



CprE 281: Digital Logic

Instructor: Alexander Stoytchev

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

Serial Adder

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

Administrative Stuff

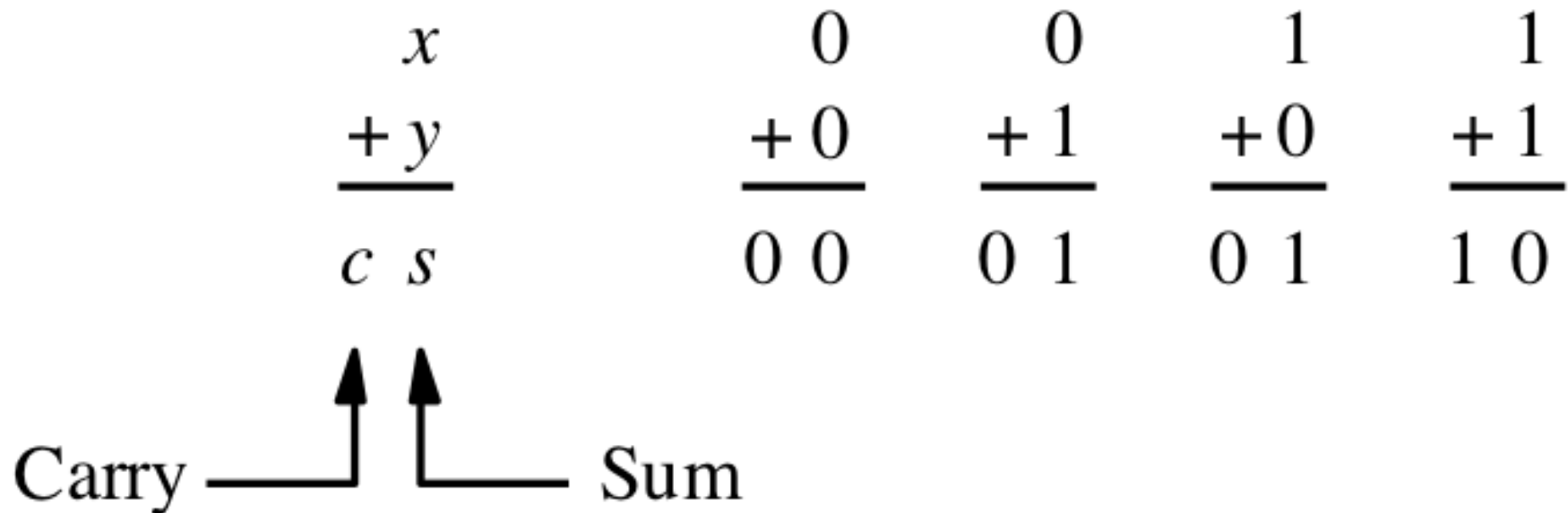
- **Homework 10 is out**
- **It is due on Monday Nov 12 @ 4pm**

Administrative Stuff

- **Final Project**

Quick Review

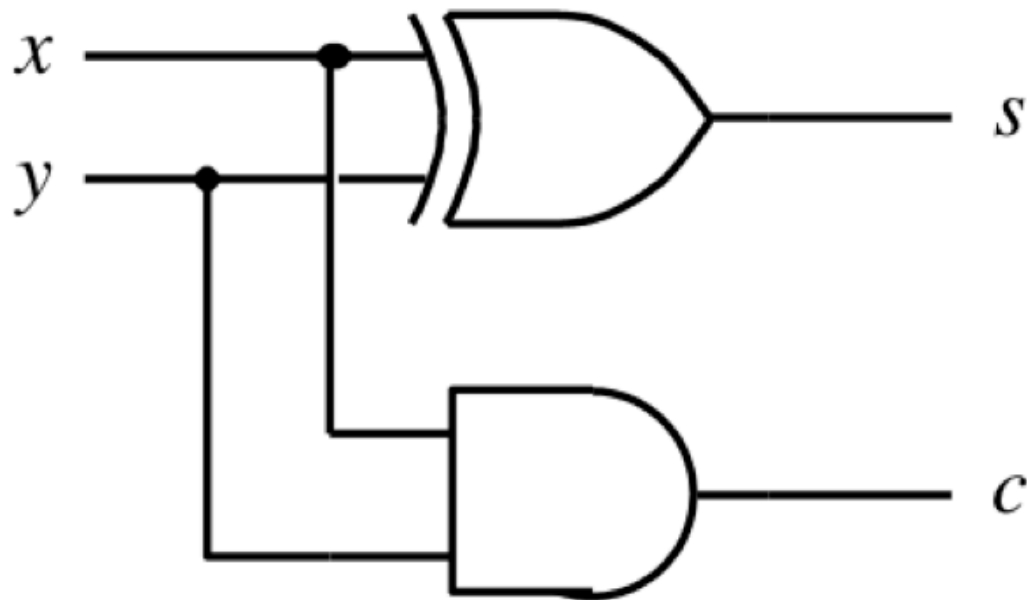
Adding two bits (there are four possible cases)



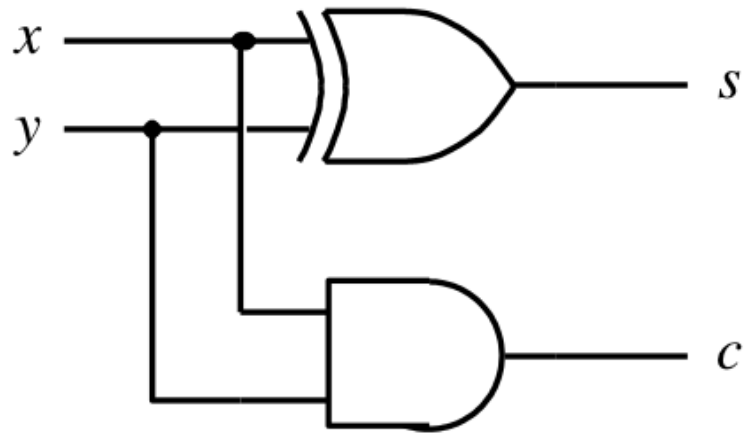
Adding two bits (the truth table)

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

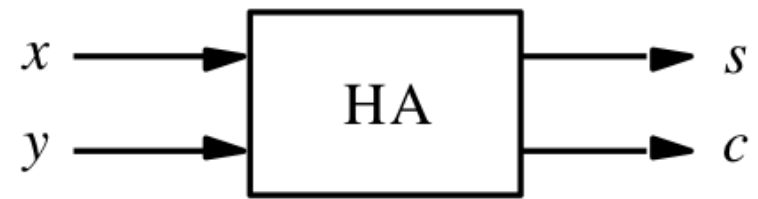
Adding two bits (the logic circuit)



The Half-Adder



(c) Circuit

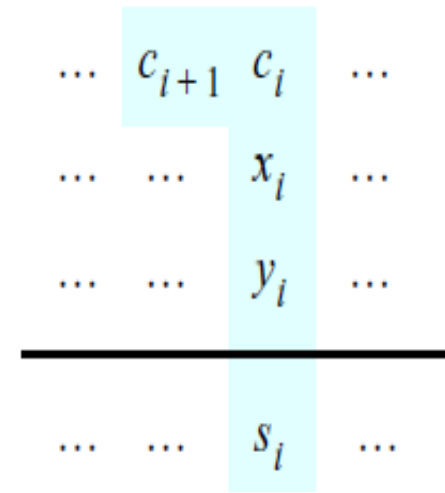


(d) Graphical symbol

Addition of multibit numbers

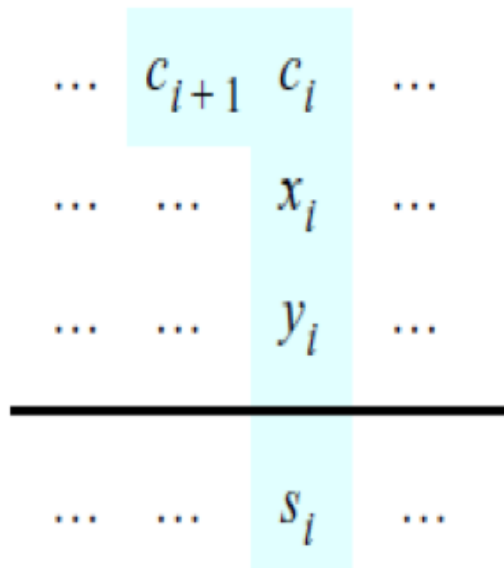
Generated carries \longrightarrow 1 1 1 0

$$\begin{array}{r}
 X = x_4x_3x_2x_1x_0 \quad 01111 \quad (15)_{10} \\
 + Y = y_4y_3y_2y_1y_0 \quad + 01010 \quad + (10)_{10} \\
 \hline
 S = s_4s_3s_2s_1s_0 \quad 11001 \quad (25)_{10}
 \end{array}$$



Bit position i

Problem Statement and Truth Table



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 3.2b from the textbook]

[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

$x_i y_i$

c_i	00	01	11	10
0				
1				

$s_i =$

$x_i y_i$

c_i	00	01	11	10
0				
1				

$c_{i+1} =$

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

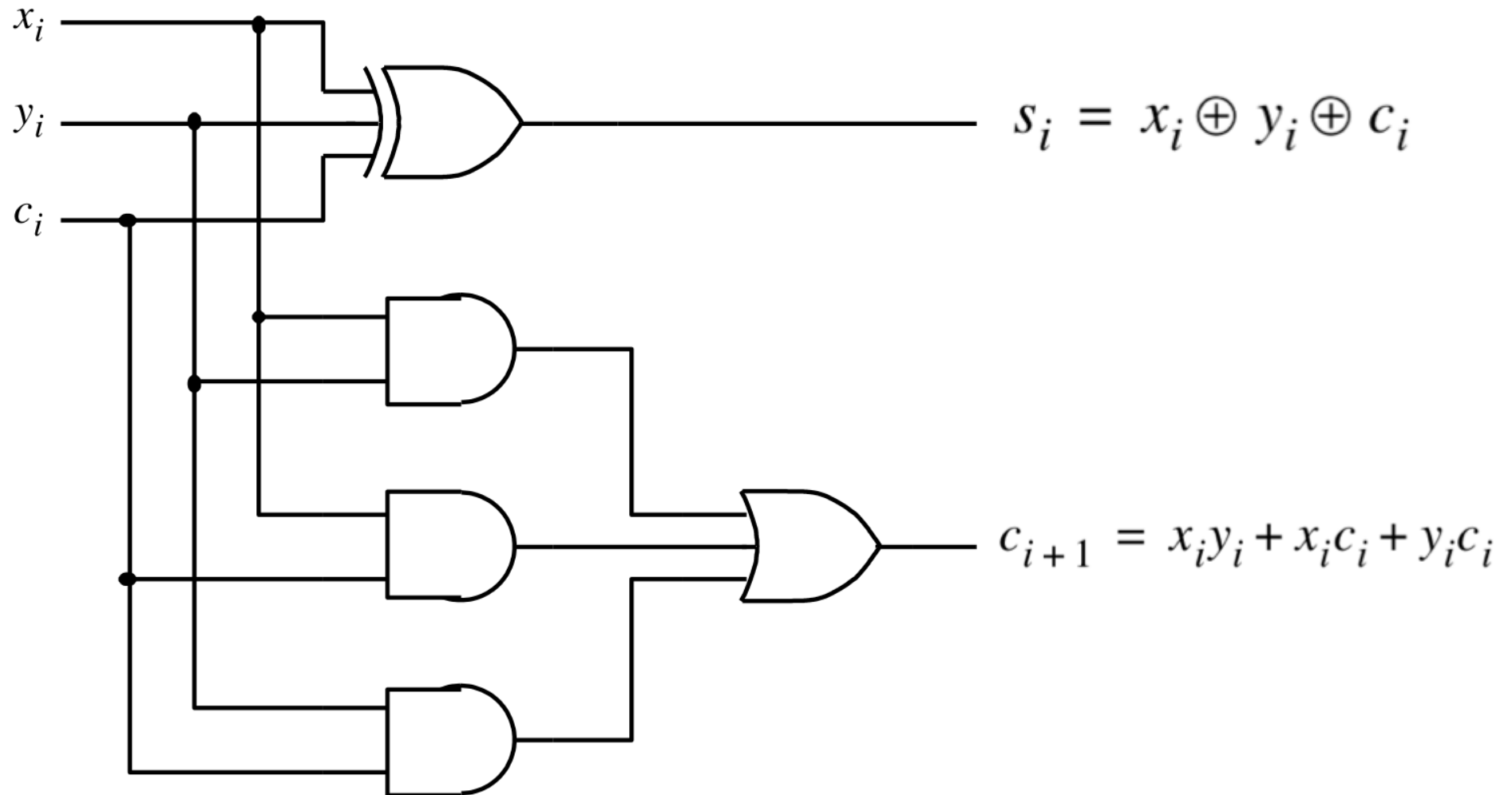
		$x_i y_i$			
		00	01	11	10
c_i	0		1		1
	1	1		1	

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

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

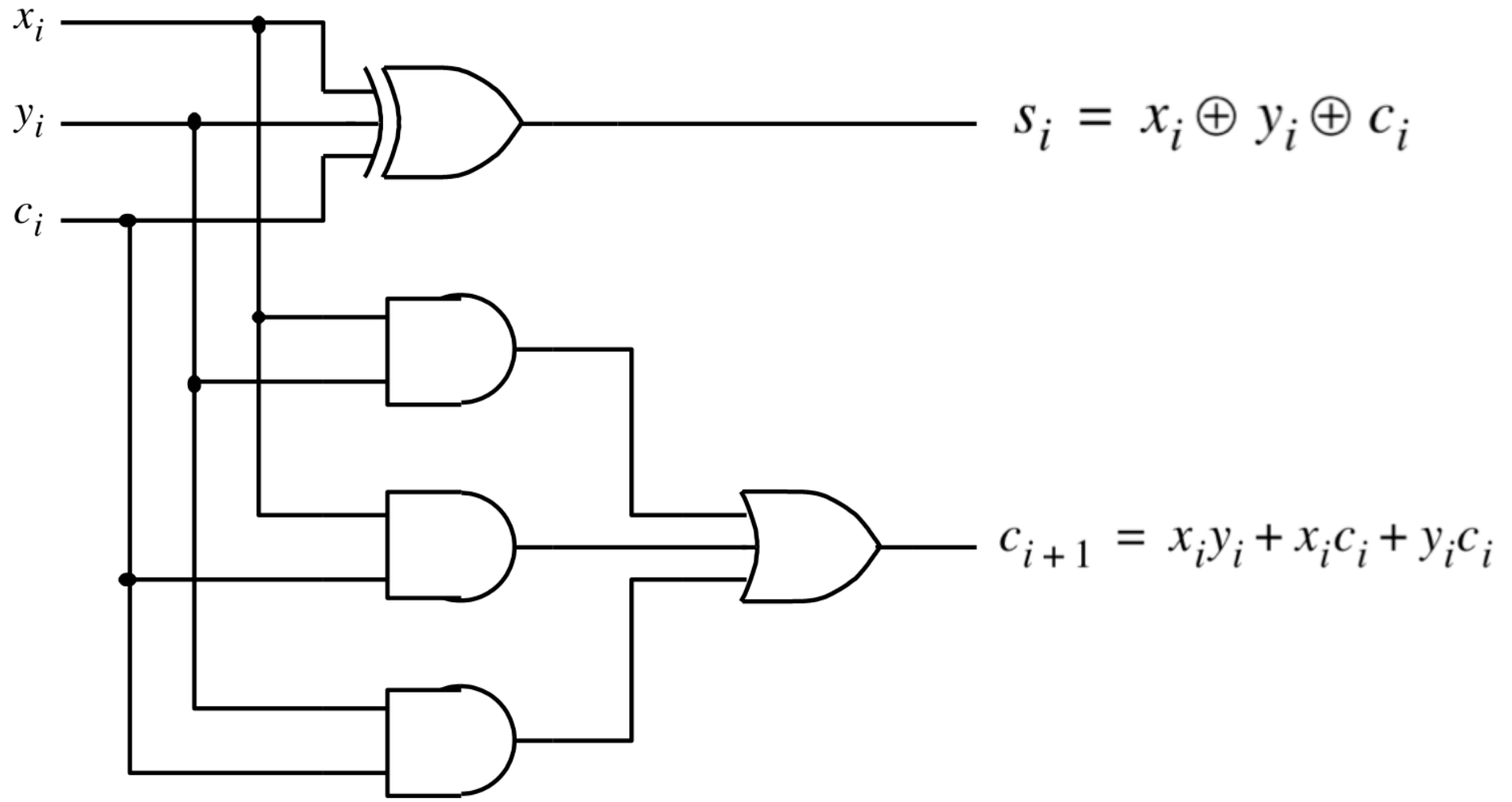
$$c_{i+1} = x_i y_i + x_i c_i + y_i c_i$$

The circuit for the two expressions



[Figure 3.3c from the textbook]

This is called the Full-Adder



[Figure 3.3c from the textbook]

XOR Magic

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

XOR Magic

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

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

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

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

XOR Magic

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

Can you prove this?

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

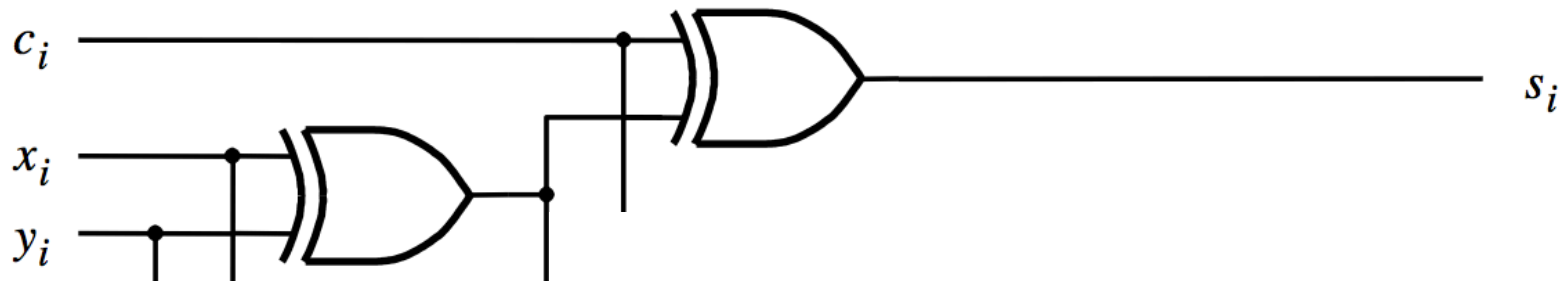
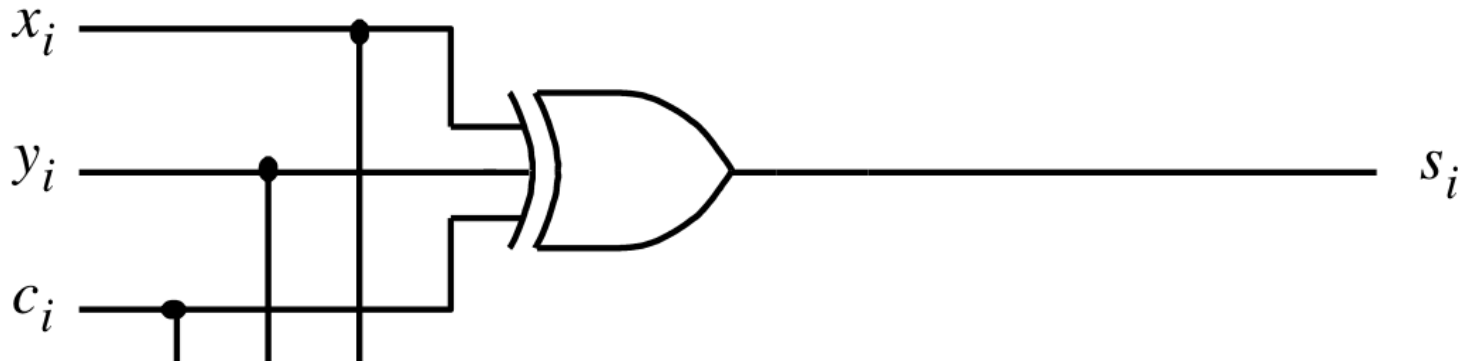
$$= (x_i \oplus y_i) \bar{c}_i + \overline{(x_i \oplus y_i)} c_i$$

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

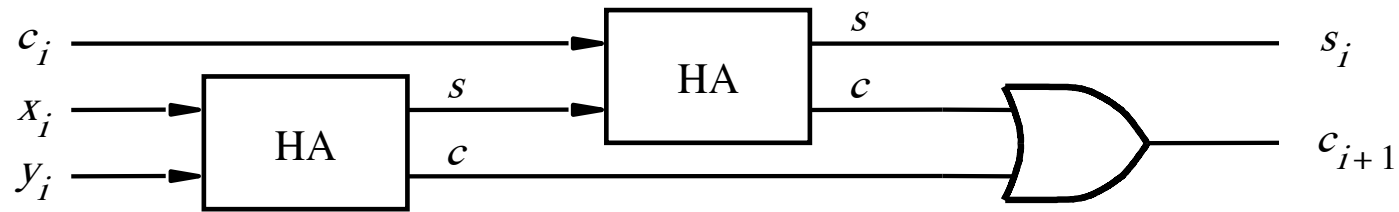
XOR Magic

(s_i can be implemented in two different ways)

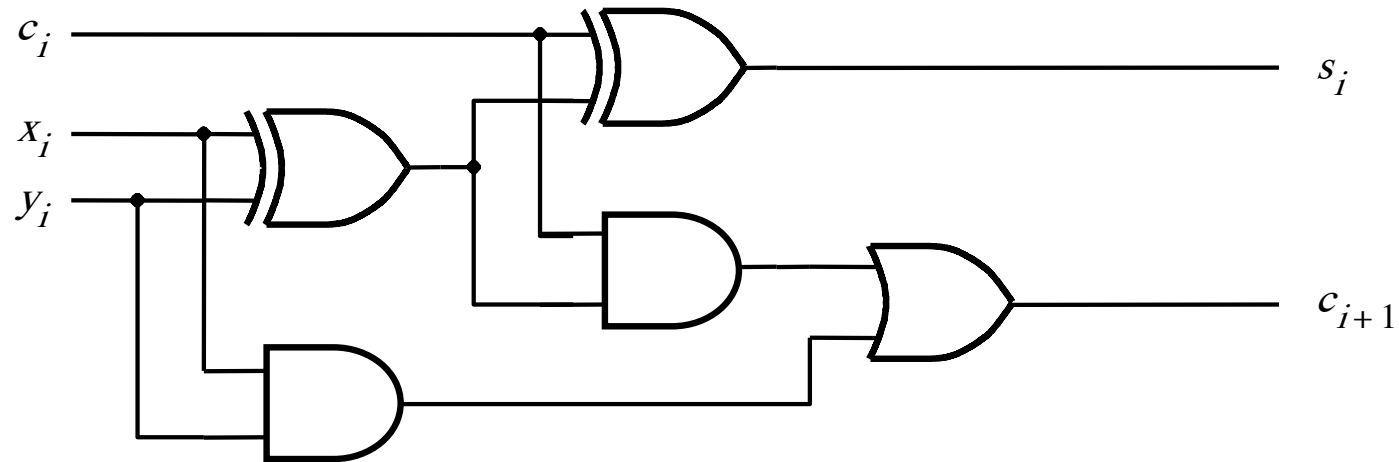
$$s_i = x_i \oplus y_i \oplus c_i$$



A decomposed implementation of the full-adder circuit

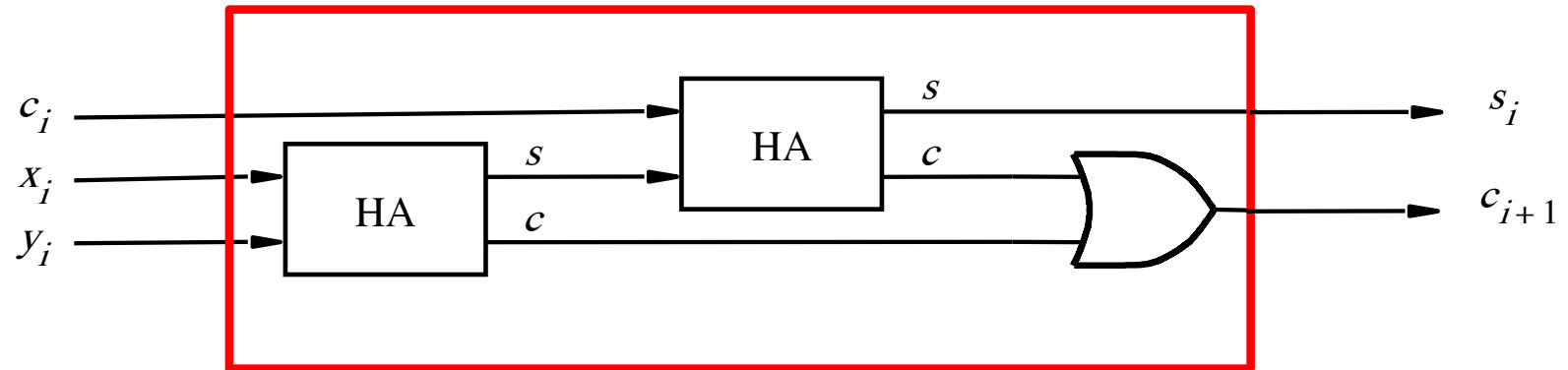


(a) Block diagram

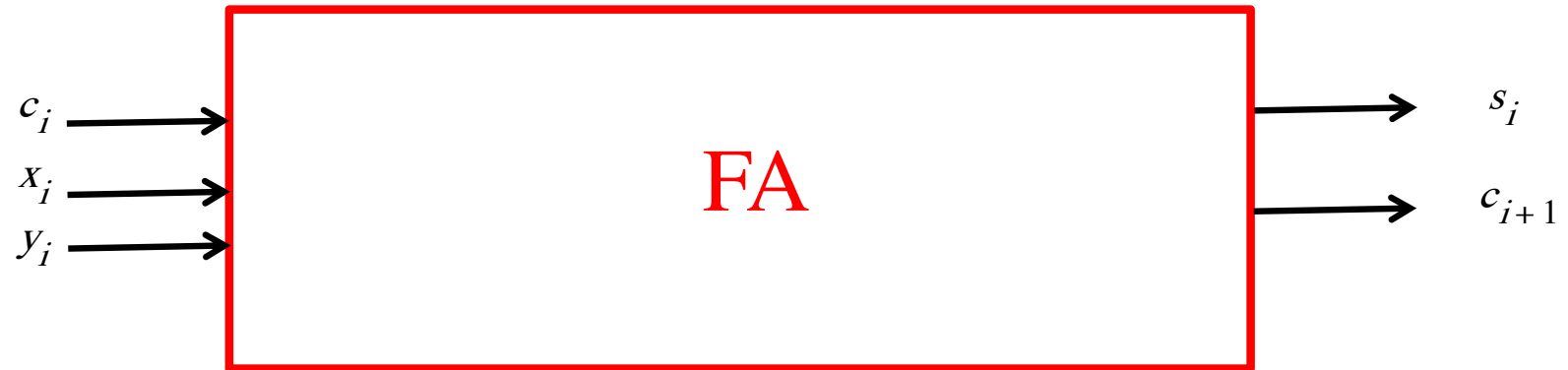


(b) Detailed diagram

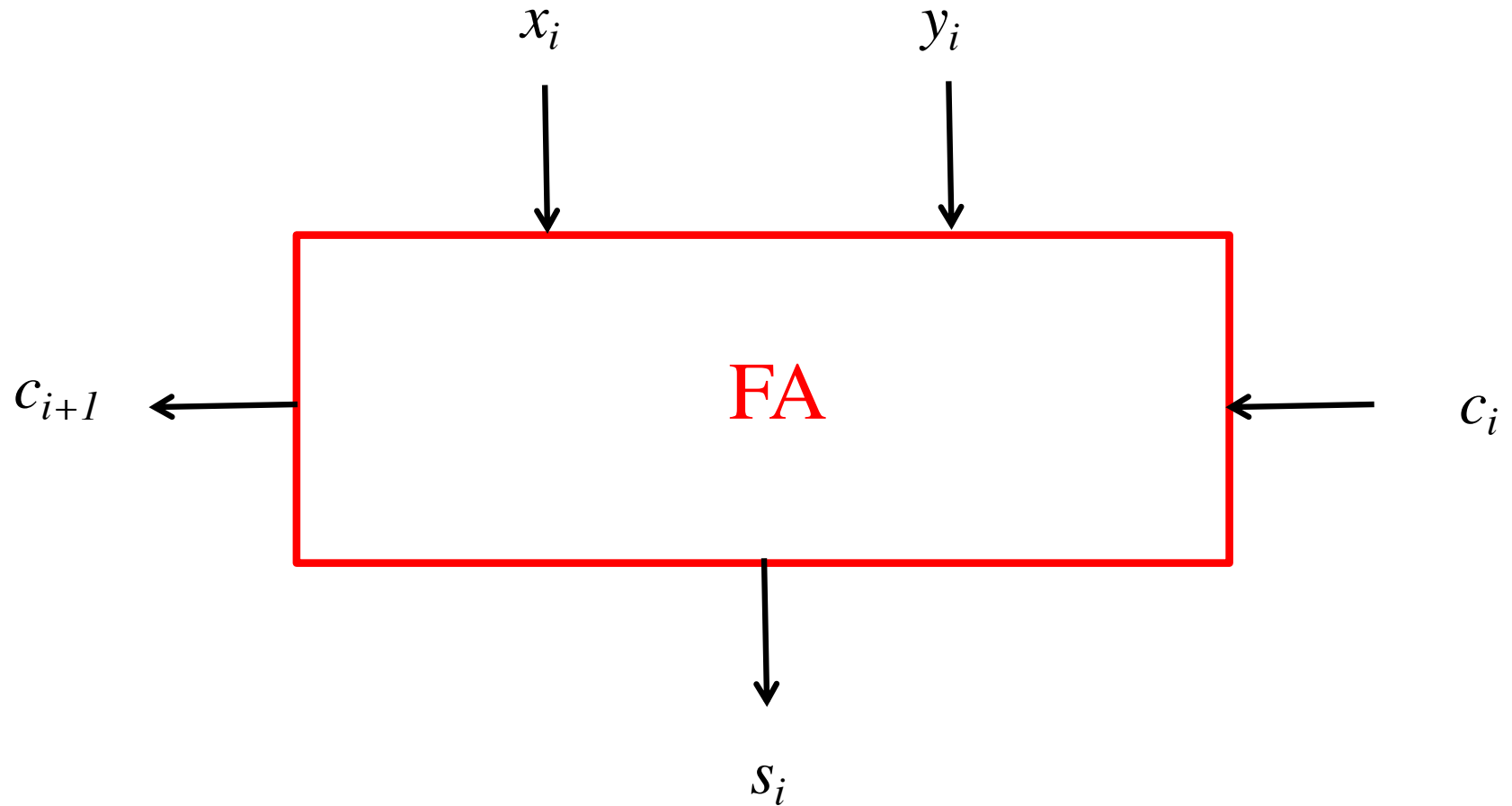
The Full-Adder Abstraction



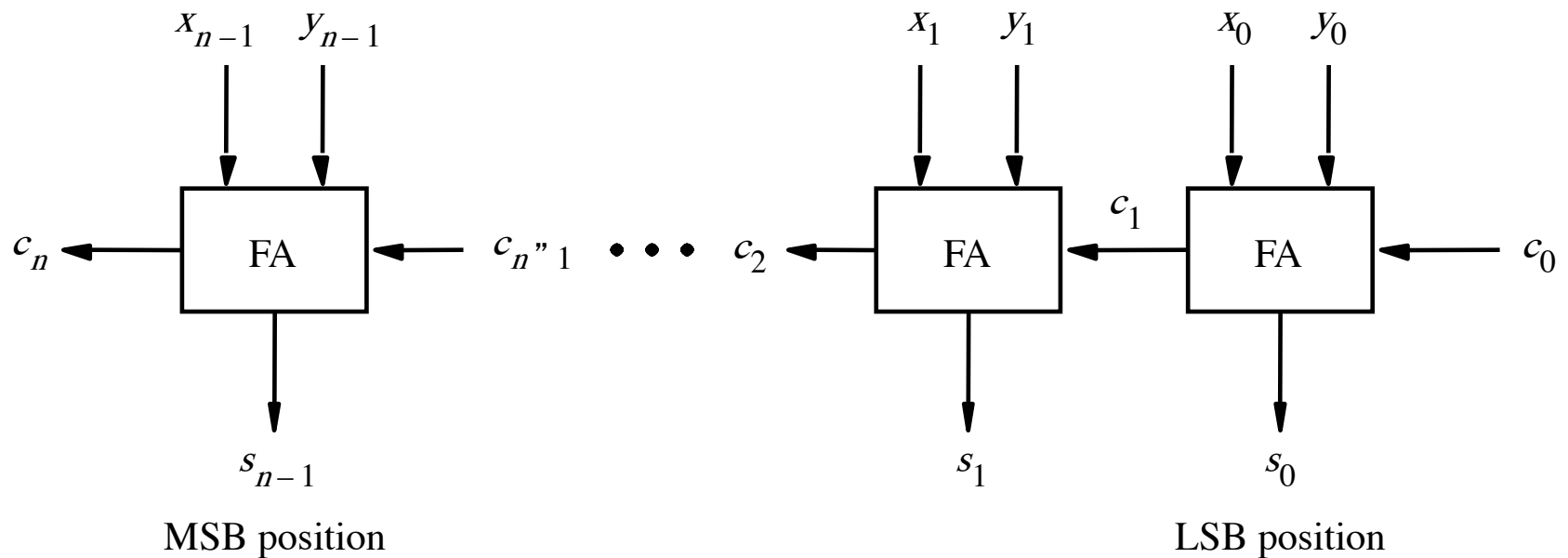
The Full-Adder Abstraction



We can place the arrows anywhere

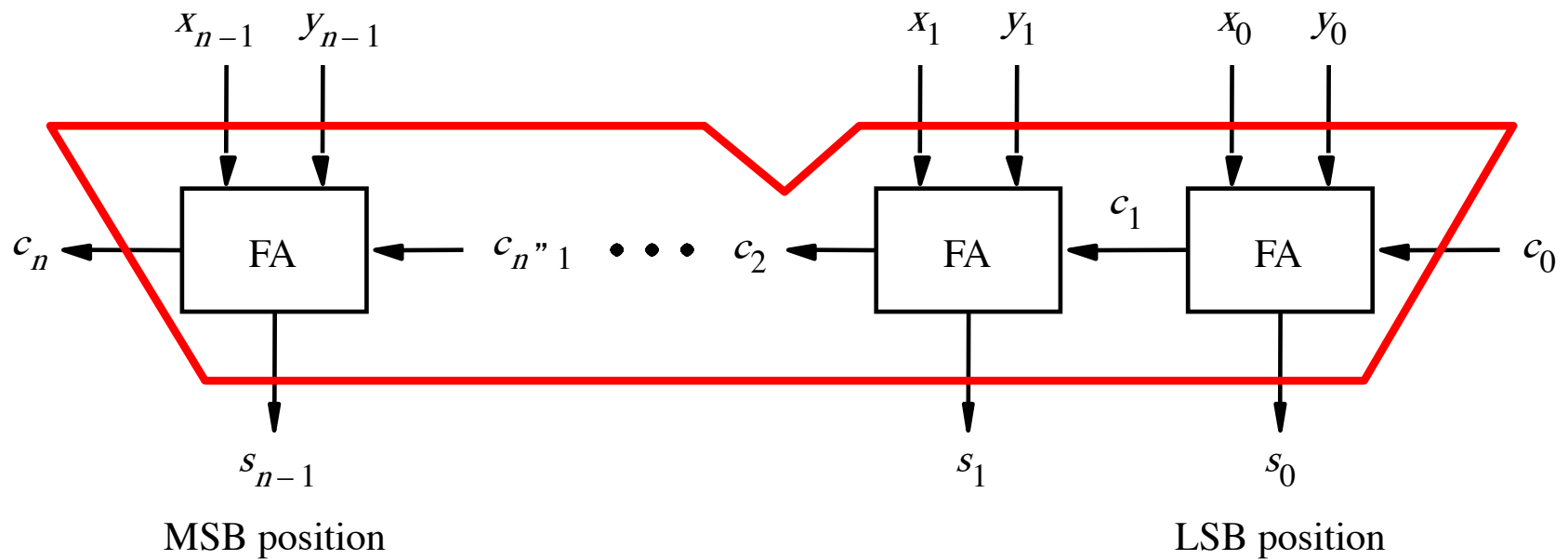


n-bit ripple-carry adder

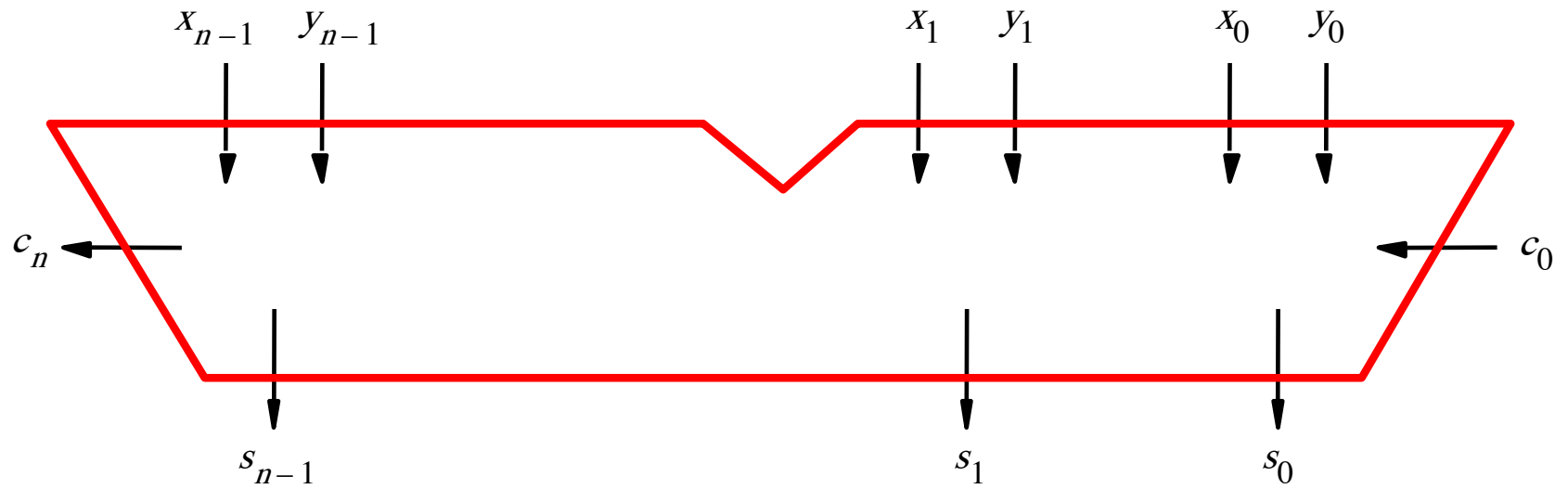


[Figure 3.5 from the textbook]

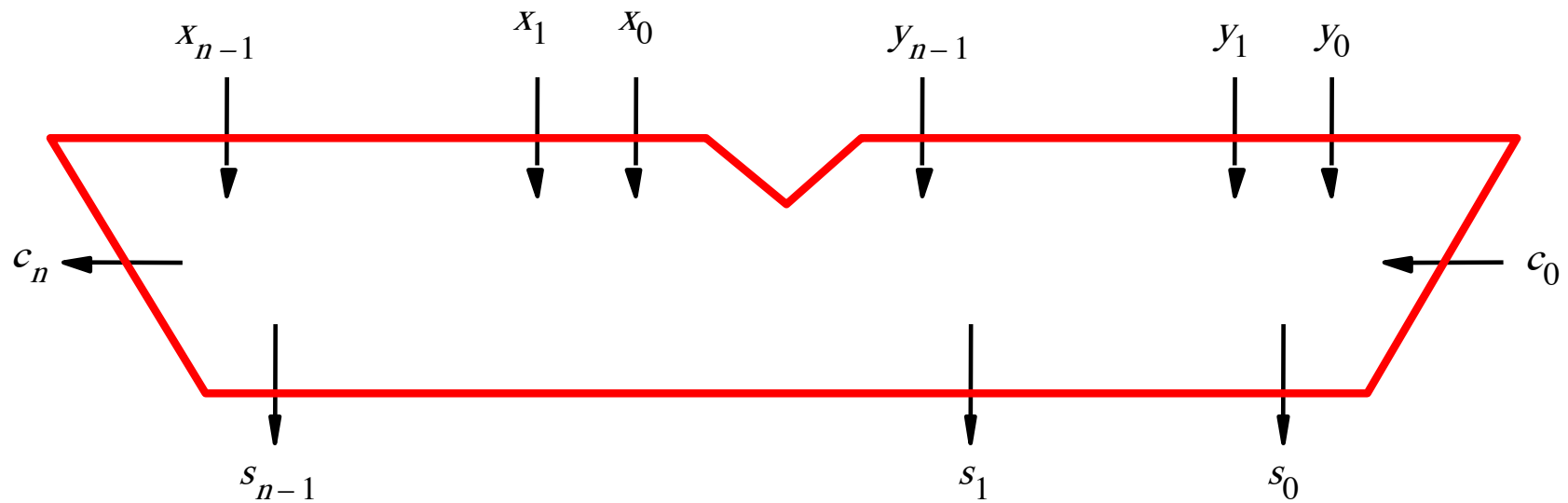
n-bit ripple-carry adder abstraction



n-bit ripple-carry adder abstraction



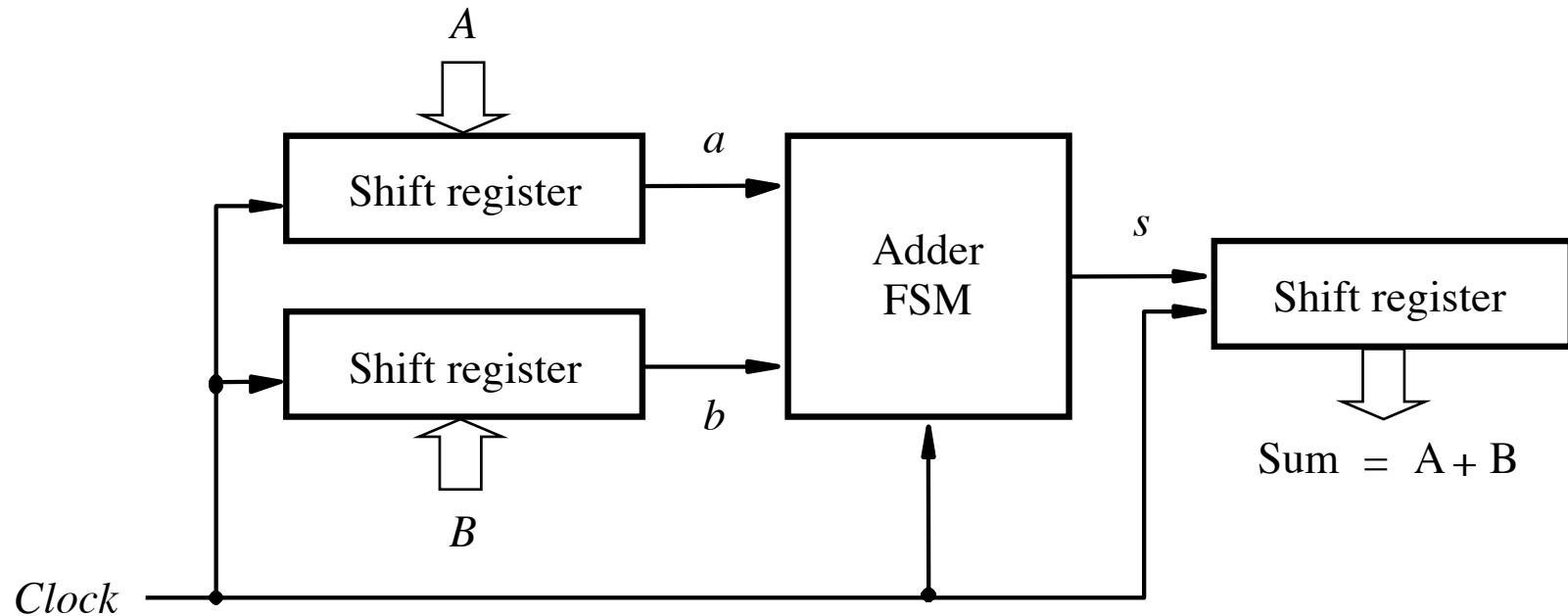
The x and y lines are typically grouped together for better visualization, but the underlying logic remains the same



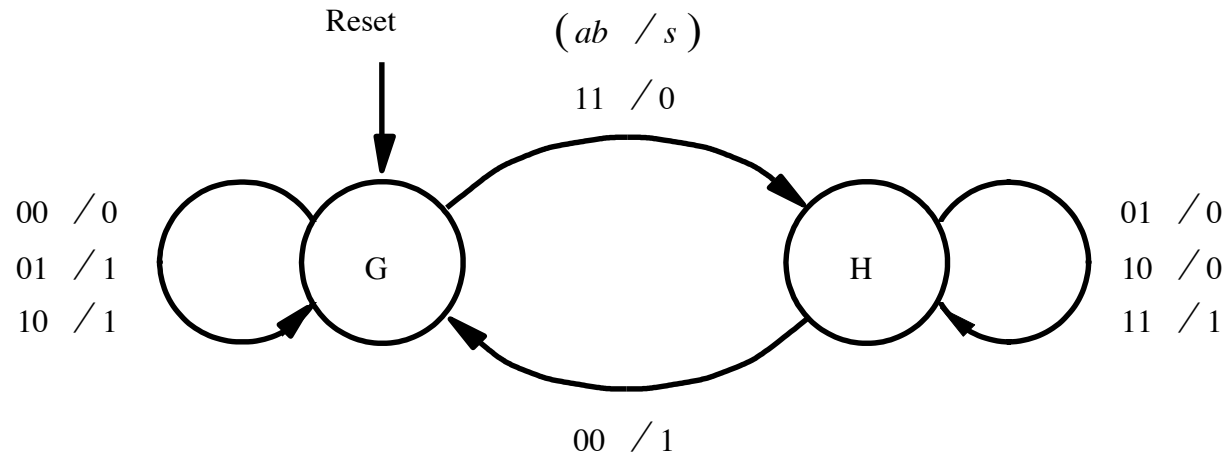
Serial Adder

- **The n-bit adder requires all bits to be provided at the same time.**
- **In some cases we may want to add the numbers as the bits come in.**
- **Also, with an n-bit adder we are limited to n-bits. Circuits for larger n are more complex.**
- **Can we add arbitrarily long numbers.**

Block diagram for the serial adder



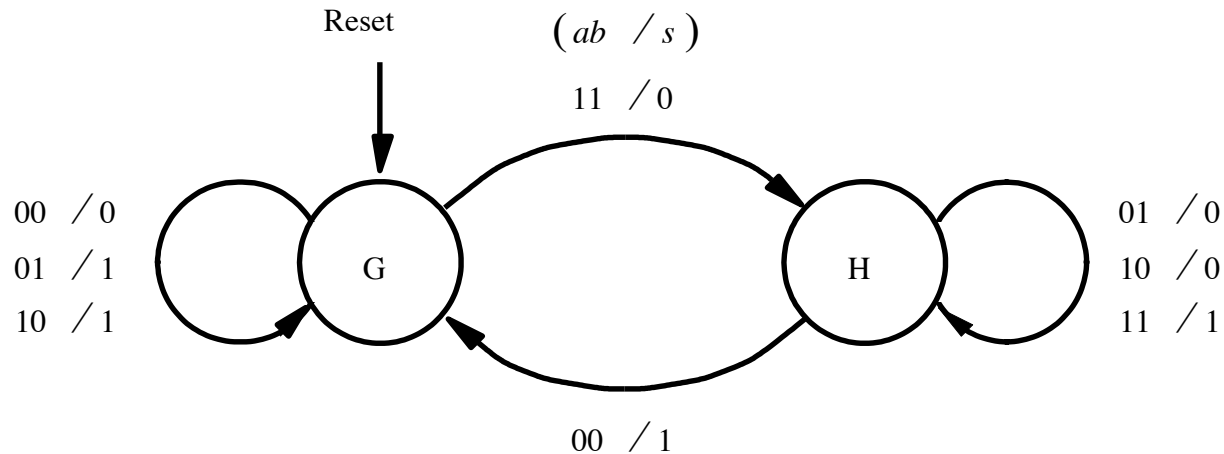
State diagram for the serial adder FSM



G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

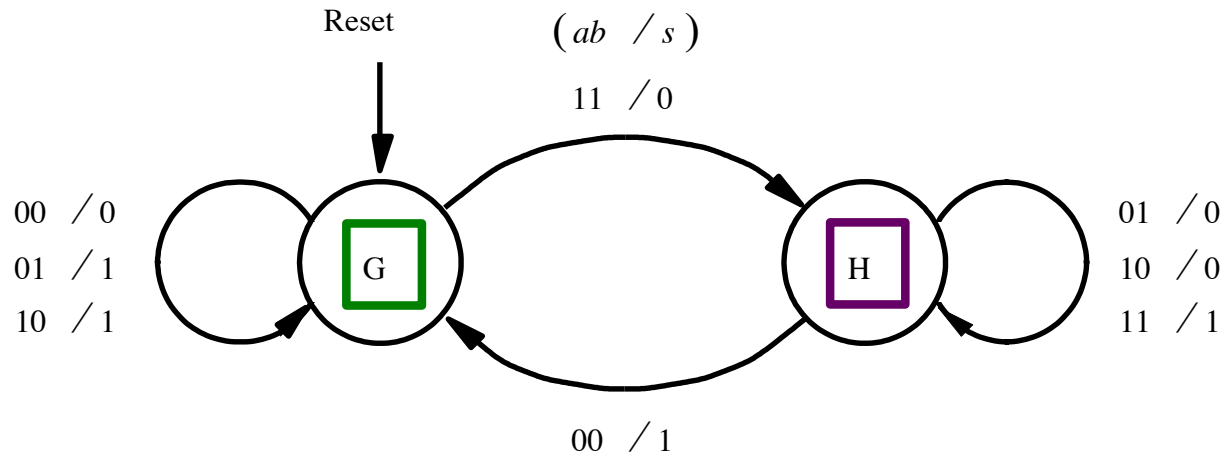


c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

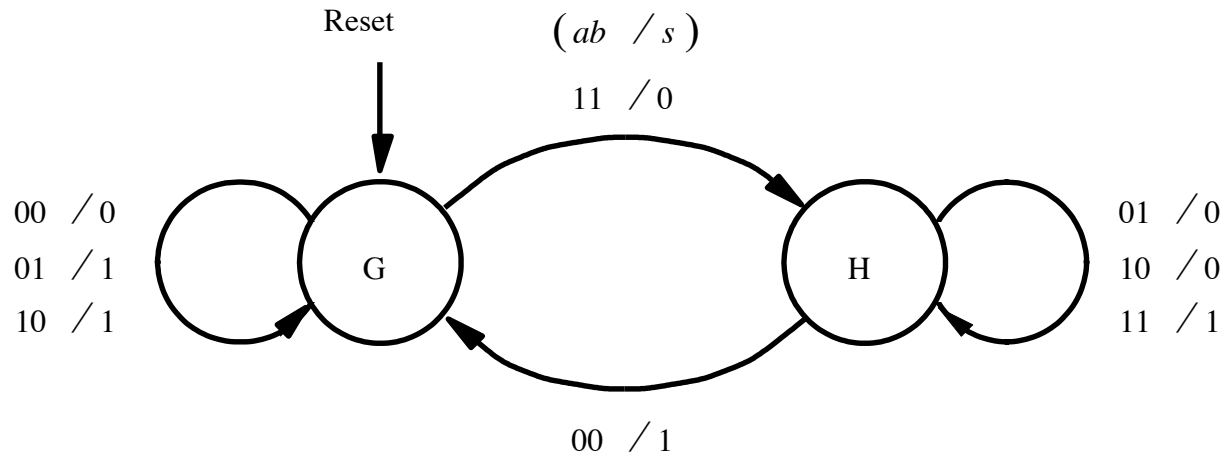


c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

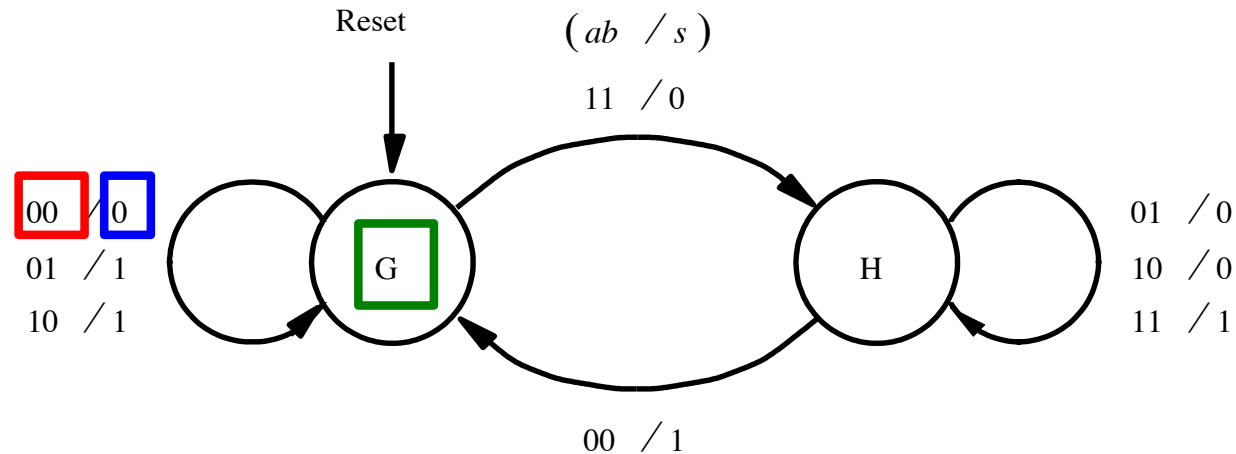


c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

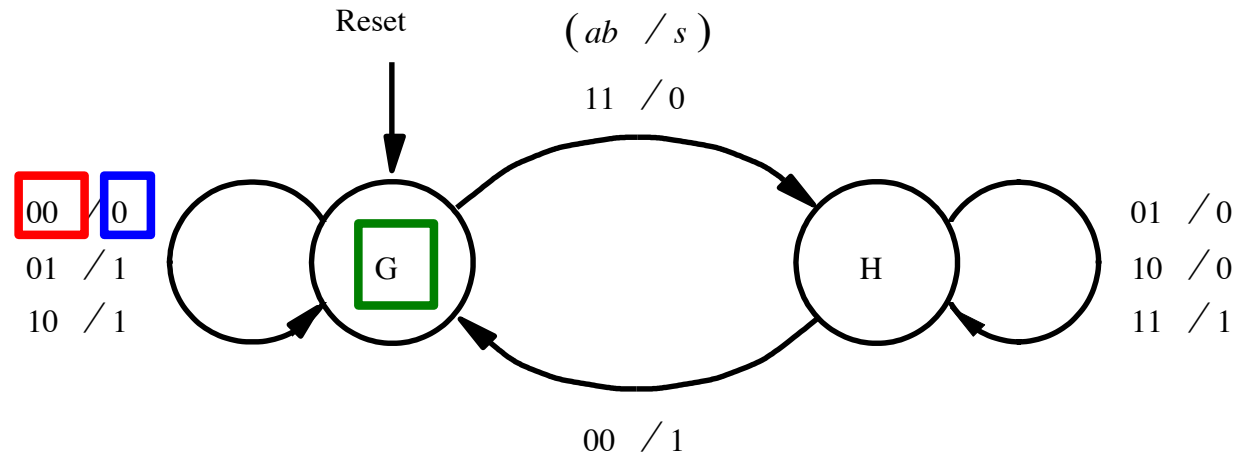


c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

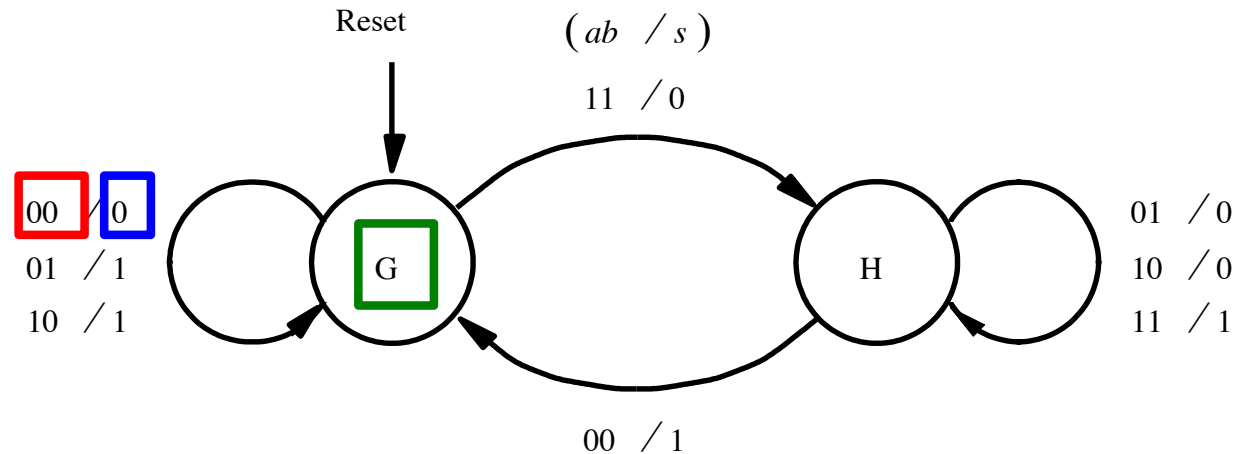


c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

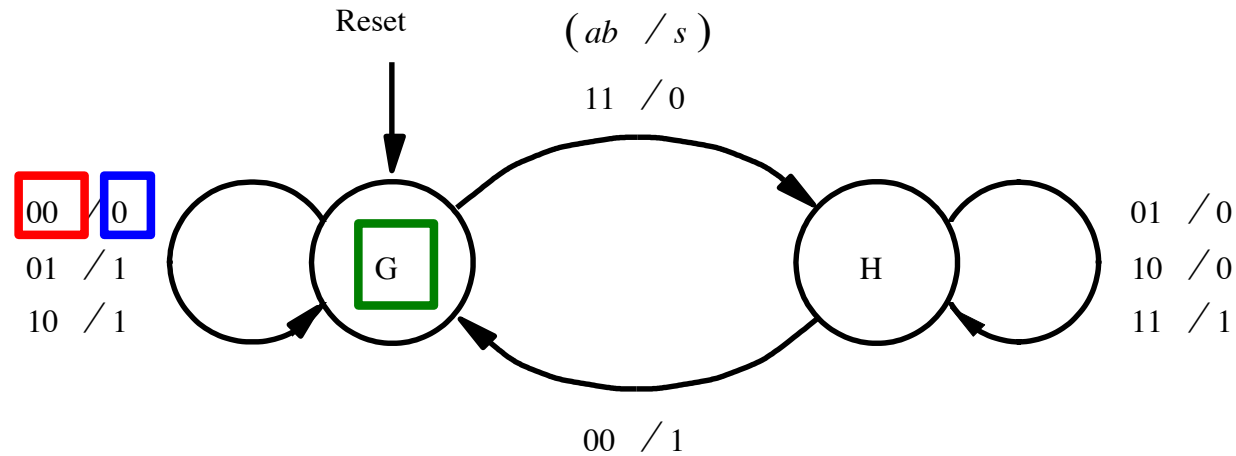


c_i	x_i	y_i	c_{i+1}	s_i
G	0	0	G	0
G	0	1	G	1
G	1	0	G	1
G	1	1	H	0
H	0	0	G	1
H	0	1	H	0
H	1	0	H	0
H	1	1	H	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

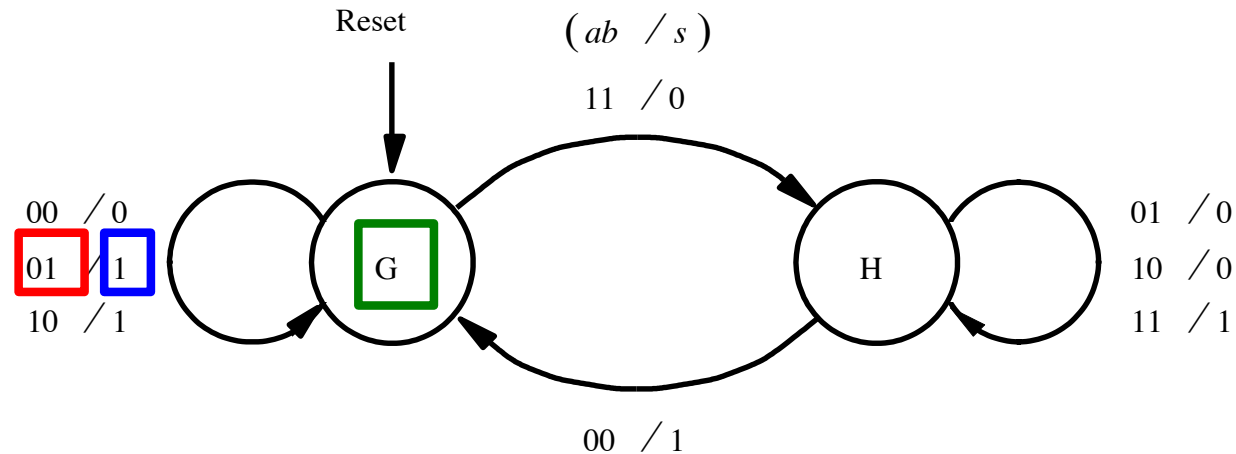


c_i	x_i	y_i	c_{i+1}	s_i
G	0	0	G	0
G	0	1	G	1
G	1	0	G	1
G	1	1	H	0
H	0	0	G	1
H	0	1	H	0
H	1	0	H	0
H	1	1	H	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

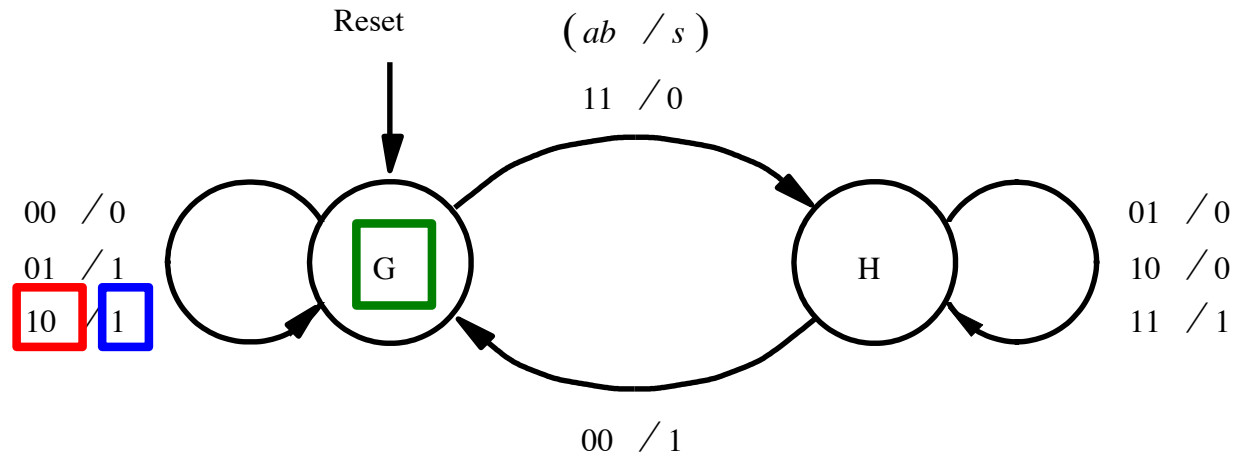


c_i	x_i	y_i	c_{i+1}	s_i
G	0	0	G	0
G	0	1	G	1
G	1	0	G	1
G	1	1	H	0
H	0	0	G	1
H	0	1	H	0
H	1	0	H	0
H	1	1	H	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

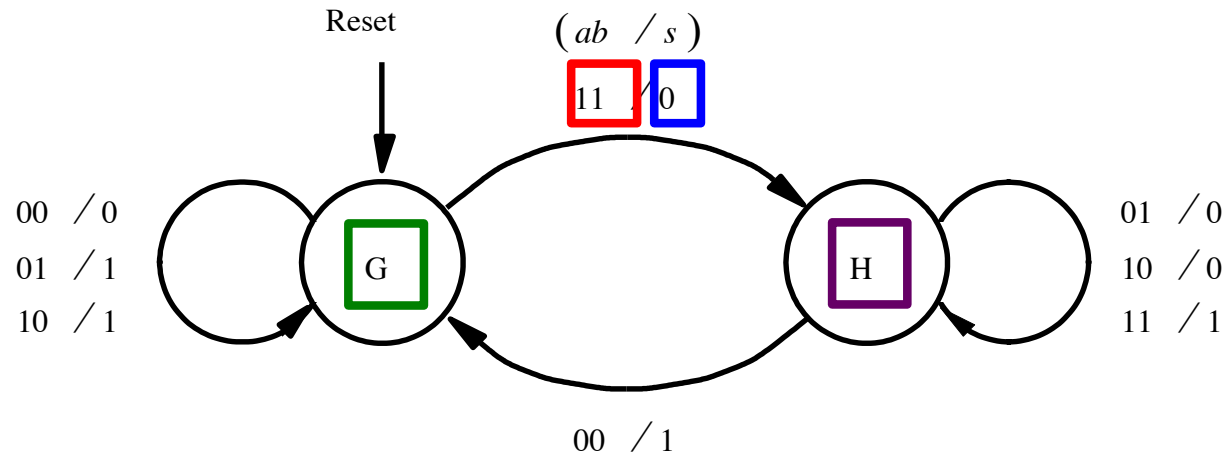


c_i	x_i	y_i	c_{i+1}	s_i
G	0	0	G	0
G	0	1	G	1
G	1	0	G	1
G	1	1	H	0
H	0	0	G	1
H	0	1	H	0
H	1	0	H	0
H	1	1	H	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

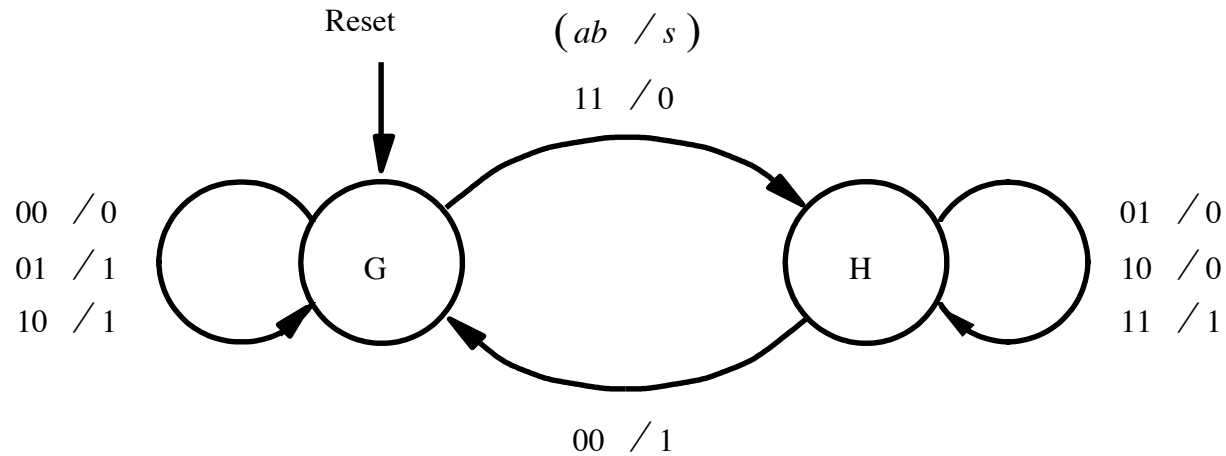


c_i	x_i	y_i	c_{i+1}	s_i
G	0	0	G	0
G	0	1	G	1
G	1	0	G	1
G	1	1	H	0
H	0	0	G	1
H	0	1	H	0
H	1	0	H	0
H	1	1	H	1

G: carry-in = 0

H: carry-in = 1

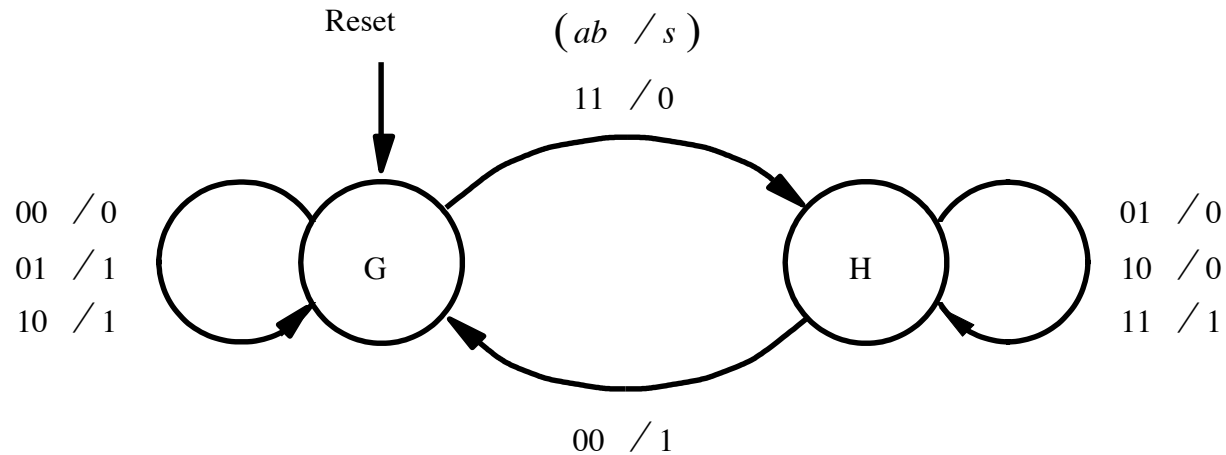
State diagram for the serial adder FSM



State table for the serial adder FSM

Present state	Next state				Output s			
	$ab = 00$	01	10	11	00	01	10	11
G								
H								

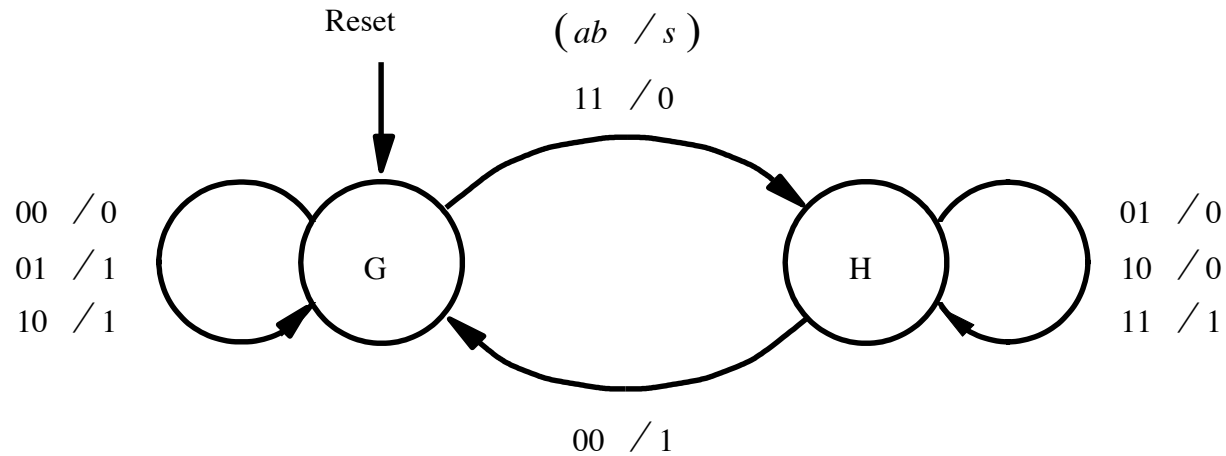
State diagram for the serial adder FSM



State table for the serial adder FSM

Present state	Next state				Output s			
	$ab = 00$	01	10	11	00	01	10	11
G	G	G	G	H				
H	G	H	H	H				

State diagram for the serial adder FSM



State table for the serial adder FSM

Present state	Next state				Output s			
	$ab = 00$	01	10	11	00	01	10	11
G	G	G	G	H	0	1	1	0
H	G	H	H	H	1	0	0	1

State table for the serial adder FSM

Present state	Next state				Output s			
	$ab = 00$	01	10	11	00	01	10	11
G	G	G	G	H	0	1	1	0
H	G	H	H	H	1	0	0	1

State table for the serial adder FSM

Present state	Next state				Output s			
	$ab = 00$	01	10	11	00	01	10	11
G	G	G	G	H	0	1	1	0
H	G	H	H	H	1	0	0	1

State-assigned table for the serial adder

Present state	Next state				Output			
	$ab = 00$	01	10	11	00	01	10	11
y	Y				s			
0								
1								

State table for the serial adder FSM

Present state	Next state				Output s			
	$ab = 00$	01	10	11	00	01	10	11
G	G	G	G	H	0	1	1	0
H	G	H	H	H	1	0	0	1

State-assigned table for the serial adder

Present state	Next state				Output			
	$ab = 00$	01	10	11	00	01	10	11
y	Y				s			
0	0	0	0	1				
1	0	1	1	1				

State table for the serial adder FSM

Present state	Next state				Output s			
	$ab = 00$	01	10	11	00	01	10	11
G	G	G	G	H	0	1	1	0
H	G	H	H	H	1	0	0	1

State-assigned table for the serial adder

Present state	Next state				Output			
	$ab = 00$	01	10	11	00	01	10	11
y	Y				s			
0	0	0	0	1	0	1	1	0
1	0	1	1	1	1	0	0	1

Derivation of Y and s

Present state	Next state				Output			
	$ab = 00$	01	10	11	00	01	10	11
y	Y				s			
0	0	0	0	1	0	1	1	0
1	0	1	1	1	1	0	0	1

Derivation of Y and s

Present state y	Next state				Output			
	$ab=00$	01	10	11	00	01	10	11
	Y				s			
0	0	0	0	1	0	1	1	0
1	0	1	1	1	1	0	0	1

y	a	b	Y	s
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Derivation of Y and s

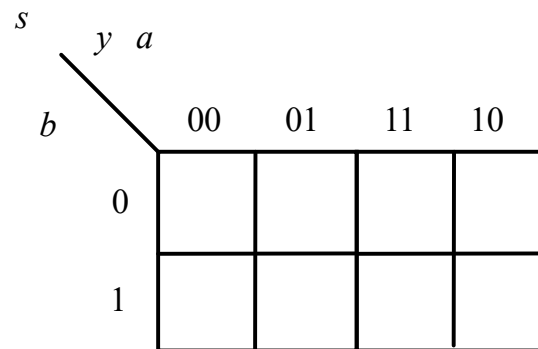
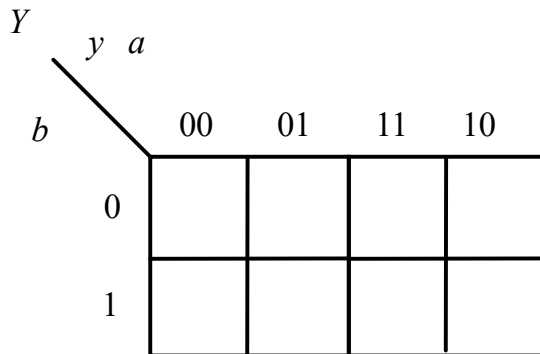
Present state y	Next state				Output			
	$ab=00$	01	10	11	00	01	10	11
	Y				s			
0	0	0	0	1	0	1	1	0
1	0	1	1	1	1	0	0	1

y	a	b	Y	s
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

Derivation of Y and s

Present state <i>y</i>	Next state				Output			
	<i>ab</i> = 00	01	10	11	00	01	10	11
	<i>Y</i>				<i>s</i>			
0	0	0	0	1	0	1	1	0
1	0	1	1	1	1	0	0	1

<i>y</i>	<i>a</i>	<i>b</i>	<i>Y</i>	<i>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



Derivation of Y and s

Present state y	Next state				Output			
	$ab=00$	01	10	11	00	01	10	11
	Y				s			
0	0	0	0	1	0	1	1	0
1	0	1	1	1	1	0	0	1

y	a	b	Y	s
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

Y

y	a			
b	00	01	11	10
0	0	0	1	0
1	0	1	1	1

s

y	a			
b	00	01	11	10
0	0	1	0	1
1	1	0	1	0

Derivation of Y and s

Present state <i>y</i>	Next state				Output			
	<i>ab</i> = 00	01	10	11	00	01	10	11
	<i>Y</i>				<i>s</i>			
0	0	0	0	1	0	1	1	0
1	0	1	1	1	1	0	0	1

<i>y</i>	<i>a</i>	<i>b</i>	<i>Y</i>	<i>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

Y

<i>y</i>	<i>a</i>			
<i>b</i>	00	01	11	10
0	0	0	1	0
1	0	1	1	1

s

<i>y</i>	<i>a</i>			
<i>b</i>	00	01	11	10
0	0	1	0	1
1	1	0	1	0

Derivation of Y and s

Present state <i>y</i>	Next state				Output			
	<i>ab</i> = 00	01	10	11	00	01	10	11
0	0	0	0	1	0	1	1	0
1	0	1	1	1	1	0	0	1

<i>y</i>	<i>a</i>	<i>b</i>	<i>Y</i>	<i>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

Y

<i>y</i>	<i>a</i>	<i>b</i>			
		00	01	11	10
0		0	0	1	0
1		0	1	1	1

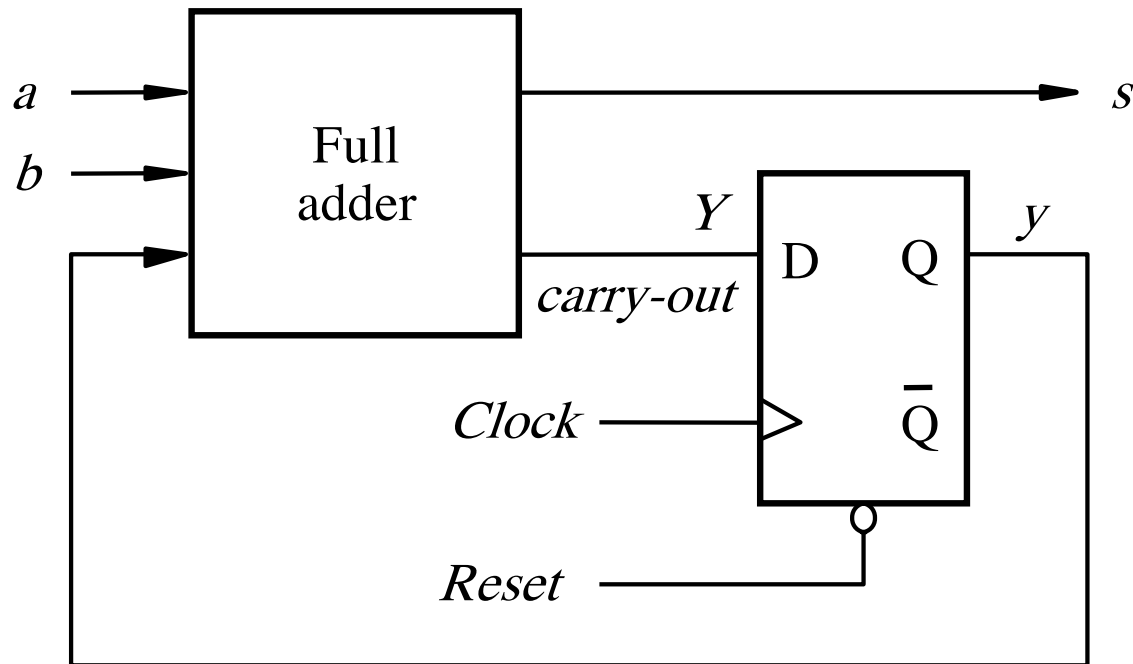
s

<i>y</i>	<i>a</i>	<i>b</i>			
		00	01	11	10
0		0	1	0	1
1		1	0	1	0

$$Y = ab + ay + by$$

$$s = \text{XOR}(\text{XOR}(a, b), y)$$

Circuit for the serial adder FSM

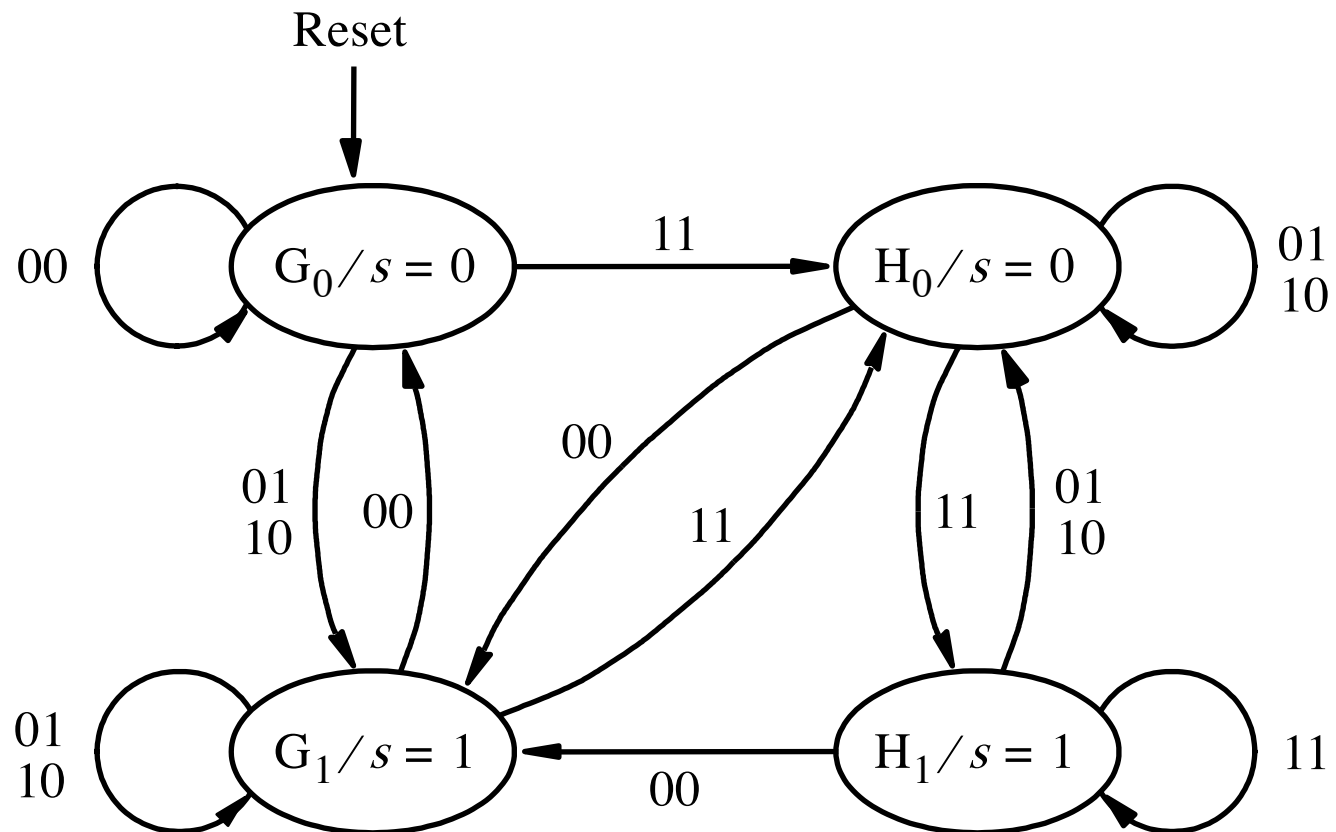


$$Y = ab + ay + by$$

$$s = \text{XOR}(\text{XOR}(a, b), y)$$

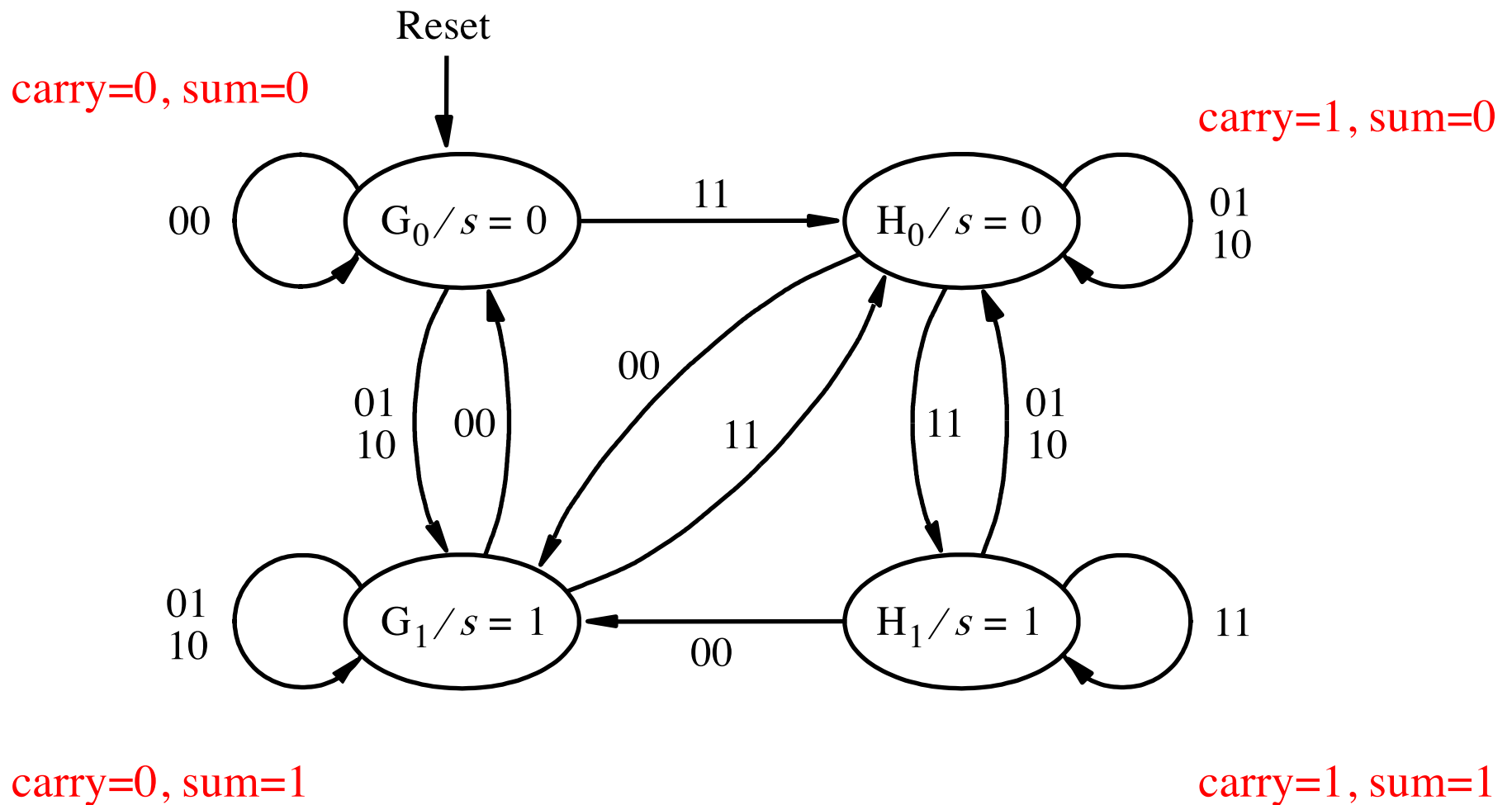
Moore Machine Implementation

State diagram for the Moore-type serial adder FSM

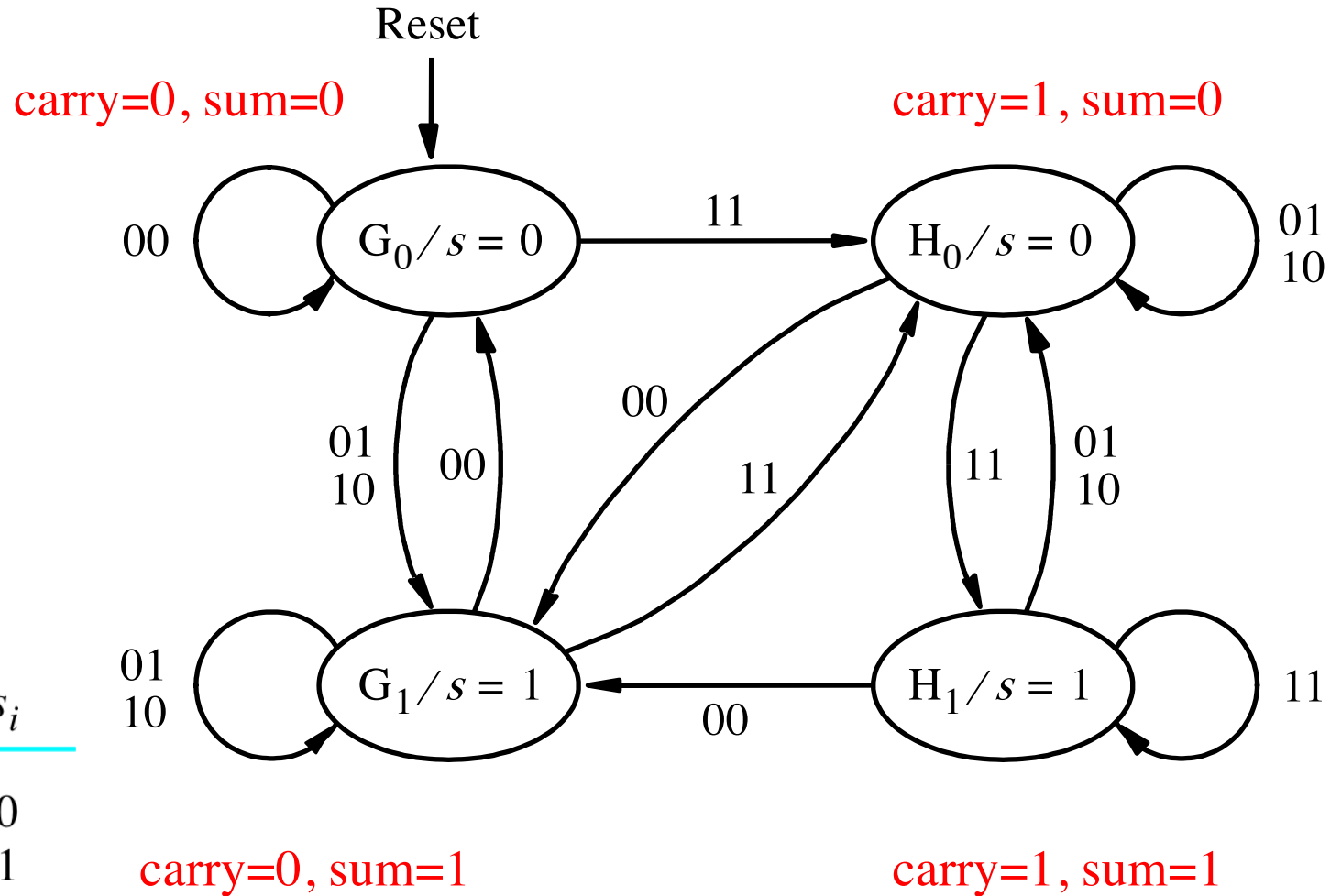


[Figure 6.44 from the textbook]

State diagram for the Moore-type serial adder FSM



State diagram for the Moore-type serial adder FSM

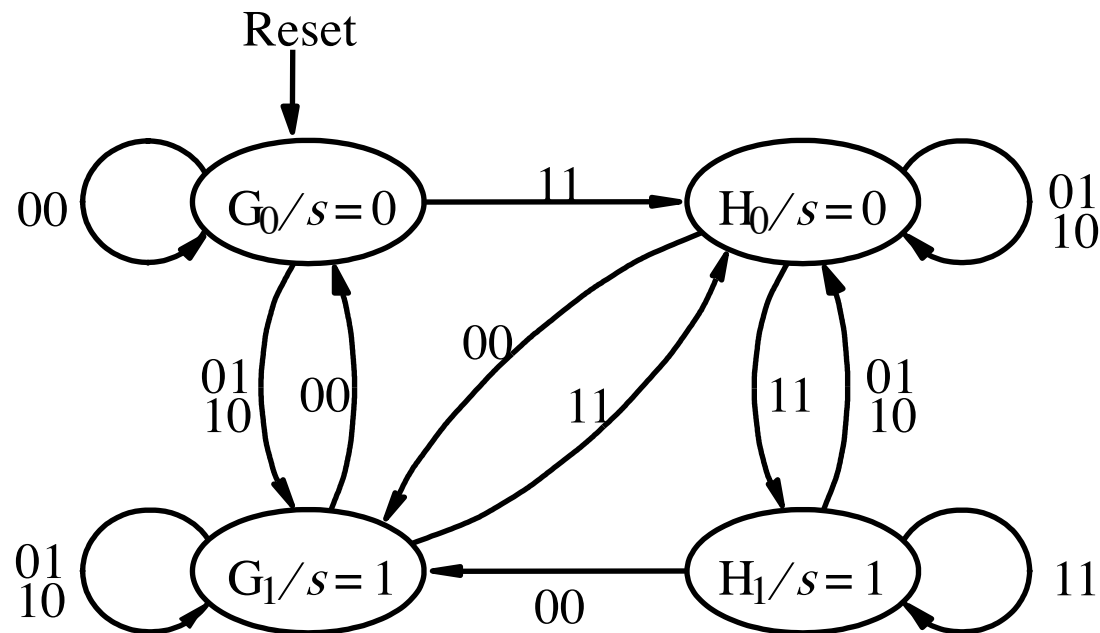


c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

[Figure 6.44 from the textbook]

State table for the Moore-type serial adder FSM

Present state	Nextstate				Output s
	$ab = 00$	01	10	11	
G_0	G_0	G_1	G_1	H_0	0
G_1	G_0	G_1	G_1	H_0	1
H_0	G_1	H_0	H_0	H_1	0
H_1	G_1	H_0	H_0	H_1	1



[Figure 6.45 from the textbook]

State table for the Moore-type serial adder FSM

Present state	Nextstate				Output s
	$ab = 00$	01	10	11	
G_0	G_0	G_1	G_1	H_0	0
G_1	G_0	G_1	G_1	H_0	1
H_0	G_1	H_0	H_0	H_1	0
H_1	G_1	H_0	H_0	H_1	1

State table for the Moore-type serial adder FSM

Present state	Nextstate				Output s
	$ab = 00$	01	10	11	
G_0	G_0	G_1	G_1	H_0	0
G_1	G_0	G_1	G_1	H_0	1
H_0	G_1	H_0	H_0	H_1	0
H_1	G_1	H_0	H_0	H_1	1

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_2Y_1				
00					
01					
10					
11					

State table for the Moore-type serial adder FSM

Present state	Nextstate				Output <i>s</i>
	<i>ab</i> = 00	01	10	11	
G ₀	G ₀	G ₁	G ₁	H ₀	0
G ₁	G ₀	G ₁	G ₁	H ₀	1
H ₀	G ₁	H ₀	H ₀	H ₁	0
H ₁	G ₁	H ₀	H ₀	H ₁	1

Present state <i>y₂y₁</i>	Nextstate				Output <i>s</i>
	<i>ab</i> = 00	01	10	11	
	<i>Y₂Y₁</i>				
00	00	01	01	10	0
01	00	01	01	10	1
10	01	10	10	11	0
11	01	10	10	11	1

State-assigned table for the Moore-type serial adder FSM

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_2Y_1				
00	00	01	01	10	0
01	00	01	01	10	1
10	01	10	10	11	0
11	01	10	10	11	1

Deriving Y1, Y2, and s

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_2Y_1				
00	00	01	01	10	0
01	00	01	01	10	1
10	01	10	10	11	0
11	01	10	10	11	1

y_2	y_1	a	b	Y_1	Y_2
0	0	0	0		
0	0	0	1		
0	0	1	0		
0	0	1	1		
0	1	0	0		
0	1	0	1		
0	1	1	0		
0	1	1	1		
1	0	0	0		
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		

Deriving Y_1

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	<div style="display: flex; align-items: center; justify-content: center;"> <div style="width: 15px; height: 15px; background-color: #ADD8E6; margin-right: 5px;"></div> Y_1 </div>				
00	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 0	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 1	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 1	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 0	0
01	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 0	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 1	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 1	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 0	1
10	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 1	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 0	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 0	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 1	0
11	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 1	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 0	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 0	<div style="width: 15px; height: 15px; background-color: #ADD8E6; display: inline-block;"></div> 1	1

y_2	y_1	a	b	Y_1	Y_2
0	0	0	0		
0	0	0	1		
0	0	1	0		
0	0	1	1		
0	1	0	0		
0	1	0	1		
0	1	1	0		
0	1	1	1		
1	0	0	0		
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		

Deriving Y_1

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	□ Y_1				
00	□ 0	□ 1	□ 1	□ 0	0
01	□ 0	□ 1	□ 1	□ 0	1
10	□ 1	□ 0	□ 0	□ 1	0
11	□ 1	□ 0	□ 0	□ 1	1

y_2	y_1	a	b	Y_1	Y_2
0	0	0	0	0	
0	0	0	1	1	
0	0	1	0	1	
0	0	1	1	0	
0	1	0	0	0	
0	1	0	1	1	
0	1	1	0	1	
0	1	1	1	0	
1	0	0	0	1	
1	0	0	1	0	
1	0	1	0	0	
1	0	1	1	1	
1	1	0	0	1	
1	1	0	1	0	
1	1	1	0	0	
1	1	1	1	1	

Deriving Y_1

Present state y_2y_1	Nextstate				Output s
	$ab=00$	01	10	11	
	Y_1				
00	0	1	1	0	0
01	0	1	1	0	1
10	1	0	0	1	0
11	1	0	0	1	1

y_2y_1	ab			
	00	01	11	10
00				
01				
11				
10				

y_2	y_1	a	b	Y_1	Y_2
0	0	0	0	0	
0	0	0	1	1	
0	0	1	0	1	
0	0	1	1	0	
0	1	0	0	0	
0	1	0	1	1	
0	1	1	0	1	
0	1	1	1	0	
1	0	0	0	1	
1	0	0	1	0	
1	0	1	0	0	
1	0	1	1	1	
1	1	0	0	1	
1	1	0	1	0	
1	1	1	0	0	
1	1	1	1	1	

Deriving Y_1

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_1				
00	0	1	1	0	0
01	0	1	1	0	1
10	1	0	0	1	0
11	1	0	0	1	1

y_2y_1	ab			
	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	0	0	1	1
10	1	1	0	0

y_2	y_1	a	b	Y_1	Y_2
0	0	0	0	0	
0	0	0	1	1	
0	0	1	0	1	
0	0	1	1	0	
0	1	0	0	0	
0	1	0	1	1	
0	1	1	0	1	
0	1	1	1	0	
1	0	0	0	1	
1	0	0	1	0	
1	0	1	0	0	
1	0	1	1	1	
1	1	0	0	1	
1	1	0	1	0	
1	1	1	0	0	
1	1	1	1	1	

Deriving Y_1

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_1				
00	0	1	1	0	0
01	0	1	1	0	1
10	1	0	0	1	0
11	1	0	0	1	1

y_2y_1	ab			
	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	0	0	1	1
10	1	1	0	0

$$Y_1 = a \oplus b \oplus y_2$$

y_2	y_1	a	b	Y_1	Y_2
0	0	0	0	0	
0	0	0	1	1	
0	0	1	0	1	
0	0	1	1	0	
0	1	0	0	0	
0	1	0	1	1	
0	1	1	0	1	
0	1	1	1	0	
1	0	0	0	1	
1	0	0	1	0	
1	0	1	0	0	
1	0	1	1	1	
1	1	0	0	1	
1	1	0	1	0	
1	1	1	0	0	
1	1	1	1	1	

Deriving Y_2

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_2 				
00	0 	0 	0 	1 	0
01	0 	0 	0 	1 	1
10	0 	1 	1 	1 	0
11	0 	1 	1 	1 	1

y_2	y_1	a	b	Y_1	Y_2
0	0	0	0	0	
0	0	0	1	1	
0	0	1	0	1	
0	0	1	1	0	
0	1	0	0	0	
0	1	0	1	1	
0	1	1	0	1	
0	1	1	1	0	
1	0	0	0	1	
1	0	0	1	0	
1	0	1	0	0	
1	0	1	1	1	
1	1	0	0	1	
1	1	0	1	0	
1	1	1	0	0	
1	1	1	1	1	

Deriving Y_2

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_2 				
00	0 	0 	0 	1 	0
01	0 	0 	0 	1 	1
10	0 	1 	1 	1 	0
11	0 	1 	1 	1 	1

y_2	y_1	a	b	Y_1	Y_2
0	0	0	0	0	0
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	0	1
0	1	0	0	0	0
0	1	0	1	1	0
0	1	1	0	1	0
0	1	1	1	0	1
1	0	0	0	1	0
1	0	0	1	0	1
1	0	1	0	0	1
1	0	1	1	1	1
1	1	0	0	1	0
1	1	0	1	0	1
1	1	1	0	0	1
1	1	1	1	1	1

Deriving Y_2

Present state y_2y_1	Nextstate				Output s
	$ab=00$	01	10	11	
	Y_2 				
00	0 	0 	0 	1 	0
01	0 	0 	0 	1 	1
10	0 	1 	1 	1 	0
11	0 	1 	1 	1 	1

		y_2y_1			
	ab	00	01	11	10
00					
01					
11					
10					

y_2	y_1	a	b	Y_1	Y_2
0	0	0	0	0	0
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	0	1
0	1	0	0	0	0
0	1	0	1	1	0
0	1	1	0	1	0
0	1	1	1	0	1
1	0	0	0	1	0
1	0	0	1	0	1
1	0	1	0	0	1
1	0	1	1	1	1
1	1	0	0	1	0
1	1	0	1	0	1
1	1	1	0	0	1
1	1	1	1	1	1






Deriving Y_2

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_2 				
00	0 	0 	0 	1 	0
01	0 	0 	0 	1 	1
10	0 	1 	1 	1 	0
11	0 	1 	1 	1 	1

		y_2y_1			
		00	01	11	10
ab	00	0	0	0	0
	01	0	0	1	1
	11	1	1	1	1
	10	0	0	1	1

y_2	y_1	a	b	Y_1	Y_2
0	0	0	0	0	0
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	0	1
0	1	0	0	0	0
0	1	0	1	1	0
0	1	1	0	1	0
0	1	1	1	0	1
1	0	0	0	1	0
1	0	0	1	0	1
1	0	1	0	0	1
1	0	1	1	1	1
1	1	0	0	1	0
1	1	0	1	0	1
1	1	1	0	0	1
1	1	1	1	1	1

Deriving Y_2

Present state y_2y_1	Nextstate				Output s
	$ab=00$	01	10	11	
	Y_2 				
00	0 	0 	0 	1 	0
01	0	0	0	1	1
10	0	1	1	1	0
11	0	1	1	1	1

		y_2y_1			
		00	01	11	10
ab	00	0	0	0	0
	01	0	0	1	1
	11	1	1	1	1
	10	0	0	1	1

y_2	y_1	a	b	Y_1	Y_2
0	0	0	0	0	0
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	0	1
0	1	0	0	0	0
0	1	0	1	1	0
0	1	1	0	1	0
0	1	1	1	0	1
1	0	0	0	1	0
1	0	0	1	0	1
1	0	1	0	0	1
1	0	1	1	1	1
1	1	0	0	1	0
1	1	0	1	0	1
1	1	1	0	0	1
1	1	1	1	1	1

Deriving Y_2

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_2				
00	0	0	0	1	0
01	0	0	0	1	1
10	0	1	1	1	0
11	0	1	1	1	1

ab	y_2y_1			
	00	01	11	10
00	0	0	0	0
01	0	0	1	1
11	1	1	1	1
10	0	0	1	1

$$Y_2 = ab + ay_2 + by_2$$

y_2	y_1	a	b	Y_1	Y_2
0	0	0	0	0	0
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	0	1
0	1	0	0	0	0
0	1	0	1	1	0
0	1	1	0	1	0
0	1	1	1	0	1
1	0	0	0	1	0
1	0	0	1	0	1
1	0	1	0	0	1
1	0	1	1	1	1
1	1	0	0	1	0
1	1	0	1	0	1
1	1	1	0	0	1
1	1	1	1	1	1

Deriving s

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_2Y_1				
00	00	01	01	10	0
01	00	01	01	10	1
10	01	10	10	11	0
11	01	10	10	11	1

y_2	y_1	s
0	0	
0	1	
1	0	
1	1	

Deriving s

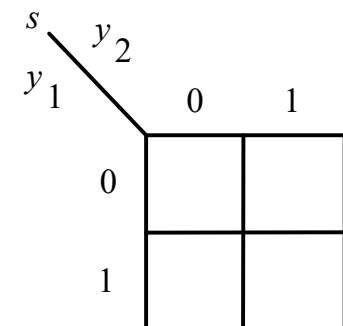
Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_2Y_1				
00	00	01	01	10	0
01	00	01	01	10	1
10	01	10	10	11	0
11	01	10	10	11	1

y_2	y_1	s
0	0	0
0	1	1
1	0	0
1	1	1

Deriving s

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_2Y_1				
00	0 0	01	0 1	10	0
01	0 0	01	0 1	10	1
10	0 1	10	1 0	11	0
11	0 1	10	1 0	11	1

y_2	y_1	s
0	0	0
0	1	1
1	0	0
1	1	1



Deriving s

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_2Y_1				
00	00	01	01	10	0
01	00	01	01	10	1
10	01	10	10	11	0
11	01	10	10	11	1

y_2	y_1	s
0	0	0
0	1	1
1	0	0
1	1	1

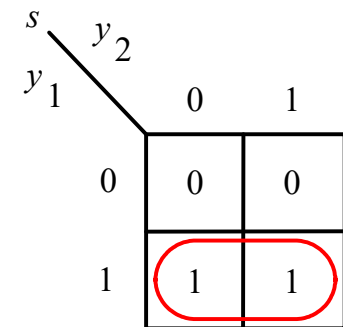
		y_2	
		0	1
y_1	0	0	0
	1	1	1

The output s is 1 for the input combinations $(y_2, y_1) = (0, 1)$ and $(1, 1)$, which are circled in red in the diagram.

Deriving s

Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_2Y_1				
00	00	01	01	10	0
01	00	01	01	10	1
10	01	10	10	11	0
11	01	10	10	11	1

y_2	y_1	s
0	0	0
0	1	1
1	0	0
1	1	1



$$s = y_1$$

State-assigned table for the Moore-type serial adder FSM

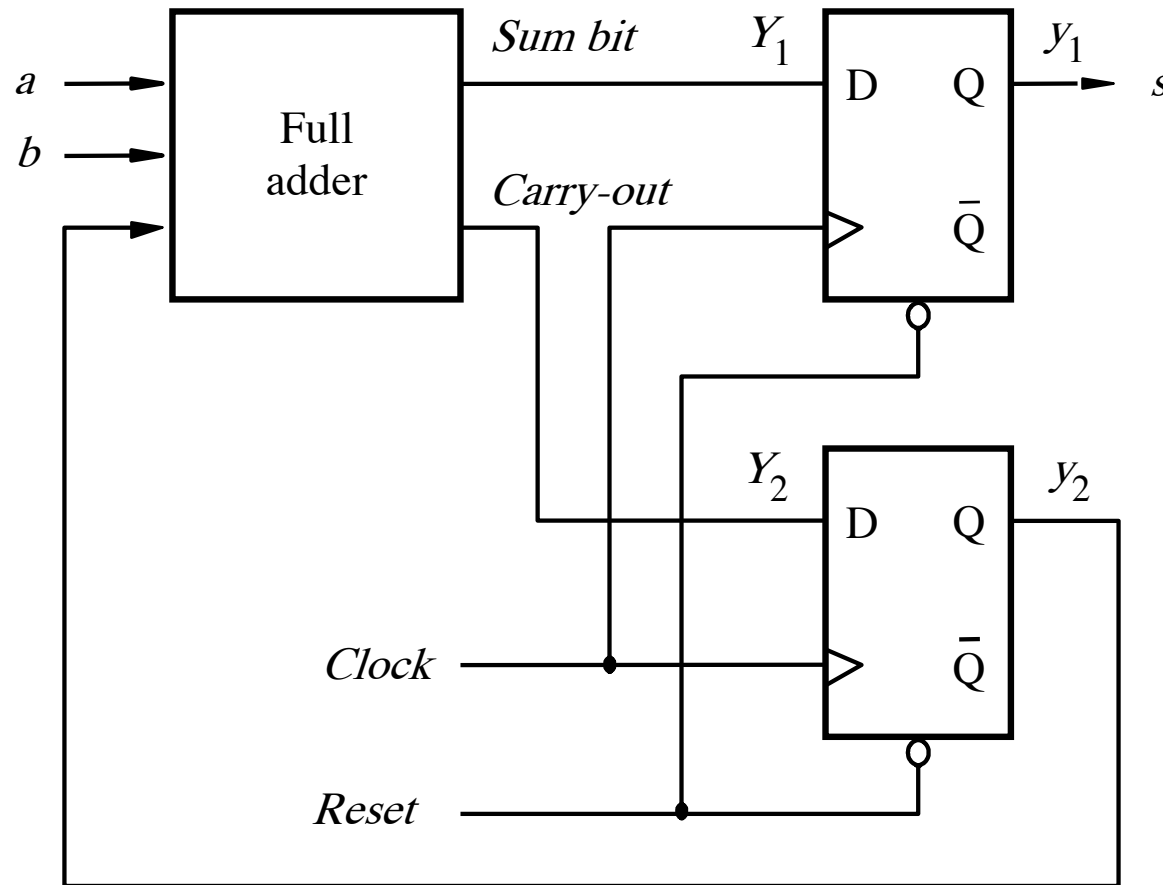
Present state y_2y_1	Nextstate				Output s
	$ab = 00$	01	10	11	
	Y_2Y_1				
00	00	01	01	10	0
01	00	01	01	10	1
10	01	10	10	11	0
11	01	10	10	11	1

$$Y_1 = a \oplus b \oplus y_2$$

$$Y_2 = ab + ay_2 + by_2$$

$$s = y_1$$

Circuit for the Moore-type serial adder FSM

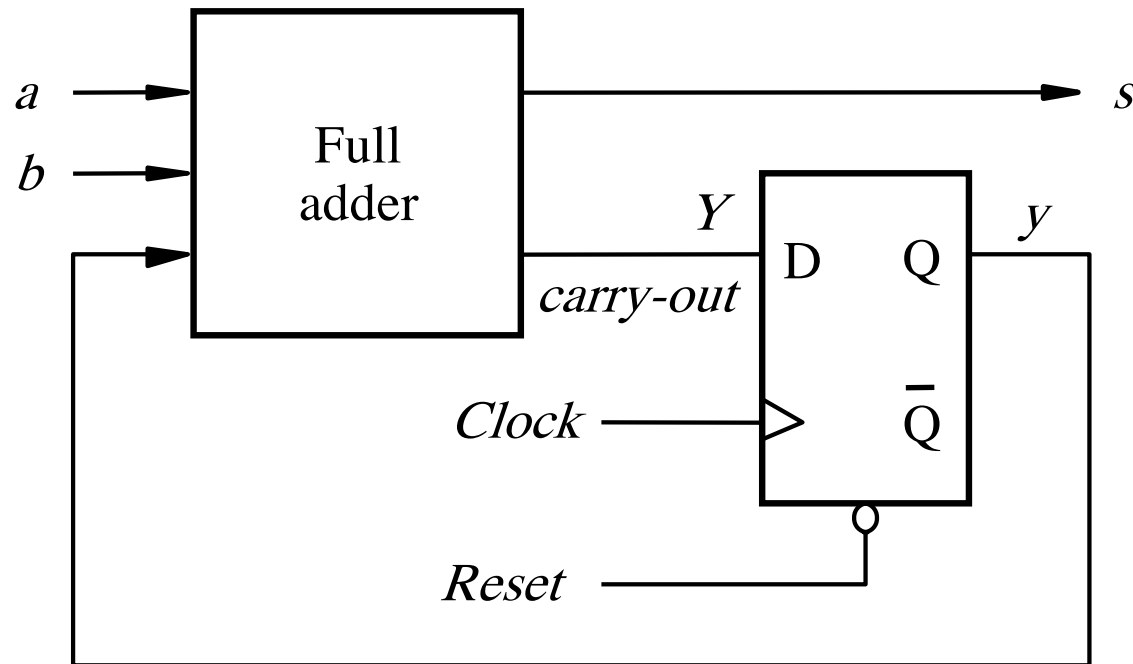


$$Y_1 = a \oplus b \oplus y_2 \quad (\text{sum bit from FA})$$

$$Y_2 = ab + ay_2 + by_2 \quad (\text{carry bit from FA})$$

$$s = y_1$$

Circuit for the Mealy-type serial adder FSM



$$s = \text{XOR}(\text{XOR}(a, b), y) \quad (\text{sum bit from FA})$$

$$Y = ab + ay + by \quad (\text{carry bit from FA})$$

Questions?

THE END