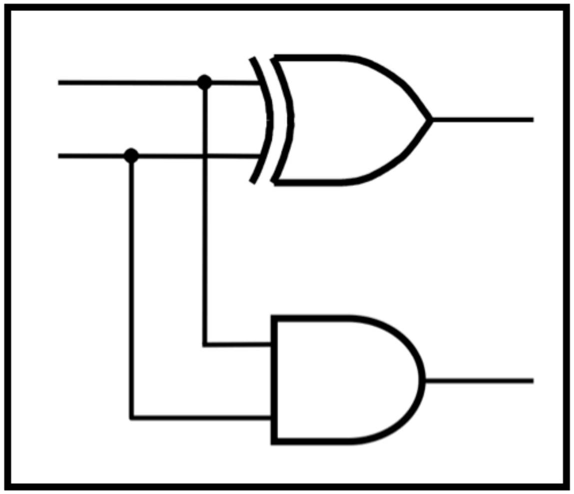


CprE 281: Digital Logic

Instructor: Alexander Stoytchev

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

Fast Adders

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

Administrative Stuff

- HW5 is out
- It is due on Monday Oct 5 @ 4pm.
- Please write clearly on the first page (in block capital letters) the following three things:
    - Your First and Last Name
    - Your Student ID Number
    - Your Lab Section Letter

Administrative Stuff

- **Labs Next Week**
- **Mini-Project**
- **This one is worth 3% of your grade.**
- **Make sure to get all the points.**
- **http://www.ece.iastate.edu/~alexs/classes/2015_Fall_281/labs/Project-Mini/**

Quick Review

The problems in which row are easier to calculate?

$$\begin{array}{r} 82 \\ - \; 61 \\ \hline ?? \end{array} \qquad \begin{array}{r} 48 \\ - \; 26 \\ \hline ?? \end{array} \qquad \begin{array}{r} 32 \\ - \; 11 \\ \hline ?? \end{array}$$

$$\begin{array}{r} 82 \\ - \; 64 \\ \hline ?? \end{array} \qquad \begin{array}{r} 48 \\ - \; 29 \\ \hline ?? \end{array} \qquad \begin{array}{r} 32 \\ - \; 13 \\ \hline ?? \end{array}$$

The problems in which row are easier to calculate?

$$\begin{array}{r} 82 \\ -\ 61 \\ \hline 21 \end{array} \qquad \begin{array}{r} 48 \\ -\ 26 \\ \hline 22 \end{array} \qquad \begin{array}{r} 32 \\ -\ 11 \\ \hline 21 \end{array}$$

Why?

$$\begin{array}{r} 82 \\ -\ 64 \\ \hline 18 \end{array} \qquad \begin{array}{r} 48 \\ -\ 29 \\ \hline 19 \end{array} \qquad \begin{array}{r} 32 \\ -\ 13 \\ \hline 19 \end{array}$$

Another Way to Do Subtraction

$$82 – 64 = 82 + 100 – 100 - 64$$

Another Way to Do Subtraction

$$82 – 64 = 82 + 100 – 100 - 64$$

$$= 82 + (100 – 64) - 100$$

Another Way to Do Subtraction

$$82 - 64 = 82 + 100 - 100 - 64$$

$$= 82 + (100 - 64) - 100$$

$$= 82 + (99 + 1 - 64) - 100$$

Another Way to Do Subtraction

$$82 – 64 = 82 + 100 – 100 - 64$$

$$= 82 + (100 – 64) - 100$$

$$= 82 + (99 + 1 – 64) - 100$$

$$= 82 + (99 – 64) +1 - 100$$

Another Way to Do Subtraction

$$82 - 64 = 82 + 100 - 100 - 64$$

$$= 82 + (100 - 64) - 100$$

$$= 82 + (99 + 1 - 64) - 100$$

Does not require borrows

$$= 82 + (99 - 64) + 1 - 100$$

9's Complement
(subtract each digit from 9)

$$\begin{array}{r} 99 \\ -\ 64 \\ \hline 35 \end{array}$$

10's Complement
(subtract each digit from 9 and add 1 to the result)

$$
\begin{array}{r}
99 \\
-\ 64 \\
\hline
35 + 1 = 36
\end{array}
$$

Another Way to Do Subtraction

$$82 - 64 = 82 + (99 - 64) + 1 - 100$$

Another Way to Do Subtraction

$$\underbrace{}_{}$$

9's complement

$$82 - 64 = 82 + (99 - 64) + 1 - 100$$

Another Way to Do Subtraction

9's complement

$$82 - 64 = 82 + (99 - 64) + 1 - 100$$

$$= 82 + 35 + 1 - 100$$

Another Way to Do Subtraction

9's complement

$$82 - 64 = 82 + (99 - 64) + 1 - 100$$

10's complement

$$= 82 + 35 + 1 - 100$$

Another Way to Do Subtraction

9's complement

$$82 - 64 = 82 + (99 - 64) + 1 - 100$$

10's complement

$$= 82 + 35 + 1 - 100$$

$$= 82 + 36 - 100$$ // add the first two

Another Way to Do Subtraction

9's complement

$$82 - 64 = 82 + (99 - 64) + 1 - 100$$

10's complement

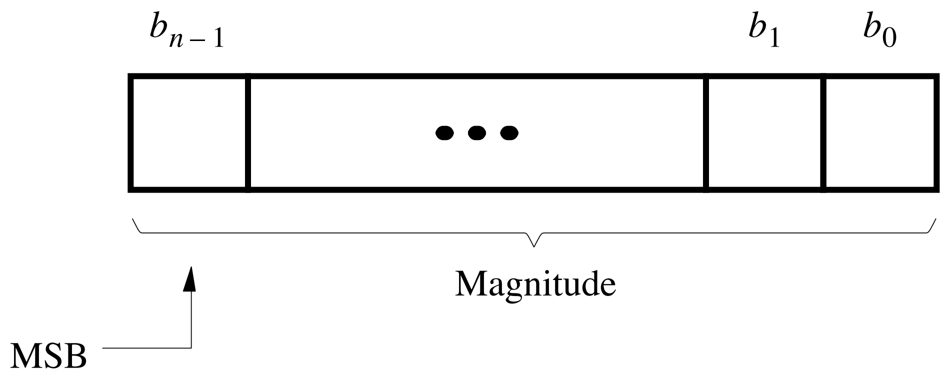
$$= 82 + 35 + 1 - 100$$

$$= 82 + 36 - 100$$ // add the first two

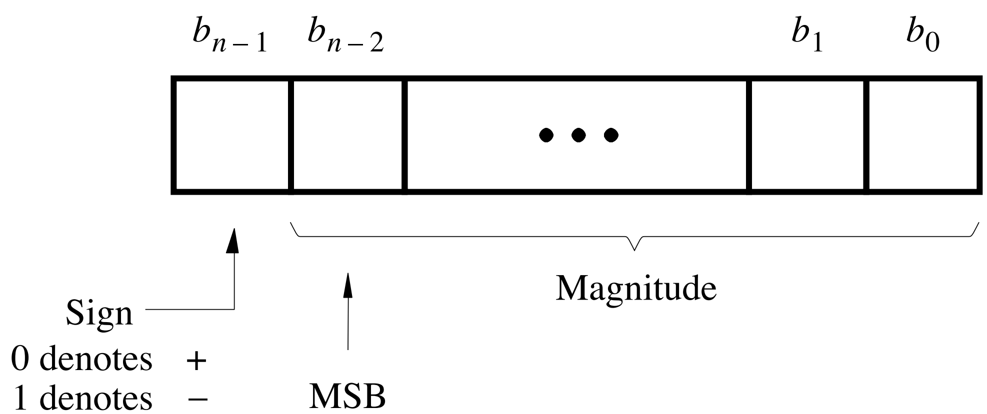
$$= 118 - 100$$ // delete the leading 1

$$= 18$$

Formats for representation of integers



(a) Unsigned number



(b) Signed number

[Figure 3.7 from the textbook]

Negative numbers can be represented in following ways

- Sign and magnitude
- 1’s complement
- 2’s complement

1's complement

Let K be the negative equivalent of an n-bit positive number P.

Then, in 1's complement representation K is obtained by subtracting P from $2^n - 1$, namely

$$K = (2^n - 1) - P$$

This means that K can be obtained by inverting all bits of P.

Find the 1's complement of …

0 1 0 1

0 0 1 0

0 0 1 1

0 1 1 1

Find the 1's complement of …

0 1 0 1	0 0 1 0
1 0 1 0	1 1 0 1
0 0 1 1	0 1 1 1
1 1 0 0	1 0 0 0

Just flip 1's to 0's and vice versa.

Example of 1's complement addition

$$
\begin{array}{r}
(+5) \\
+\,(+2) \\
\hline
(+7)
\end{array}
\qquad
\begin{array}{r}
0\,1\,0\,1 \\
+\,0\,0\,1\,0 \\
\hline
0\,1\,1\,1
\end{array}
$$

Example of 1's complement addition

$$\begin{array}{rr} (-5) & 1\,0\,1\,0 \\ +\,(+2) & +\,0\,0\,1\,0 \\ \hline (-3) & 1\,1\,0\,0 \end{array}$$

[Figure 3.8 from the textbook]

Example of 1's complement addition

$$
\begin{array}{r}
(+5) \\
+(-2) \\
\hline
(+3)
\end{array}
\qquad\qquad
\begin{array}{r}
0\,1\,0\,1 \\
+\,1\,1\,0\,1 \\
\hline
1\,0\,0\,1\,0 \\
1 \\
\hline
0\,0\,1\,1
\end{array}
$$

[Figure 3.8 from the textbook]

Example of 1's complement addition

$$\begin{array}{r} (-5) \\ +\,(-2) \\ \hline (-7) \end{array} \qquad \begin{array}{r} 1\,0\,1\,0 \\ +\,1\,1\,0\,1 \\ \hline 1\,0\,1\,1\,1 \\ 1 \\ \hline 1\,0\,0\,0 \end{array}$$

[Figure 3.8 from the textbook]

2's complement

Let K be the negative equivalent of an n-bit positive number P.

Then, in 2's complement representation K is obtained by subtracting P from 2^n , namely

$$K = 2^n - P$$

Deriving 2's complement

For a positive n-bit number P, let K_1 and K_2 denote its 1's and 2's complements, respectively.

$$K_1 = (2^n - 1) - P$$
$$K_2 = 2^n - P$$

Since $K_2 = K_1 + 1$, it is evident that in a logic circuit the 2's complement can computed by inverting all bits of P and then adding 1 to the resulting 1's-complement number.

Find the 2' s complement of …

0 1 0 1

0 0 1 0

0 0 1 1

0 1 1 1

Find the 2's complement of …

0 1 0 1

1 0 1 1

0 0 1 0

1 1 1 0

0 0 1 1

1 1 0 1

0 1 1 1

1 0 0 1

Quick Way to find 2's complement

- Scan the binary number from right to left
- Copy all bits that are 0 from right to left
- Stop at the first 1
- Copy that 1 as well
- Invert all remaining bits

Interpretation of four-bit signed integers

$b_3b_2b_1b_0$	Sign and magnitude	1's complement	2's complement
0111	+7	+7	+7
0110	+6	+6	+6
0101	+5	+5	+5
0100	+4	+4	+4
0011	+3	+3	+3
0010	+2	+2	+2
0001	+1	+1	+1
0000	+0	+0	+0
1000	−0	−7	−8
1001	−1	−6	−7
1010	−2	−5	−6
1011	−3	−4	−5
1100	−4	−3	−4
1101	−5	−2	−3
1110	−6	−1	−2
1111	−7	−0	−1

[Table 3.2 from the textbook]

Example of 2's complement addition

$(+5)$	$0\,1\,0\,1$
$+\ (+2)$	$+\ 0\,0\,1\,0$
$(+7)$	$0\,1\,1\,1$

[Figure 3.9 from the textbook]

Example of 2's complement addition

(-5)	$1\,0\,1\,1$
$+\ (+2)$	$+\ 0\,0\,1\,0$
(-3)	$1\,1\,0\,1$

[Figure 3.9 from the textbook]

Example of 2's complement addition

$$\begin{array}{rr} (+5) & 0\ 1\ 0\ 1 \\ +\ \ (-2) & +\ 1\ 1\ 1\ 0 \\ \hline (+3) & 1\ \ 0\ 0\ 1\ 1 \end{array}$$

ignore

[Figure 3.9 from the textbook]

Example of 2' s complement addition

$$\begin{array}{rr} & (-5) \\ + & (-2) \\ \hline & (-7) \end{array} \qquad \begin{array}{rr} & 1\,0\,1\,1 \\ + & 1\,1\,1\,0 \\ \hline & 1\,1\,0\,0\,1 \end{array}$$

ignore

[Figure 3.9 from the textbook]

Example of 2's complement subtraction

$$\begin{array}{r} (+5) \\ -\ (+2) \\ \hline (+3) \end{array} \qquad \begin{array}{r} 0\,1\,0\,1 \\ -\ 0\,0\,1\,0 \\ \hline \end{array} \qquad \Longrightarrow \qquad \begin{array}{r} 0\,1\,0\,1 \\ +\ 1\,1\,1\,0 \\ \hline 1\,0\,0\,1\,1 \end{array}$$

ignore

[Figure 3.10 from the textbook]

Example of 2’s complement subtraction

$$
\begin{array}{r}
(-5) \\
-\ (+2) \\
\hline
(-7)
\end{array}
\qquad
\begin{array}{r}
1\,0\,1\,1 \\
-\ 0\,0\,1\,0 \\
\hline
\end{array}
\qquad \Longrightarrow \qquad
\begin{array}{r}
1\,0\,1\,1 \\
+\ 1\,1\,1\,0 \\
\hline
1\,1\,0\,0\,1
\end{array}
$$

ignore

[Figure 3.10 from the textbook]

Example of 2’s complement subtraction

$$
\begin{array}{r}
(+5) \\
-\ (-2) \\
\hline
(+7)
\end{array}
\qquad
\begin{array}{r}
0\,1\,0\,1 \\
-\ 1\,1\,1\,0 \\
\hline
\end{array}
\quad \Longrightarrow \quad
\begin{array}{r}
0\,1\,0\,1 \\
+\ 0\,0\,1\,0 \\
\hline
0\,1\,1\,1
\end{array}
$$

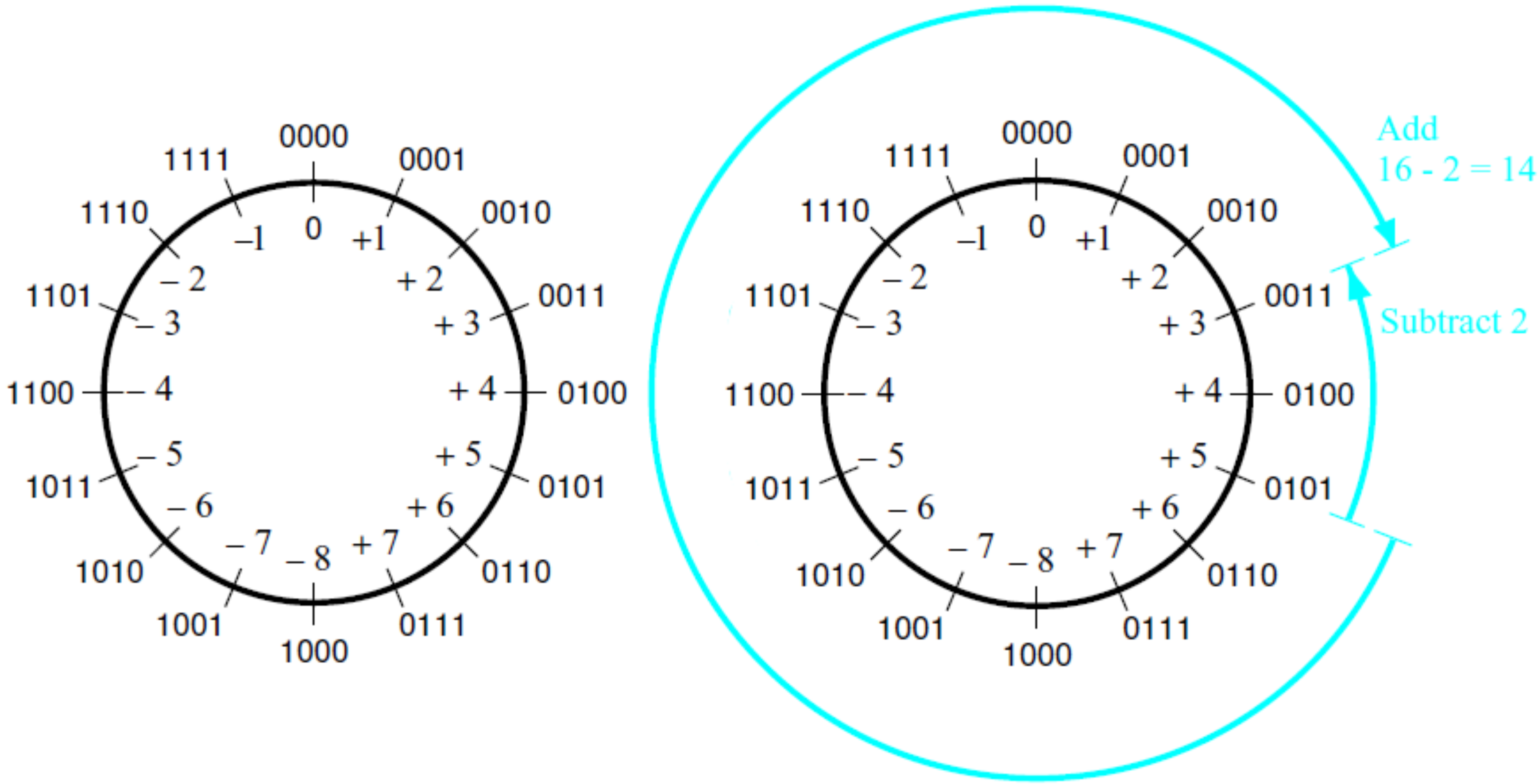
[Figure 3.10 from the textbook]

Example of 2’s complement subtraction

$$
\begin{array}{rr}
 & (-5) \\
- & (-2) \\
\hline
 & (-3)
\end{array}
\qquad
\begin{array}{rr}
 & 1\,0\,1\,1 \\
- & 1\,1\,1\,0 \\
\hline
\end{array}
\qquad \Longrightarrow \qquad
\begin{array}{rr}
 & 1\,0\,1\,1 \\
+ & 0\,0\,1\,0 \\
\hline
 & 1\,1\,0\,1
\end{array}
$$

[Figure 3.10 from the textbook]

Graphical interpretation of four-bit 2's complement numbers



(a) The number circle (b) Subtracting 2 by adding its 2's complement

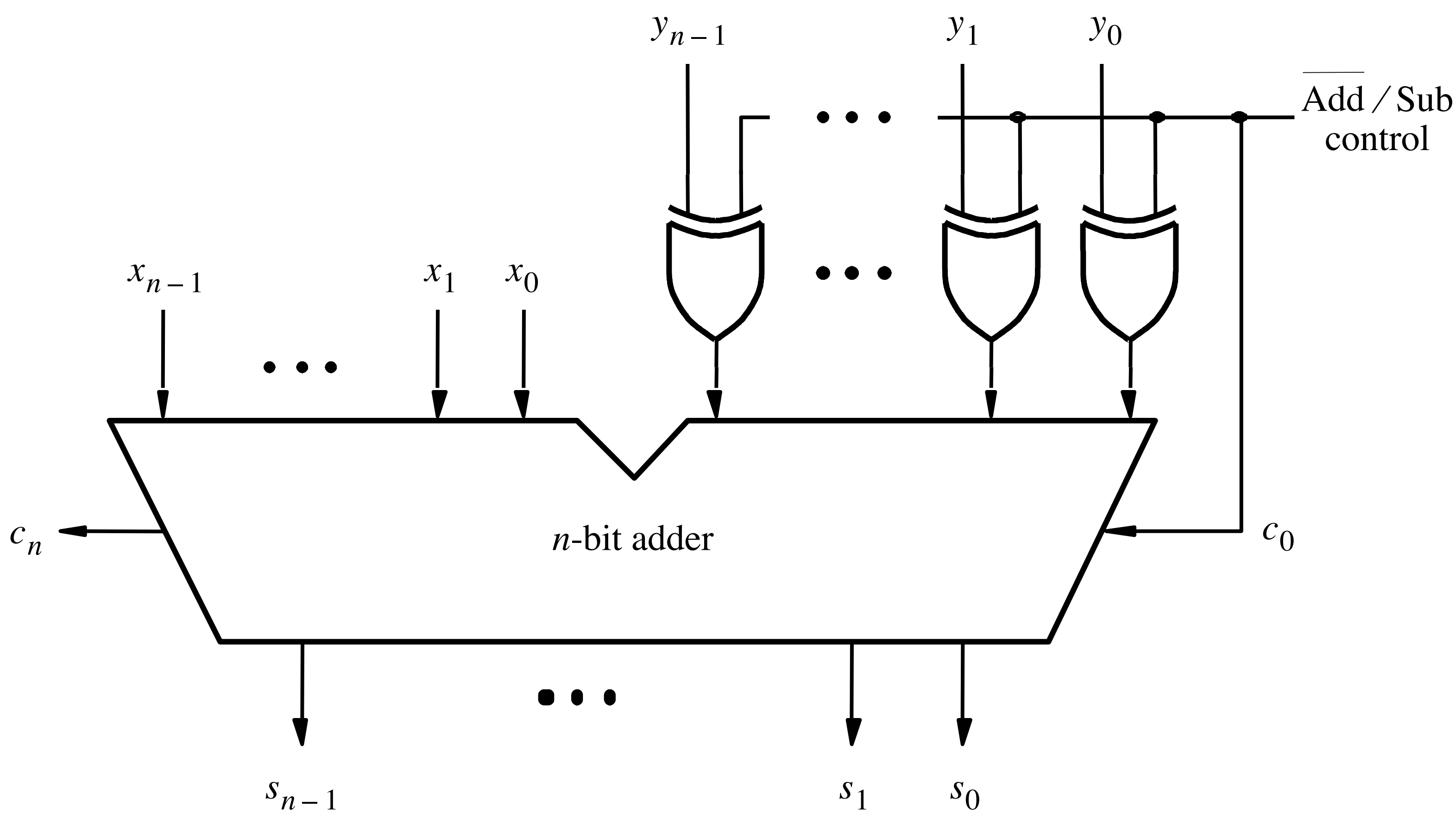
[Figure 3.11 from the textbook]

Take Home Message

- Subtraction can be performed by simply adding the 2’s complement of the second number, regardless of the signs of the two numbers.

- Thus, the same adder circuit can be used to perform both addition and subtraction !!!

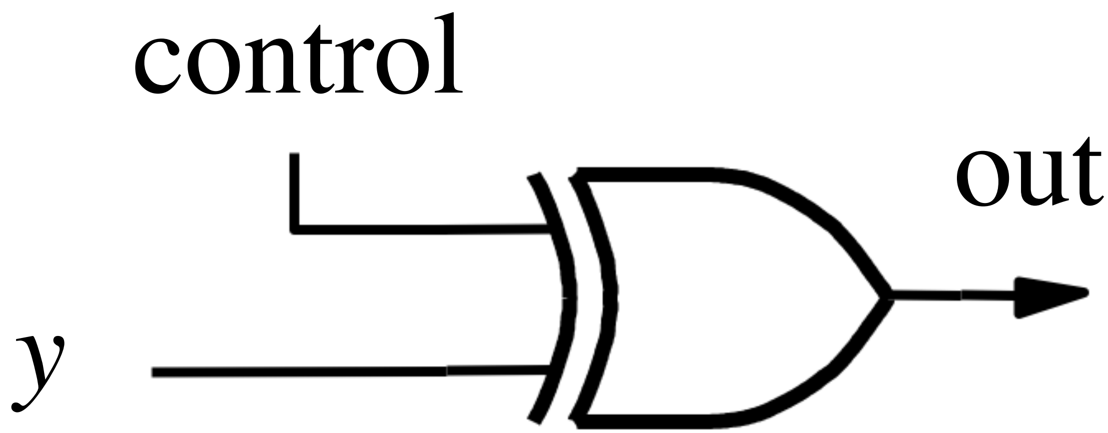
Adder/subtractor unit



[Figure 3.12 from the textbook]

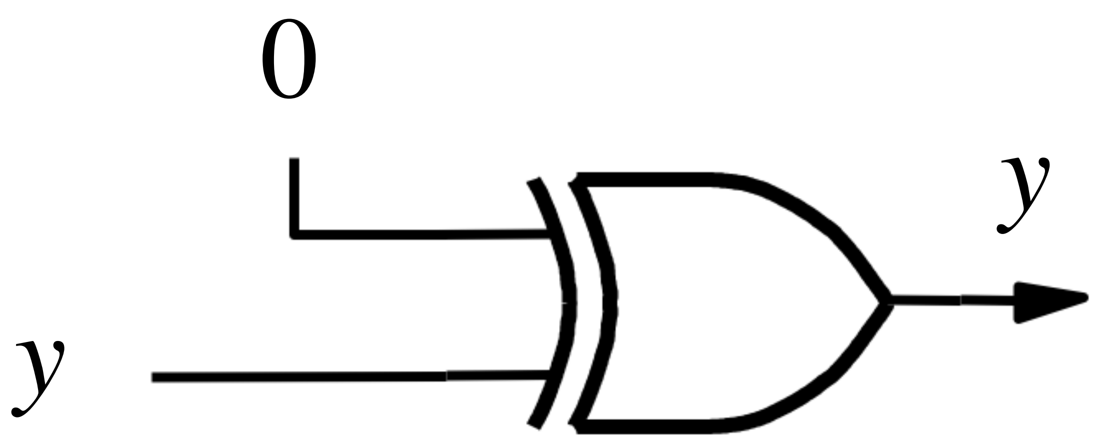
XOR Tricks

control	y	out
0	0	0
0	1	1
1	0	1
1	1	0



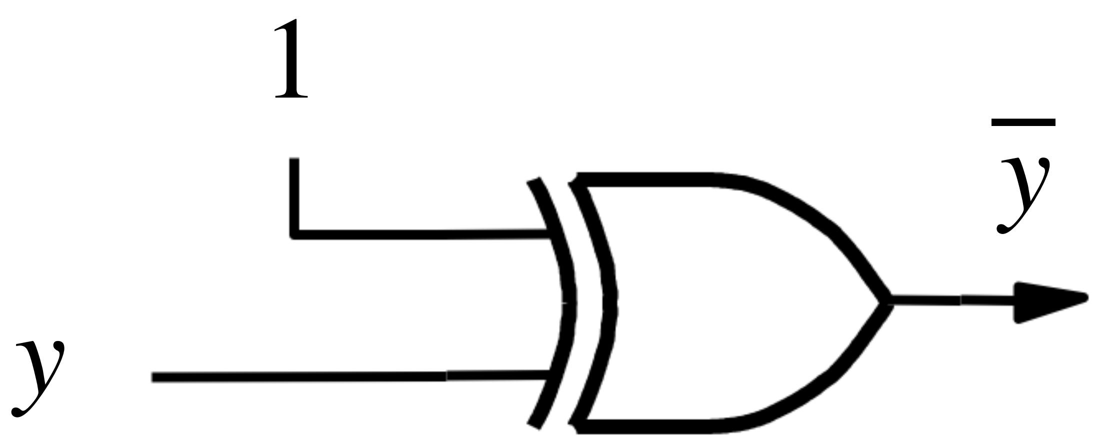
XOR as a repeater

control	y	out
0	0	0
0	1	1
1	0	1
1	1	0

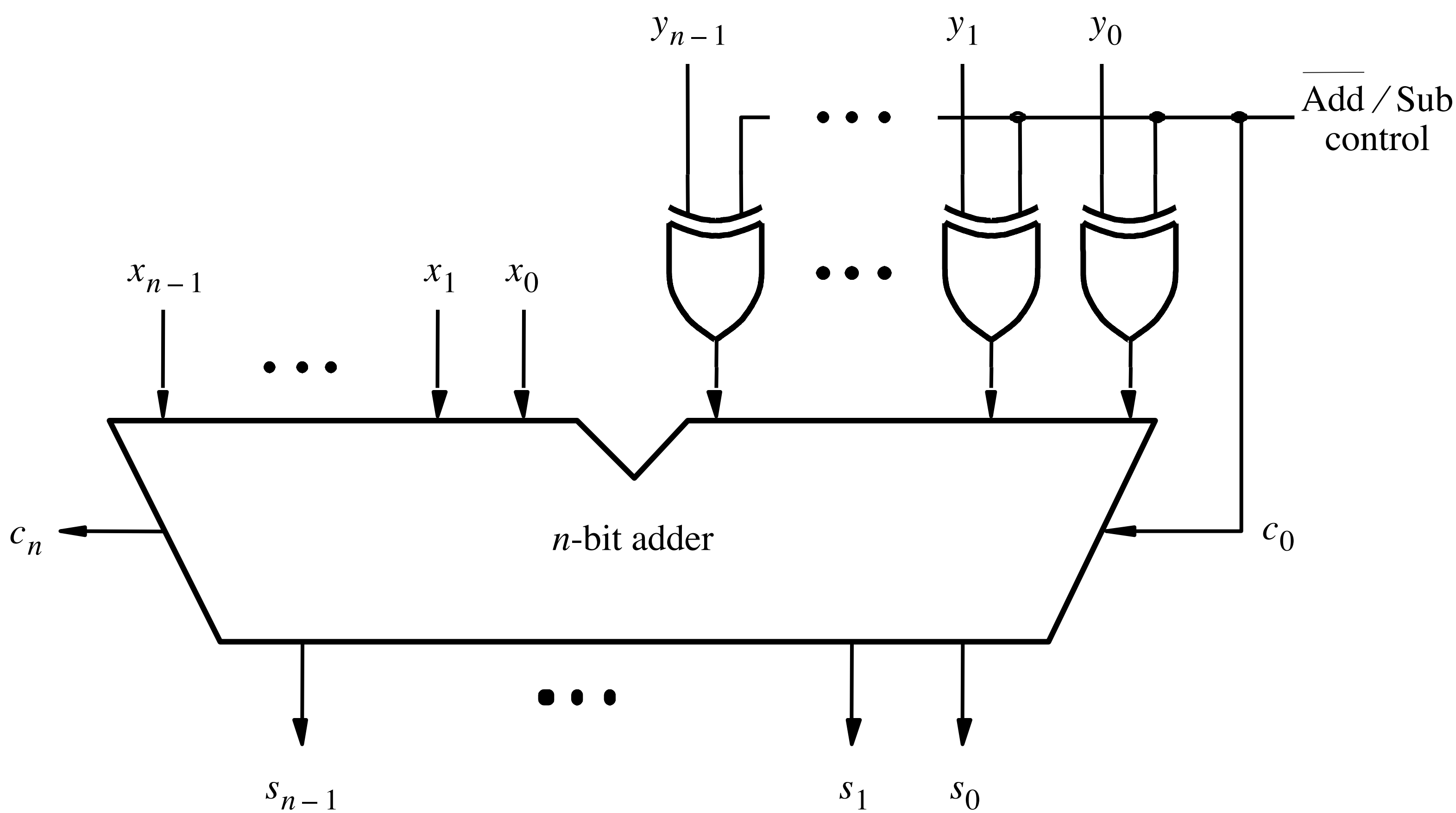


XOR as an inverter

control	y	out
0	0	0
0	1	1
1	0	1
1	1	0

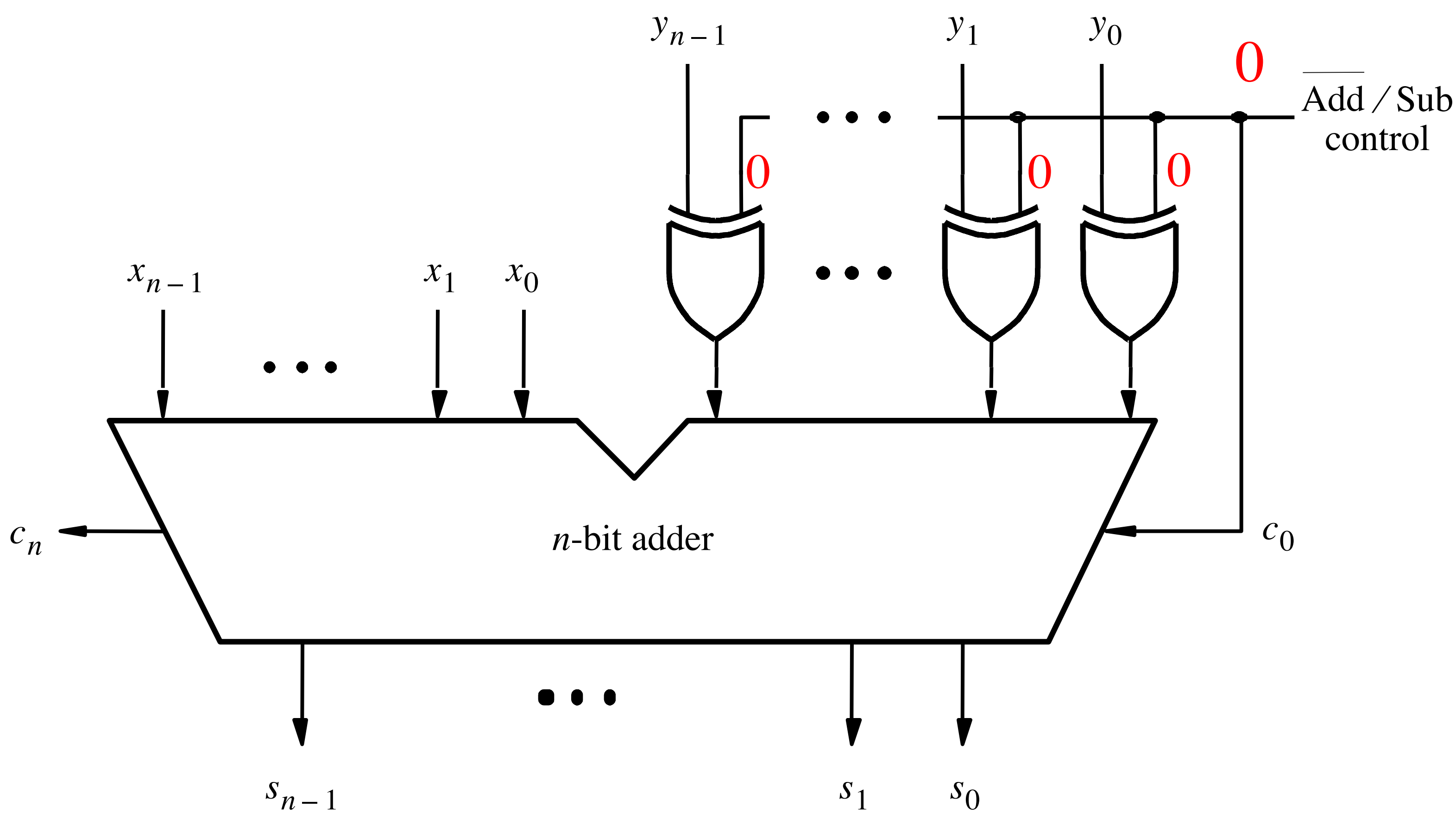


Addition: when control = 0



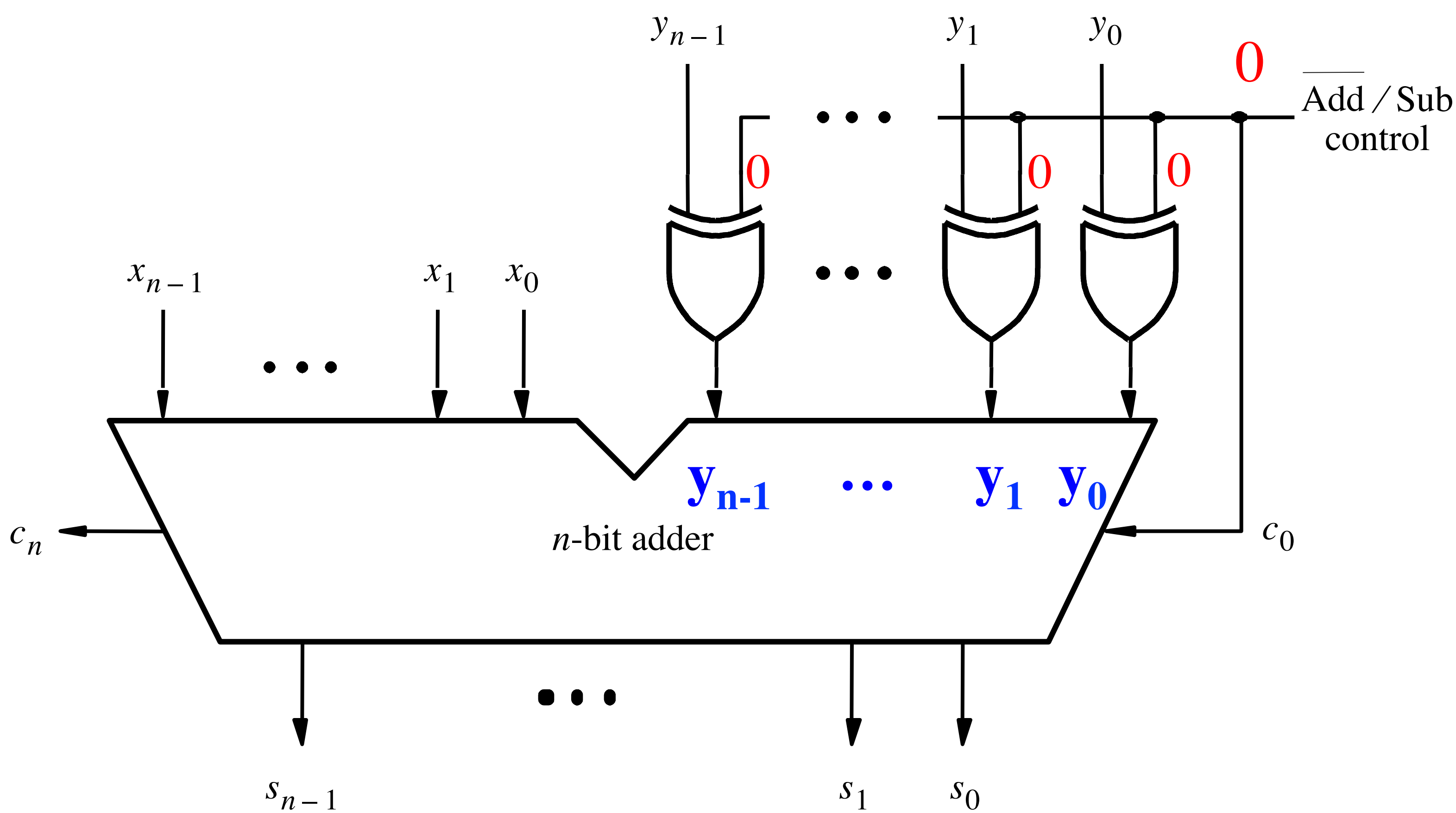
[Figure 3.12 from the textbook]

Addition: when control = 0



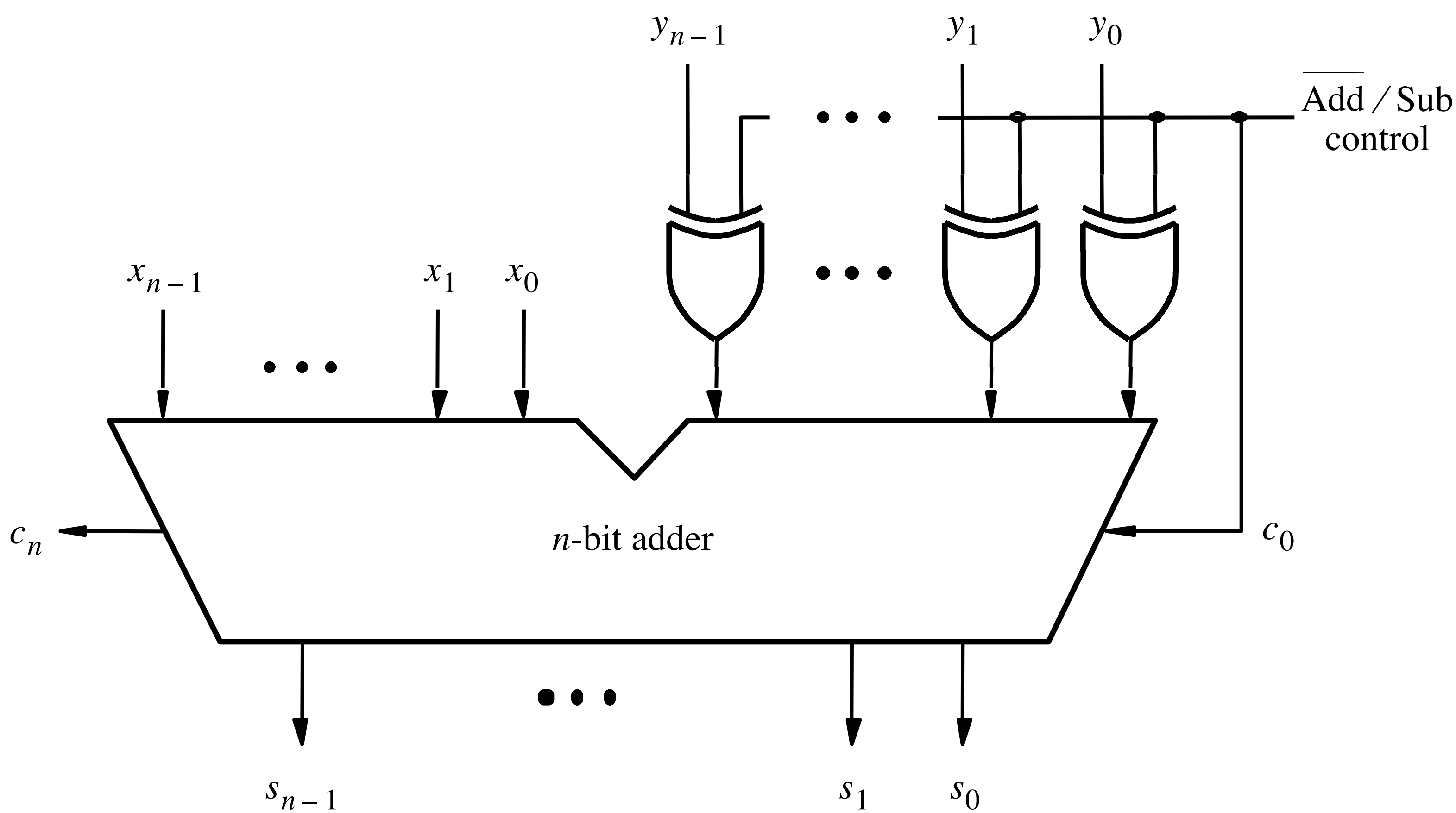
[Figure 3.12 from the textbook]

Addition: when control = 0



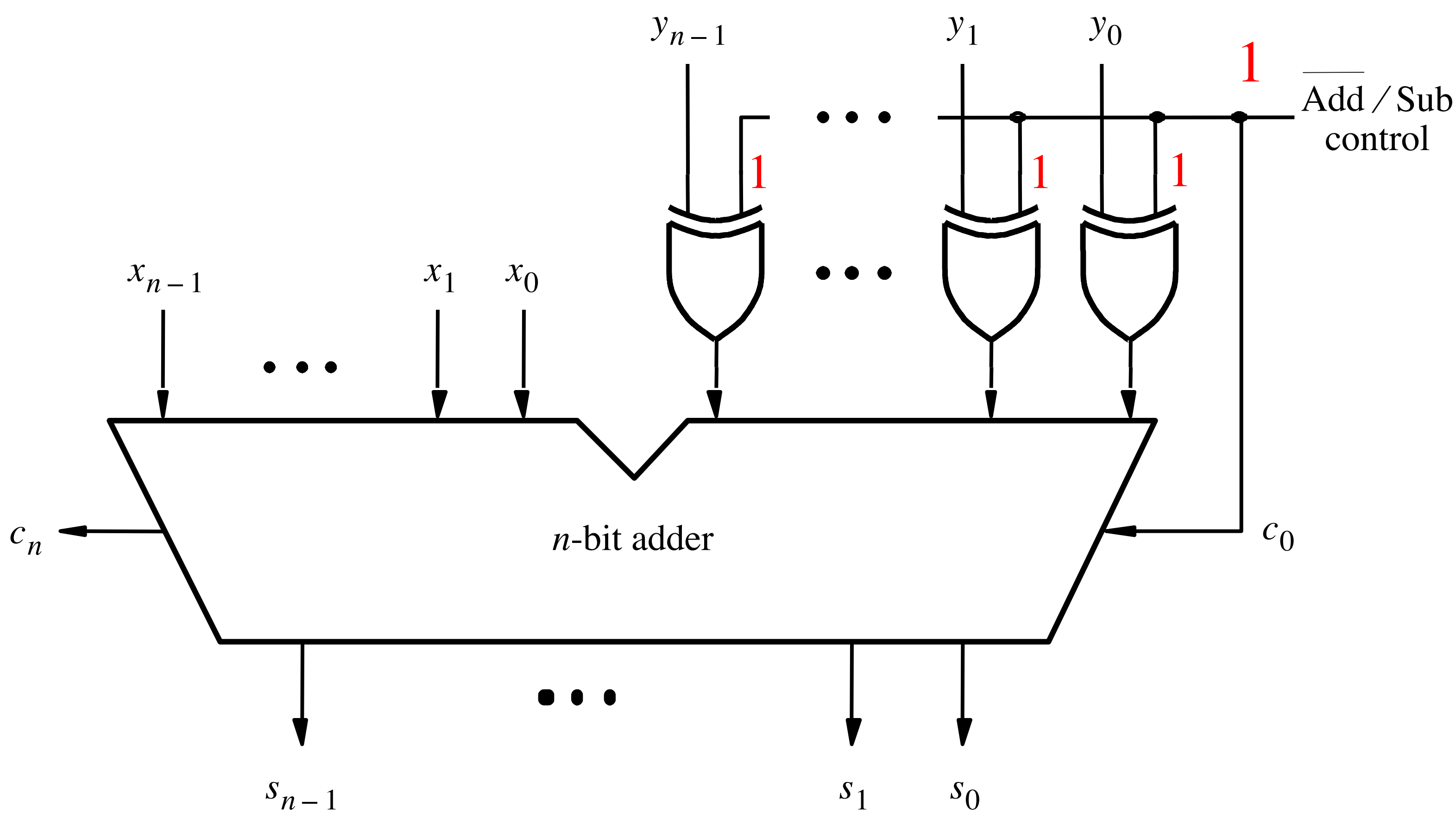
[Figure 3.12 from the textbook]

Subtraction: when control = 1



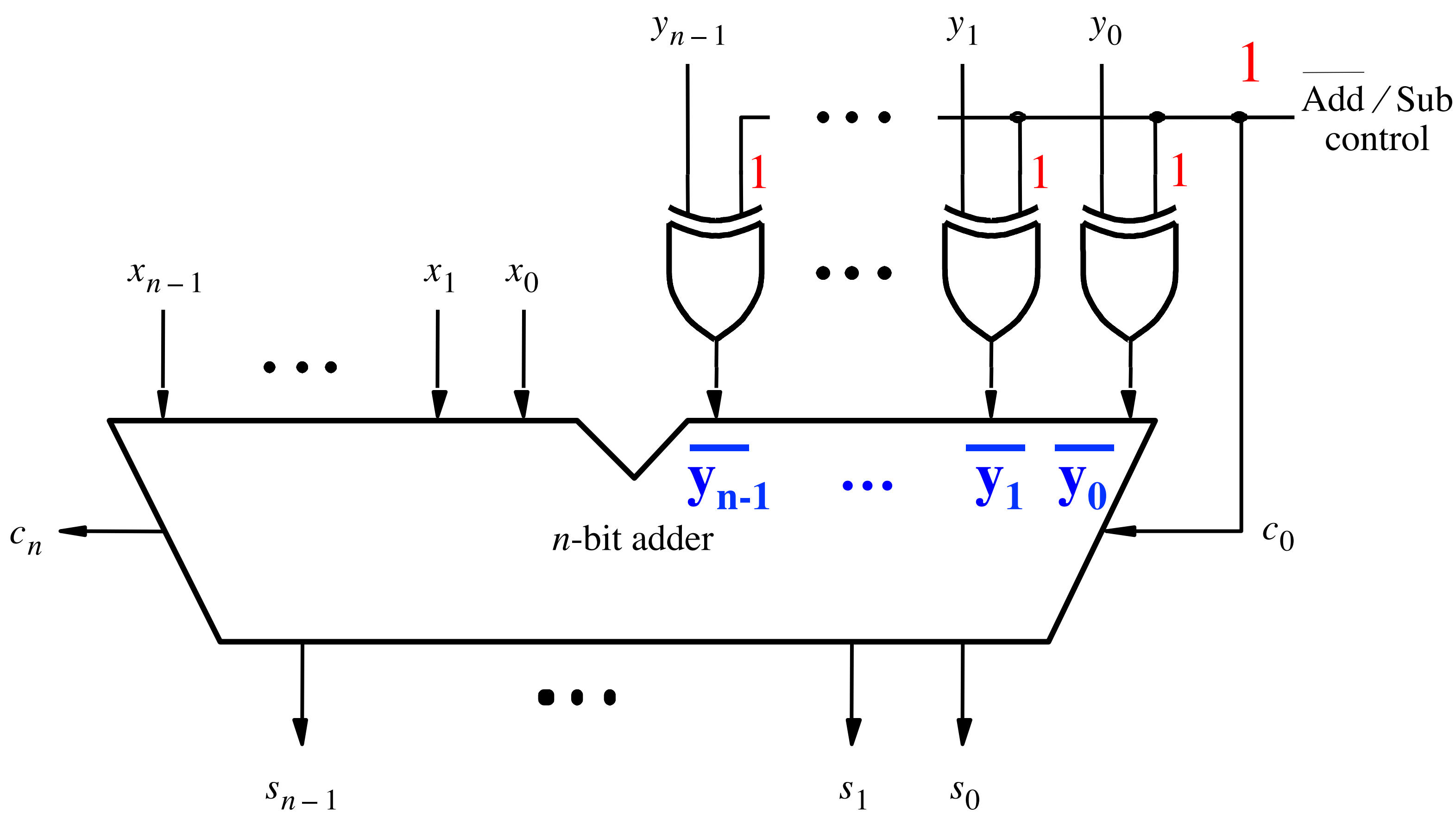
[Figure 3.12 from the textbook]

Subtraction: when control = 1



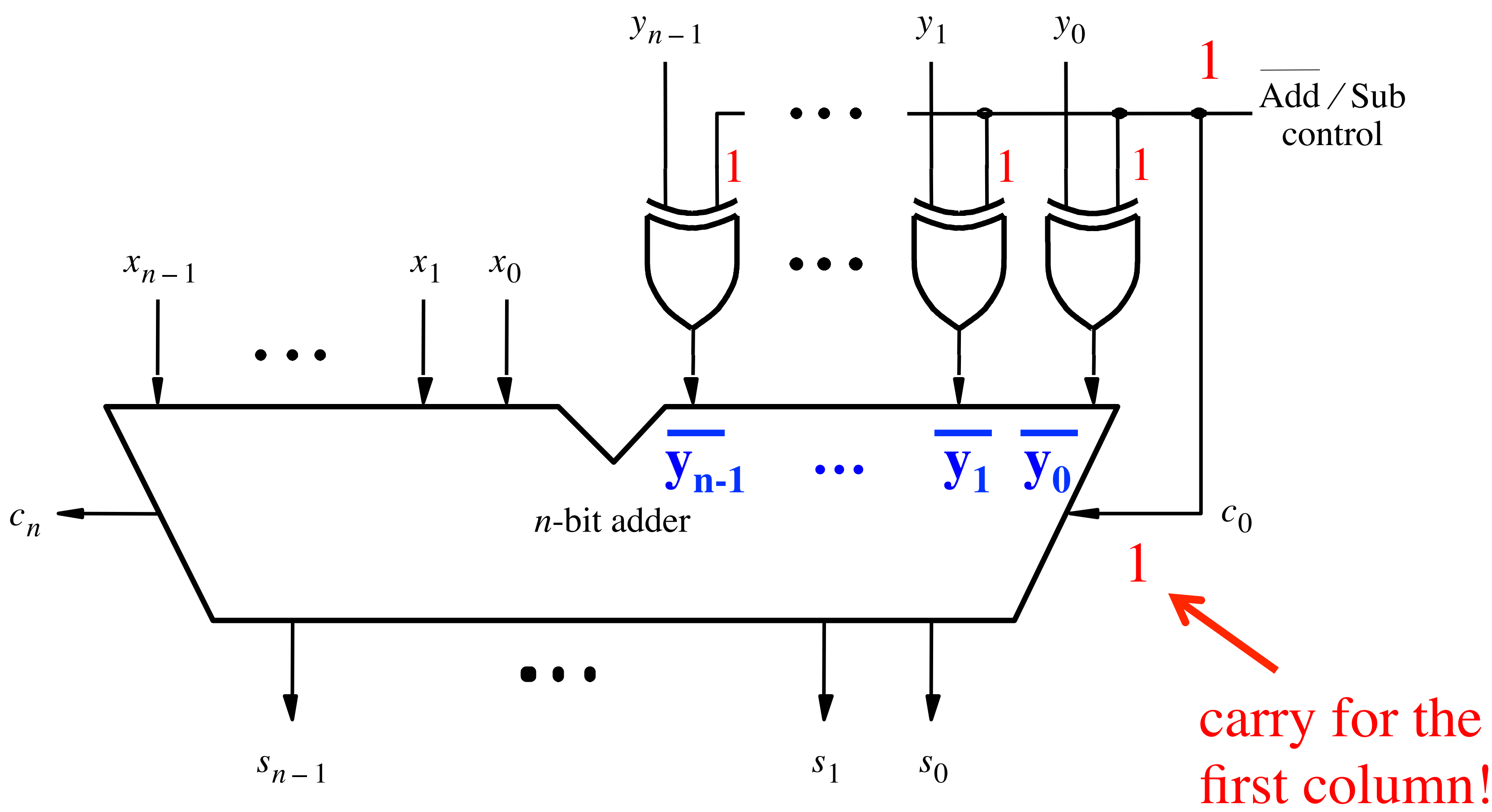
[Figure 3.12 from the textbook]

Subtraction: when control = 1



[Figure 3.12 from the textbook]

Subtraction: when control = 1



[Figure 3.12 from the textbook]

Examples of determination of overflow

$$
\begin{array}{r}
(+7) \\
+\ (+2) \\
\hline
(+9)
\end{array}
\qquad
\begin{array}{r}
0\,1\,1\,1 \\
+\ 0\,0\,1\,0 \\
\hline
1\,0\,0\,1
\end{array}
\qquad\qquad
\begin{array}{r}
(-7) \\
+\ (+2) \\
\hline
(-5)
\end{array}
\qquad
\begin{array}{r}
1\,0\,0\,1 \\
+\ 0\,0\,1\,0 \\
\hline
1\,0\,1\,1
\end{array}
$$

$$
c_4 = 0 \qquad\qquad\qquad c_4 = 0
$$
$$
c_3 = 1 \qquad\qquad\qquad c_3 = 0
$$

$$
\begin{array}{r}
(+7) \\
+\ (-2) \\
\hline
(+5)
\end{array}
\qquad
\begin{array}{r}
0\,1\,1\,1 \\
+\ 1\,1\,1\,0 \\
\hline
1\,0\,1\,0\,1
\end{array}
\qquad\qquad
\begin{array}{r}
(-7) \\
+\ (-2) \\
\hline
(-9)
\end{array}
\qquad
\begin{array}{r}
1\,0\,0\,1 \\
+\ 1\,1\,1\,0 \\
\hline
1\,0\,1\,1\,1
\end{array}
$$

$$
c_4 = 1 \qquad\qquad\qquad c_4 = 1
$$
$$
c_3 = 1 \qquad\qquad\qquad c_3 = 0
$$

[Figure 3.13 from the textbook]

Examples of determination of overflow

$$\begin{array}{rr} (+7) & 0111 \\ +\ (+2) & +\ 0010 \\ \hline (+9) & 1001 \end{array} \qquad \begin{array}{l} c_4 = 0 \\ c_3 = 1 \end{array}$$

$$\begin{array}{rr} (-7) & 1001 \\ +\ (+2) & +\ 0010 \\ \hline (-5) & 1011 \end{array} \qquad \begin{array}{l} c_4 = 0 \\ c_3 = 0 \end{array}$$

Overflow occurs only in these two cases.

$$\begin{array}{rr} (+7) & 0111 \\ +\ (-2) & +\ 1110 \\ \hline (+5) & 10101 \end{array} \qquad \begin{array}{l} c_4 = 1 \\ c_3 = 1 \end{array}$$

$$\begin{array}{rr} (-7) & 1001 \\ +\ (-2) & +\ 1110 \\ \hline (-9) & 10111 \end{array} \qquad \begin{array}{l} c_4 = 1 \\ c_3 = 0 \end{array}$$

[Figure 3.13 from the textbook]

Examples of determination of overflow

$$
\begin{array}{rr}
(+7) & 0111 \\
+\ (+2) & +\ 0010 \\
\hline
(+9) & 1001
\end{array}
\qquad c_4 = 0,\ c_3 = 1
$$

$$
\begin{array}{rr}
(-7) & 1001 \\
+\ (+2) & +\ 0010 \\
\hline
(-5) & 1011
\end{array}
\qquad c_4 = 0,\ c_3 = 0
$$

Overflow occurs only in these two cases.

$$
\begin{array}{rr}
(+7) & 0111 \\
+\ (-2) & +\ 1110 \\
\hline
(+5) & 10101
\end{array}
\qquad c_4 = 1,\ c_3 = 1
$$

$$
\begin{array}{rr}
(-7) & 1001 \\
+\ (-2) & +\ 1110 \\
\hline
(-9) & 10111
\end{array}
\qquad c_4 = 1,\ c_3 = 0
$$

$$\text{Overflow} = c_3\overline{c}_4 + \overline{c}_3 c_4$$

[Figure 3.13 from the textbook]

Examples of determination of overflow

$$\begin{array}{rr} (+7) \\ +\ (+2) \\ \hline (+9) \end{array} \qquad \begin{array}{r} 0\,1\,1\,1 \\ +\ 0\,0\,1\,0 \\ \hline 1\,0\,0\,1 \end{array} \qquad \begin{array}{l} c_4 = 0 \\ c_3 = 1 \end{array}$$

$$\begin{array}{rr} (-7) \\ +\ (+2) \\ \hline (-5) \end{array} \qquad \begin{array}{r} 1\,0\,0\,1 \\ +\ 0\,0\,1\,0 \\ \hline 1\,0\,1\,1 \end{array} \qquad \begin{array}{l} c_4 = 0 \\ c_3 = 0 \end{array}$$

Overflow occurs only in these two cases.

$$\begin{array}{rr} (+7) \\ +\ (-2) \\ \hline (+5) \end{array} \qquad \begin{array}{r} 0\,1\,1\,1 \\ +\ 1\,1\,1\,0 \\ \hline 1\,0\,1\,0\,1 \end{array} \qquad \begin{array}{l} c_4 = 1 \\ c_3 = 1 \end{array}$$

$$\begin{array}{rr} (-7) \\ +\ (-2) \\ \hline (-9) \end{array} \qquad \begin{array}{r} 1\,0\,0\,1 \\ +\ 1\,1\,1\,0 \\ \hline 1\,0\,1\,1\,1 \end{array} \qquad \begin{array}{l} c_4 = 1 \\ c_3 = 0 \end{array}$$

$$\text{Overflow} = \underbrace{c_3\overline{c}_4 + \overline{c}_3 c_4}_{\text{XOR}}$$

[Figure 3.13 from the textbook]

Calculating overflow for 4-bit numbers with only three significant bits

$$\begin{aligned} \text{Overflow} &= c_3\overline{c}_4 + \overline{c}_3 c_4 \\ &= c_3 \oplus c_4 \end{aligned}$$

Calculating overflow for n-bit numbers with only n-1 significant bits

$$\text{Overflow} = c_{n-1} \oplus c_n$$

Another way to look at the overflow issue

$$
\begin{array}{r}
X = x_3 x_2 x_1 x_0 \\
Y = y_3 y_2 y_1 y_0 \\
\hline
S = s_3 s_2 s_1 s_0
\end{array}
$$

Another way to look at the overflow issue

$$
\begin{array}{r}
X = x_3 x_2 x_1 x_0 \\
Y = y_3 y_2 y_1 y_0 \\
\hline
S = s_3 s_2 s_1 s_0
\end{array}
$$

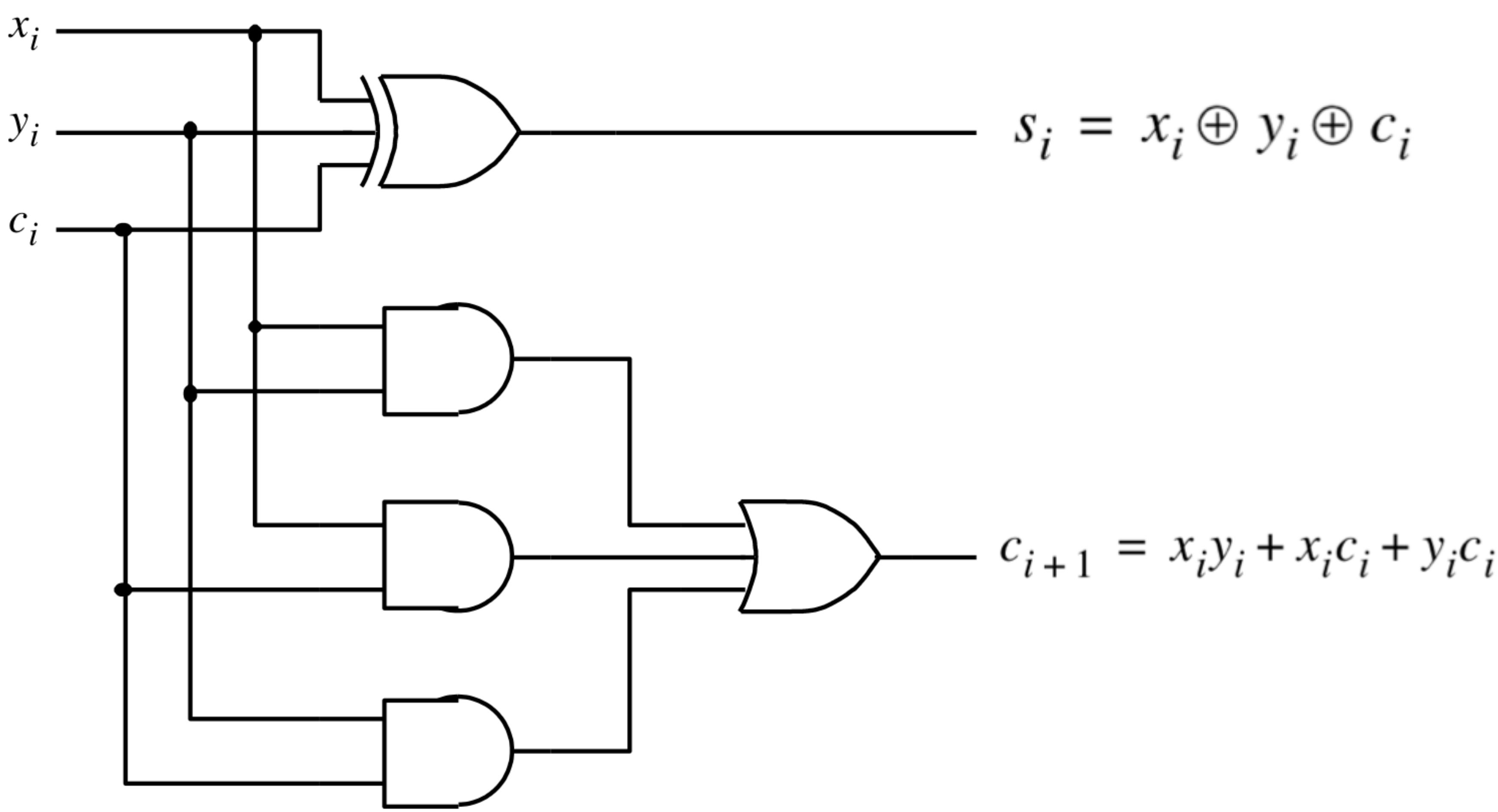
If both numbers that we are adding have the same sign but the sum does not, then we have an overflow.

$$\text{Overflow} = x_3 y_3 \bar{s}_3 + \bar{x}_3 \bar{y}_3 s_3$$

Can we perform addition even faster?

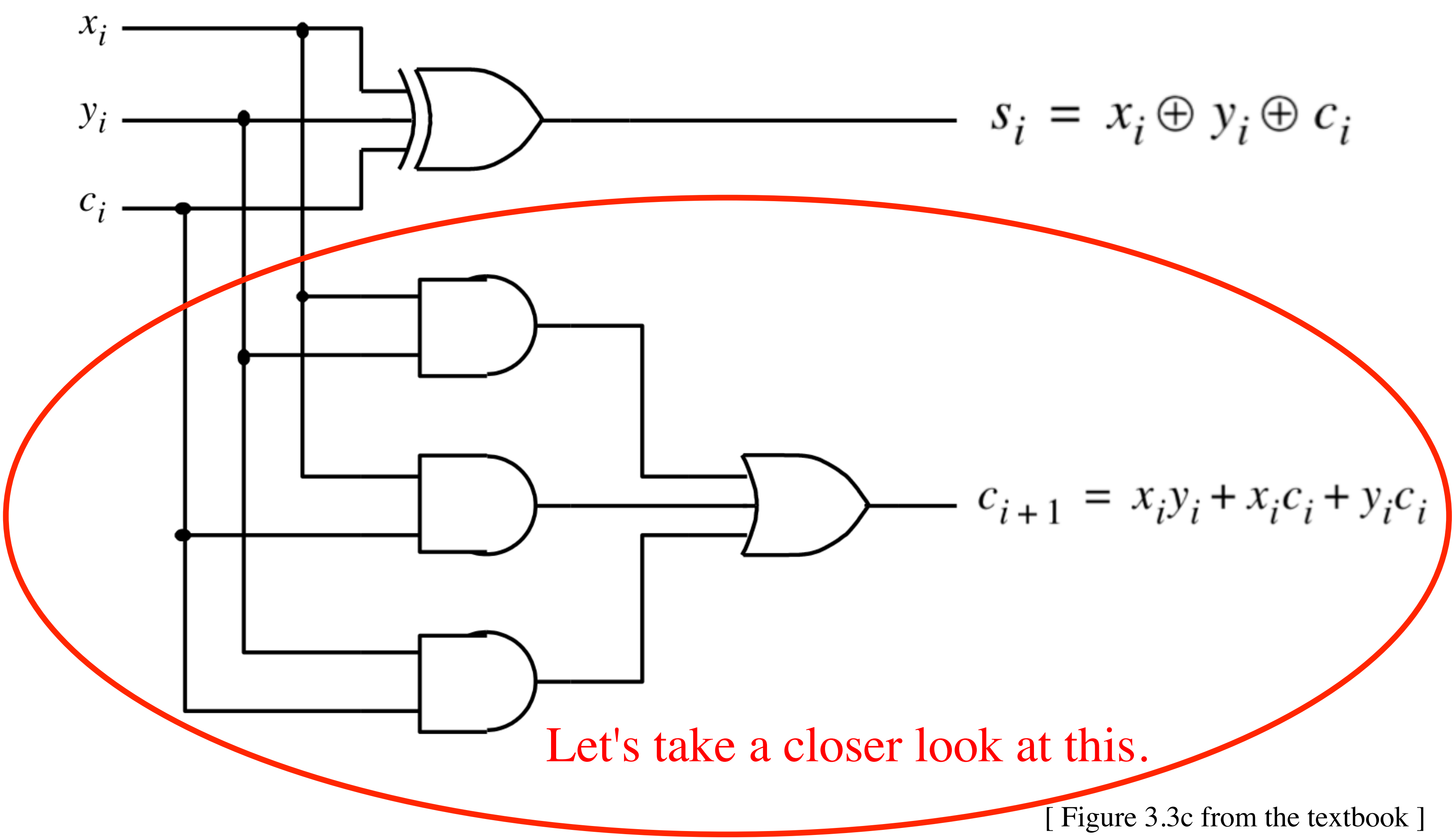
The goal is to evaluate very fast if the carry from the previous stage will be equal to 0 or 1.

The Full-Adder Circuit



[Figure 3.3c from the textbook]

The Full-Adder Circuit



[Figure 3.3c from the textbook]

Decomposing the Carry Expression

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

Decomposing the Carry Expression

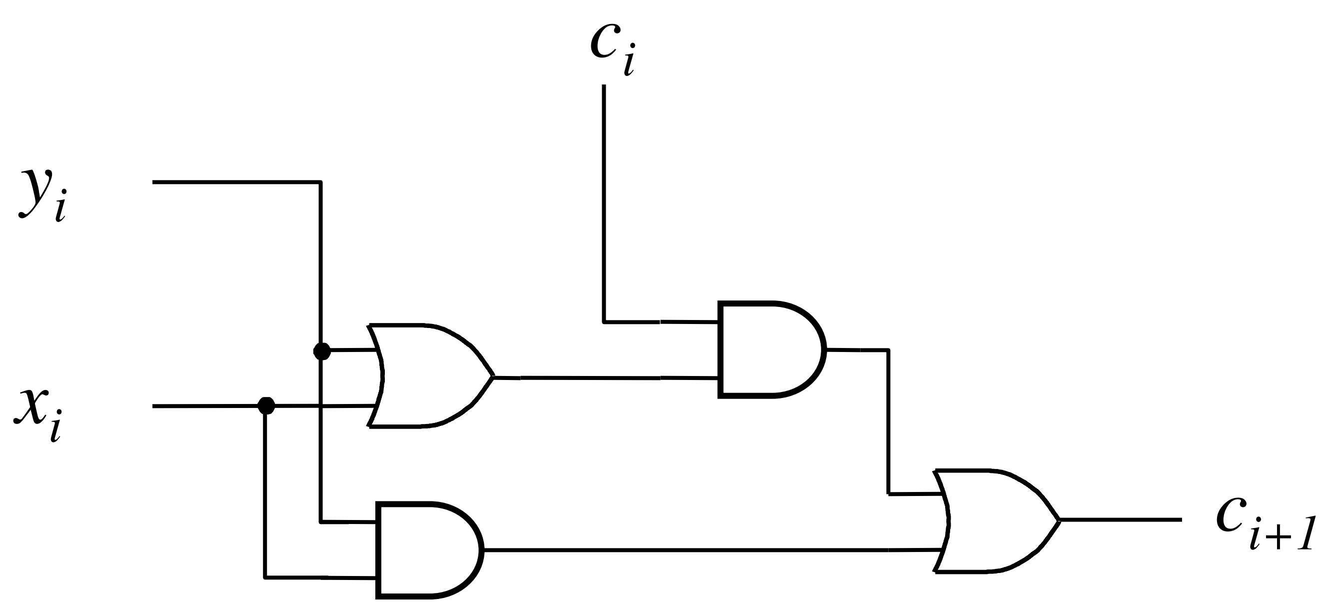
$$c_{i+1} = x_i y_i + x_i c_i + y_i c_i$$

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

Decomposing the Carry Expression

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

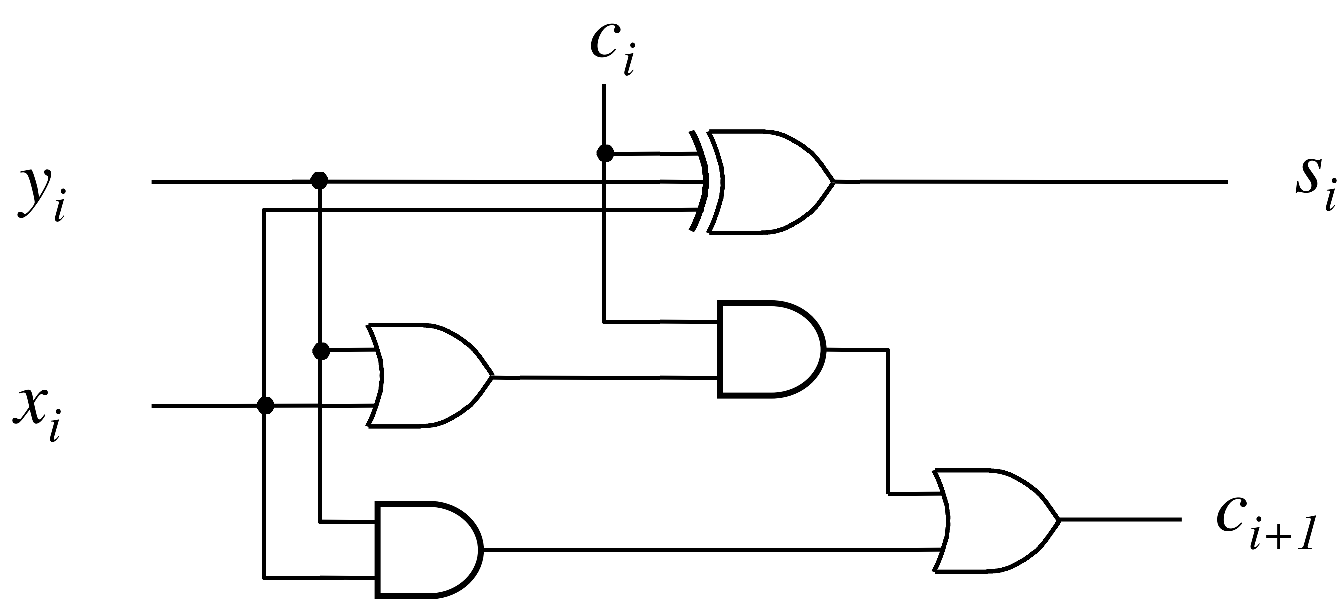
$$c_{i+1} = x_i\, y_i + (x_i + y_i)c_i$$



Another Way to Draw the Full-Adder Circuit

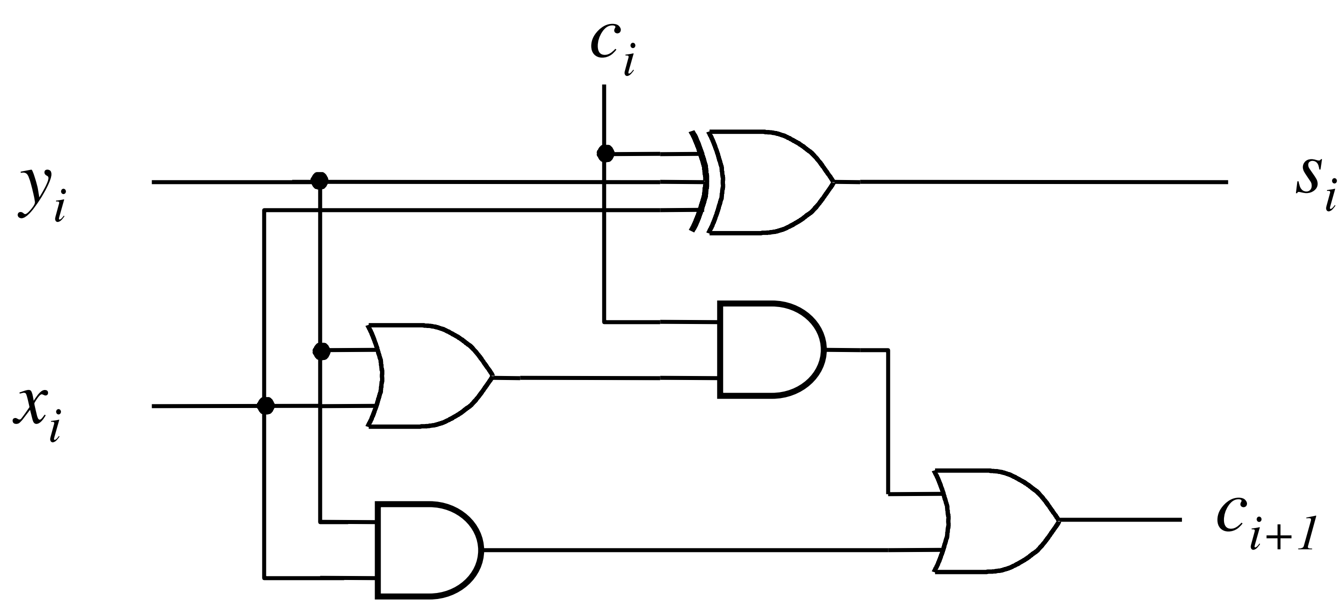
$$c_{i+1} = x_i y_i + x_i c_i + y_i c_i$$

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



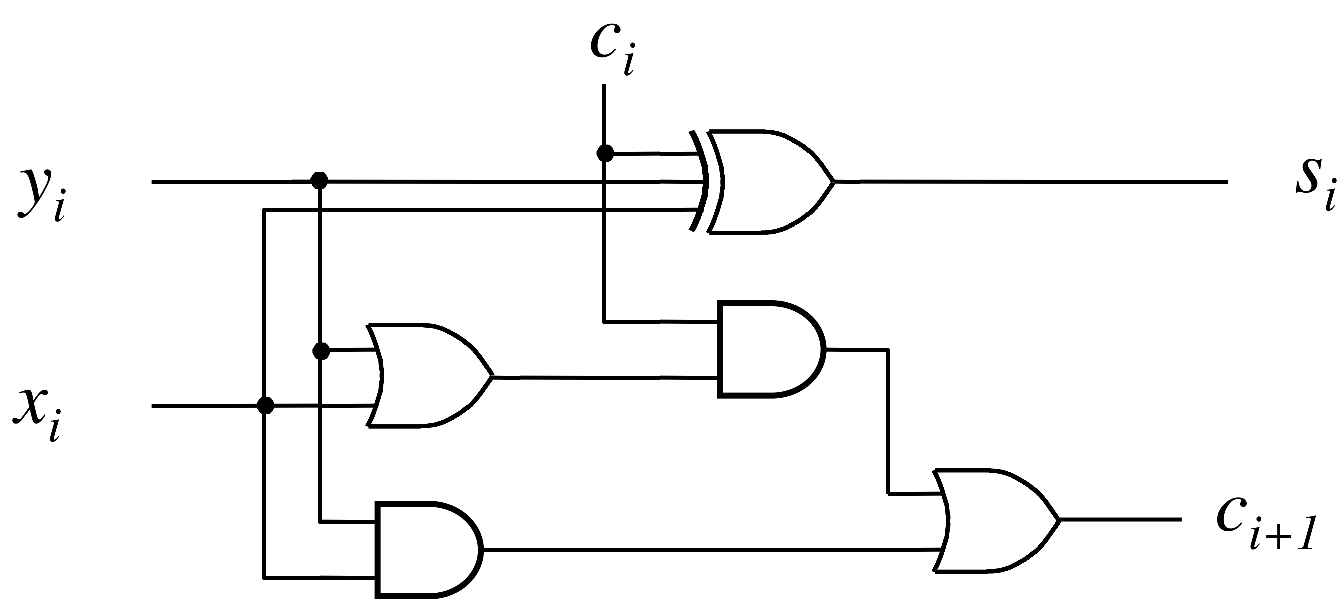
Another Way to Draw the Full-Adder Circuit

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



Another Way to Draw the Full-Adder Circuit

$$c_{i+1} = \underbrace{x_i\, y_i}_{g_i} + \underbrace{(x_i + y_i)}_{p_i} c_i$$

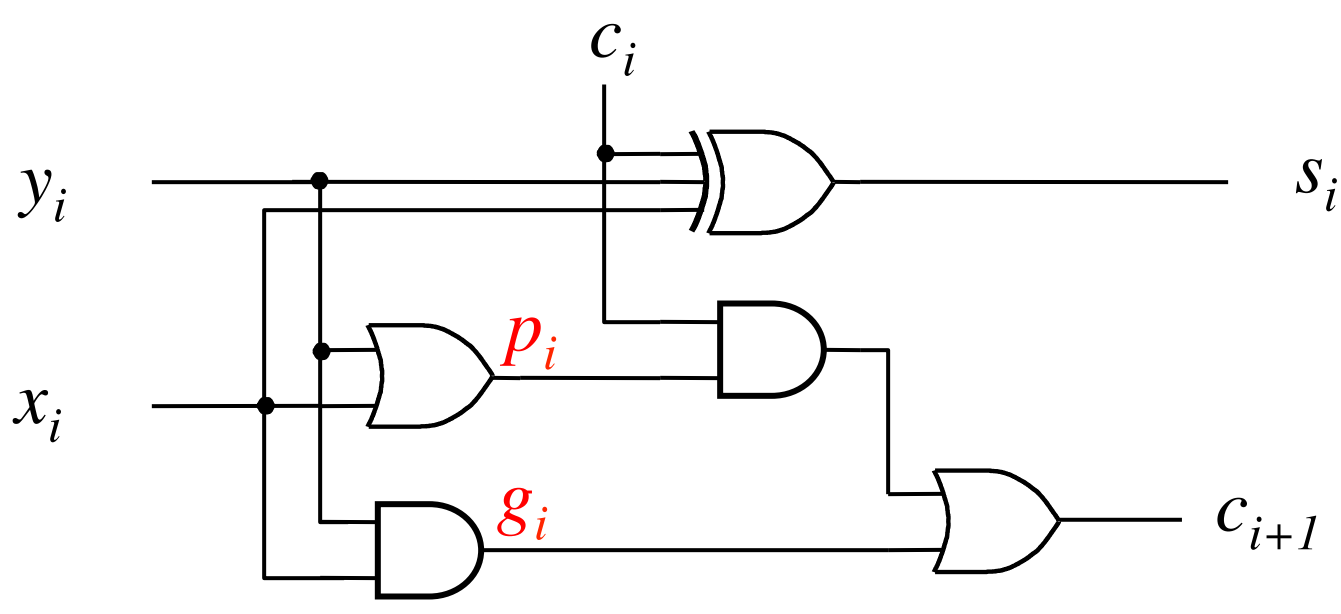


Another Way to Draw the Full-Adder Circuit

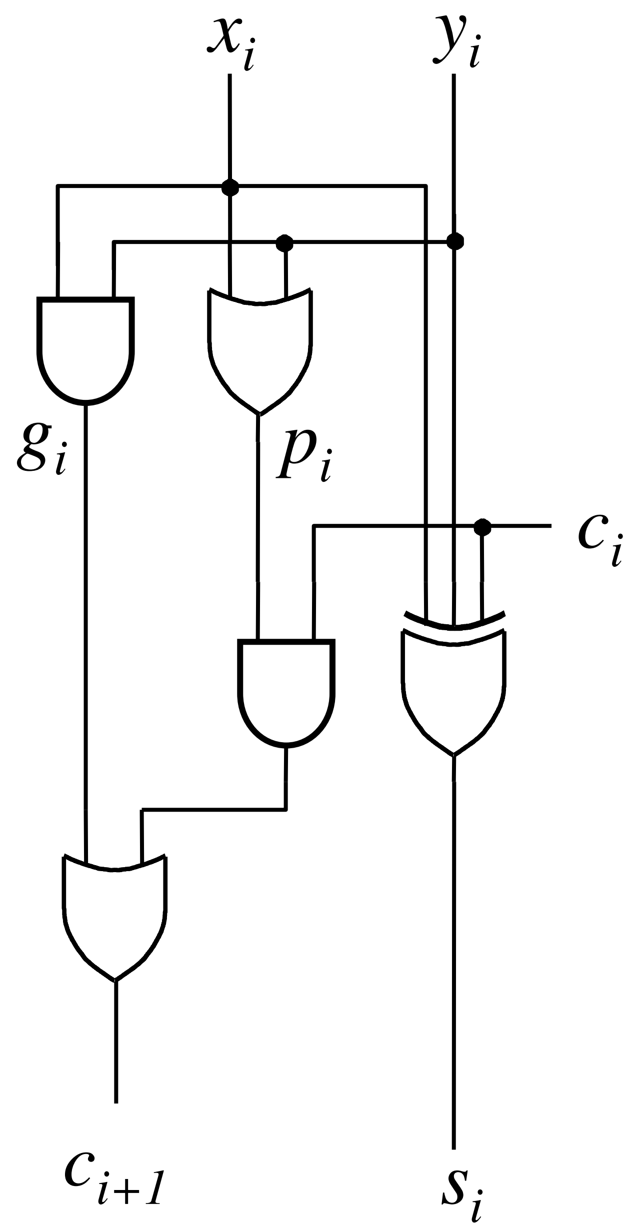
g - generate

p - propagate

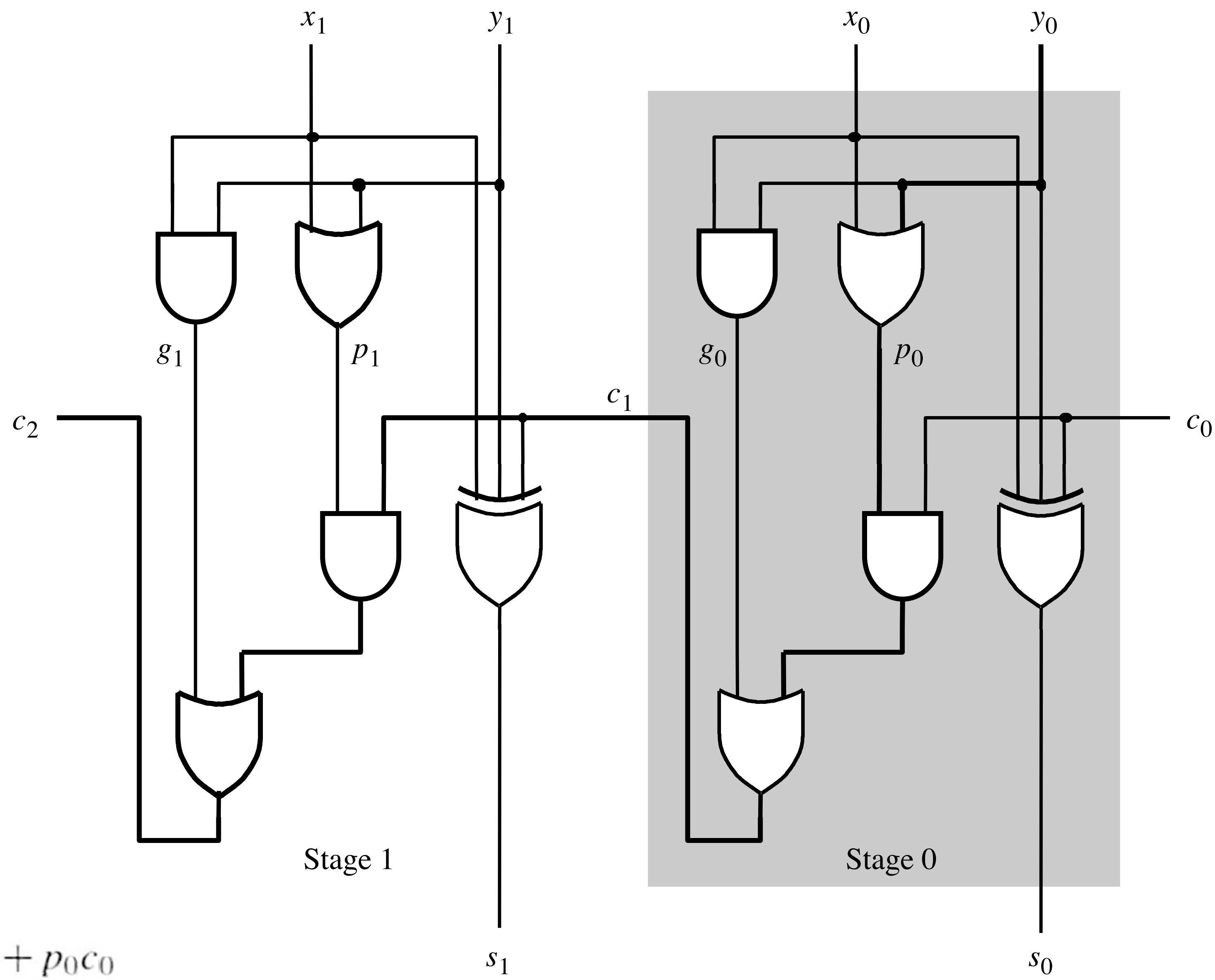
$$c_{i+1} = \underbrace{x_i y_i}_{g_i} + \underbrace{(x_i + y_i)}_{p_i} c_i$$



Yet Another Way to Draw It (Just Rotate It)



Now we can Build a Ripple-Carry Adder

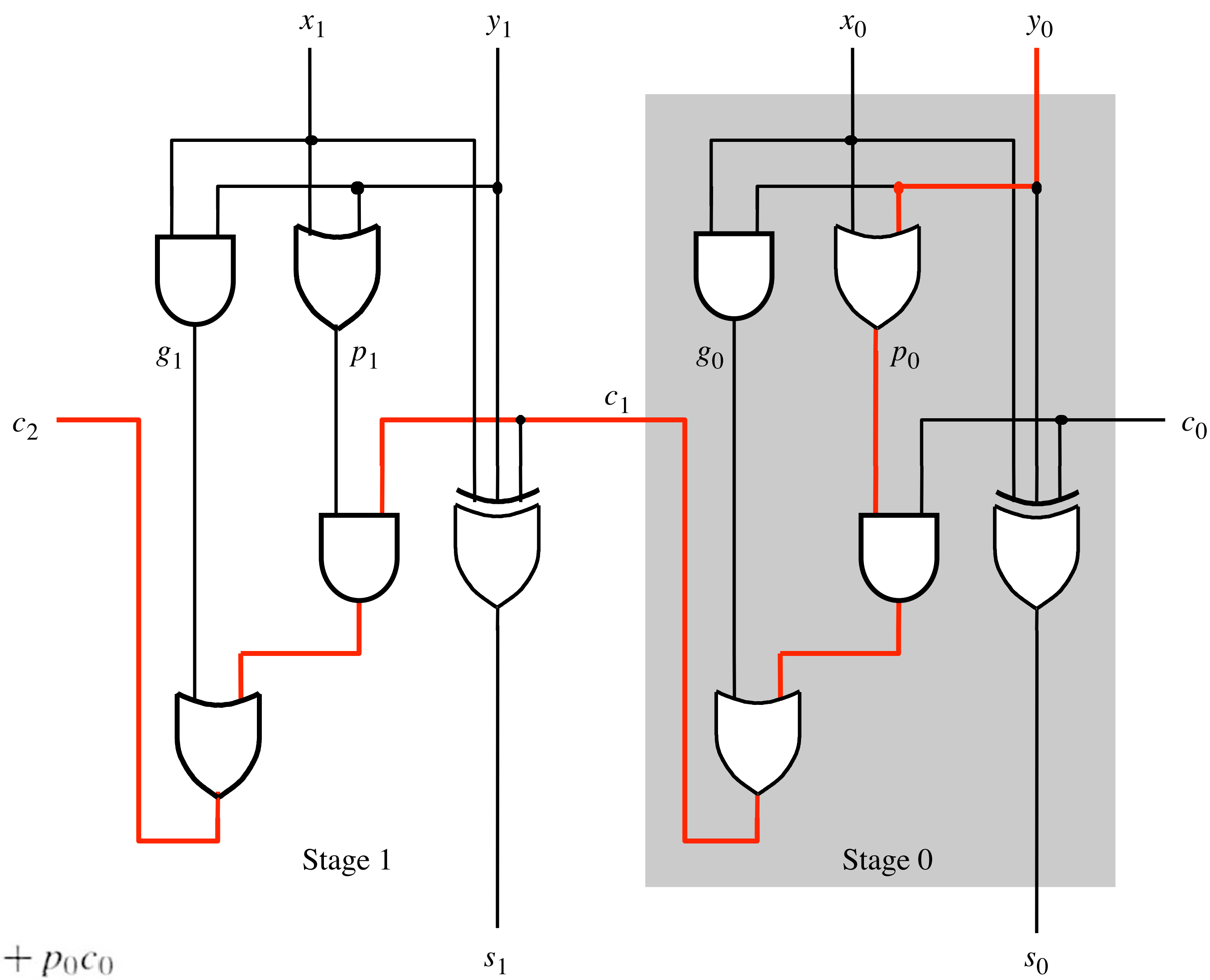


$$c_1 = g_0 + p_0 c_0$$

$$c_2 = g_1 + p_1 g_0 + p_1 p_0 c_0$$

[Figure 3.14 from the textbook]

Now we can Build a Ripple-Carry Adder

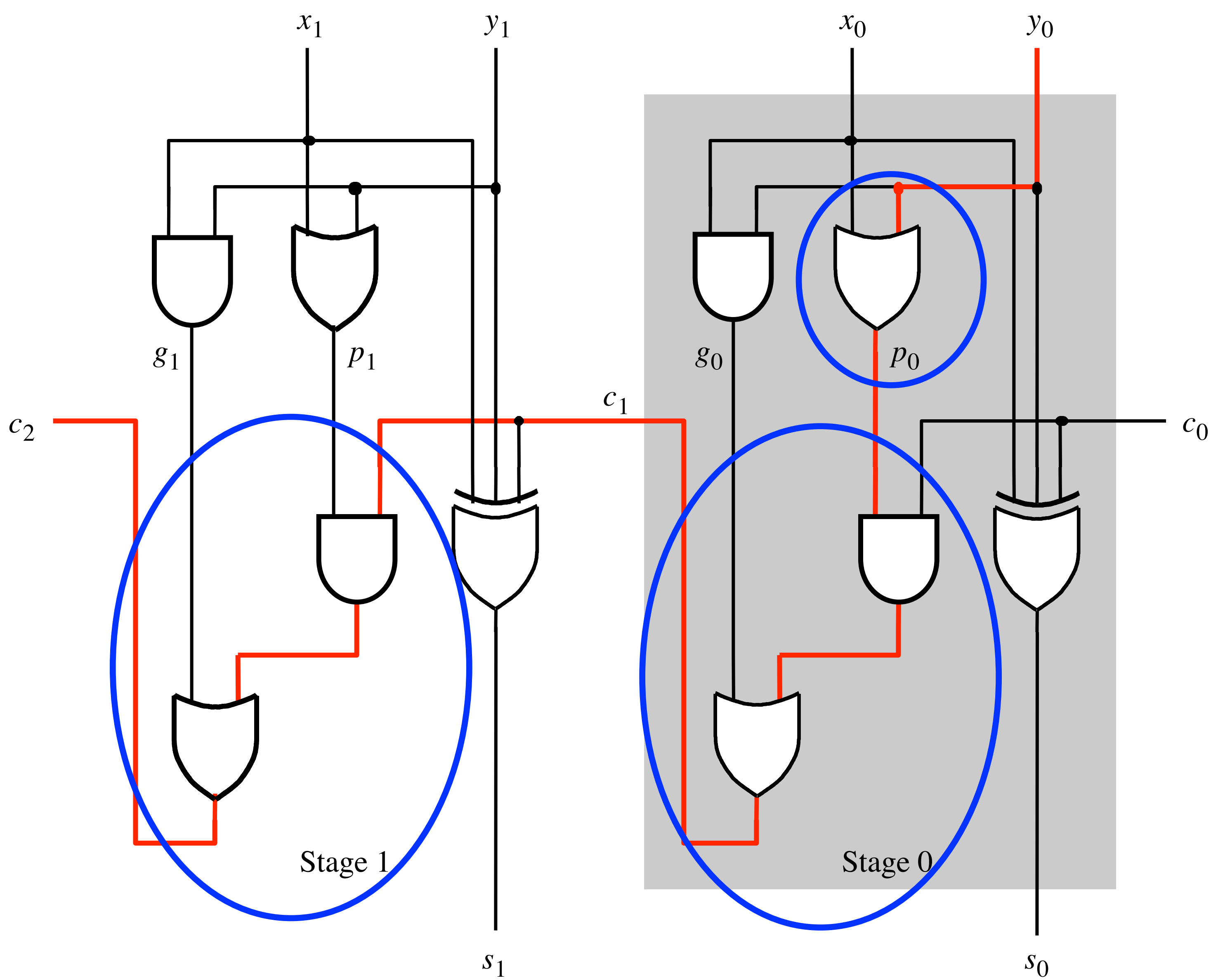


$c_1 = g_0 + p_0 c_0$

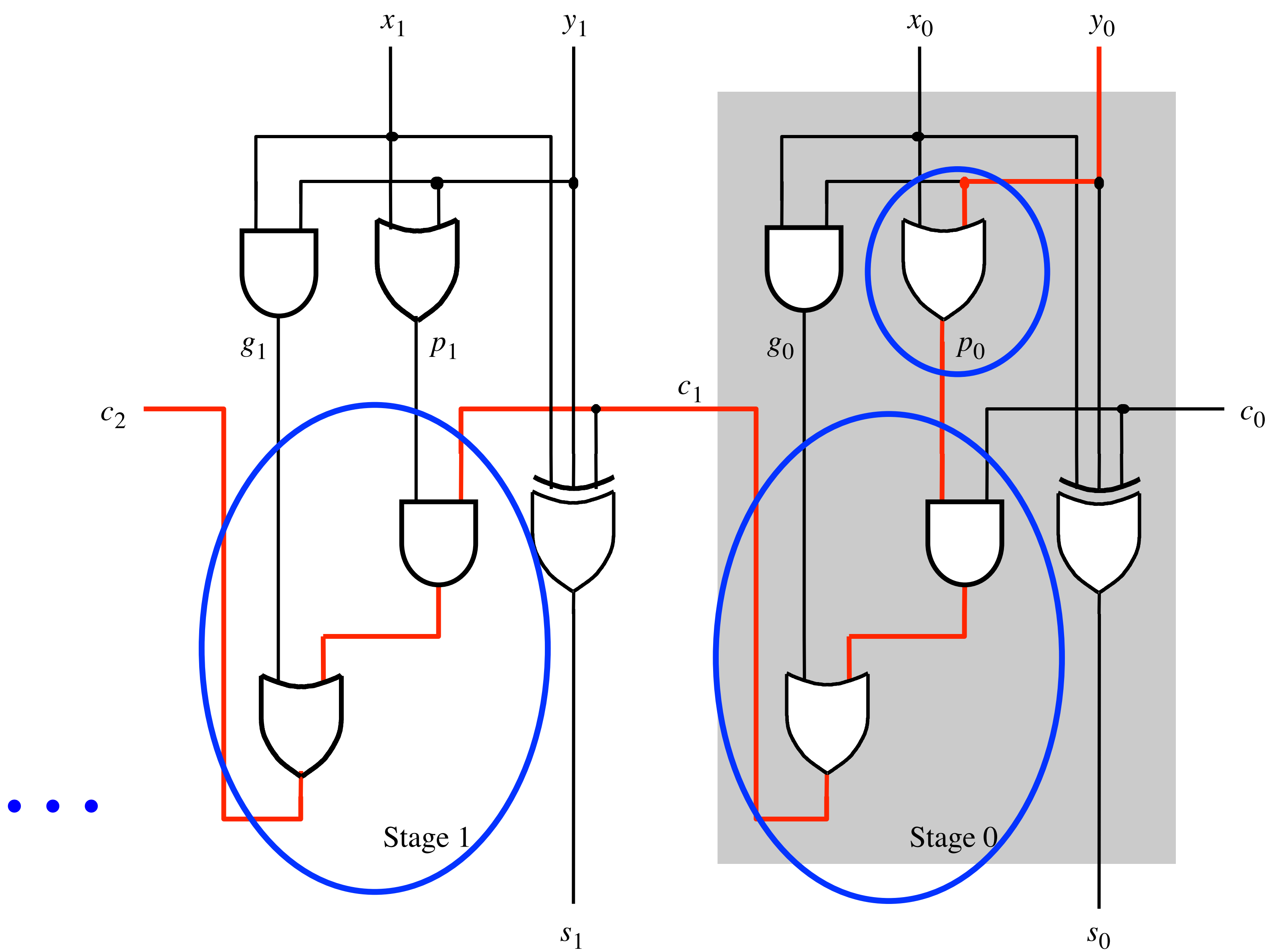
$c_2 = g_1 + p_1 g_0 + p_1 p_0 c_0$

[Figure 3.14 from the textbook]

The delay is 5 gates (1+2+2)



n-bit ripple-carry adder: 2n+1 gate delays



Decomposing the Carry Expression

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

$$c_{i+1} = \underbrace{x_i y_i}_{g_i} + \underbrace{(x_i + y_i)}_{p_i} c_i$$

$$c_{i+1} = g_i + p_i c_i$$

$$c_{i+1} = g_i + p_i (g_{i-1} + p_{i-1} c_{i-1})$$

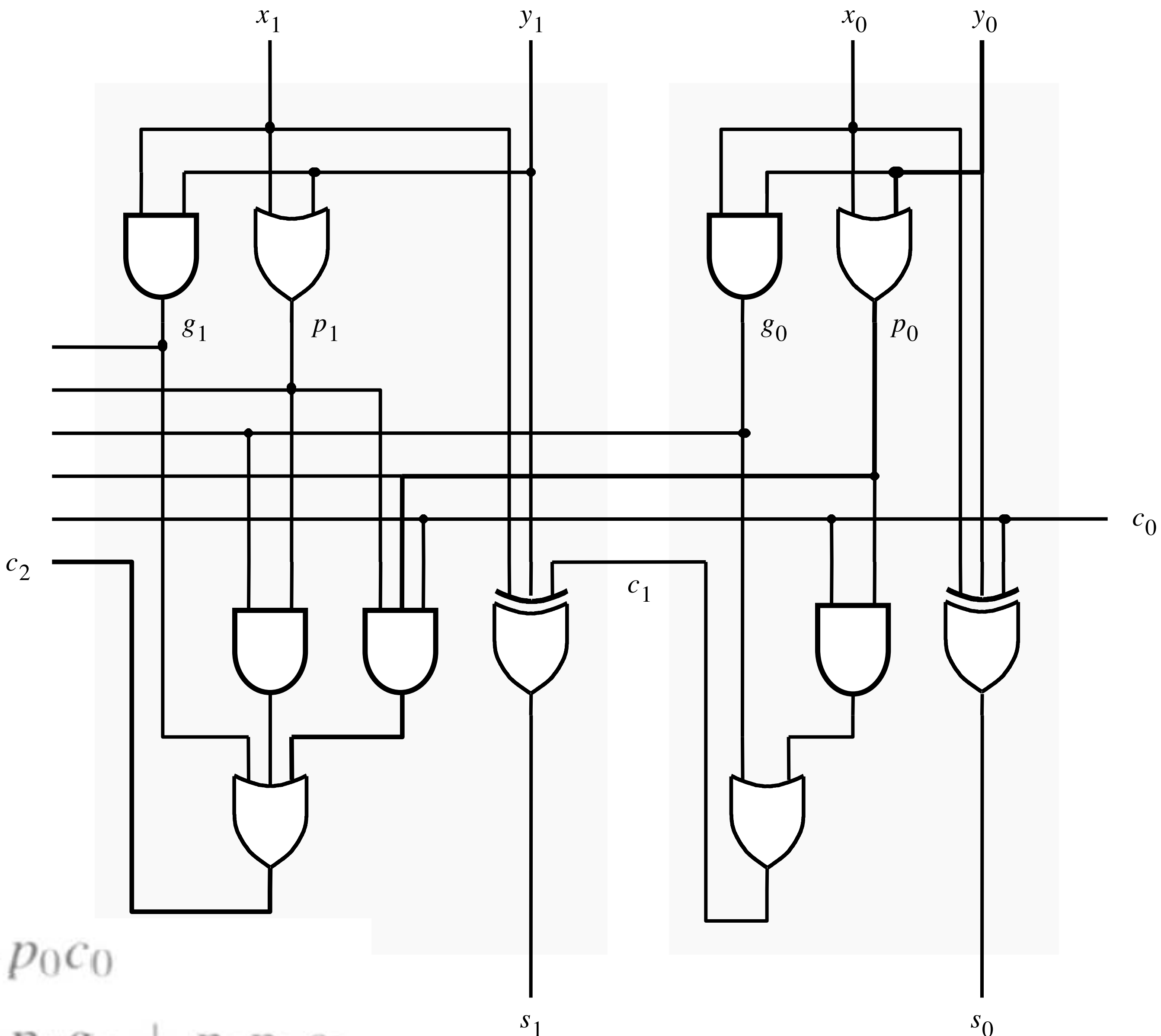
$$= g_i + p_i g_{i-1} + p_i p_{i-1} c_{i-1}$$

Carry for the first two stages

$$c_1 = g_0 + p_0 c_0$$

$$c_2 = g_1 + p_1 g_0 + p_1 p_0 c_0$$

The first two stages of a carry-lookahead adder

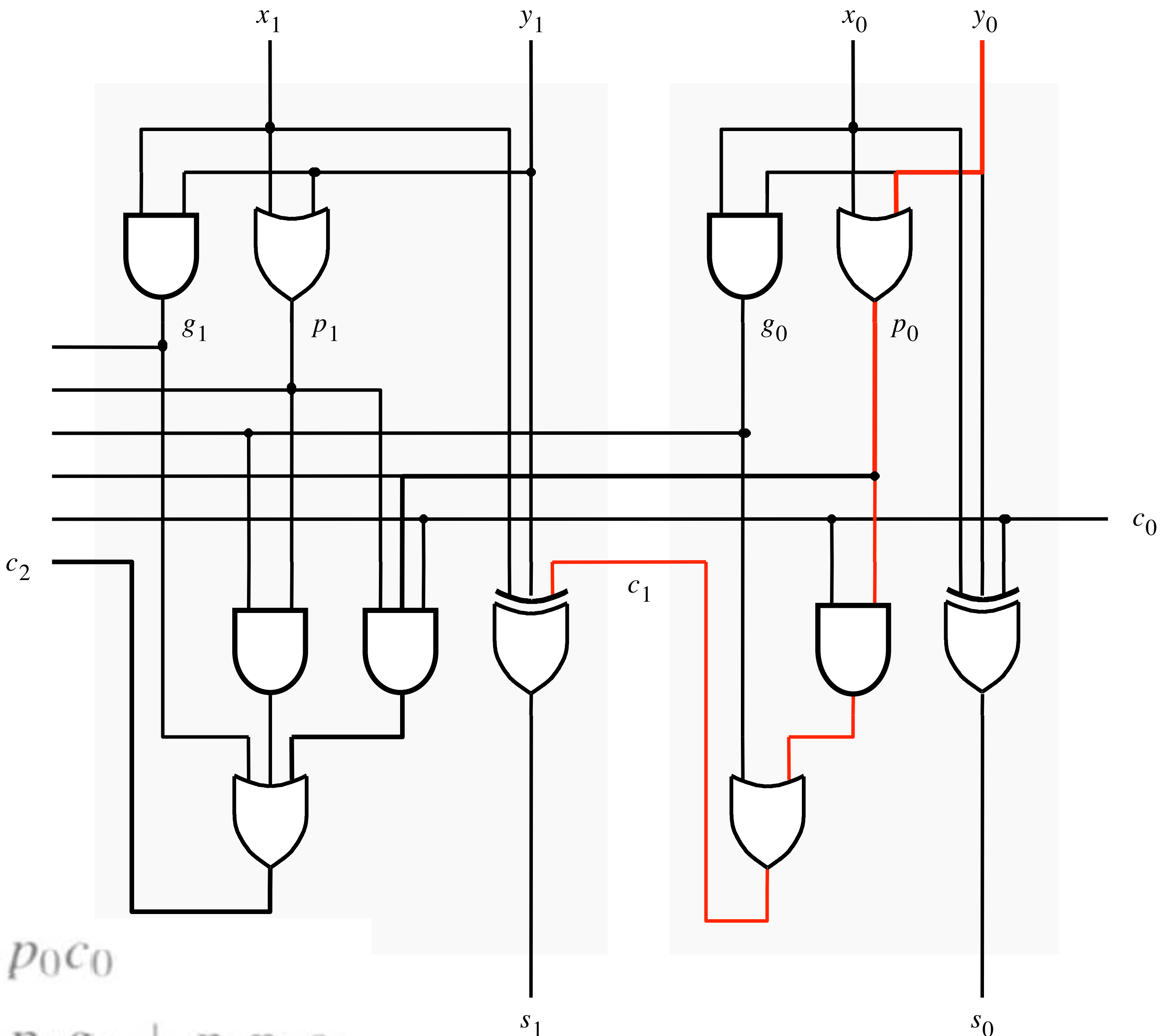


$$c_1 = g_0 + p_0c_0$$

$$c_2 = g_1 + p_1g_0 + p_1p_0c_0$$

[Figure 3.15 from the textbook]

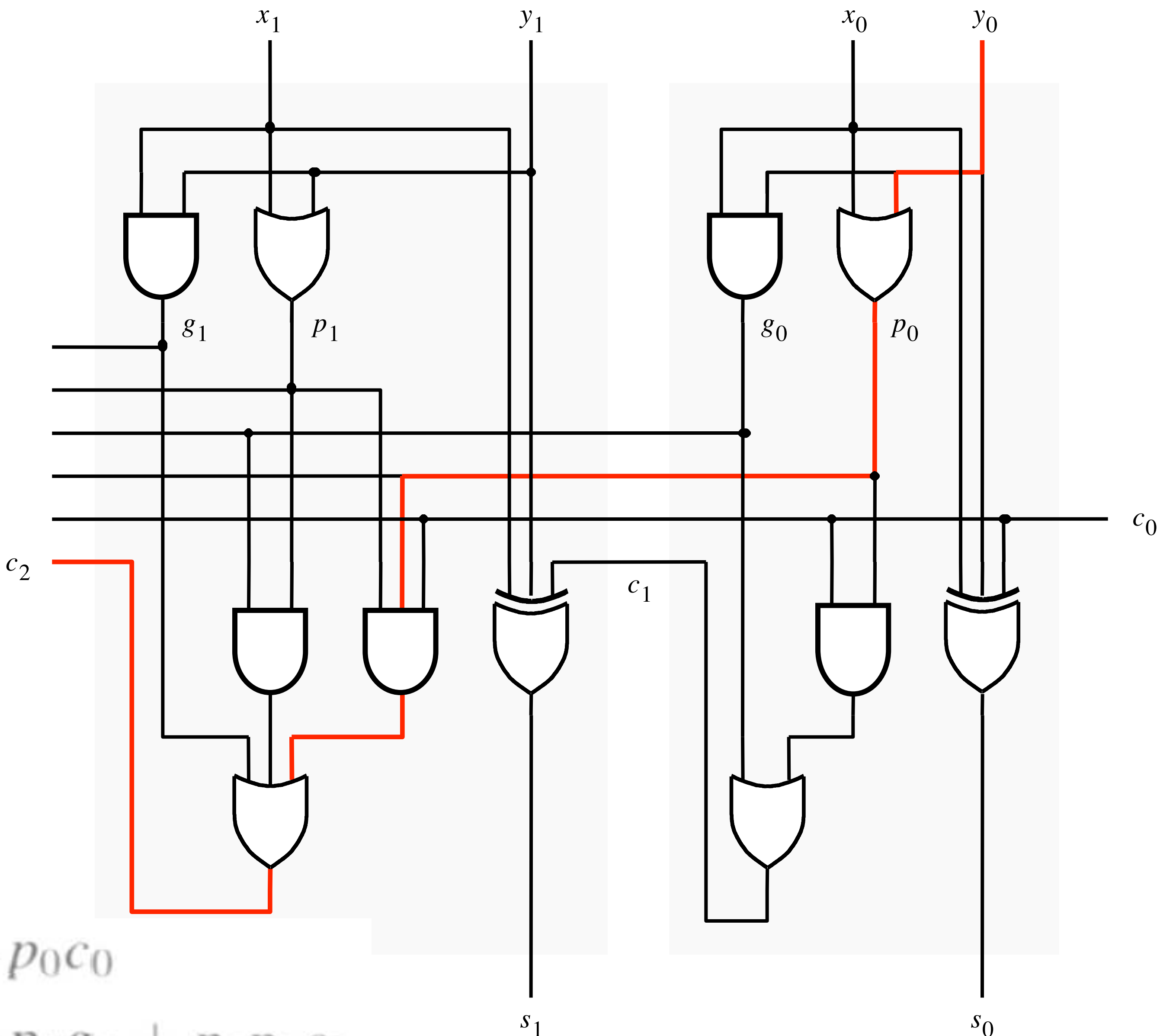
It takes 3 gate delays to generate c_1



$$c_1 = g_0 + p_0 c_0$$

$$c_2 = g_1 + p_1 g_0 + p_1 p_0 c_0$$

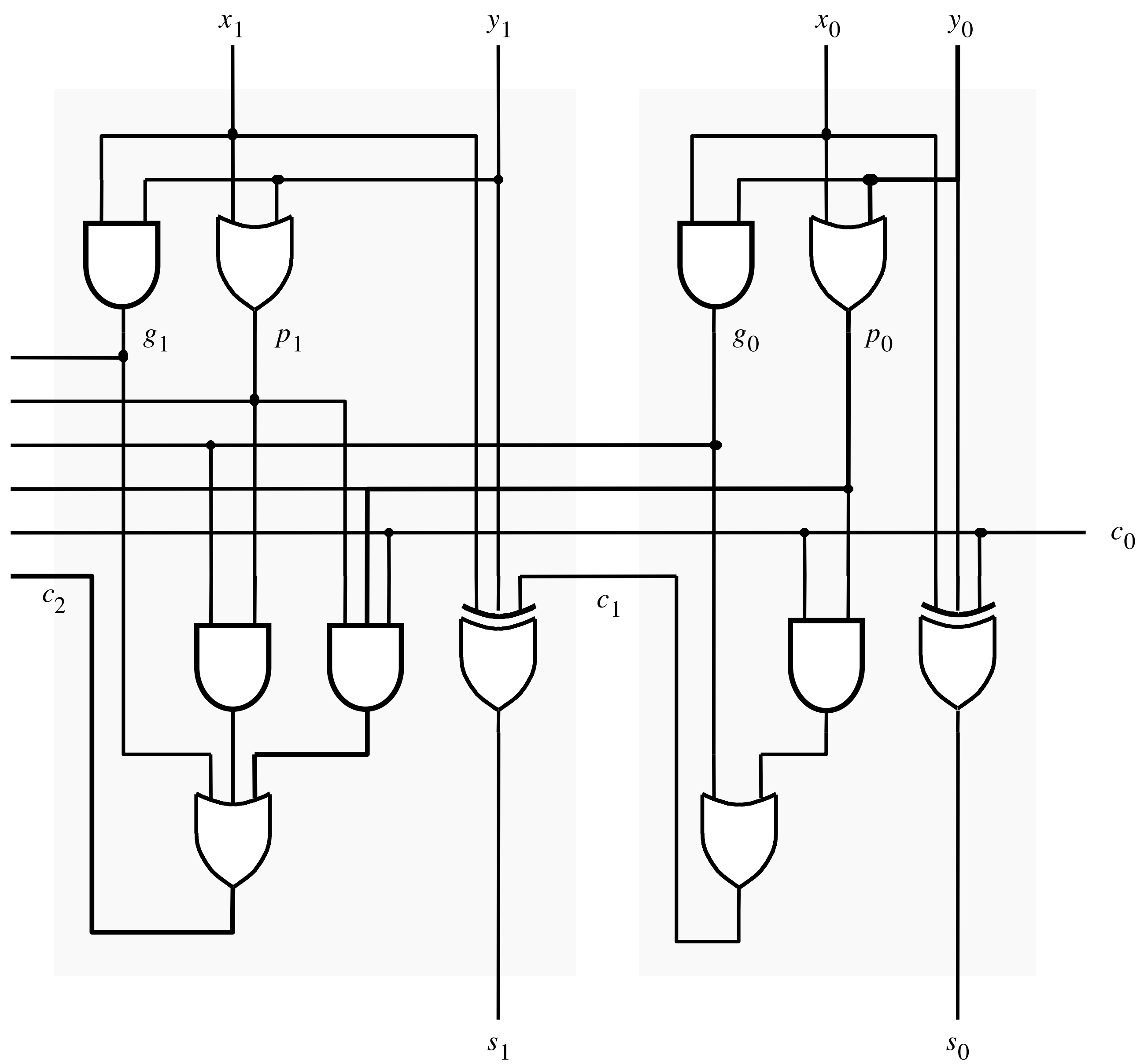
It takes 3 gate delays to generate c_2



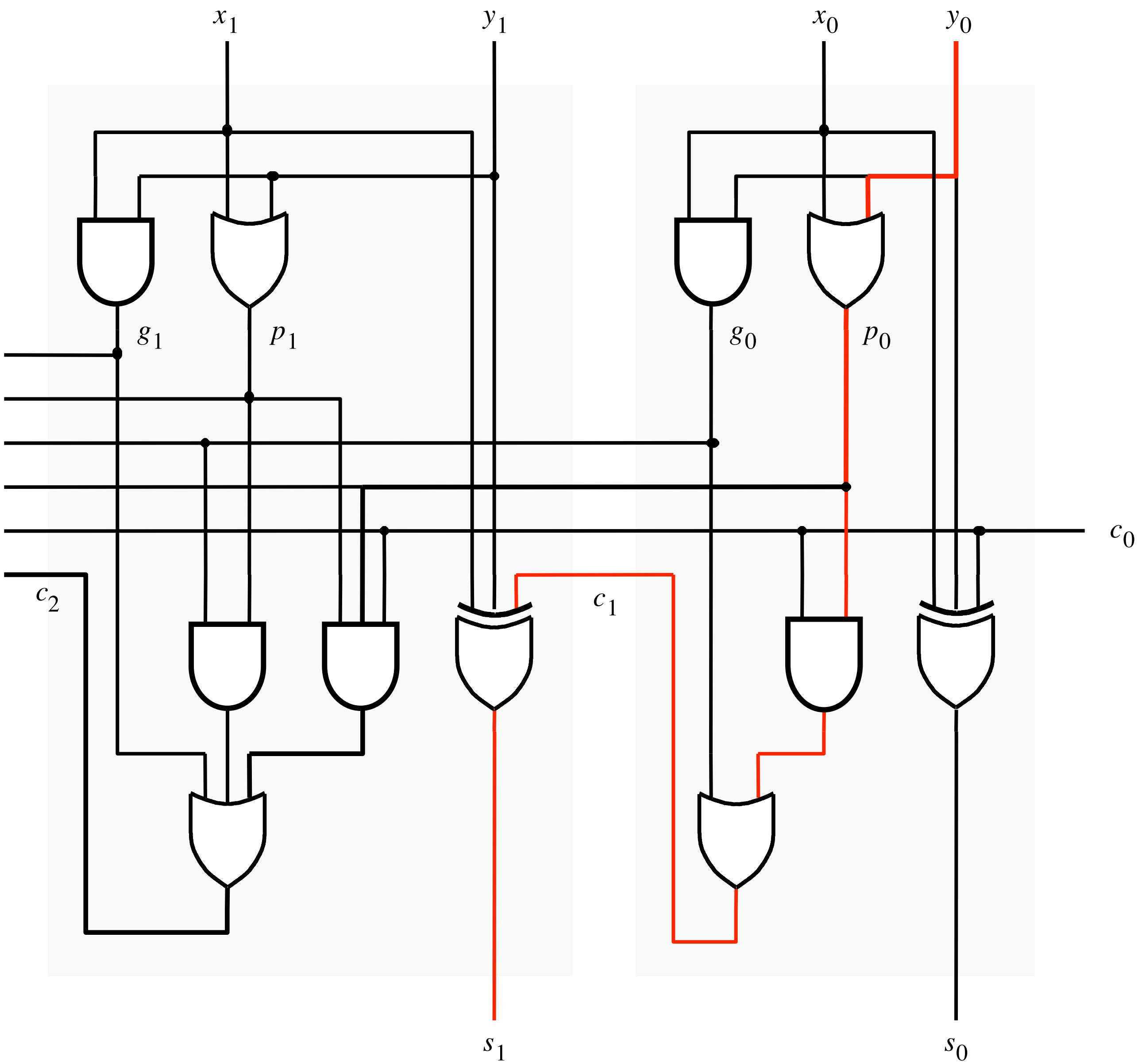
$$c_1 = g_0 + p_0 c_0$$

$$c_2 = g_1 + p_1 g_0 + p_1 p_0 c_0$$

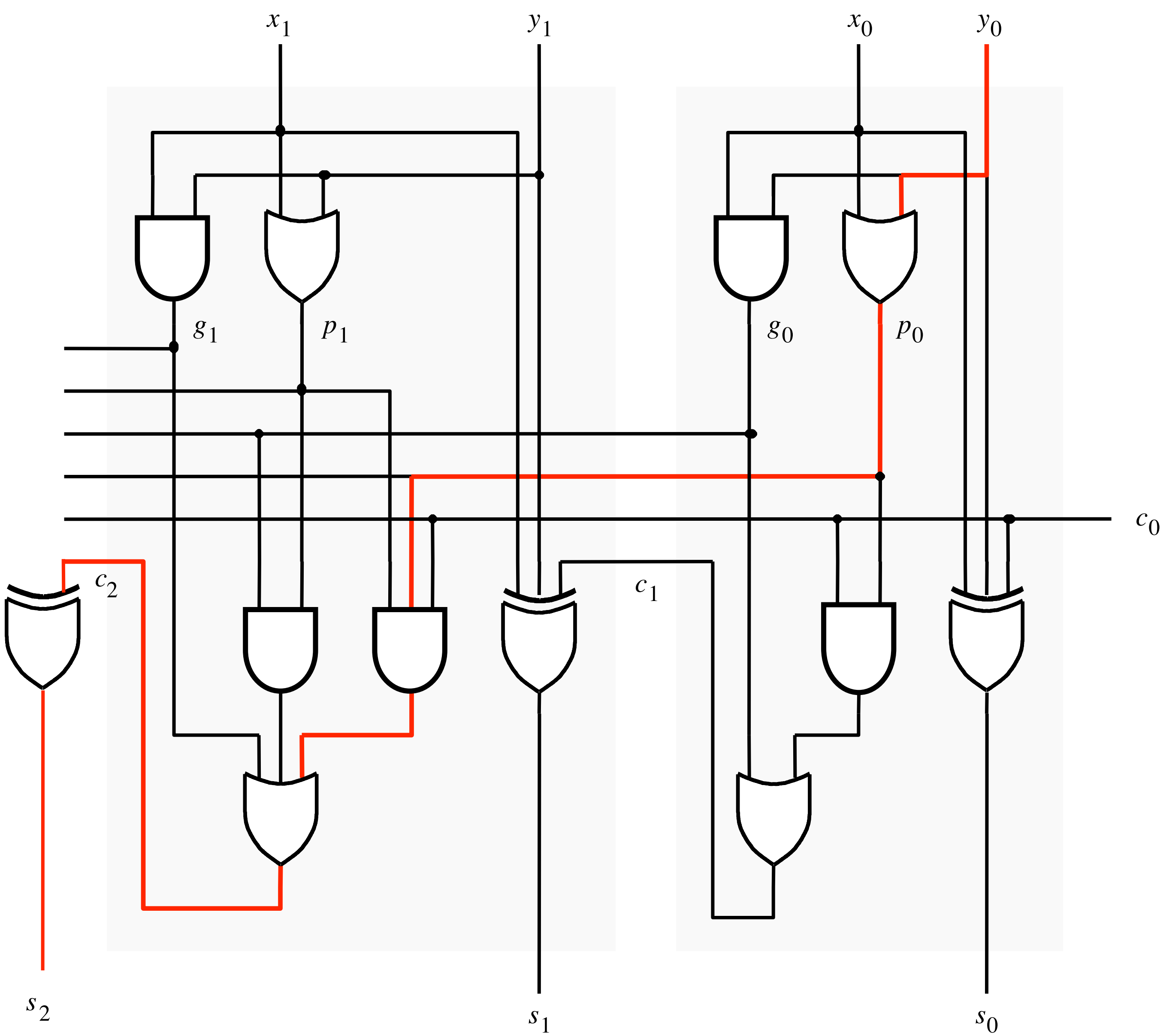
The first two stages of a carry-lookahead adder



It takes 4 gate delays to generate s_1



It takes 4 gate delays to generate s_2



N-bit Carry-Lookahead Adder

- It takes 3 gate delays to generate all carry signals
- It takes 1 more gate delay to generate all sum bits
- Thus, the total delay through an n-bit carry-lookahead adder is only 4 gate delays!

Expanding the Carry Expression

$$c_{i+1} = g_i + p_i c_i$$

$$c_1 = g_0 + p_0 c_0$$

$$c_2 = g_1 + p_1 g_0 + p_1 p_0 c_0$$

$$c_3 = g_2 + p_2 g_1 + p_2 p_1 g_0 + p_2 p_1 p_0 c_0$$

$$\cdots$$

$$\begin{aligned} c_8 = {} & g_7 + p_7 g_6 + p_7 p_6 g_5 + p_7 p_6 p_5 g_4 \\ & + p_7 p_6 p_5 p_4 g_3 + p_7 p_6 p_5 p_4 p_3 g_2 \\ & + p_7 p_6 p_5 p_4 p_3 p_2 g_1 + p_7 p_6 p_5 p_4 p_3 p_2 p_1 g_0 \\ & + p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0 c_0 \end{aligned}$$

Expanding the Carry Expression

$$c_{i+1} = g_i + p_i c_i$$

$$c_1 = g_0 + p_0 c_0$$

$$c_2 = g_1 + p_1 g_0 + p_1 p_0 c_0$$

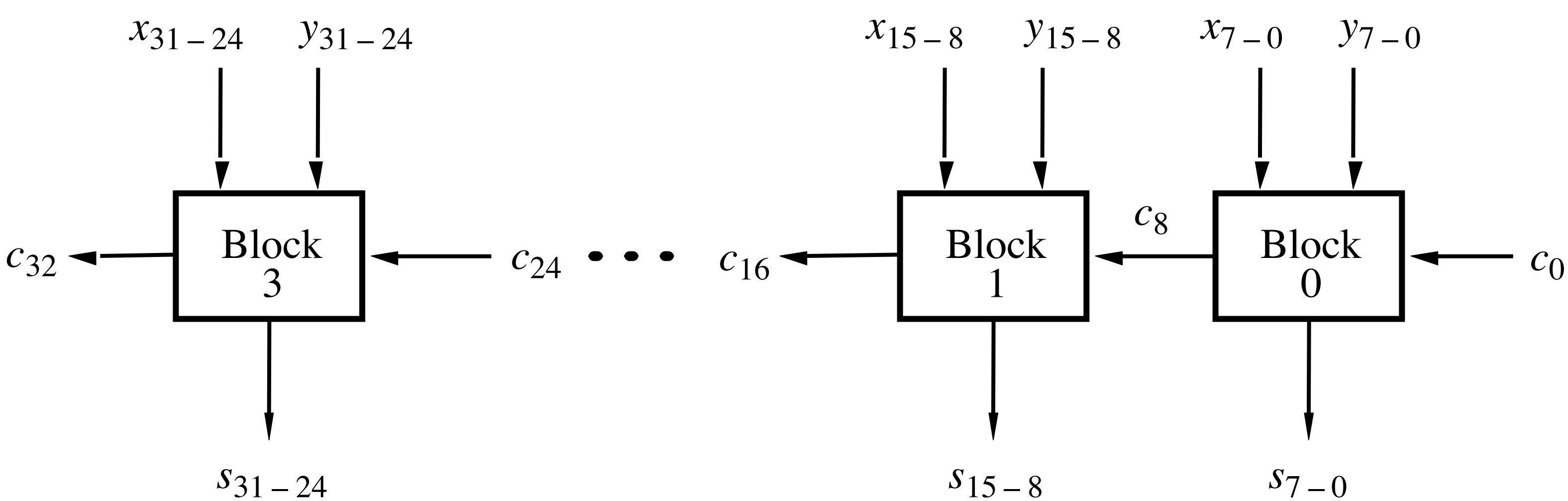
$$c_3 = g_2 + p_2 g_1 + p_2 p_1 g_0 + p_2 p_1 p_0 c_0$$

$$\ldots$$

$$\begin{aligned} c_8 = \ & g_7 + p_7 g_6 + p_7 p_6 g_5 + p_7 p_6 p_5 g_4 \\ & + p_7 p_6 p_5 p_4 g_3 + p_7 p_6 p_5 p_4 p_3 g_2 \\ & + p_7 p_6 p_5 p_4 p_3 p_2 g_1 + p_7 p_6 p_5 p_4 p_3 p_2 p_1 g_0 \\ & + p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0 c_0 \end{aligned}$$

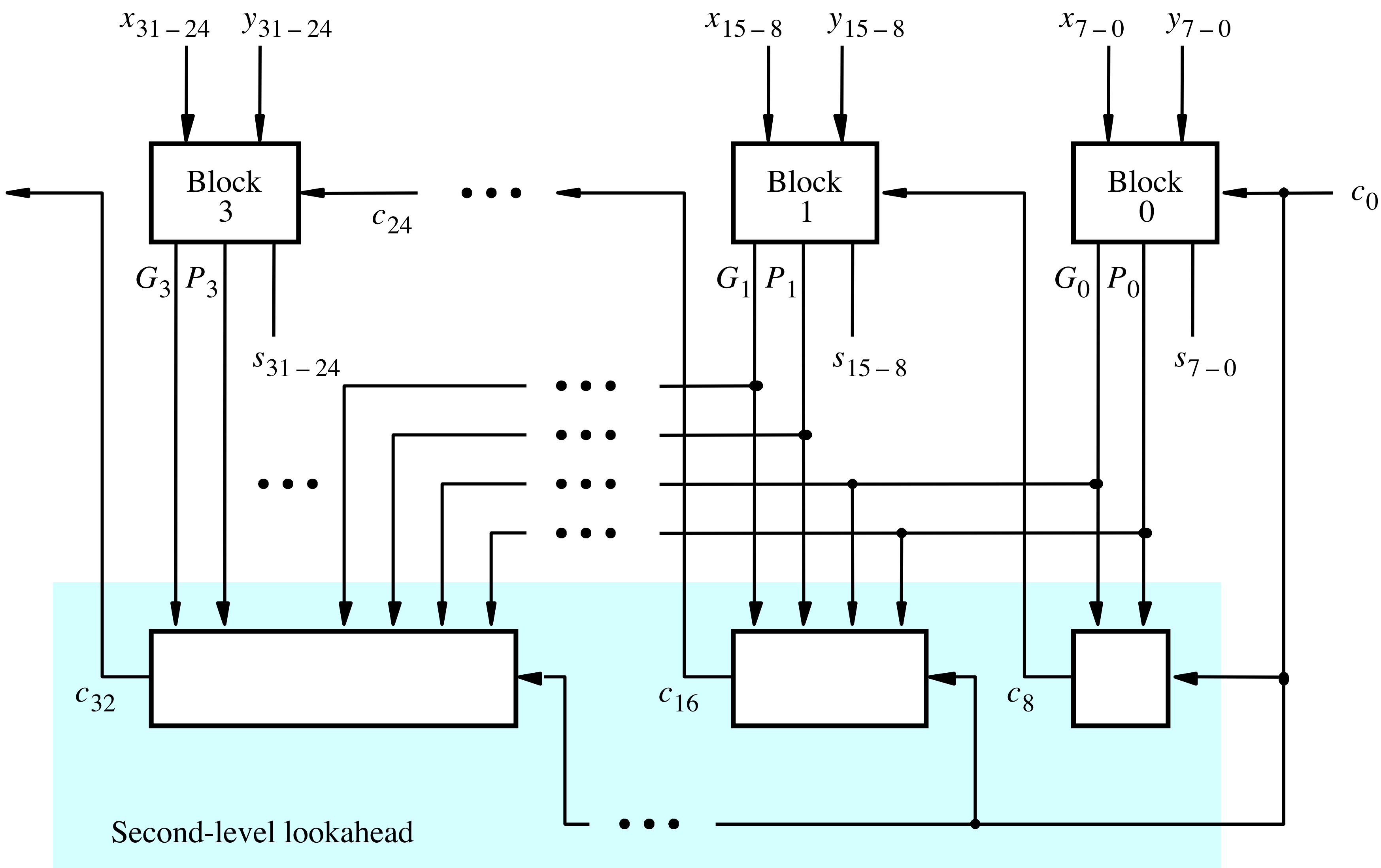
Even this takes only 3 gate delays

A hierarchical carry-lookahead adder with ripple-carry between blocks



[Figure 3.16 from the textbook]

A hierarchical carry-lookahead adder



[Figure 3.17 from the textbook]

The Hierarchical Carry Expression

$$
\begin{aligned}
c_8 = {} & g_7 + p_7 g_6 + p_7 p_6 g_5 + p_7 p_6 p_5 g_4 \\
& + p_7 p_6 p_5 p_4 g_3 + p_7 p_6 p_5 p_4 p_3 g_2 \\
& + p_7 p_6 p_5 p_4 p_3 p_2 g_1 + p_7 p_6 p_5 p_4 p_3 p_2 p_1 g_0 \\
& + p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0 c_0
\end{aligned}
$$

The Hierarchical Carry Expression

$$c_8 = \boxed{\begin{aligned} & g_7 + p_7 g_6 + p_7 p_6 g_5 + p_7 p_6 p_5 g_4 \\ & + p_7 p_6 p_5 p_4 g_3 + p_7 p_6 p_5 p_4 p_3 g_2 \\ & + p_7 p_6 p_5 p_4 p_3 p_2 g_1 + p_7 p_6 p_5 p_4 p_3 p_2 p_1 g_0 \end{aligned}} + \boxed{p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0} c_0$$

The Hierarchical Carry Expression

$$
\begin{aligned}
c_8 = {} & \boxed{\begin{aligned} & g_7 + p_7 g_6 + p_7 p_6 g_5 + p_7 p_6 p_5 g_4 \\ & + p_7 p_6 p_5 p_4 g_3 + p_7 p_6 p_5 p_4 p_3 g_2 \\ & + p_7 p_6 p_5 p_4 p_3 p_2 g_1 + p_7 p_6 p_5 p_4 p_3 p_2 p_1 g_0 \end{aligned}} \\
& + \boxed{p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0}\, c_0
\end{aligned}
$$

$\mathbf{G_0}$

$\mathbf{P_0}$

The Hierarchical Carry Expression

$$
\begin{aligned}
c_8 = {} & \underbrace{g_7 + p_7g_6 + p_7p_6g_5 + p_7p_6p_5g_4 + p_7p_6p_5p_4g_3 + p_7p_6p_5p_4p_3g_2 + p_7p_6p_5p_4p_3p_2g_1 + p_7p_6p_5p_4p_3p_2p_1g_0}_{G_0} \\
& + \underbrace{p_7p_6p_5p_4p_3p_2p_1p_0}_{P_0}c_0
\end{aligned}
$$

$$c_8 = G_0 + P_0c_0$$

The Hierarchical Carry Expression

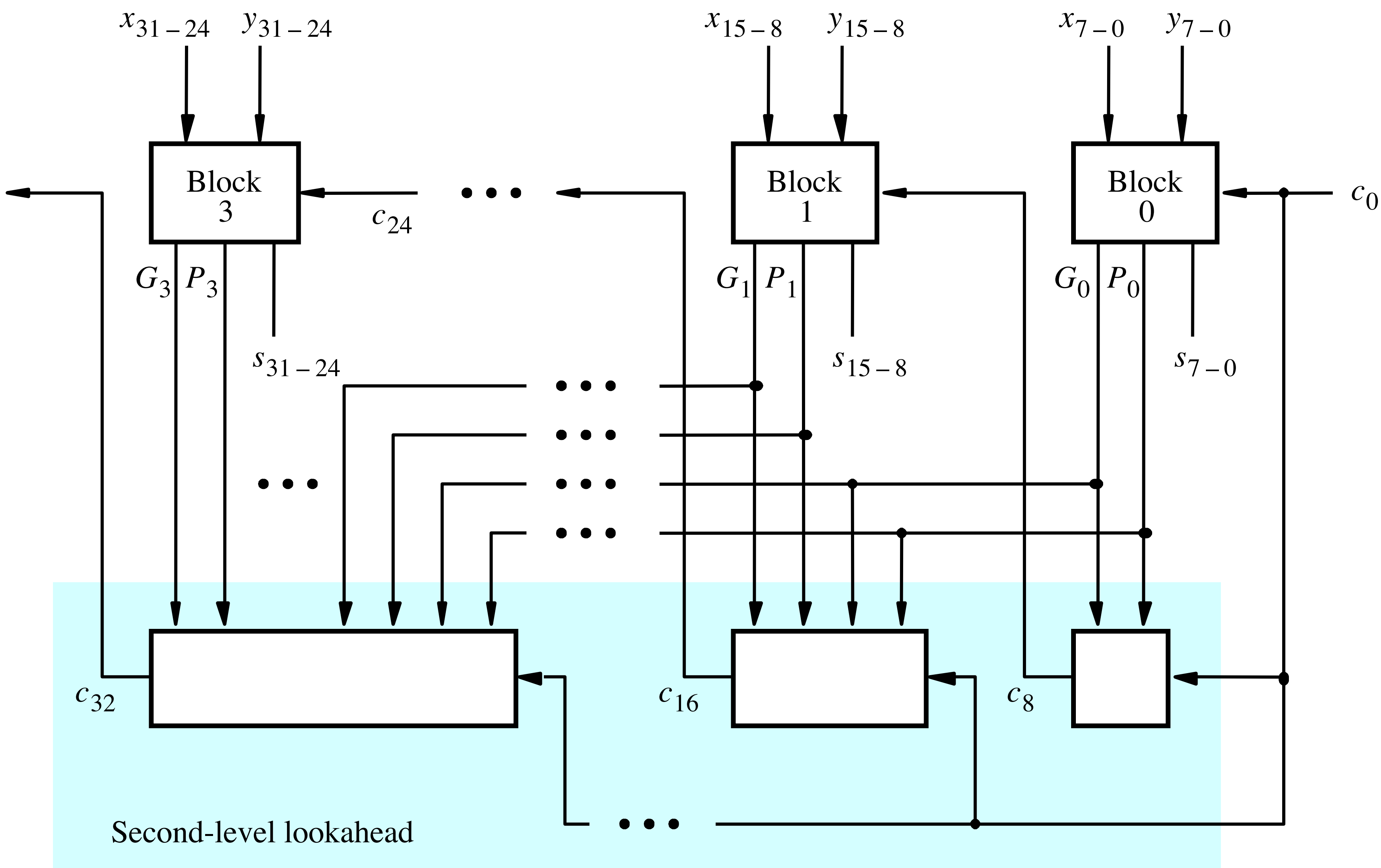
$$c_8 = G_0 + P_0 c_0$$

$$\begin{aligned} c_{16} &= G_1 + P_1 c_8 \\ &= G_1 + P_1 G_0 + P_1 P_0 c_0 \end{aligned}$$

$$c_{24} = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 c_0$$

$$c_{32} = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 c_0$$

A hierarchical carry-lookahead adder

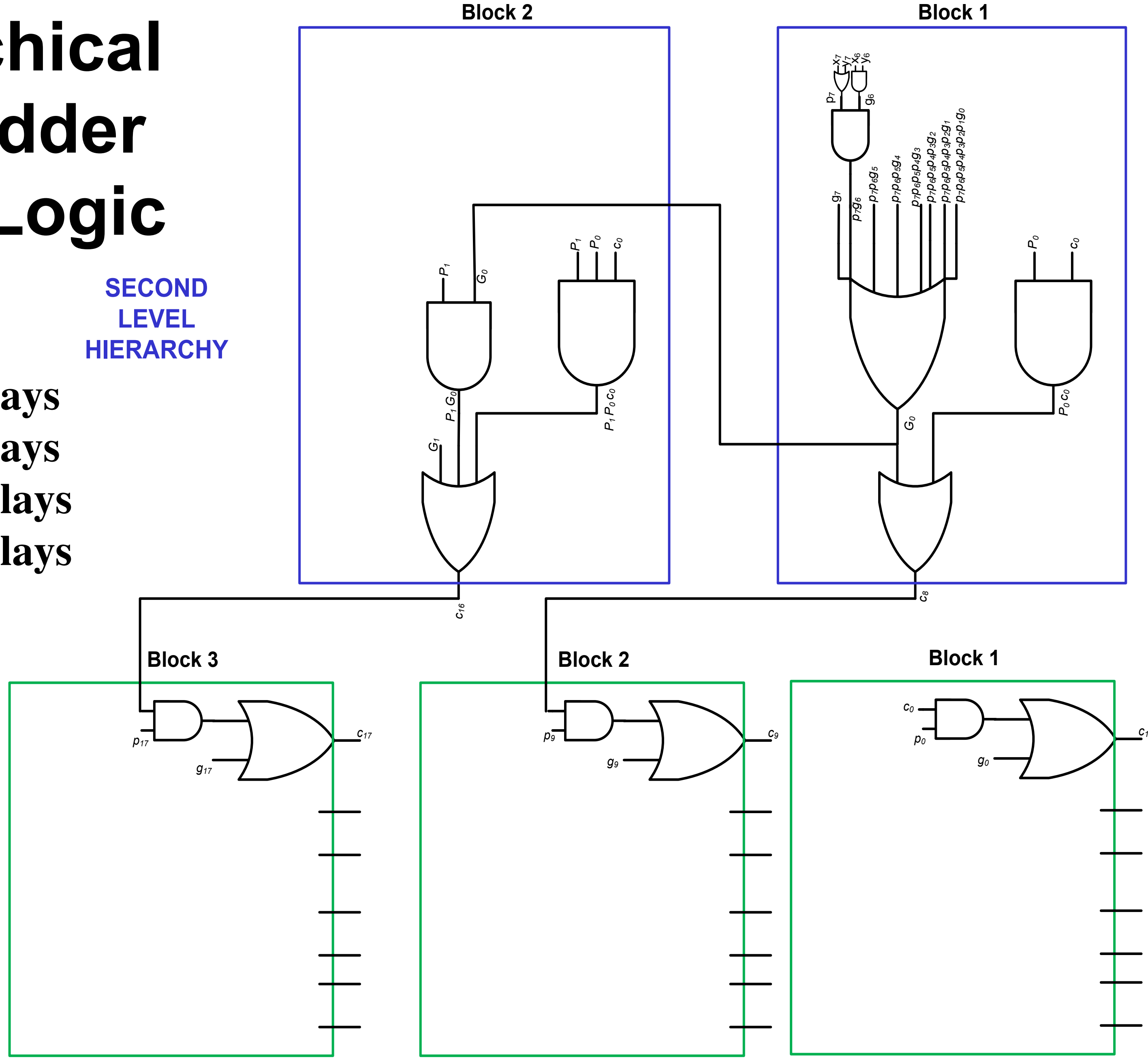


[Figure 3.17 from the textbook]

Hierarchical CLA Adder Carry Logic

SECOND LEVEL HIERARCHY

C8 – 5 gate delays
C16 – 5 gate delays
C24 – 5 Gate delays
C32 – 5 Gate delays



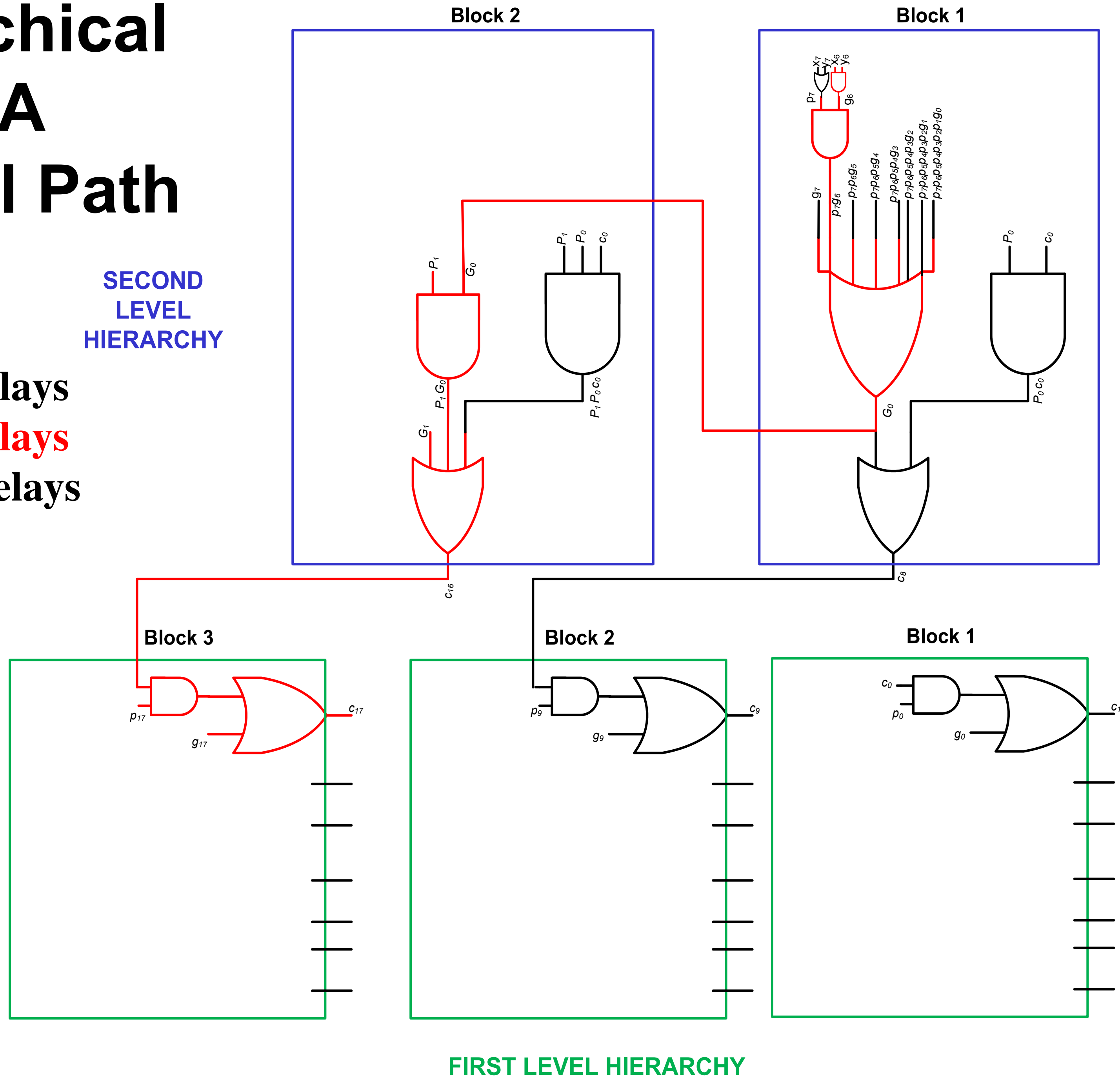
FIRST LEVEL HIERARCHY

Hierarchical CLA Critical Path

C9 – 7 gate delays

C17 – 7 gate delays

C25 – 7 Gate delays



Total Gate Delay Through a Hierarchical Carry-Lookahead Adder

- Is 8 gates
  - 3 to generate all Gj and Pj
  - +2 to generate c8, c16, c24, and c32
  - +2 to generate internal carries in the blocks
  - +1 to generate the sum bits (one extra XOR)

Questions?

THE END