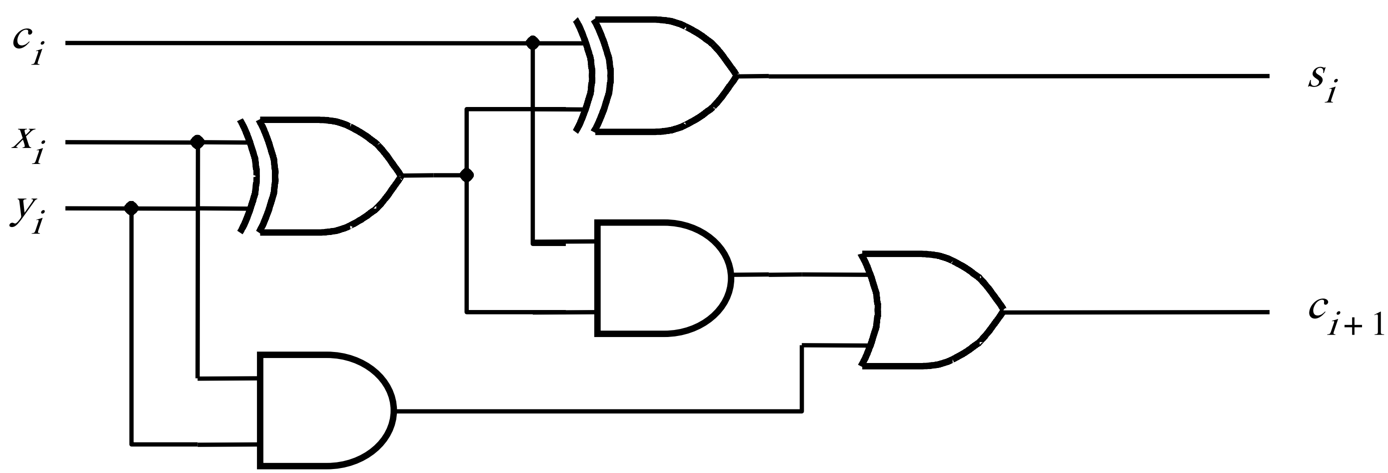


original version



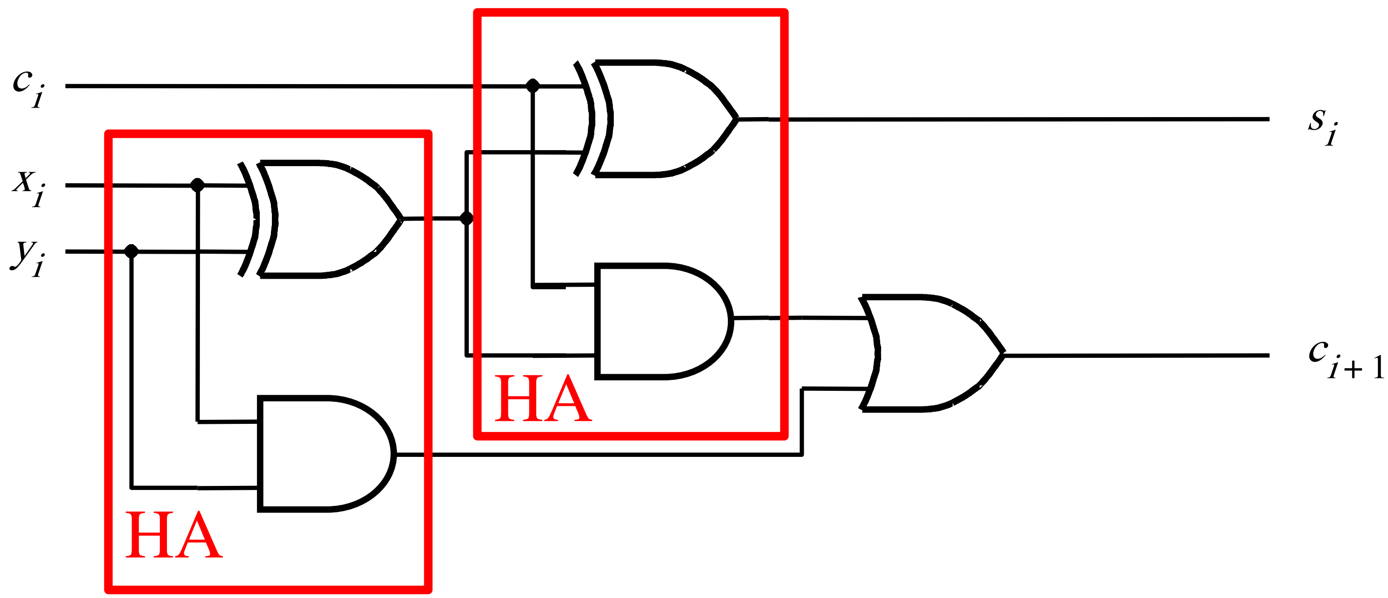
alternative version

The Full-Adder Circuit (alternative drawing)



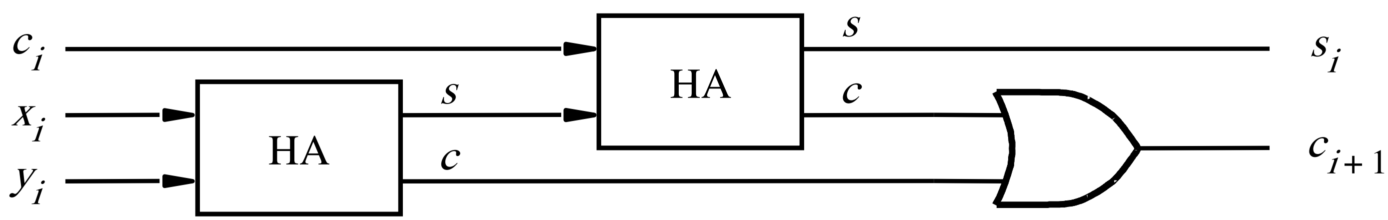
[Figure 3.4b from the textbook]

The Full-Adder Circuit (alternative drawing)

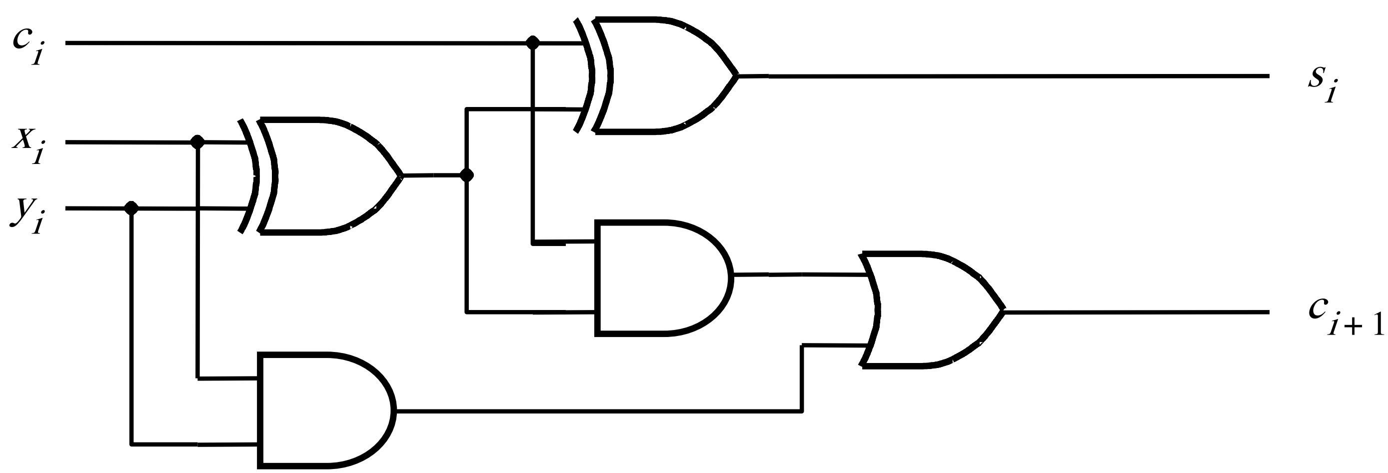


[Figure 3.4b from the textbook]

The Full-Adder Circuit (alternative drawing)



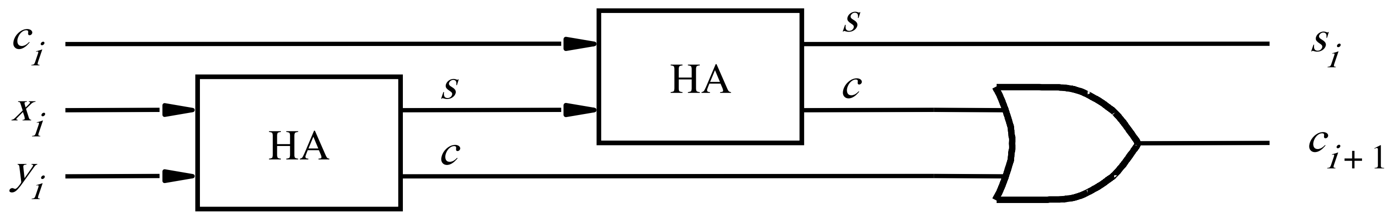
(a) Block diagram



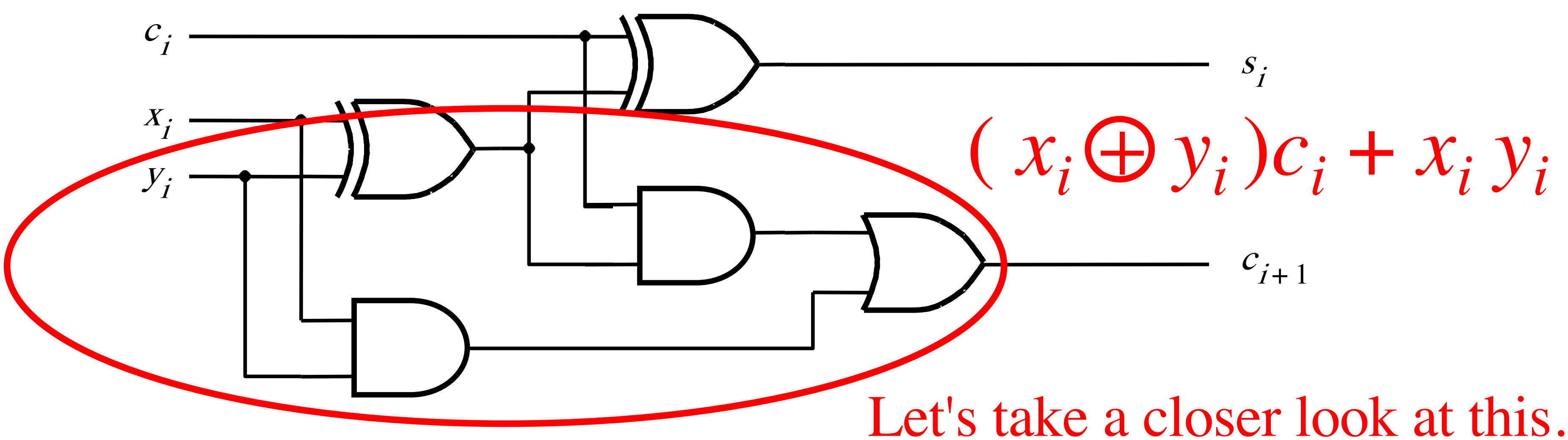
(b) Detailed diagram

[Figure 3.4 from the textbook]

The Full-Adder Circuit (alternative drawing)



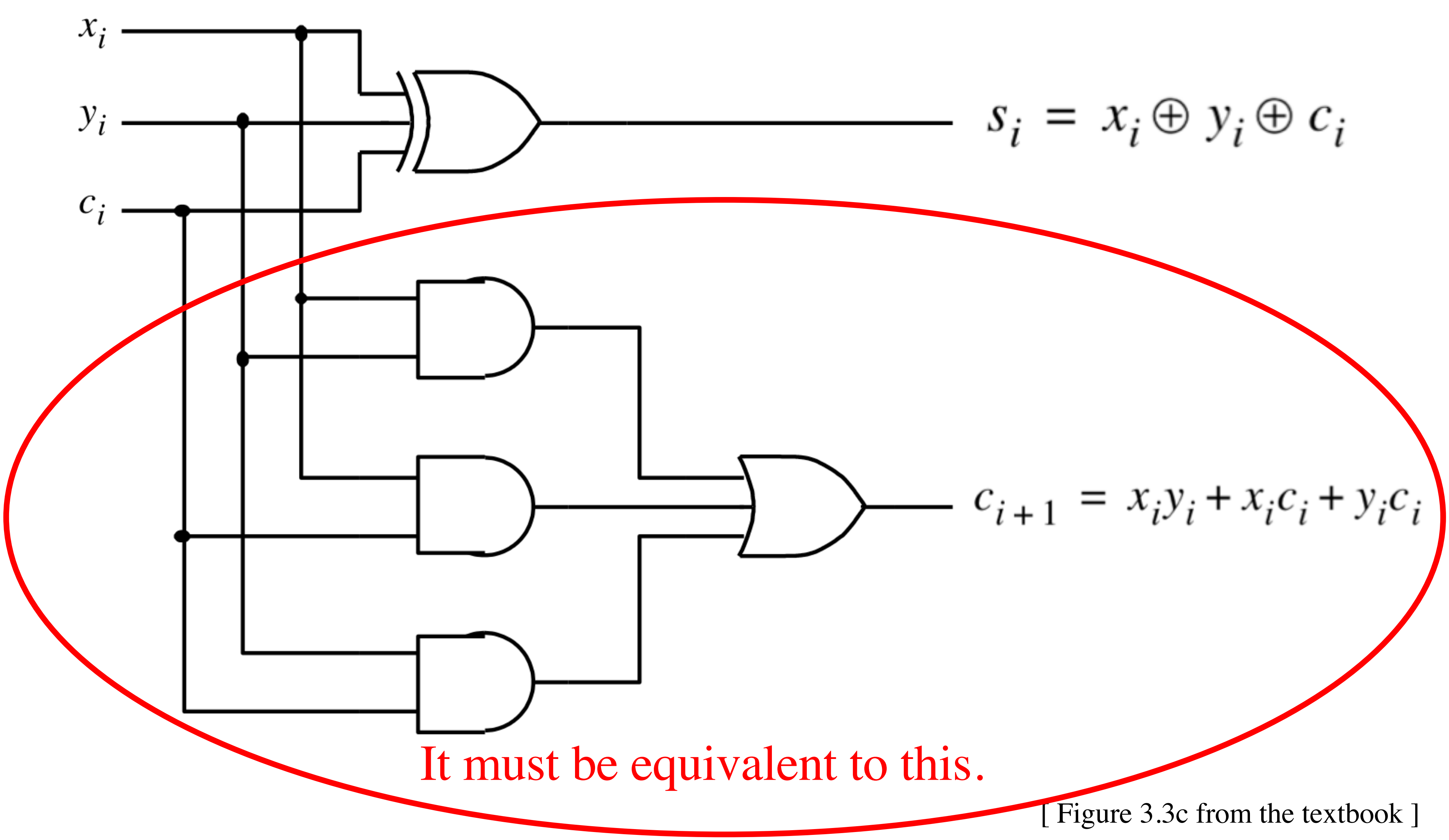
(a) Block diagram



(b) Detailed diagram

[Figure 3.4 from the textbook]

The Full-Adder Circuit



[Figure 3.3c from the textbook]

Let's Prove This

$$(x_i \oplus y_i)c_i + x_i y_i \stackrel{?}{=} x_i y_i + x_i c_i + c_i y_i$$

Let's Prove This

$$(x_i \oplus y_i)c_i + x_i y_i =$$

Let's Prove This

$$(x_i \oplus y_i)c_i + x_i y_i = (\overline{x_i} y_i + x_i \overline{y_i})c_i + x_i y_i$$

Let's Prove This

$$(x_i \oplus y_i)c_i + x_i y_i = (\overline{x_i} y_i + x_i \overline{y_i})c_i + x_i y_i$$

$$= \overline{x_i} y_i c_i + x_i \overline{y_i} c_i + x_i y_i + x_i y_i$$

double this term

Let's Prove This

$$(x_i \oplus y_i)c_i + x_i y_i = (\overline{x_i} y_i + x_i \overline{y_i})c_i + x_i y_i$$

$$= \overline{x_i} y_i c_i + x_i \overline{y_i} c_i + x_i y_i + x_i y_i$$

Let's Prove This

$$(x_i \oplus y_i)c_i + x_i y_i = (\overline{x_i} y_i + x_i \overline{y_i})c_i + x_i y_i$$

$$= \overline{x_i} y_i c_i + x_i \overline{y_i} c_i + x_i y_i + x_i y_i$$

$$= (\overline{x_i} c_i + x_i) y_i + x_i (\overline{y_i} c_i + y_i)$$

Let's Prove This

$$(x_i \oplus y_i)c_i + x_i y_i = (\overline{x_i} y_i + x_i \overline{y_i})c_i + x_i y_i$$

$$= \overline{x_i} y_i c_i + x_i \overline{y_i} c_i + x_i y_i + x_i y_i$$

$$= (\overline{x_i} c_i + x_i) y_i + x_i (\overline{y_i} c_i + y_i)$$

use Theorem 16a twice

$$= (c_i + x_i) y_i + x_i (c_i + y_i)$$

Let's Prove This

$$(x_i \oplus y_i)c_i + x_i y_i = (\overline{x_i} y_i + x_i \overline{y_i})c_i + x_i y_i$$

$$= \overline{x_i} y_i c_i + x_i \overline{y_i} c_i + x_i y_i + x_i y_i$$

$$= (\overline{x_i} c_i + x_i) y_i + x_i (\overline{y_i} c_i + y_i)$$

$$= (c_i + x_i) y_i + x_i (c_i + y_i)$$

$$= c_i y_i + x_i y_i + x_i c_i + x_i y_i$$

Let's Prove This

$$(x_i \oplus y_i)c_i + x_i y_i = (\overline{x_i} y_i + x_i \overline{y_i})c_i + x_i y_i$$

$$= \overline{x_i} y_i c_i + x_i \overline{y_i} c_i + x_i y_i + x_i y_i$$

$$= (\overline{x_i} c_i + x_i) y_i + x_i (\overline{y_i} c_i + y_i)$$

$$= (c_i + x_i) y_i + x_i (c_i + y_i)$$

$$= c_i y_i + x_i y_i + x_i c_i + x_i y_i$$

remove one copy of this doubled term

Let's Prove This

$$(x_i \oplus y_i)c_i + x_i y_i = (\overline{x_i} y_i + x_i \overline{y_i})c_i + x_i y_i$$

$$= \overline{x_i} y_i c_i + x_i \overline{y_i} c_i + x_i y_i + x_i y_i$$

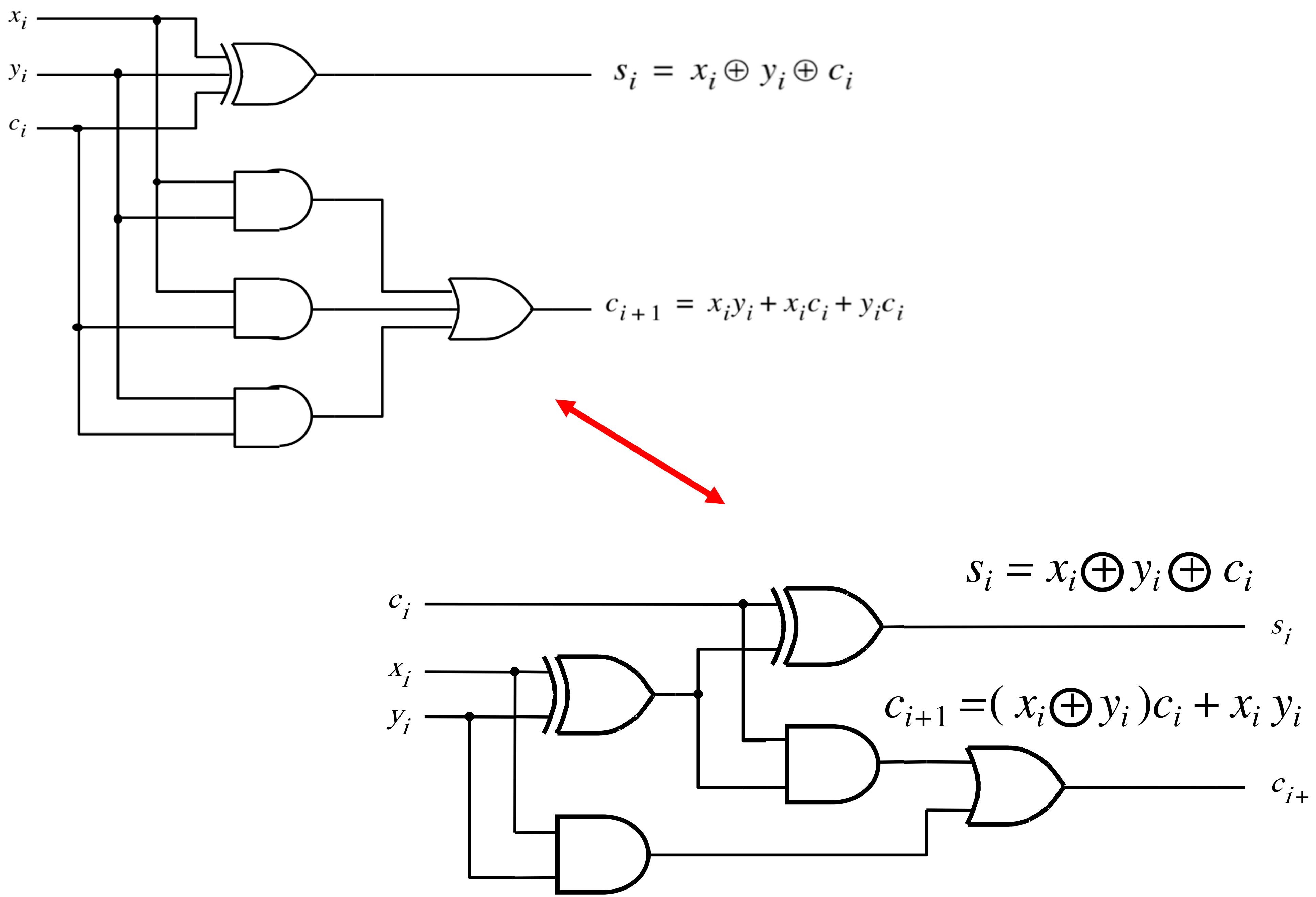
$$= (\overline{x_i} c_i + x_i) y_i + x_i (\overline{y_i} c_i + y_i)$$

$$= (c_i + x_i) y_i + x_i (c_i + y_i)$$

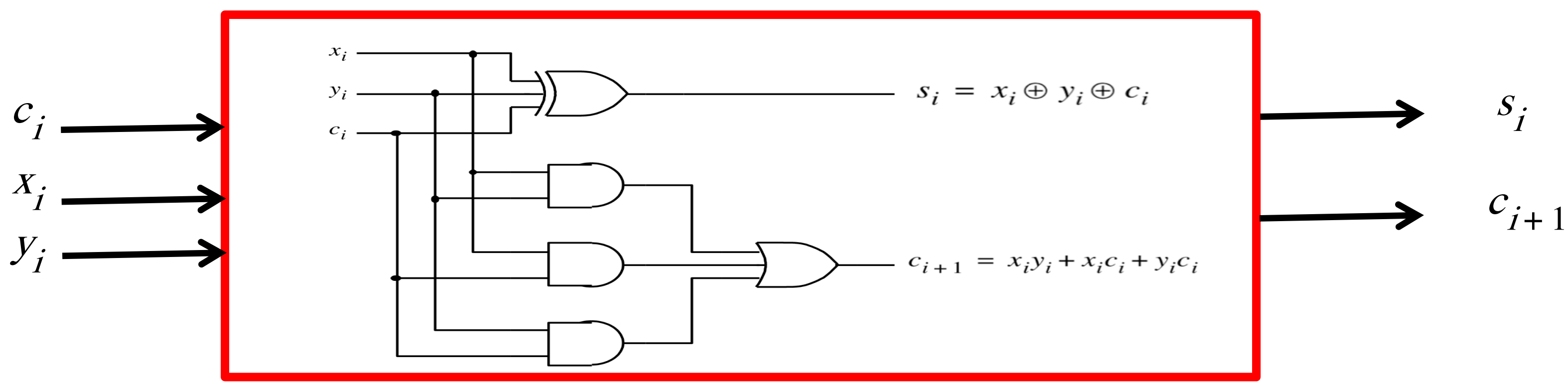
$$= c_i y_i + x_i y_i + x_i c_i + x_i y_i$$

$$= c_i y_i + x_i y_i + x_i c_i$$

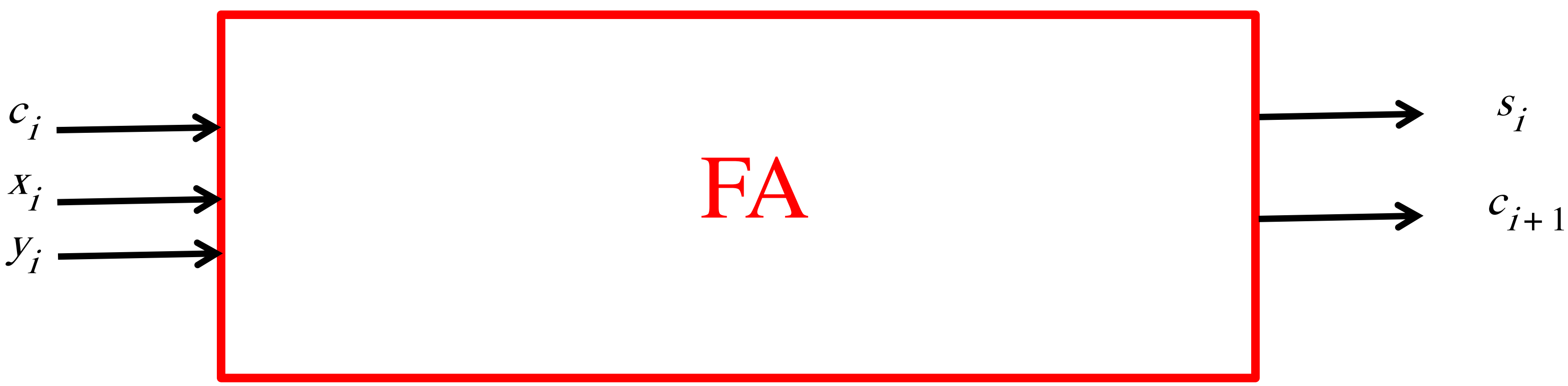
Therefore, these circuits are equivalent



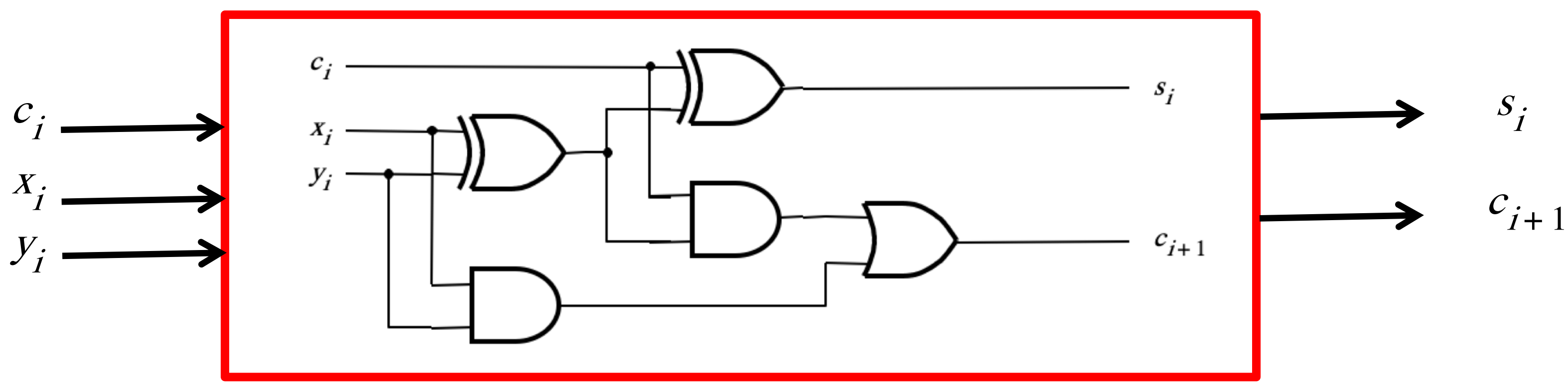
The Full-Adder Abstraction



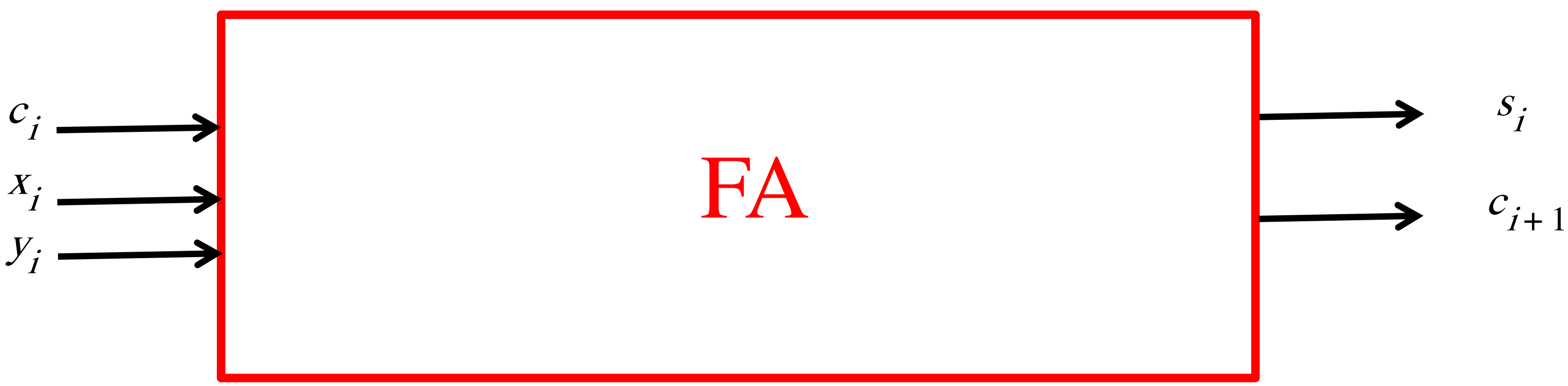
The Full-Adder Abstraction



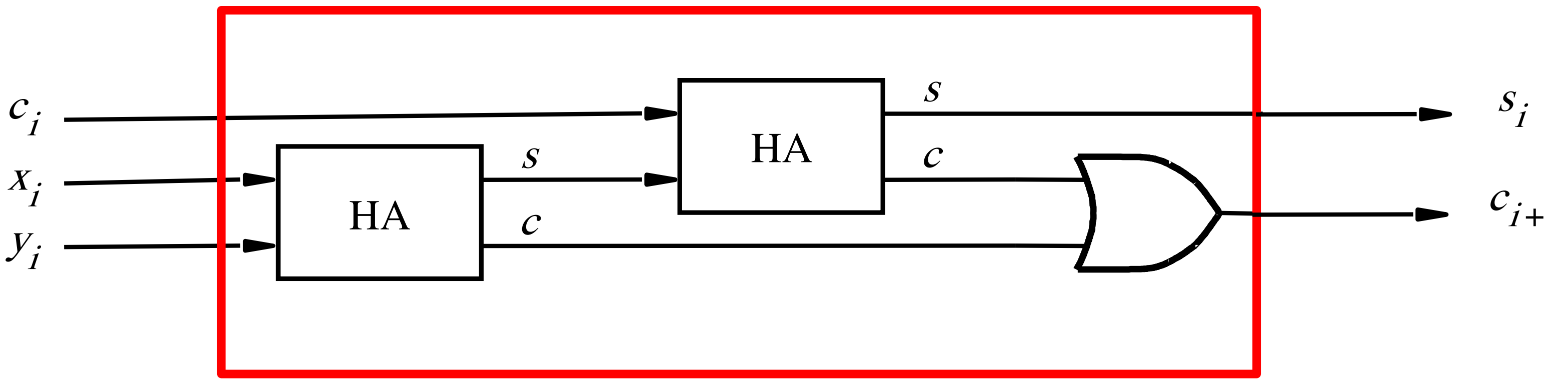
The Full-Adder Abstraction



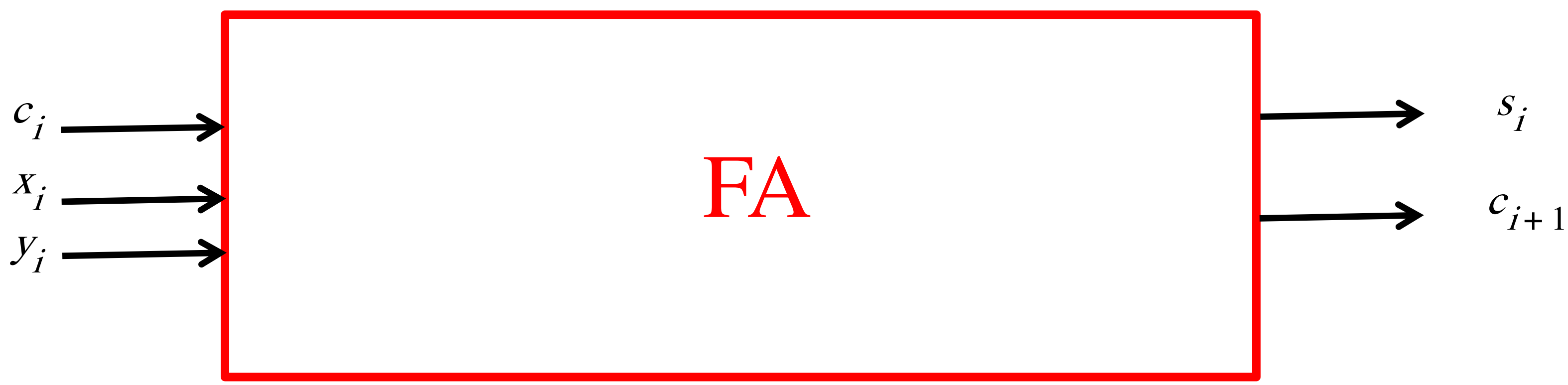
The Full-Adder Abstraction



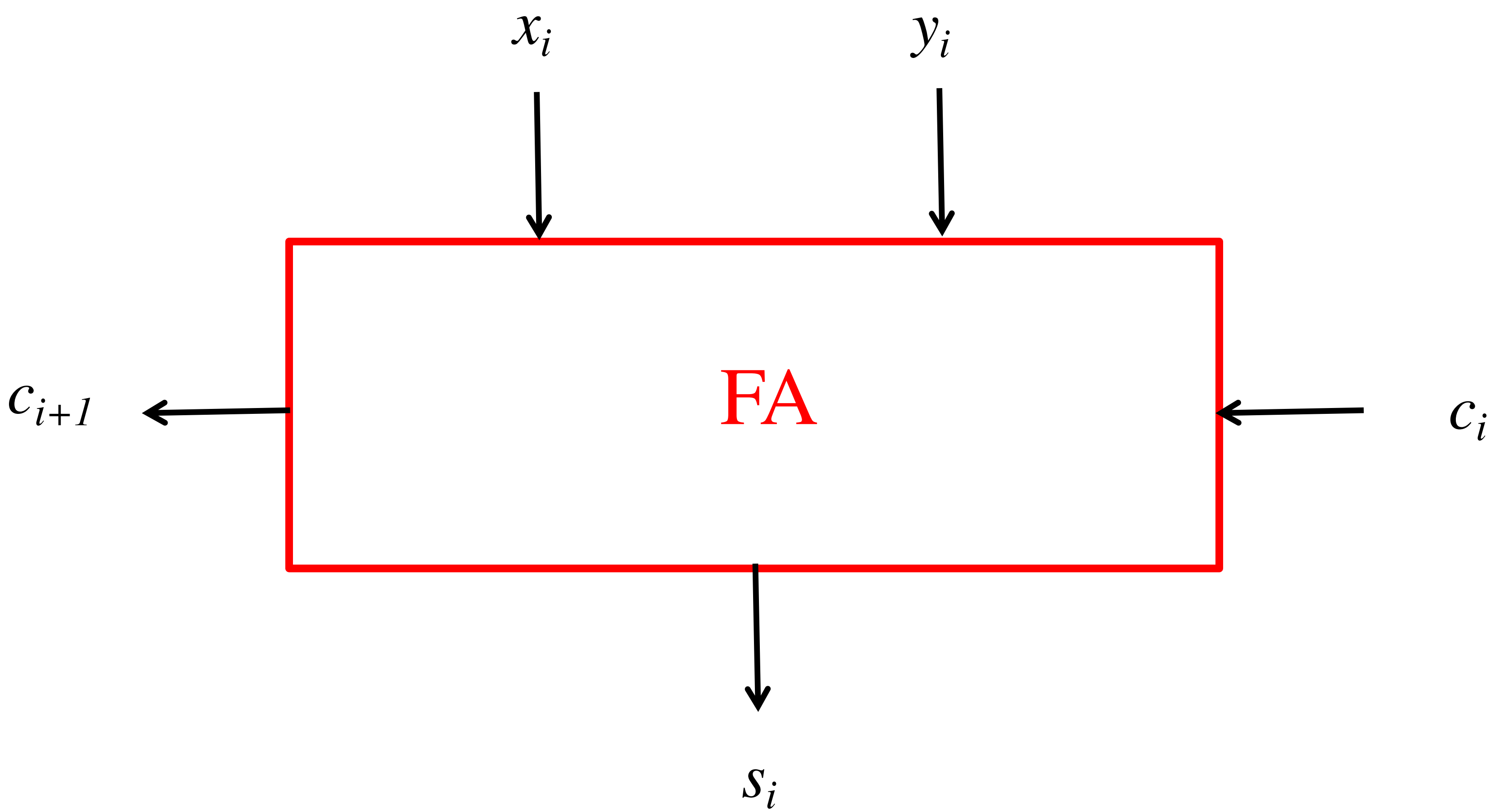
The Full-Adder Abstraction



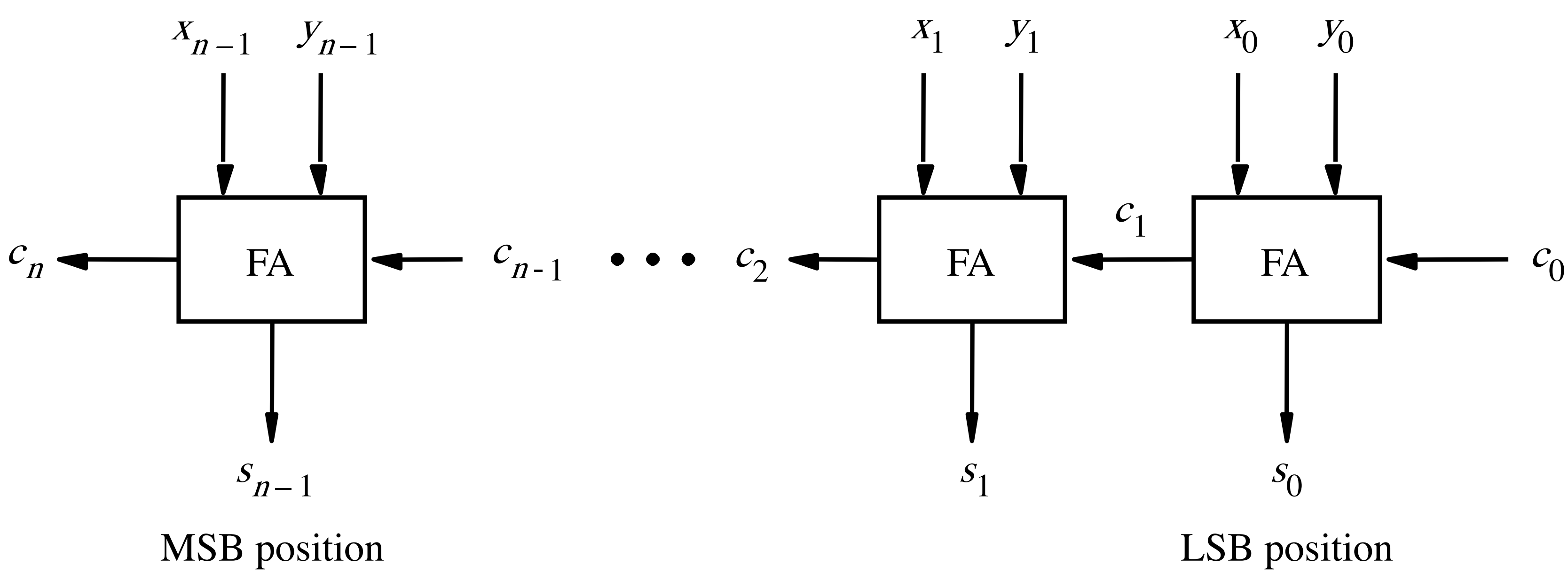
The Full-Adder Abstraction



We can place the arrows anywhere

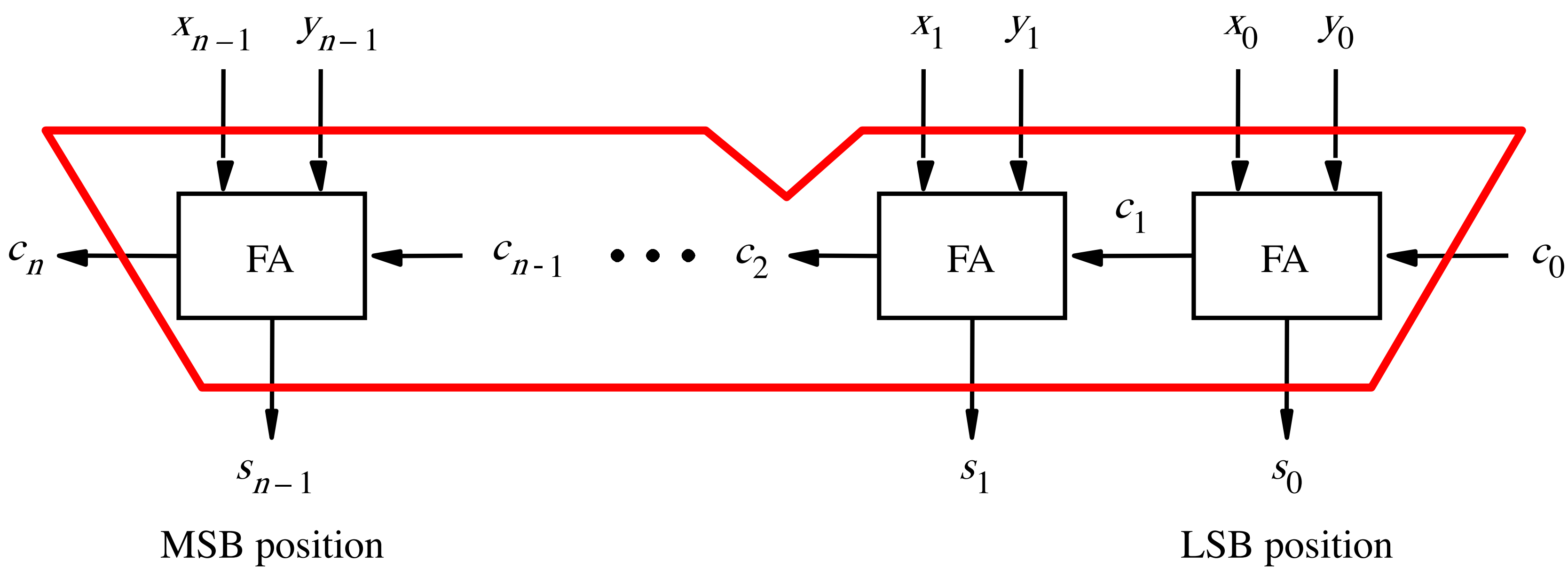


n-bit ripple-carry adder

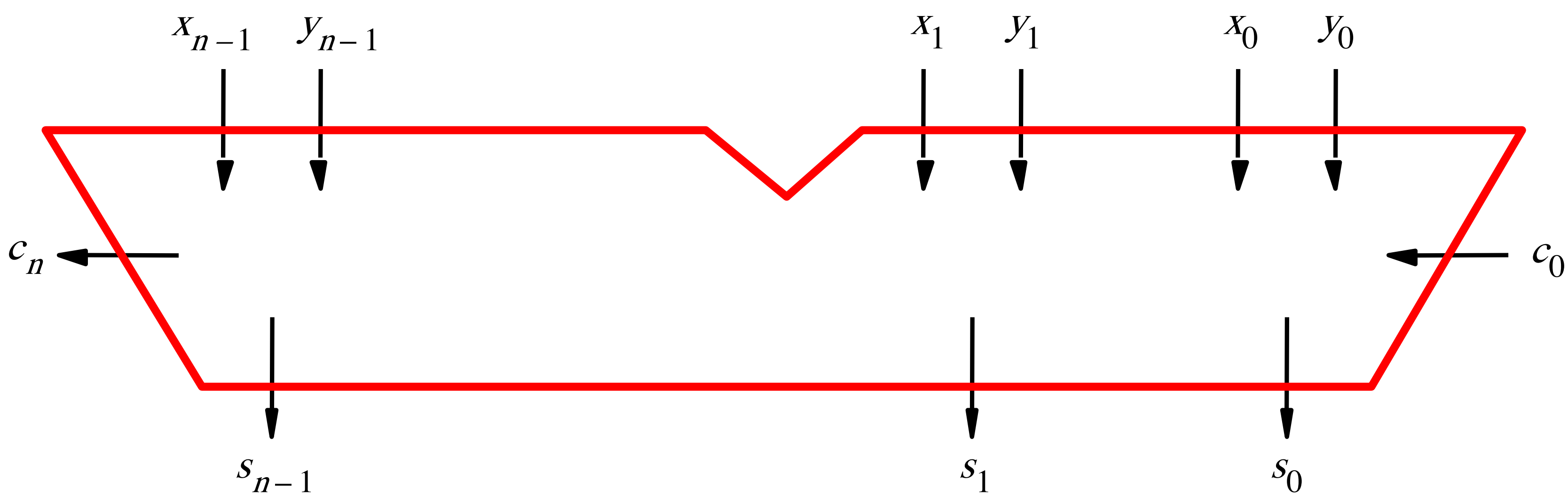


[Figure 3.5 from the textbook]

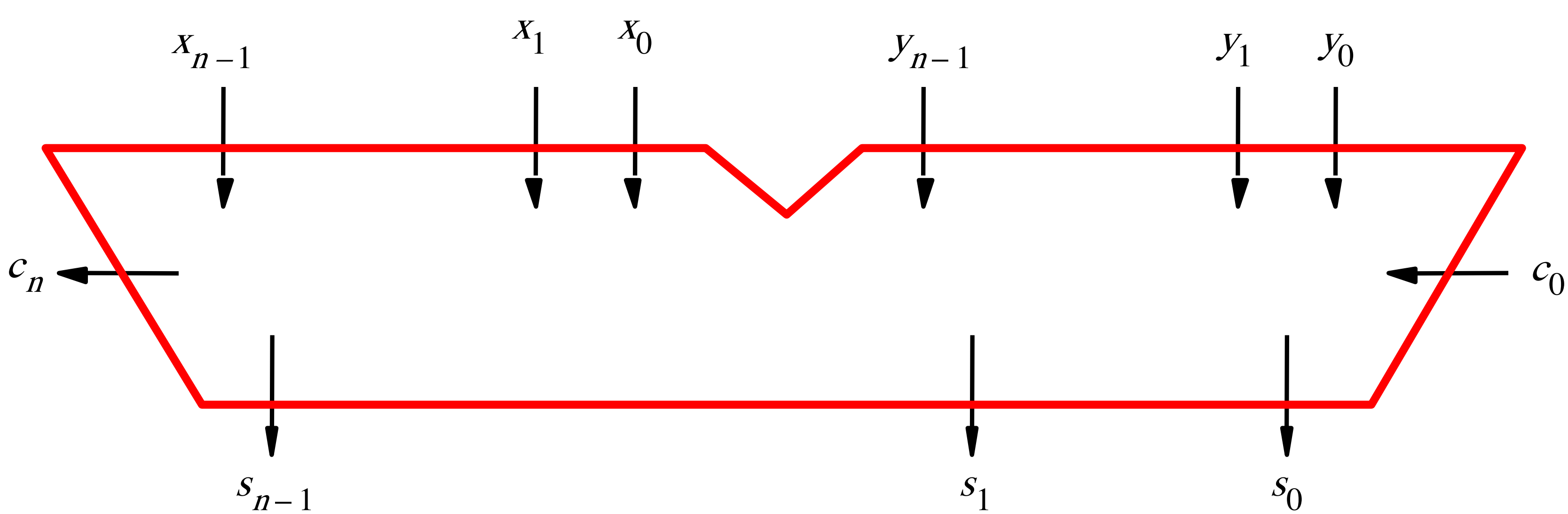
n-bit ripple-carry adder abstraction



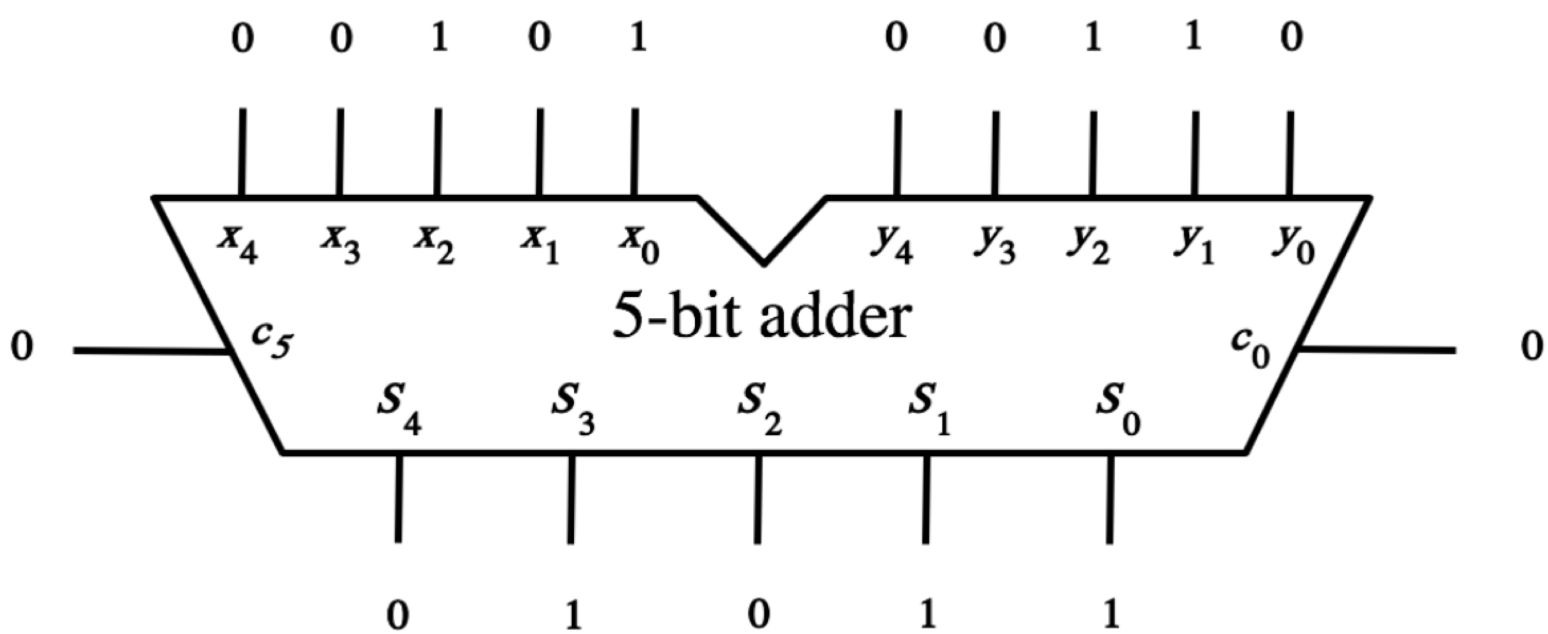
n-bit ripple-carry adder abstraction



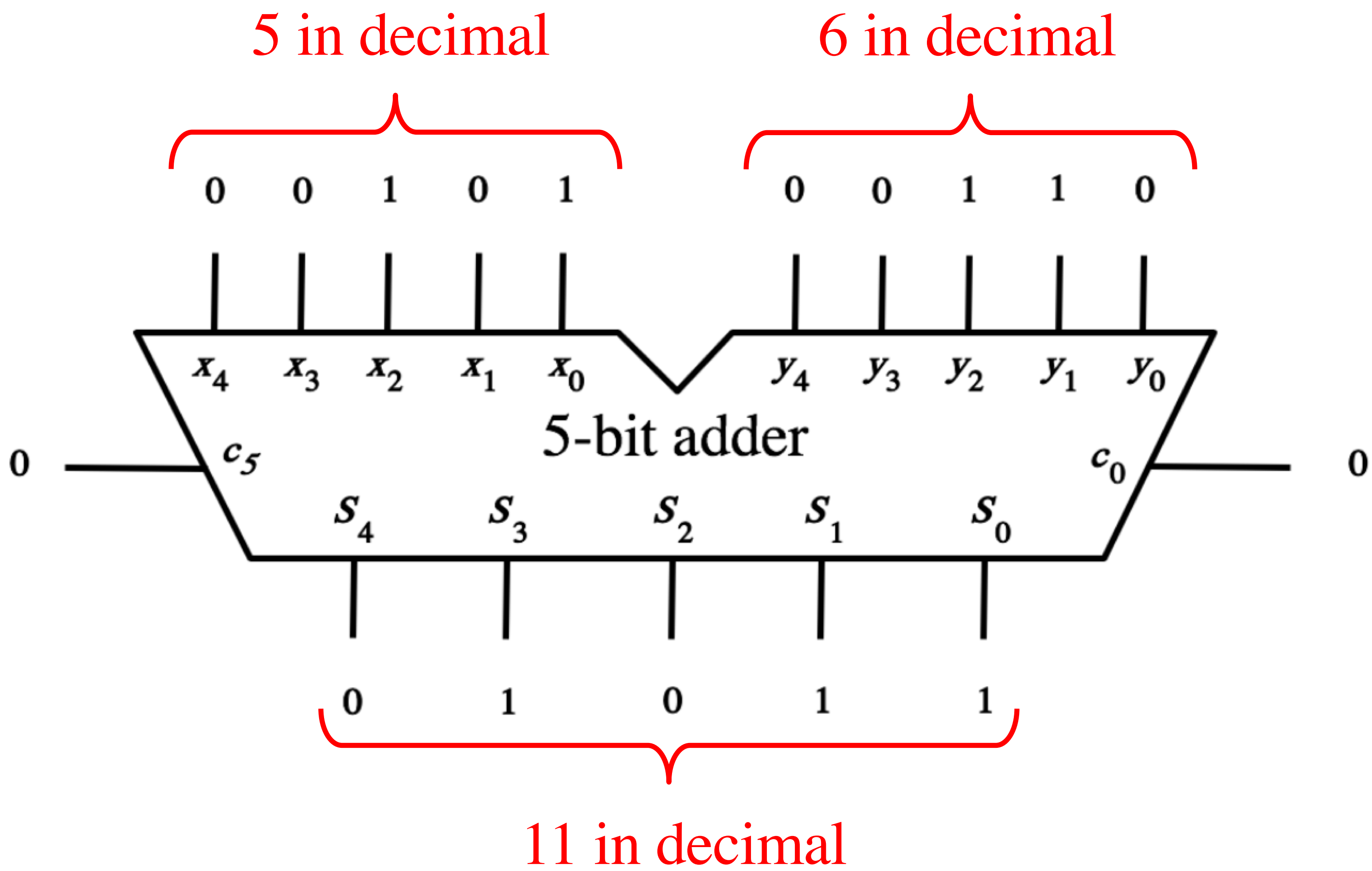
The x and y lines are typically grouped together for better visualization, but the underlying logic remains the same



Example:
Computing 5+6 using a 5-bit adder



Example:
Computing 5+6 using a 5-bit adder

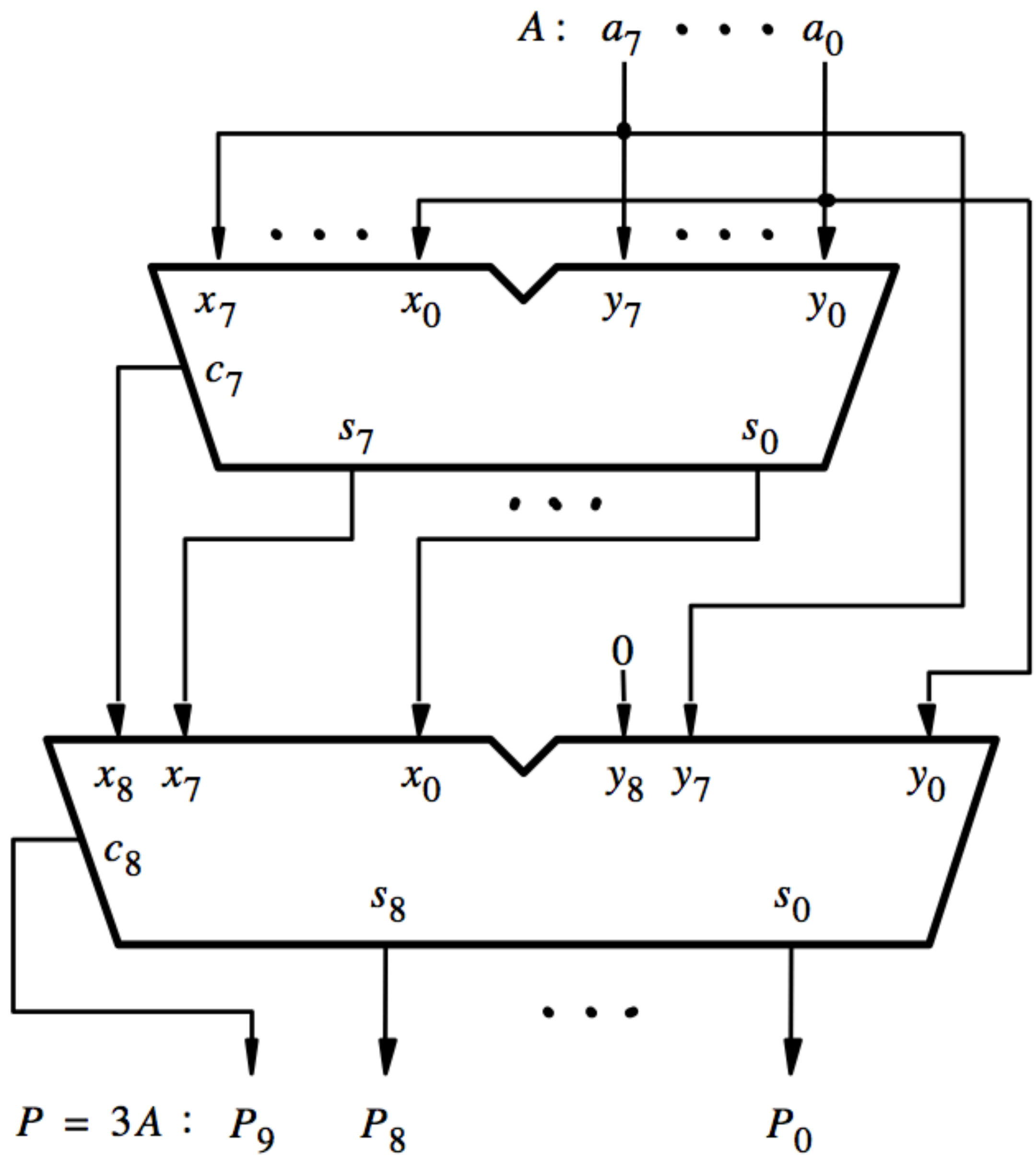


Design Example:

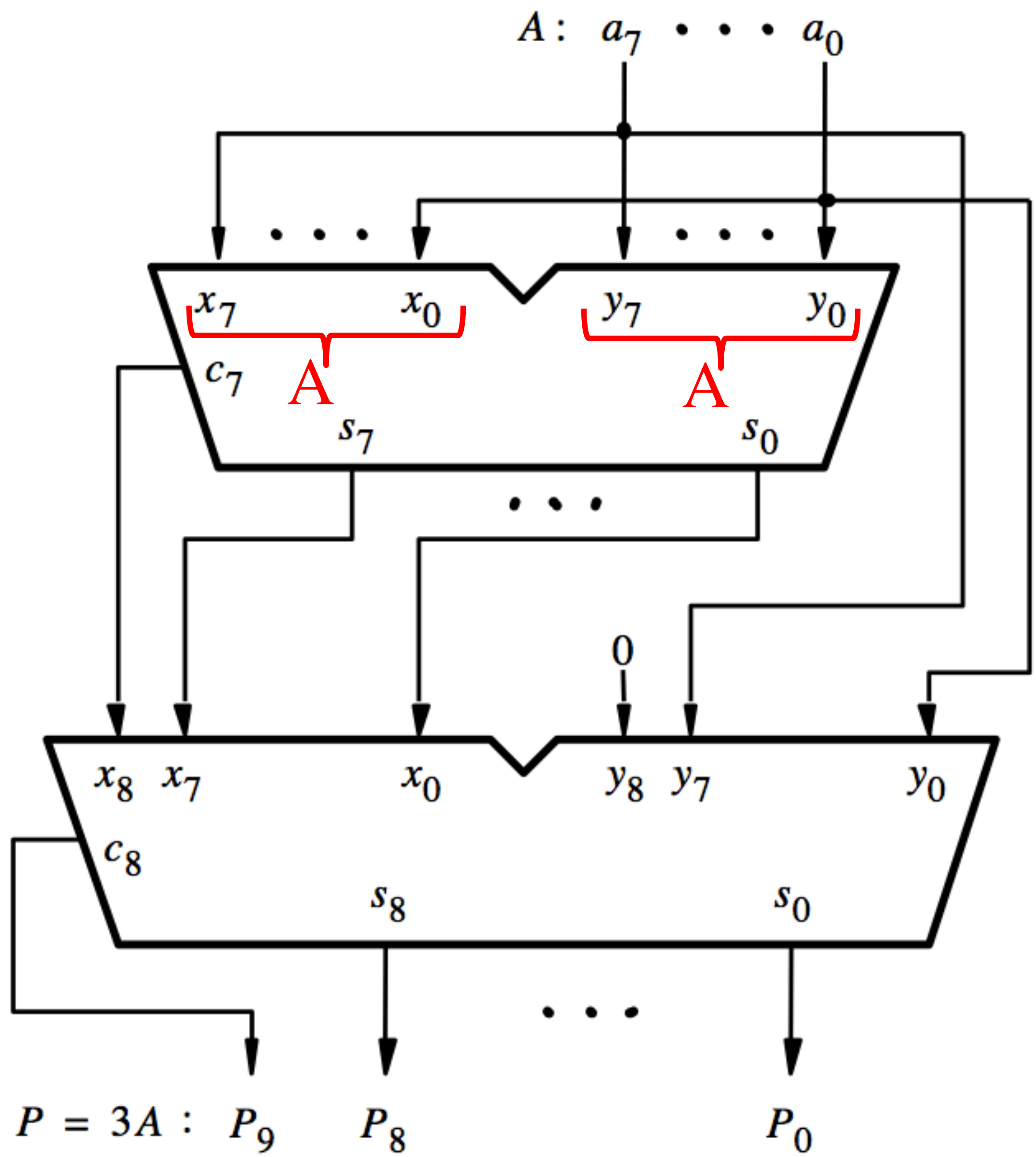
Create a circuit that multiplies a number by 3

How to Get 3A from A?

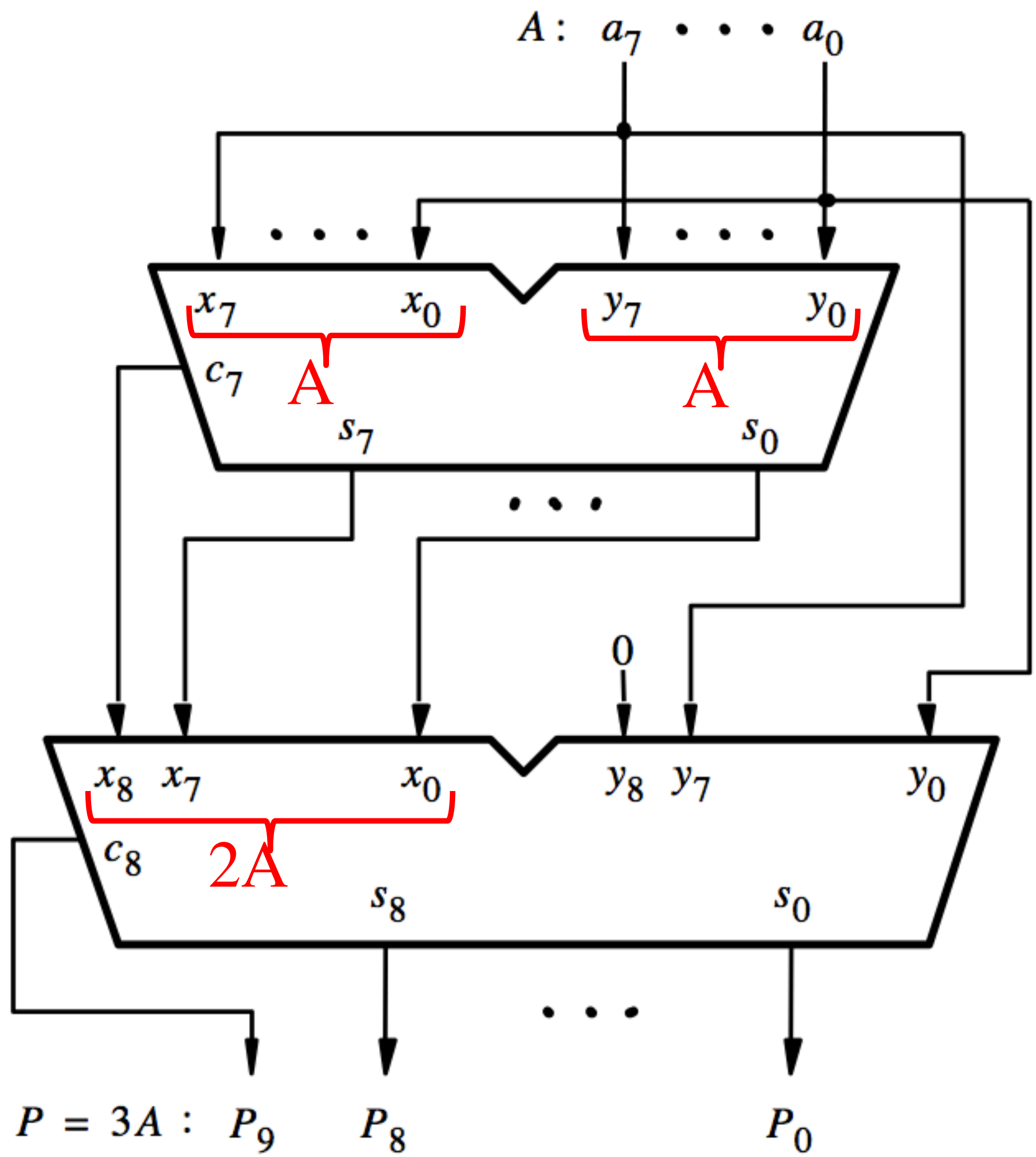
- 3A = A + A + A
- 3A = (A+A) + A
- 3A = 2A +A



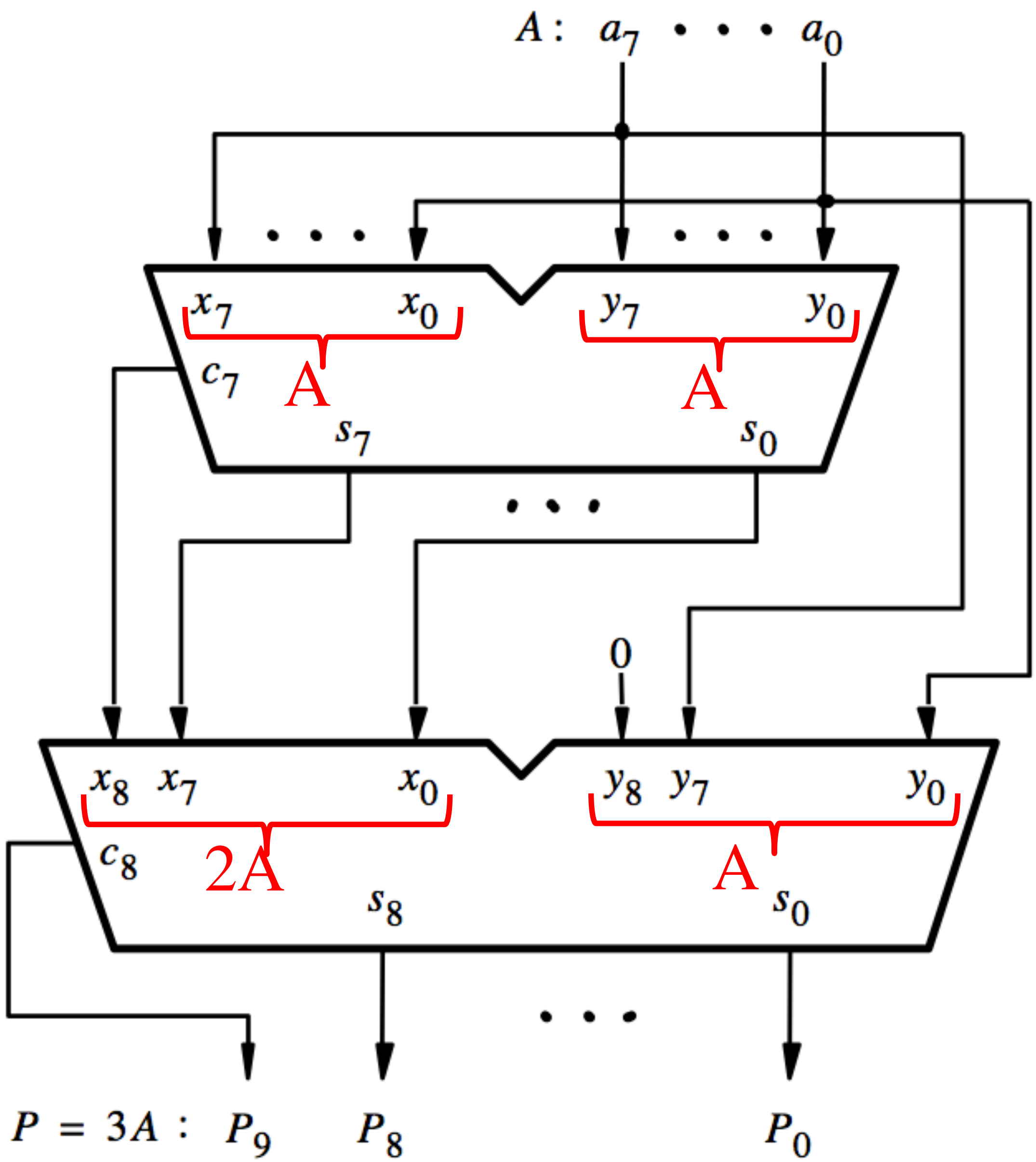
[Figure 3.6a from the textbook]



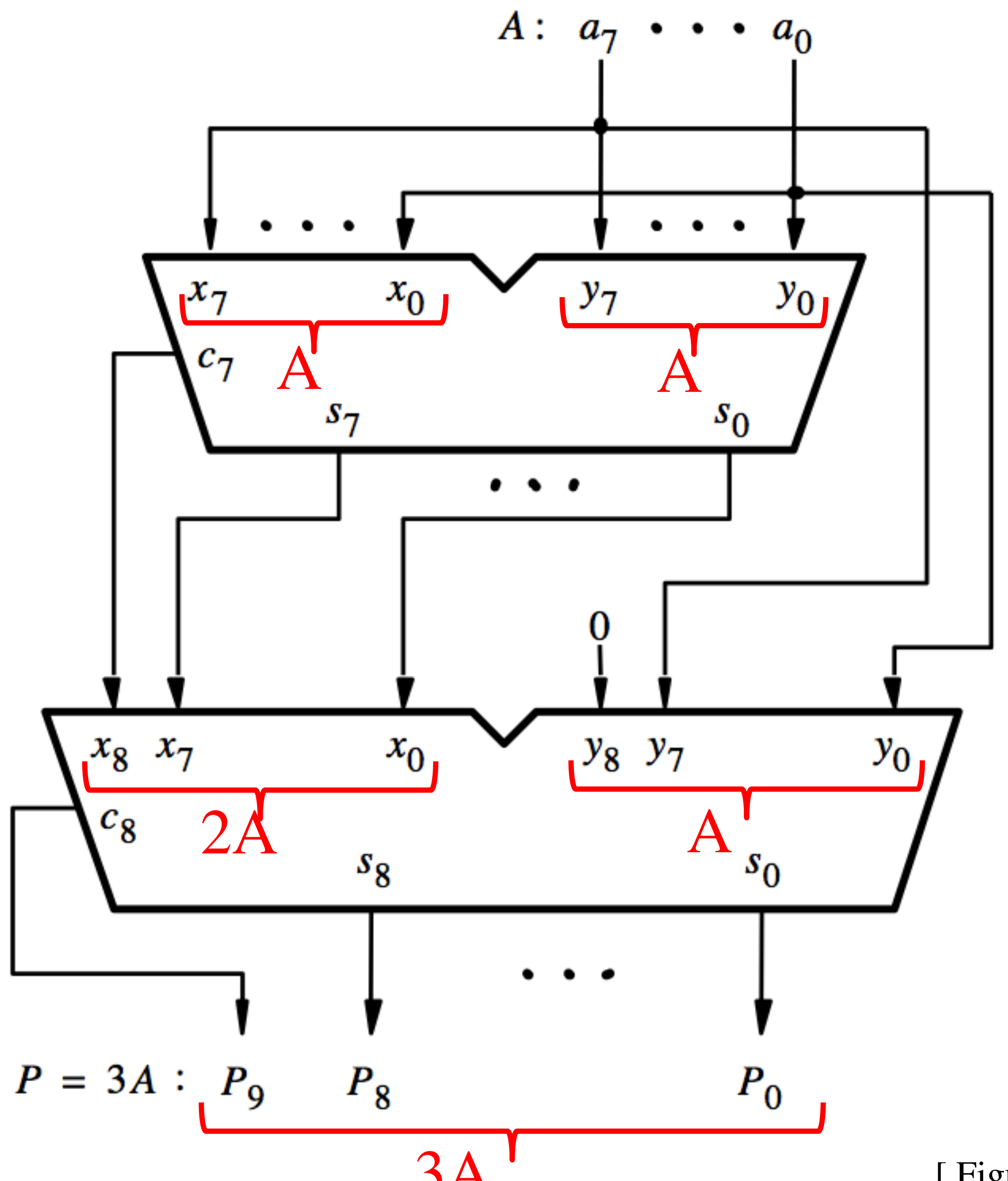
[Figure 3.6a from the textbook]



[Figure 3.6a from the textbook]



[Figure 3.6a from the textbook]



[Figure 3.6a from the textbook]

Decimal Multiplication by 10

What happens when we multiply a number by 10?

4 x 10 = ?

542 x 10 = ?

1245 x 10 = ?

Decimal Multiplication by 10

What happens when we multiply a number by 10?

4 x 10 = 40

542 x 10 = 5420

1245 x 10 = 12450

Decimal Multiplication by 10

What happens when we multiply a number by 10?

4 x 10 = 40

542 x 10 = 5420

1245 x 10 = 12450

You simply add a zero as the rightmost number

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 times 2 = 0110

101 times 2 = 1010

110011 times 2 = 1100110

Binary Multiplication by 2

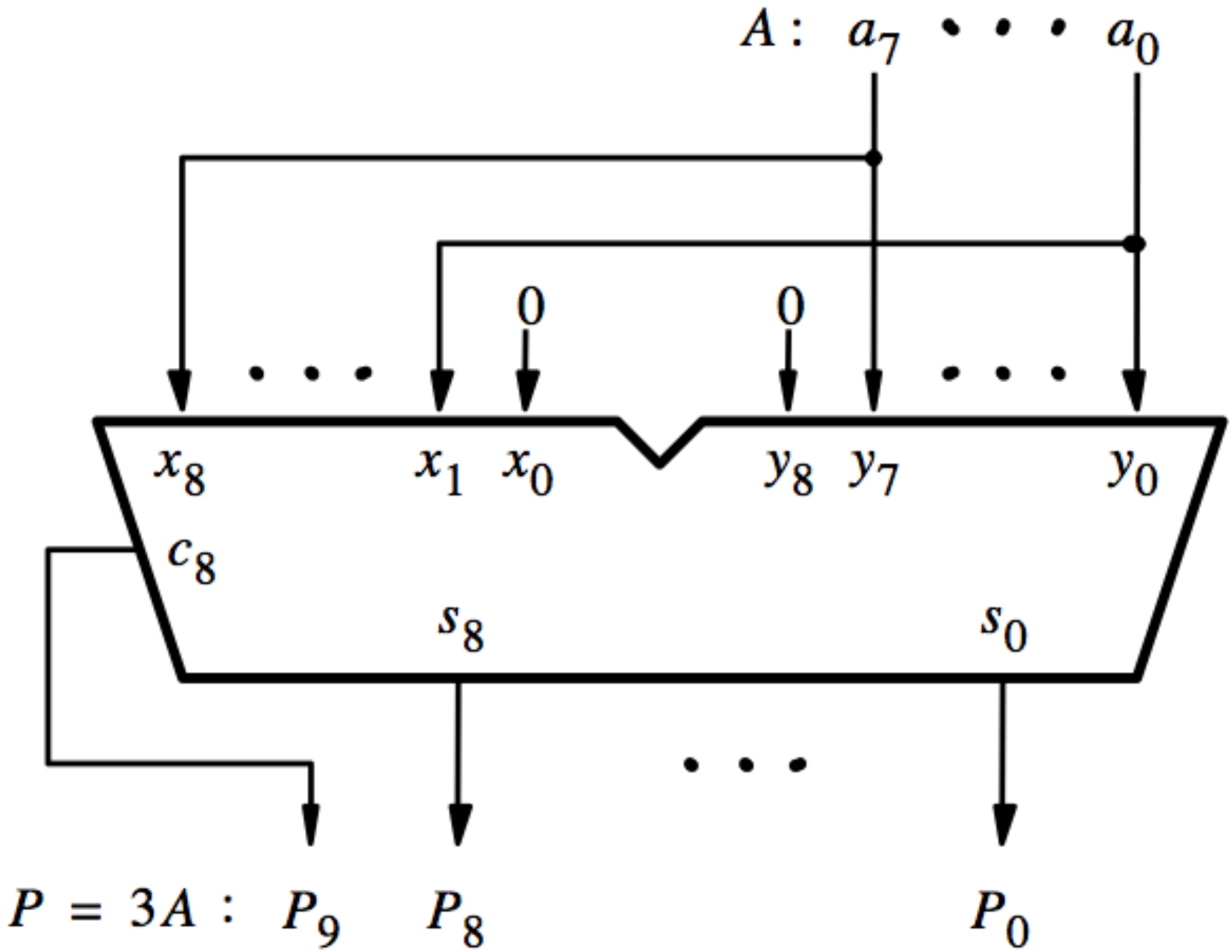
What happens when we multiply a number by 2?

011 times 2 = 0110

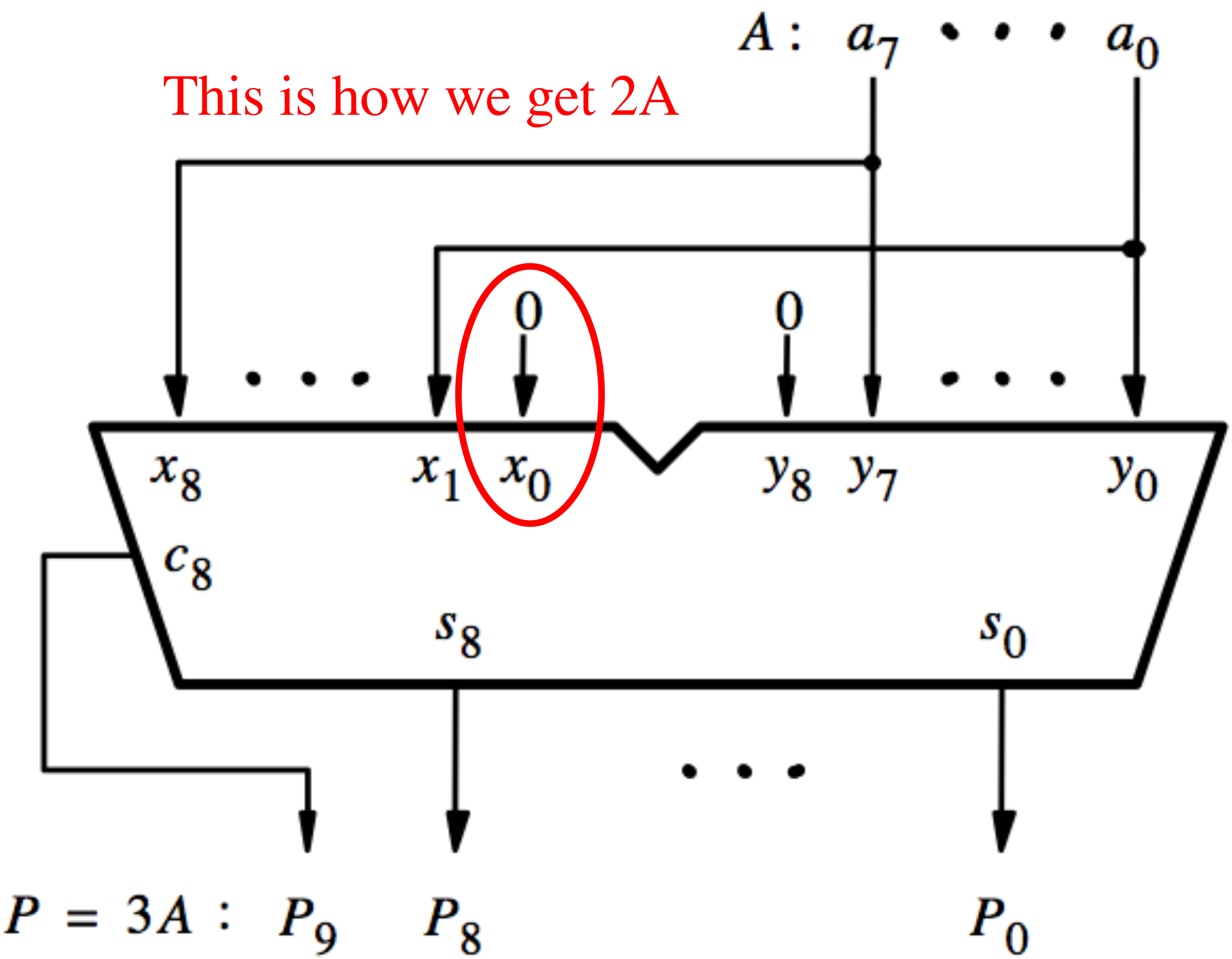
101 times 2 = 1010

110011 times 2 = 1100110

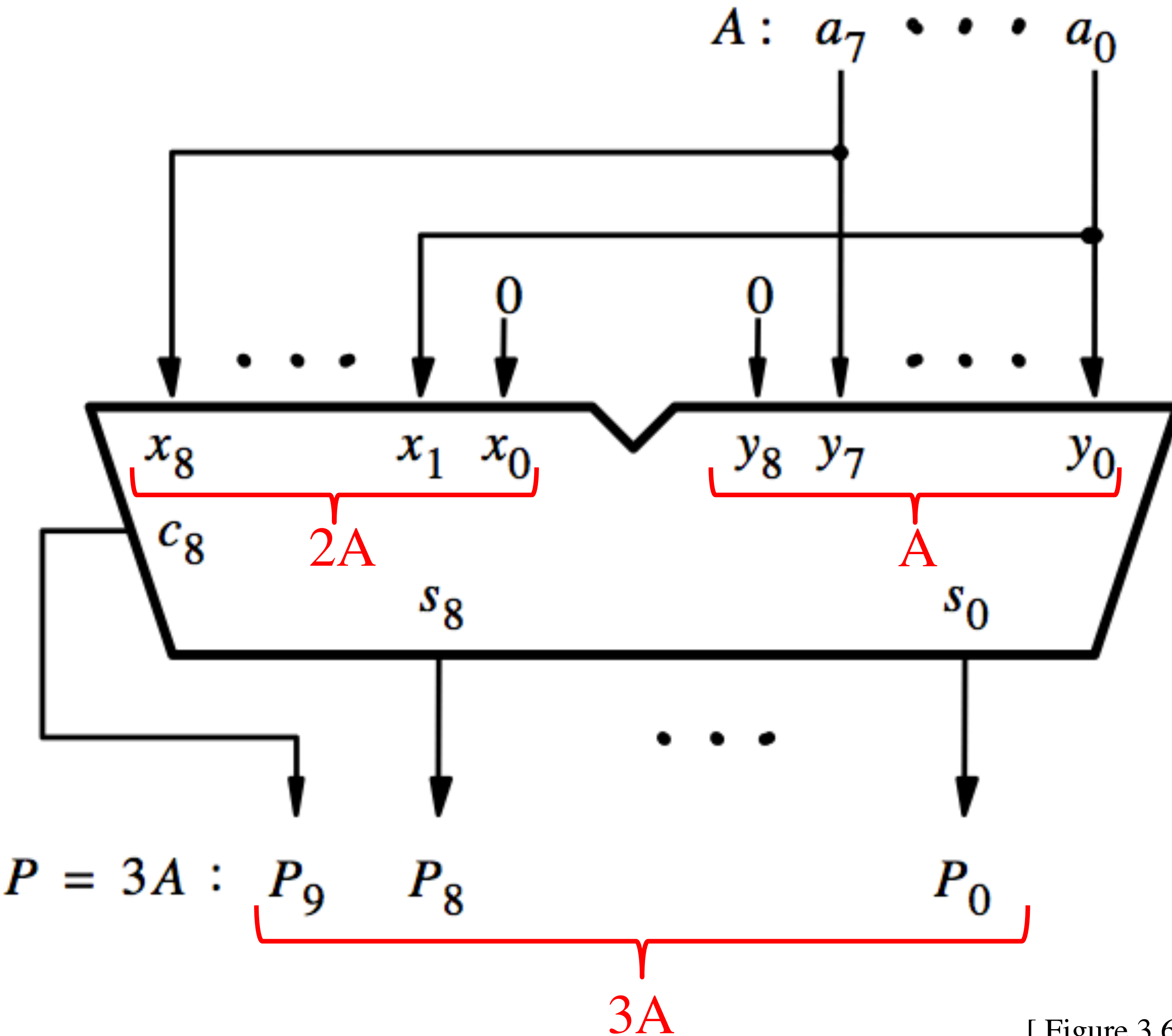
You simply add a zero as the rightmost number



[Figure 3.6b from the textbook]



[Figure 3.6b from the textbook]



[Figure 3.6b from the textbook]

Questions?

THE END