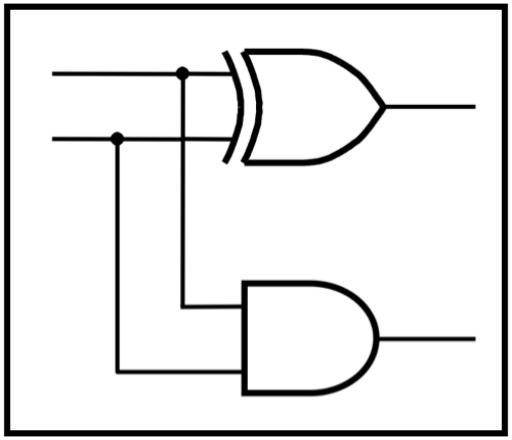


CprE 281: Digital Logic

Instructor: Alexander Stoytchev

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

Serial Adder

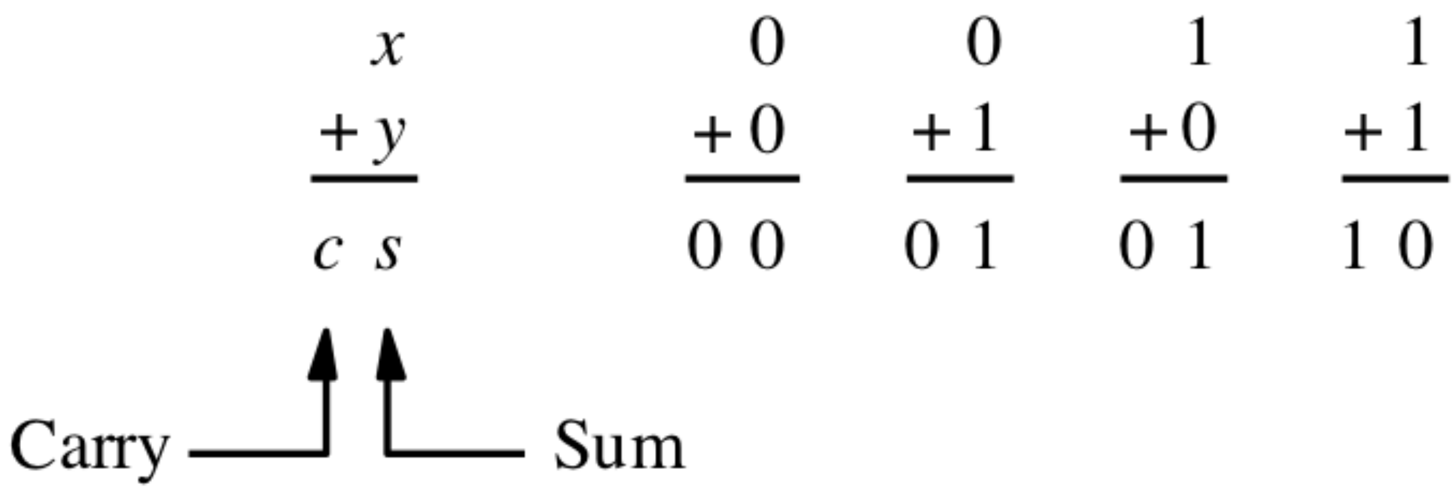
CprE 281: Digital Logic
Iowa State University, Ames, IA

Administrative Stuff

- **Homework 11 is due on Monday Nov 18 @ 10pm**

Quick Review

Adding two bits
(there are four possible cases)



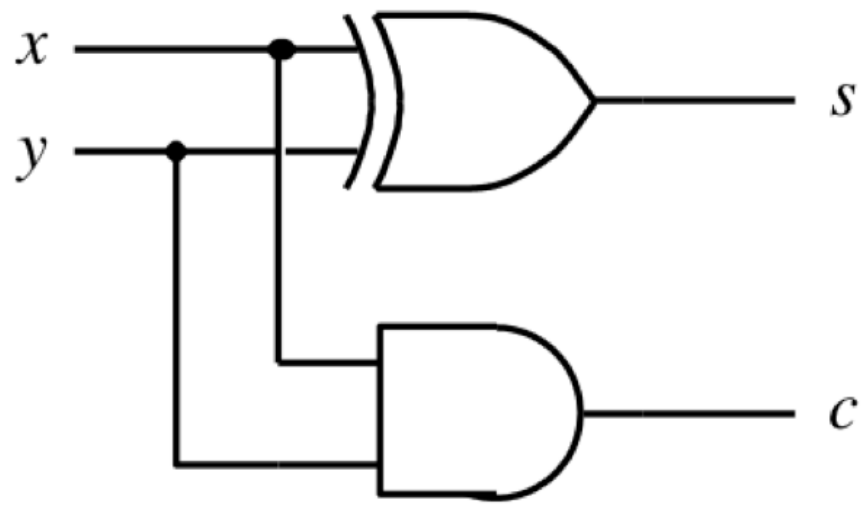
[Figure 3.1a from the textbook]

Adding two bits (the truth table)

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

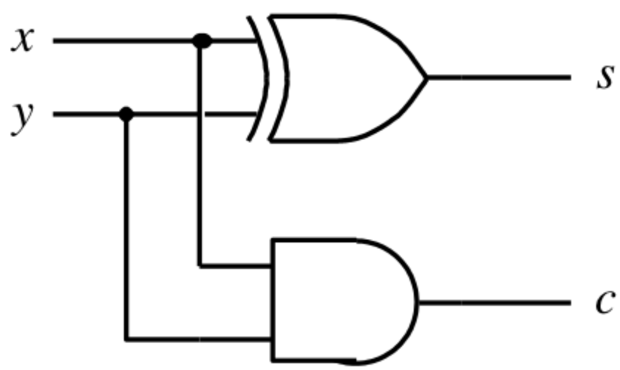
[Figure 3.1b from the textbook]

Adding two bits
(the logic circuit)

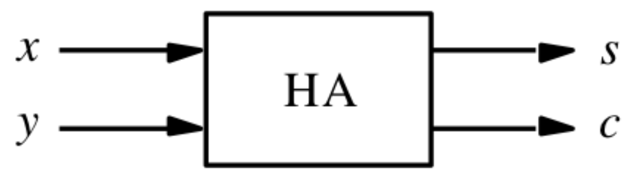


[Figure 3.1c from the textbook]

The Half-Adder



(c) Circuit

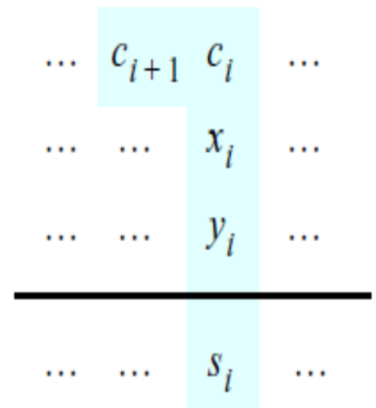


(d) Graphical symbol

[Figure 3.1c-d from the textbook]

Addition of multibit numbers

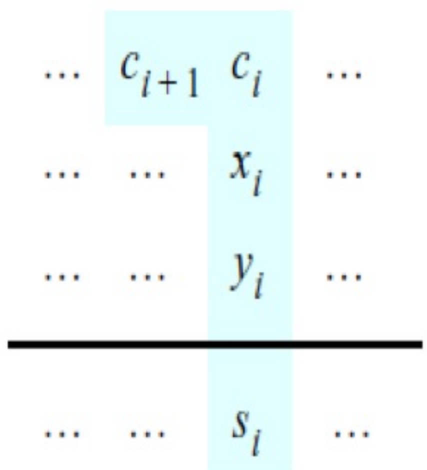
Generated carries →	1 1 1 0	
$X = x_4 x_3 x_2 x_1 x_0$	0 1 1 1 1	$(15)_{10}$
$+ Y = y_4 y_3 y_2 y_1 y_0$	+ 0 1 0 1 0	$+ (10)_{10}$
$S = s_4 s_3 s_2 s_1 s_0$	1 1 0 0 1	$(25)_{10}$



Bit position i

[Figure 3.2 from the textbook]

Problem Statement and Truth Table



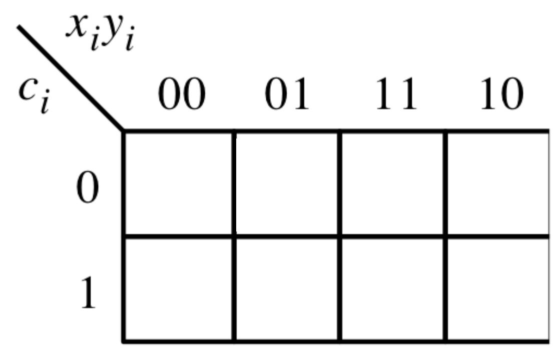
[Figure 3.2b from the textbook]

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

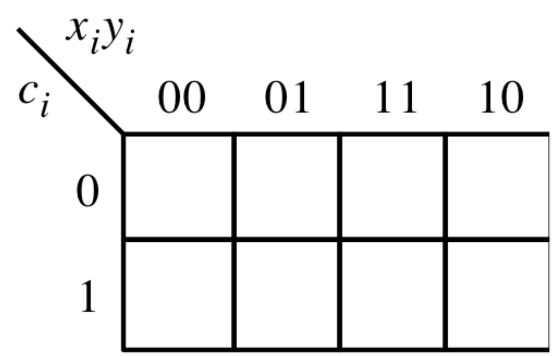
[Figure 3.3a from the textbook]

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

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	

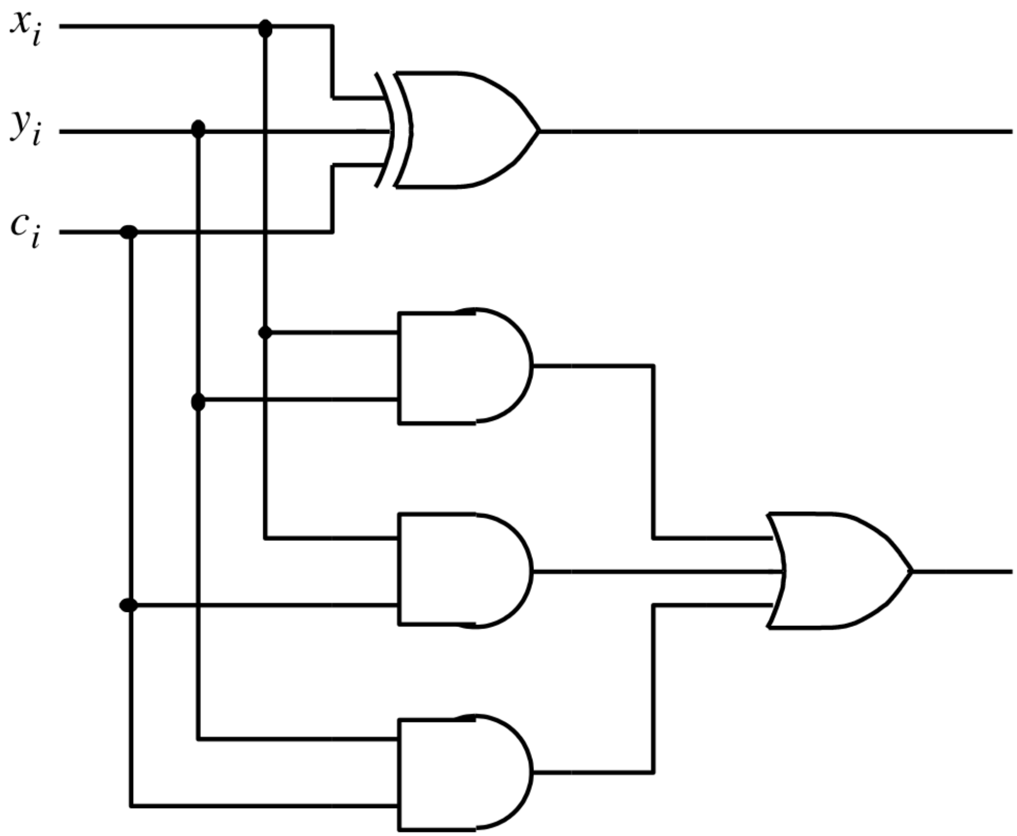
$$s_i = x_i \oplus y_i \oplus c_i$$

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]

The circuit for the two expressions

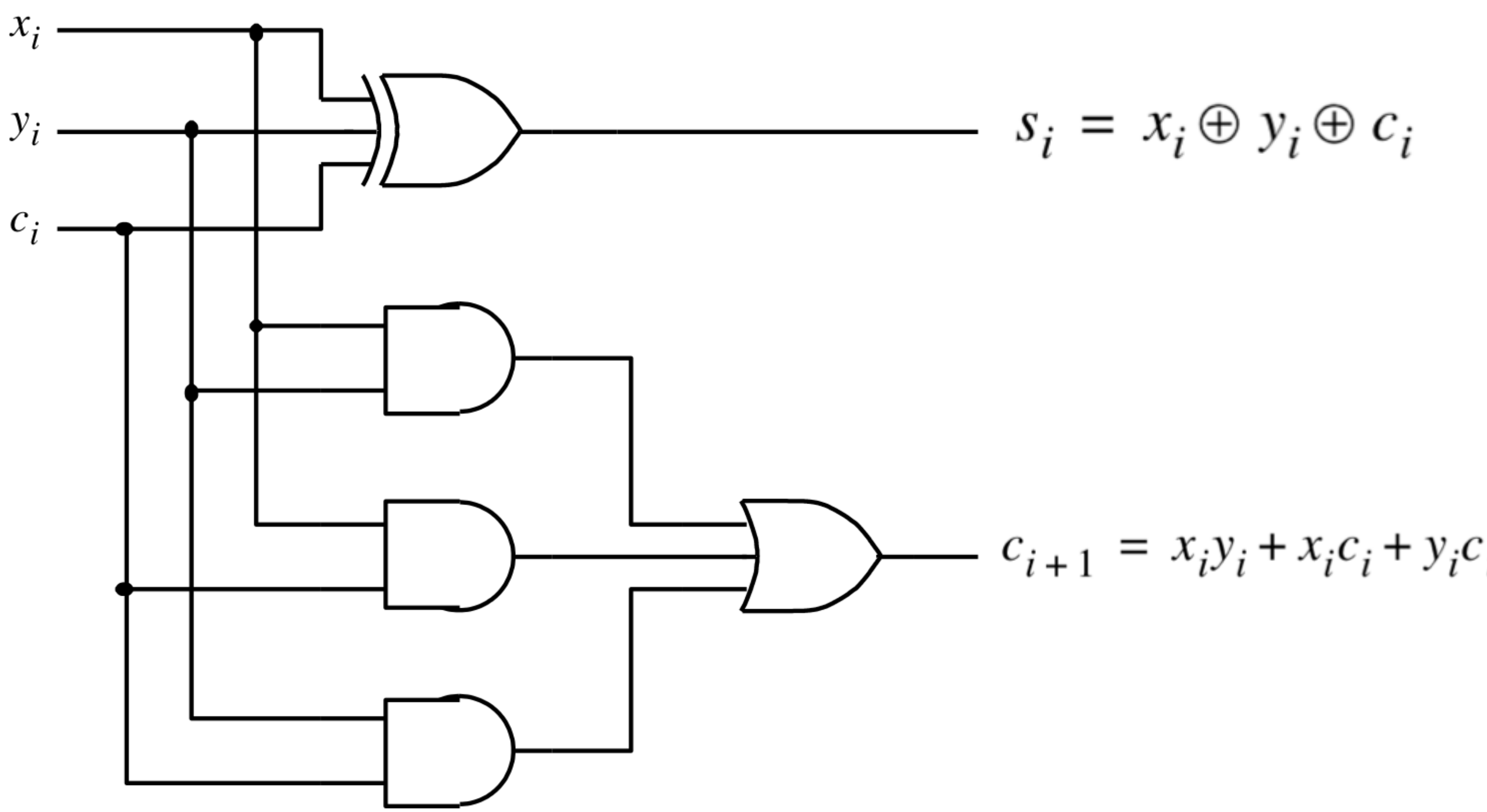


$$s_i = x_i \oplus y_i \oplus c_i$$

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

[Figure 3.3c from the textbook]

This is called the Full-Adder



[Figure 3.3c from the textbook]

XOR Magic

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

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?

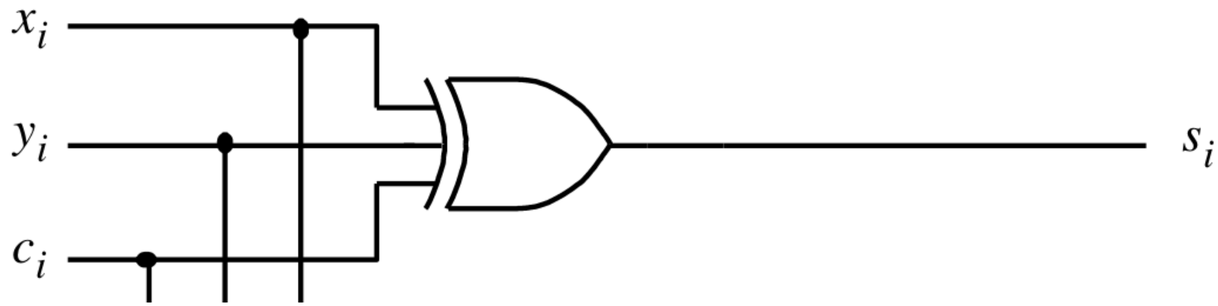
$$s_i = (\bar{x}_i y_i + x_i \bar{y}_i)\bar{c}_i + (\bar{x}_i \bar{y}_i + x_i y_i) c_i$$

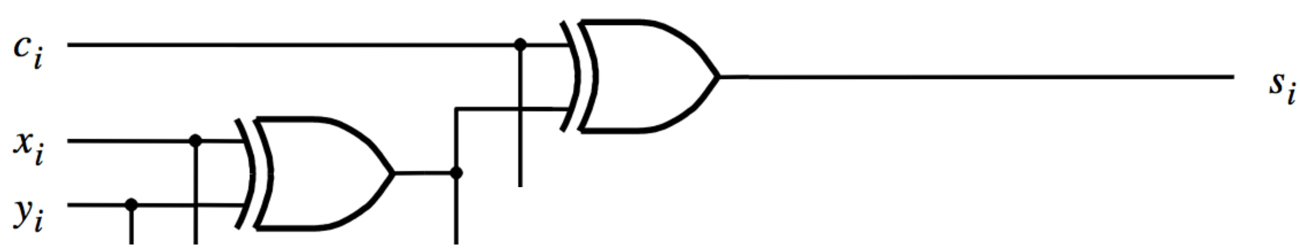
$$= (x_i \oplus y_i)\bar{c}_i + \overline{(x_i \oplus y_i)} c_i$$

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

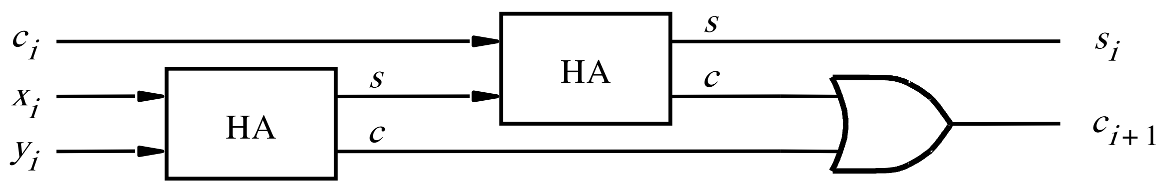
XOR Magic
(s_i can be implemented in two different ways)

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

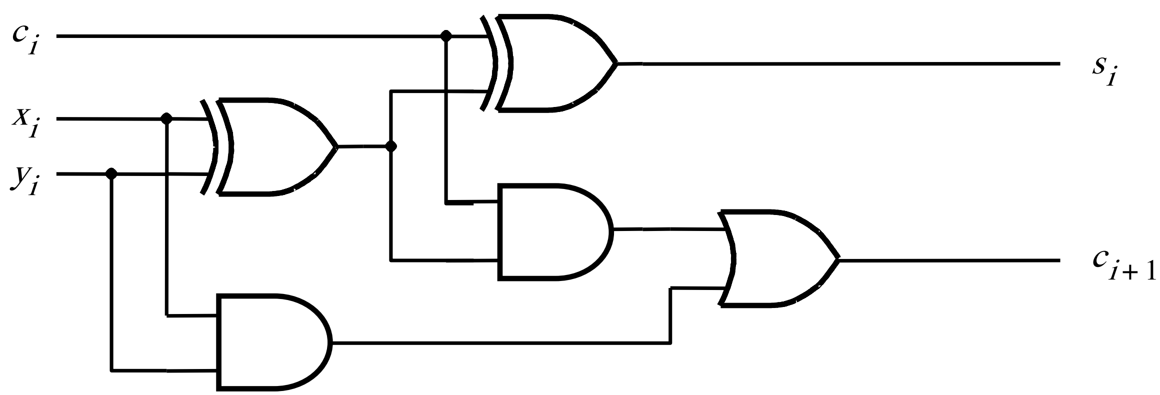




A decomposed implementation of the full-adder circuit



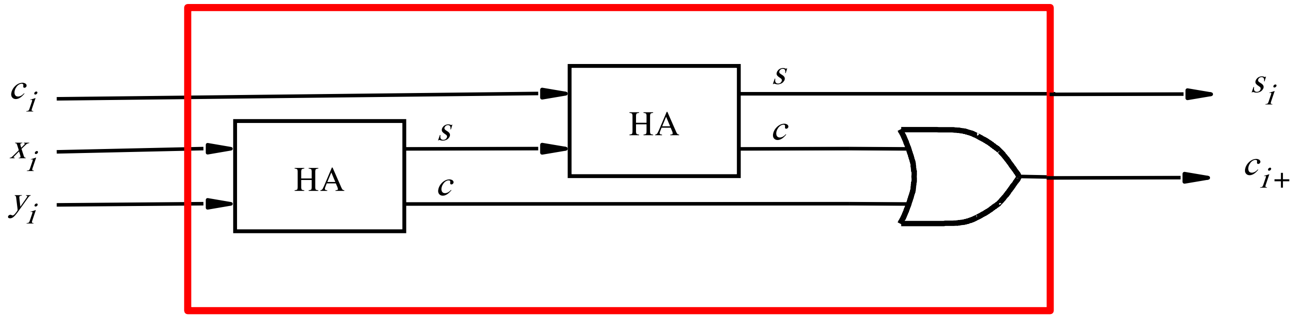
(a) Block diagram



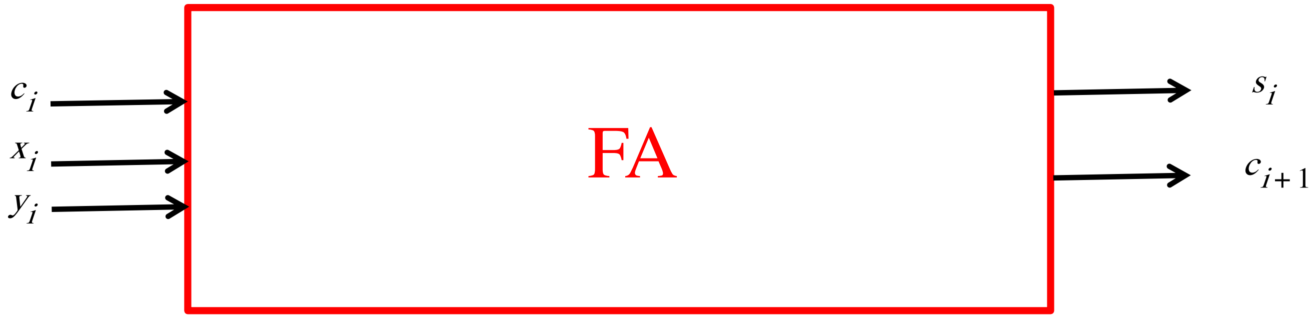
(b) Detailed diagram

[Figure 3.4 from the textbook]

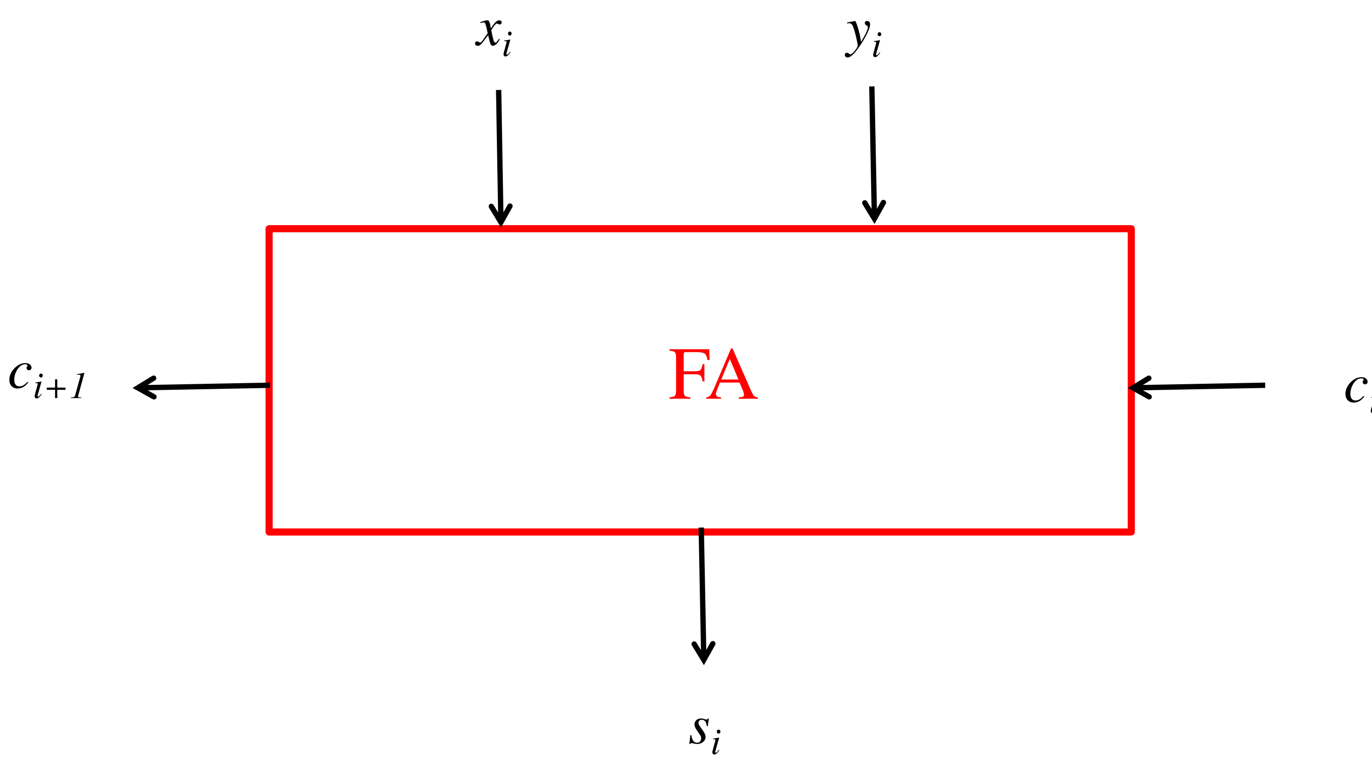
The Full-Adder Abstraction



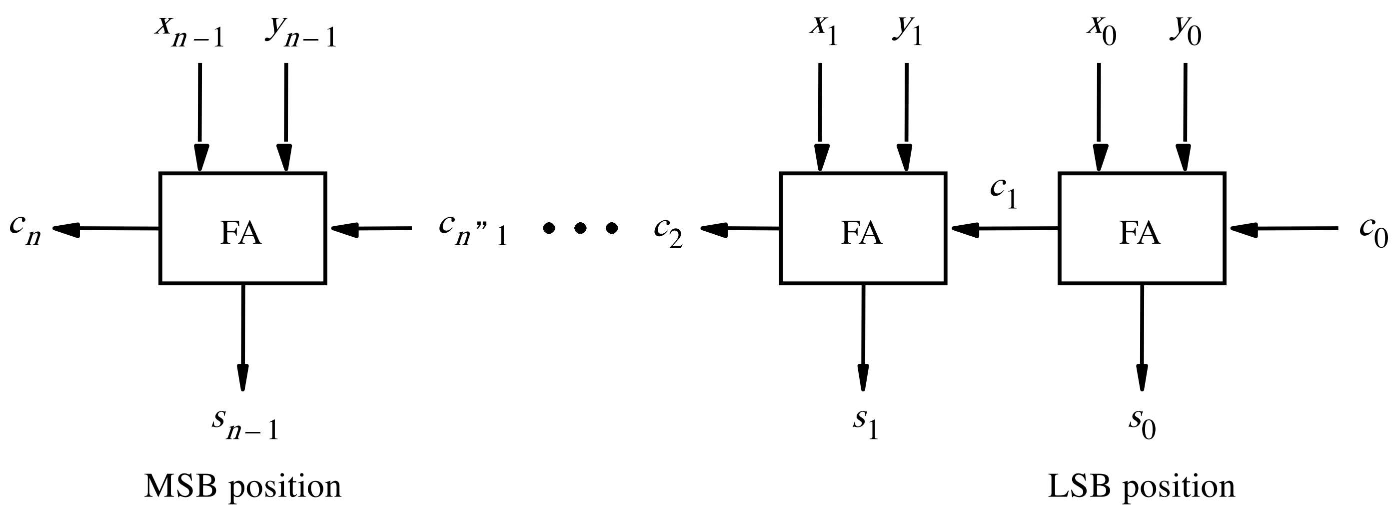
The Full-Adder Abstraction



We can place the arrows anywhere

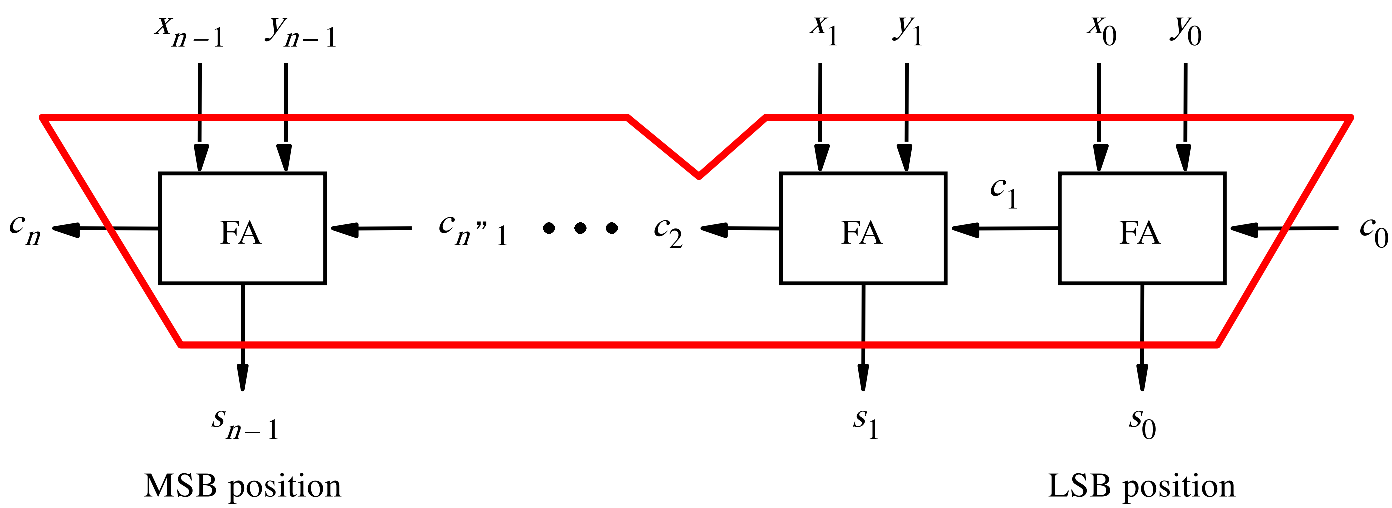


n-bit ripple-carry adder

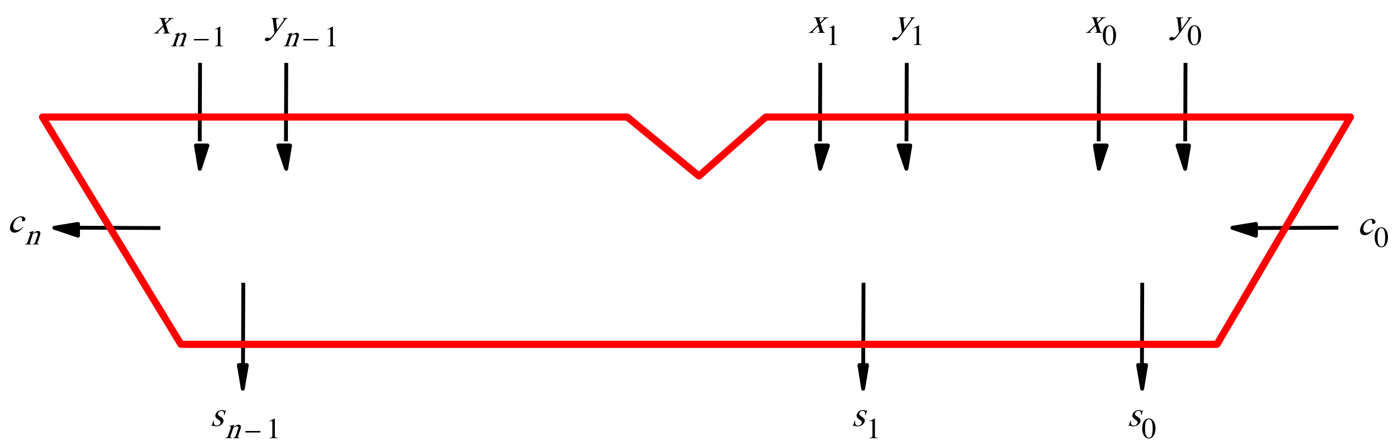


[Figure 3.5 from the textbook]

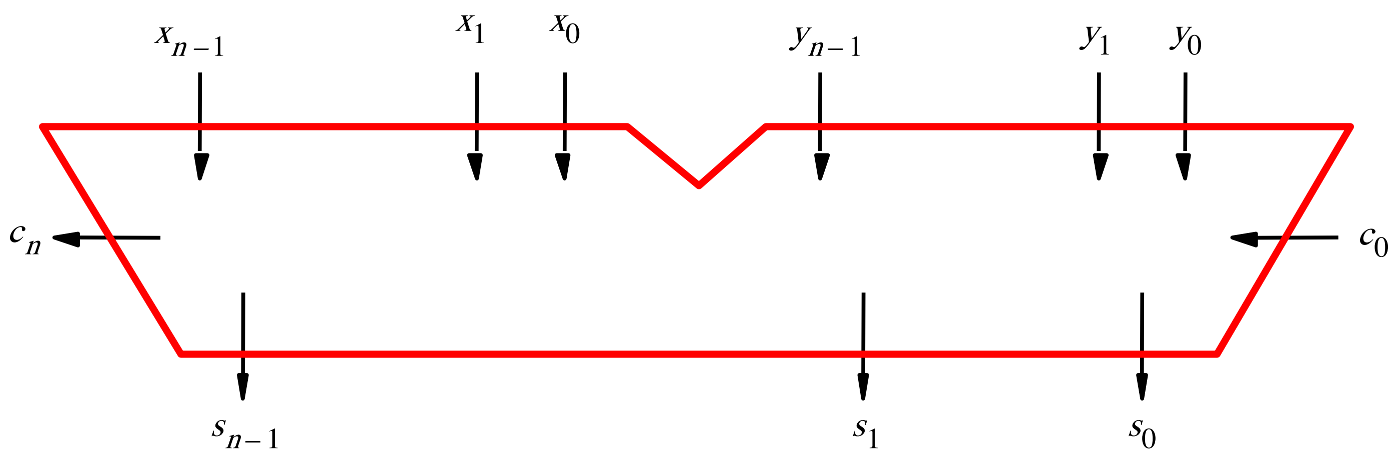
n-bit ripple-carry adder abstraction



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

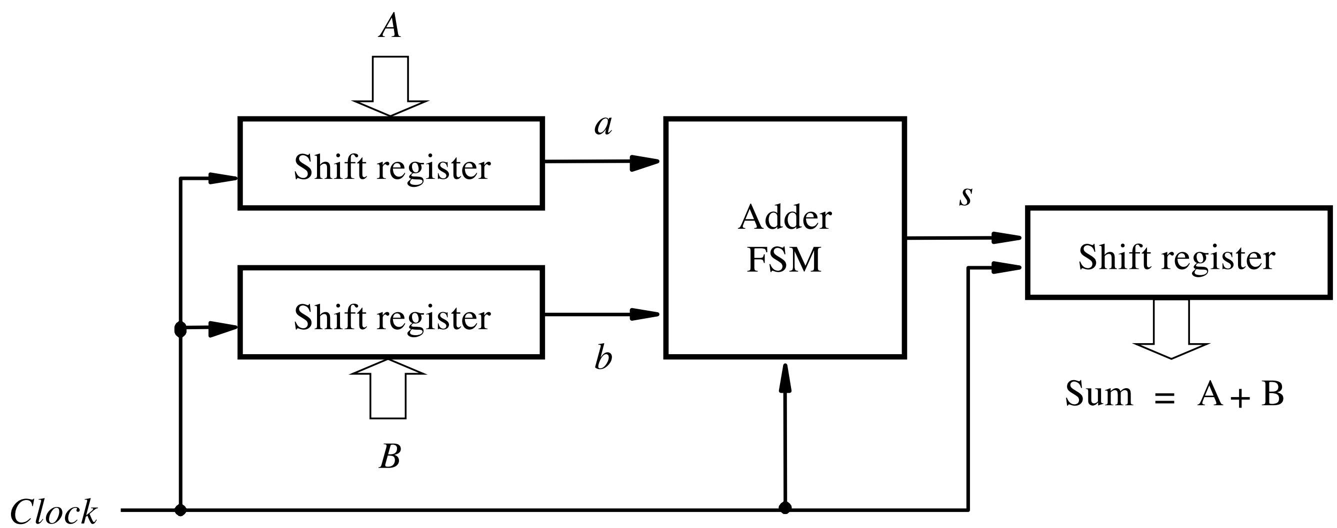
- **The n-bit adder requires all bits to be provided at the same time.**

- **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.**

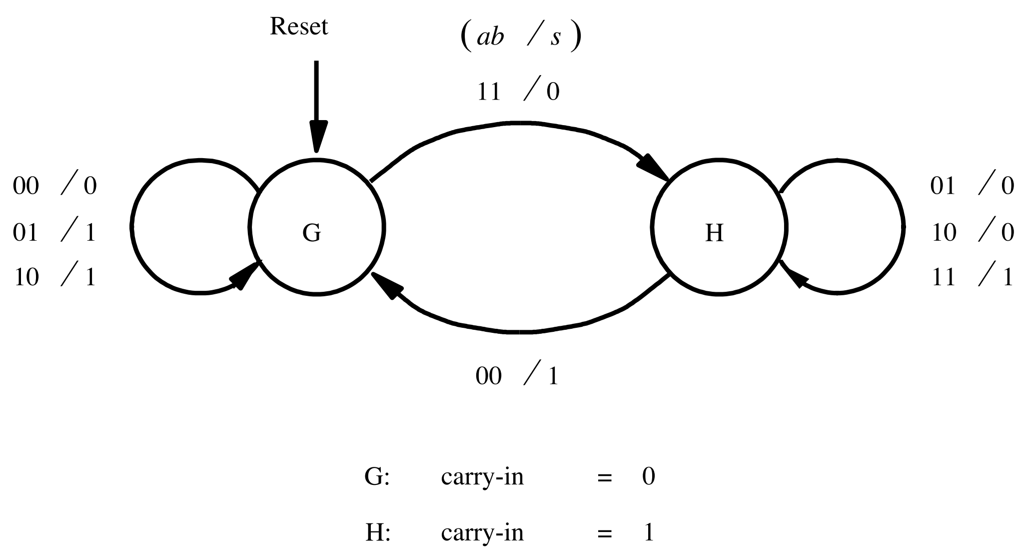
Block diagram for the serial adder



[Figure 6.39 from the textbook]

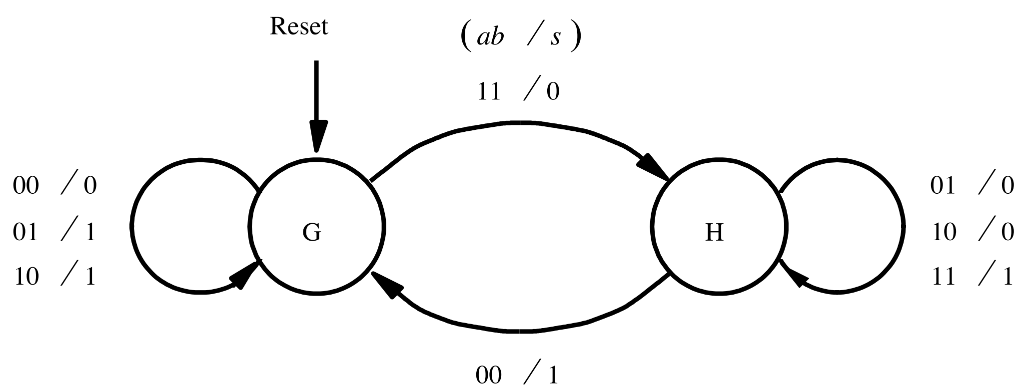
Mealy Machine Implementation

State diagram for the serial adder FSM



[Figure 6.40 from the textbook]

State diagram for the serial adder FSM



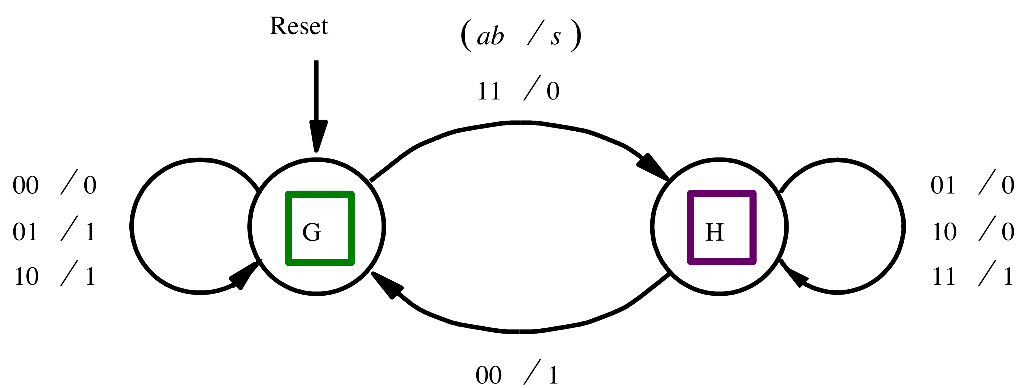
c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

G: carry-in = 0

H: carry-in = 1

[Figure 6.40 from the textbook]

State diagram for the serial adder FSM



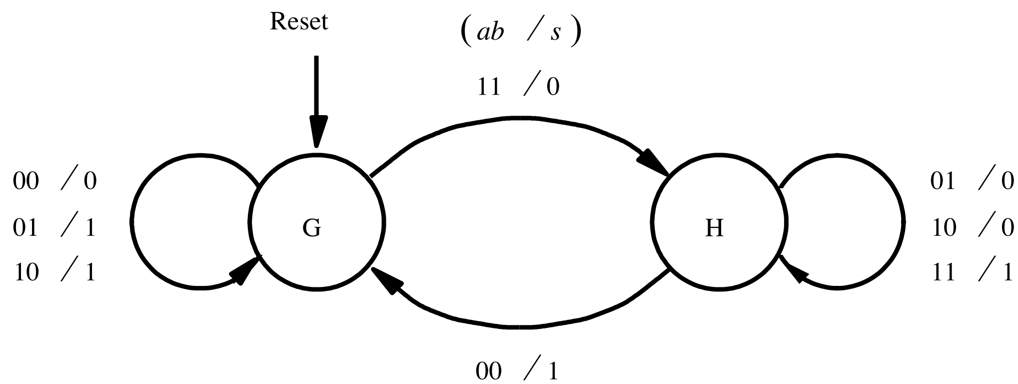
c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

G: carry-in = 0

H: carry-in = 1

[Figure 6.40 from the textbook]

State diagram for the serial adder FSM

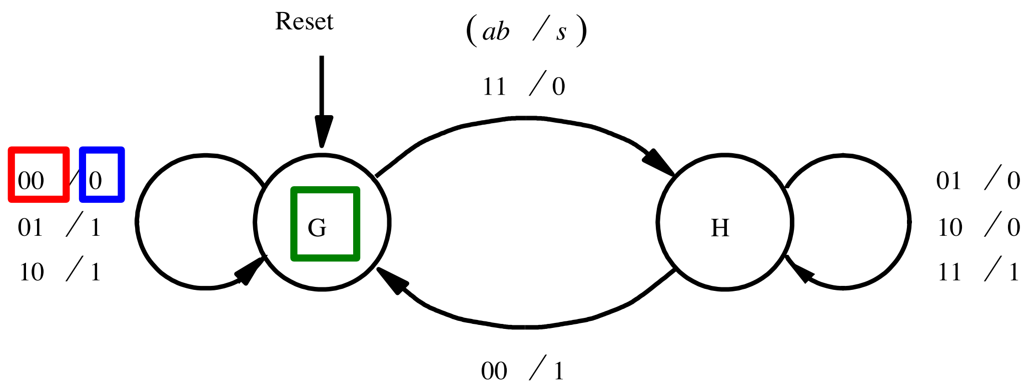


c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

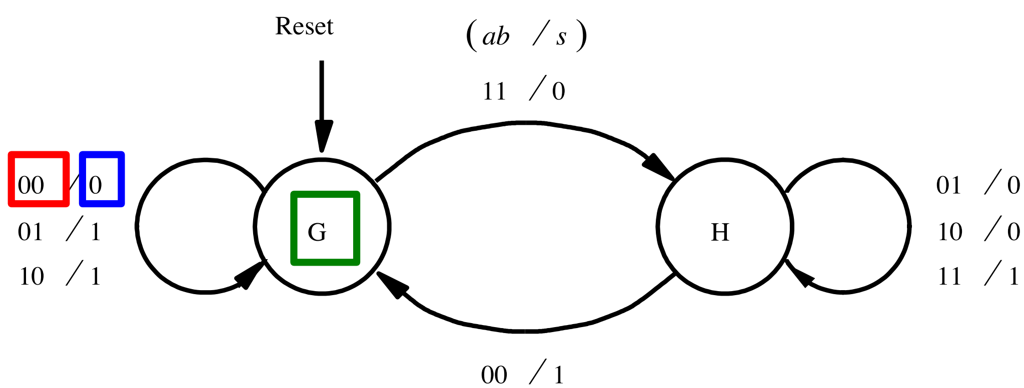


c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

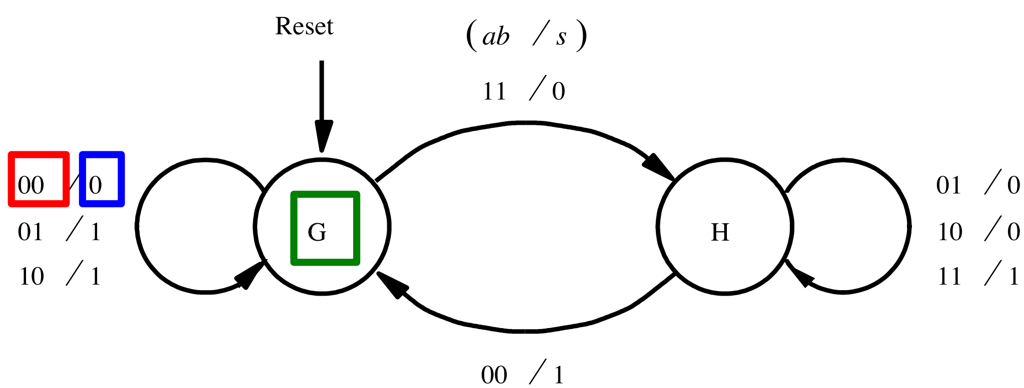


c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

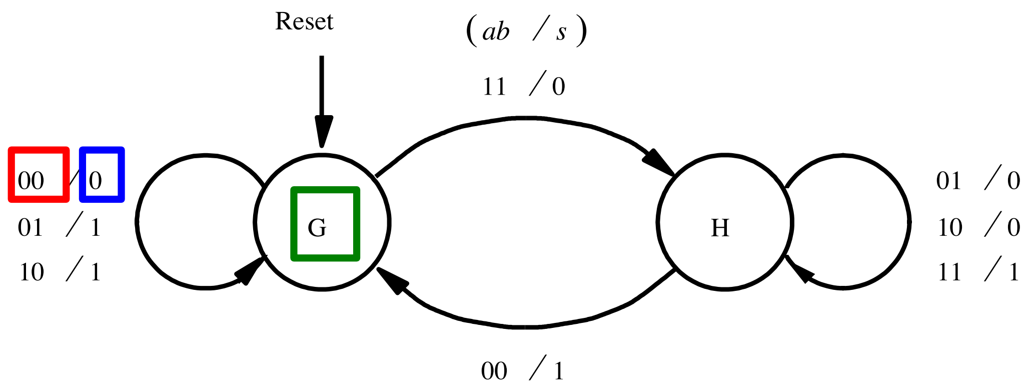


c_i	x_i	y_i	c_{i+1}	s_i
G	0	0	G	0
G	0	1	G	1
G	1	0	G	1
G	1	1	H	0
H	0	0	G	1
H	0	1	H	0
H	1	0	H	0
H	1	1	H	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

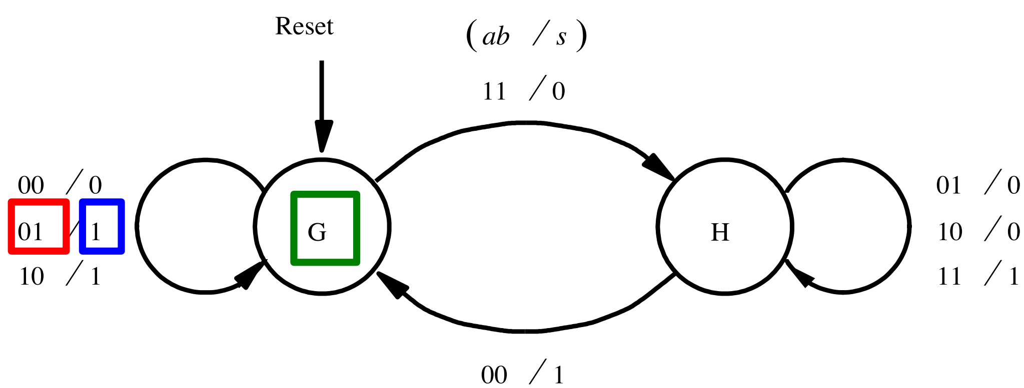


c_i	x_i	y_i	c_{i+1}	s_i
G	0	0	G	0
G	0	1	G	1
G	1	0	G	1
G	1	1	H	0
H	0	0	G	1
H	0	1	H	0
H	1	0	H	0
H	1	1	H	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

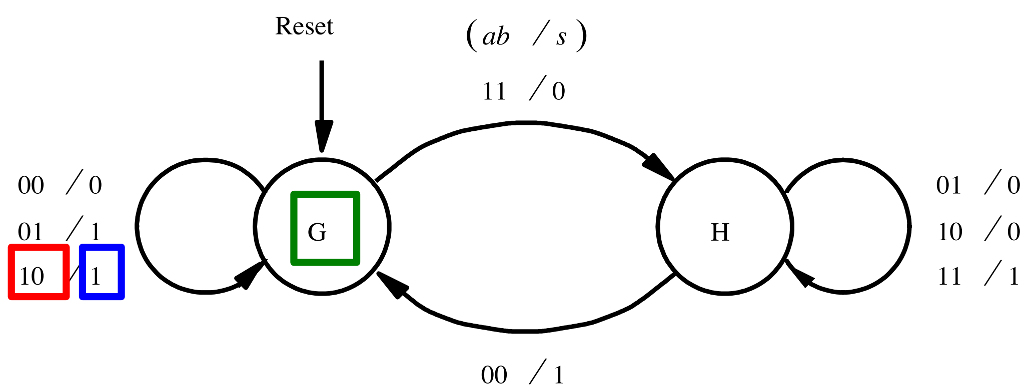


c_i	x_i	y_i	c_{i+1}	s_i
G	0	0	G	0
G	0	1	G	1
G	1	0	G	1
G	1	1	H	0
H	0	0	G	1
H	0	1	H	0
H	1	0	H	0
H	1	1	H	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

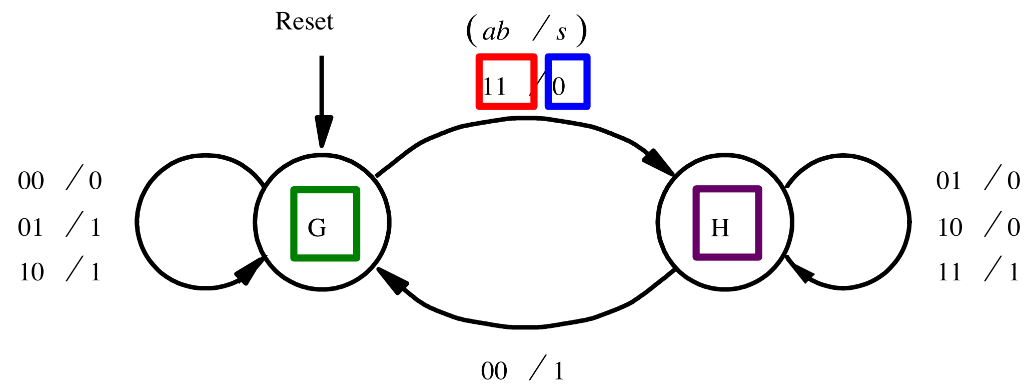


c_i	x_i	y_i	c_{i+1}	s_i
G	0	0	G	0
G	0	1	G	1
G	1	0	G	1
G	1	1	H	0
H	0	0	G	1
H	0	1	H	0
H	1	0	H	0
H	1	1	H	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

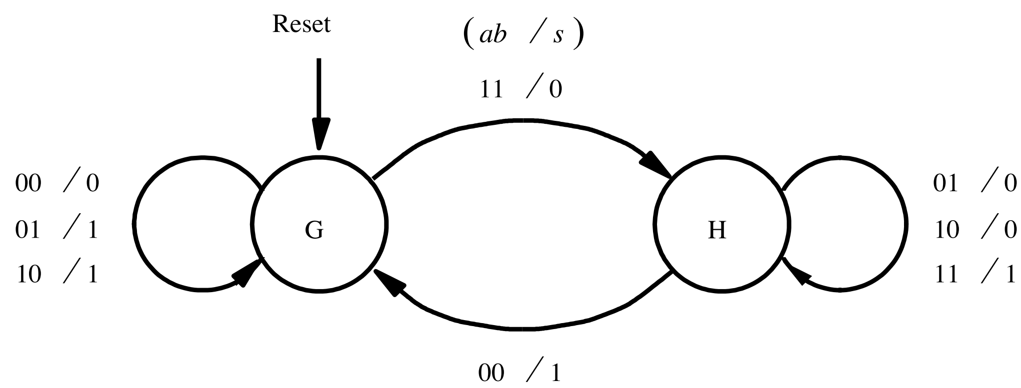


c_i	x_i	y_i	c_{i+1}	s_i
G	0	0	G	0
G	0	1	G	1
G	1	0	G	1
G	1	1	H	0
H	0	0	G	1
H	0	1	H	0
H	1	0	H	0
H	1	1	H	1

G: carry-in = 0

H: carry-in = 1

State diagram for the serial adder FSM

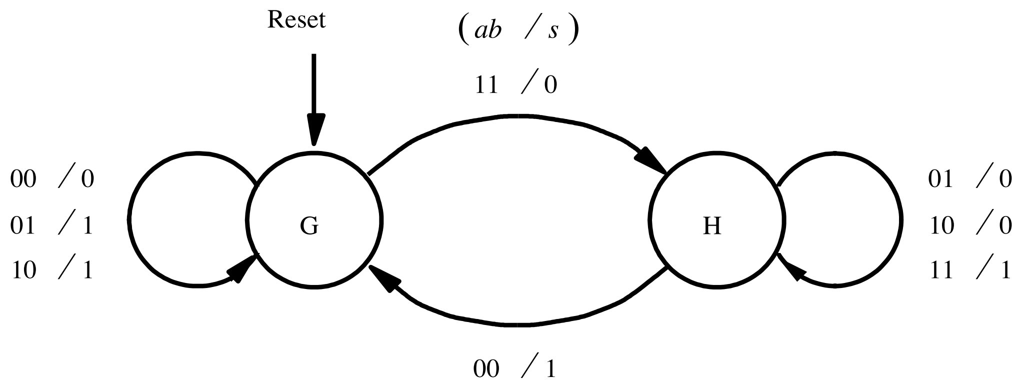


State table for the serial adder FSM

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

[Figure 6.40 & 6.41 from the textbook]

State diagram for the serial adder FSM

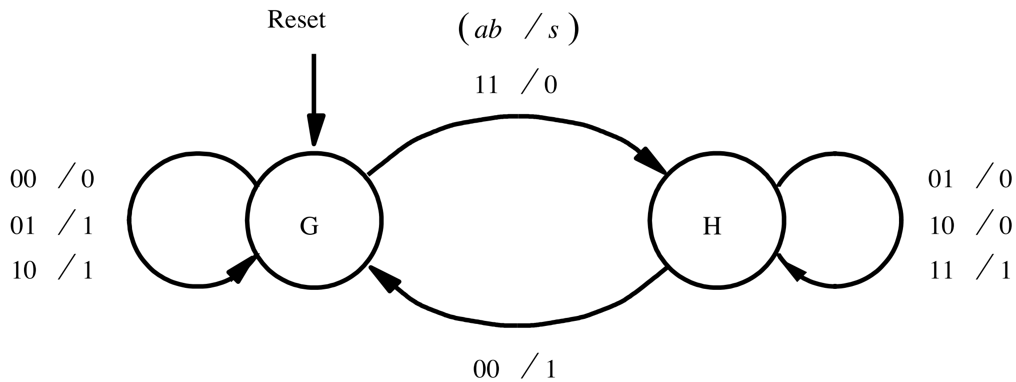


State table for the serial adder FSM

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				

[Figure 6.40 & 6.41 from the textbook]

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

[Figure 6.40 & 6.41 from the textbook]

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

[Figure 6.41 from the textbook]

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								

[Figure 6.41 & 6.42 from the textbook]

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				

[Figure 6.41 & 6.42 from the textbook]

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

[Figure 6.41 & 6.42 from the textbook]

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	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

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	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

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	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

Y

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

s

b \ *y a*	00	01	11	10
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	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

Y

b \ $y\,a$	00	01	11	10
0	0	0	1	0
1	0	1	1	1

s

b \ $y\,a$	00	01	11	10
0	0	1	0	1
1	1	0	1	0

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	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

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

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

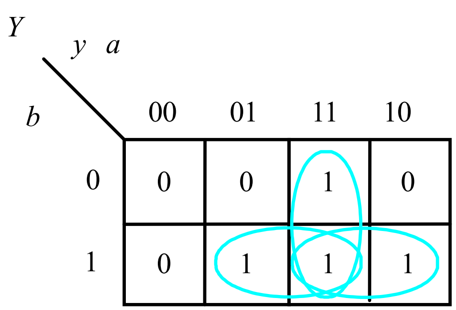
s

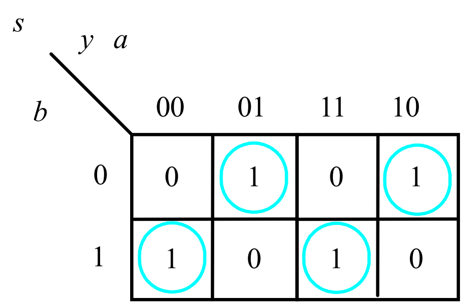
b \ $y\ a$	00	01	11	10
0	0	1	0	1
1	1	0	1	0

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

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

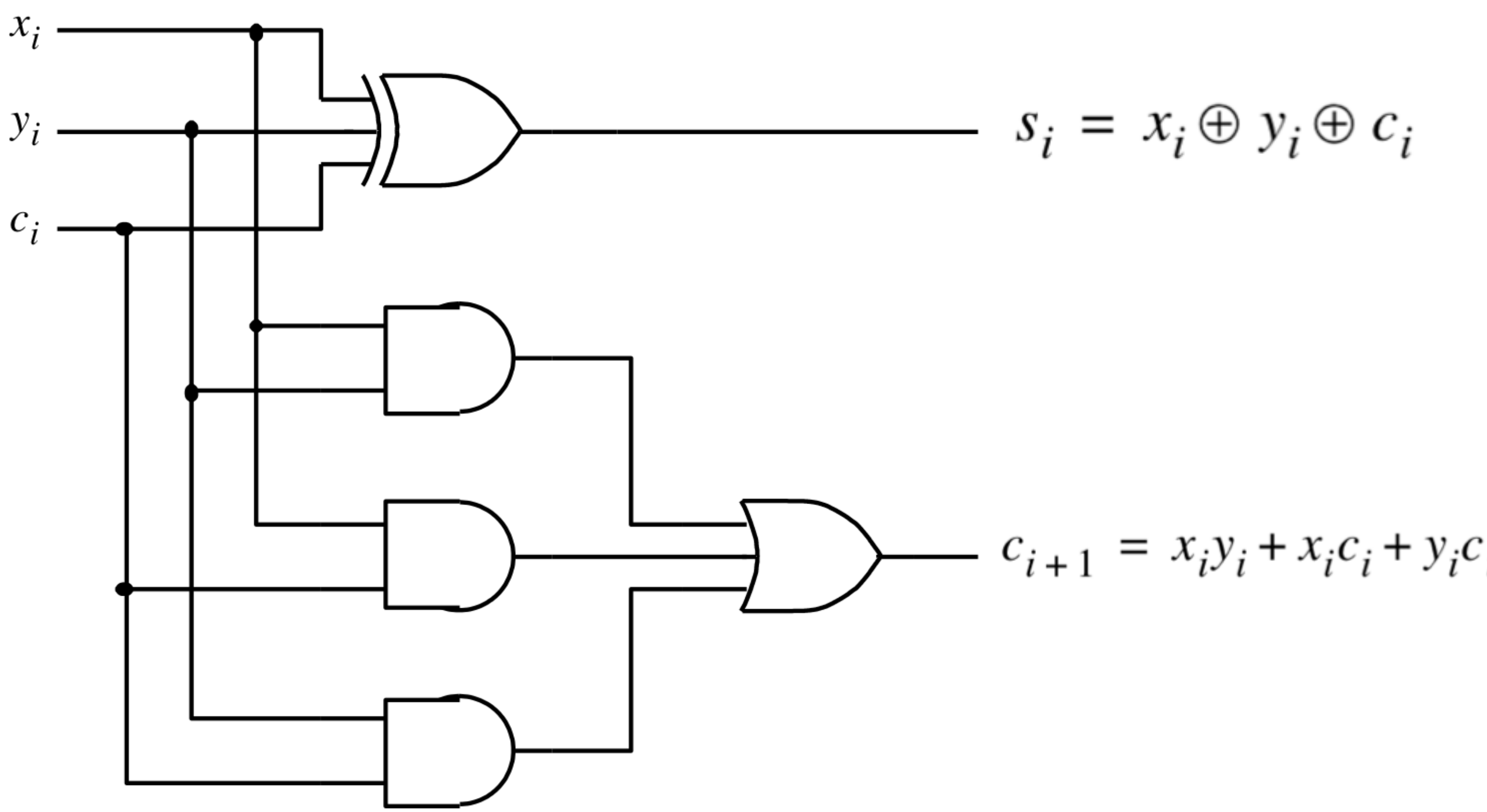




Y = ab + ay + by

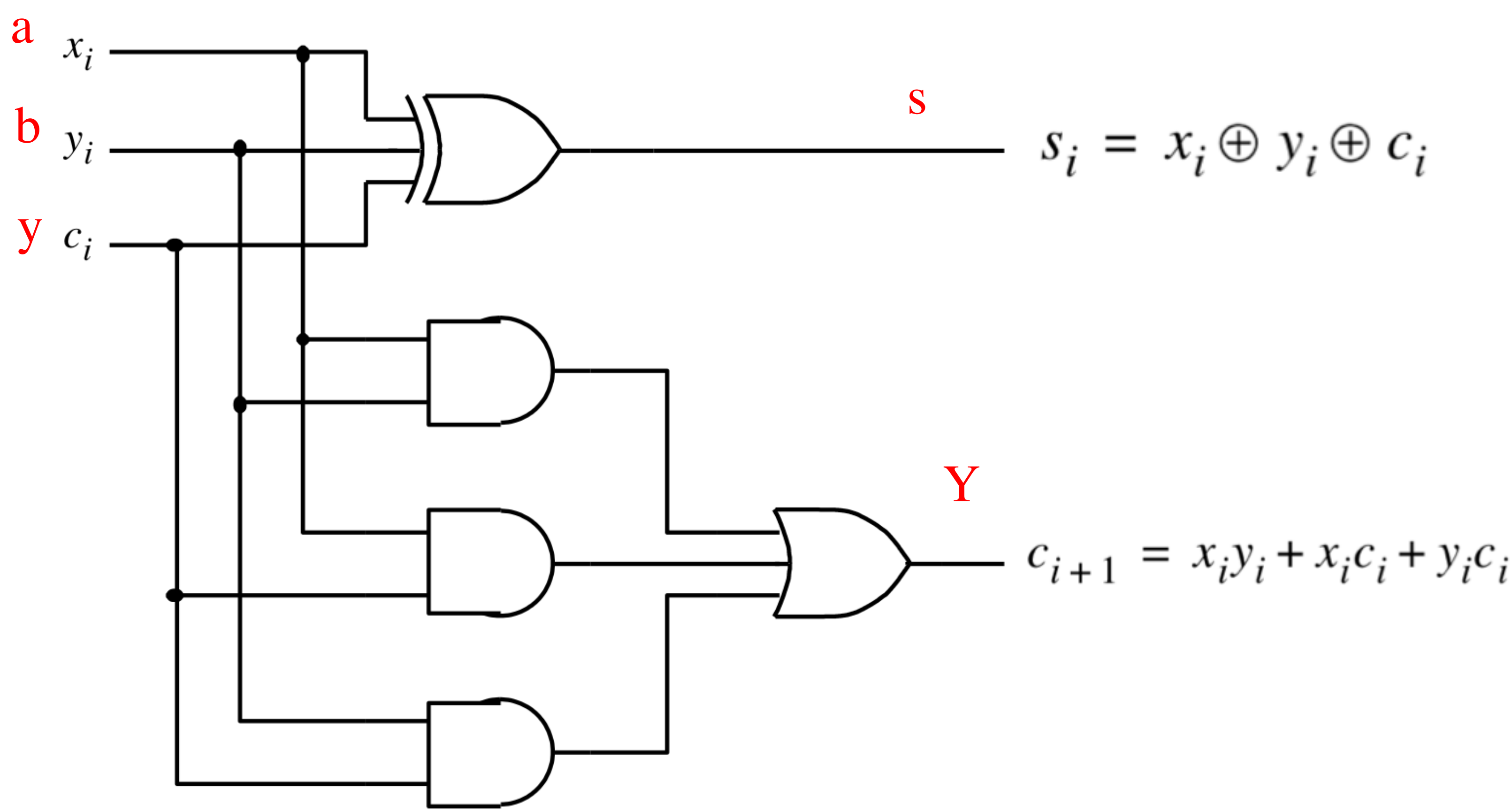
s = XOR(XOR(a, b), y)

The circuit for the two expressions



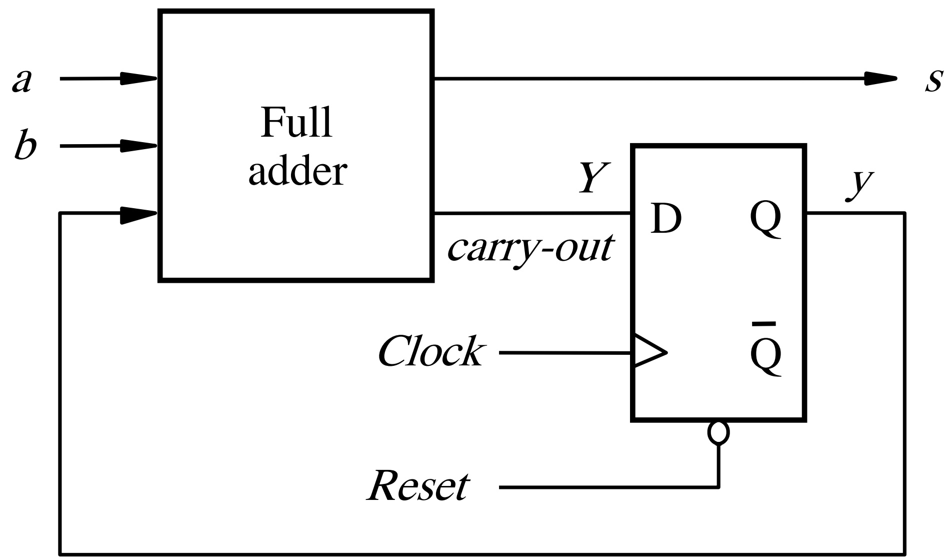
[Figure 3.3c from the textbook]

The circuit for the two expressions



[Figure 3.3c from the textbook]

Circuit for the serial adder FSM



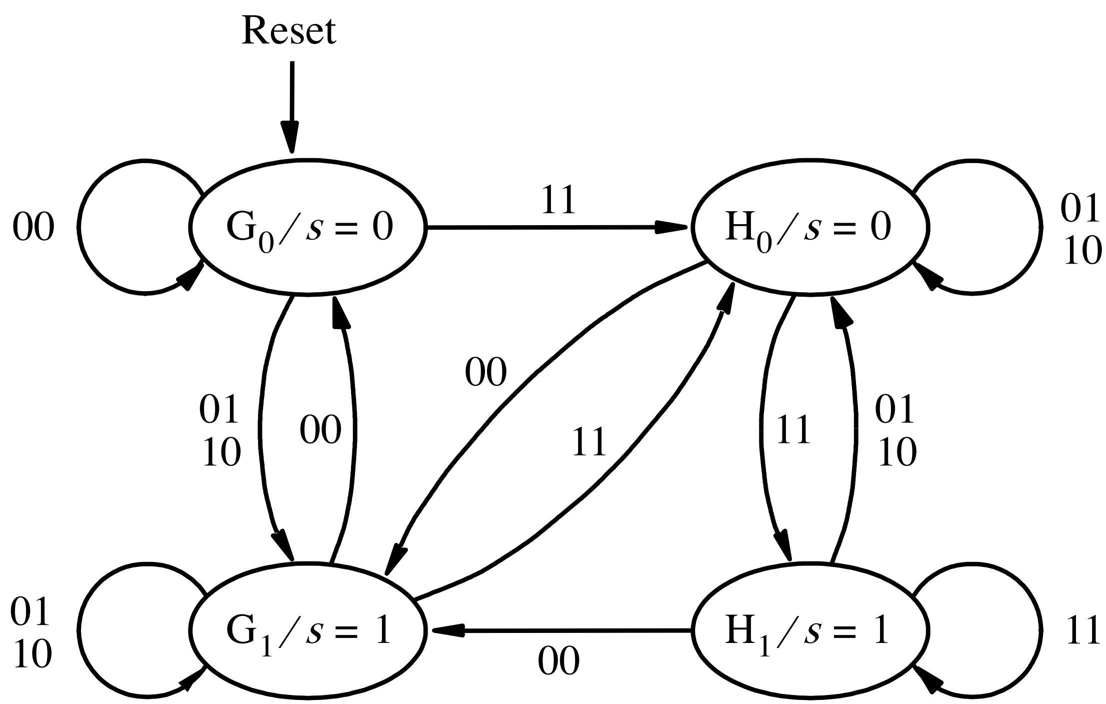
$Y = ab + ay + by$ (carry bit from FA)

$s = XOR(XOR(a, b), y)$ (sum bit from FA)

[Figure 6.43 from the textbook]

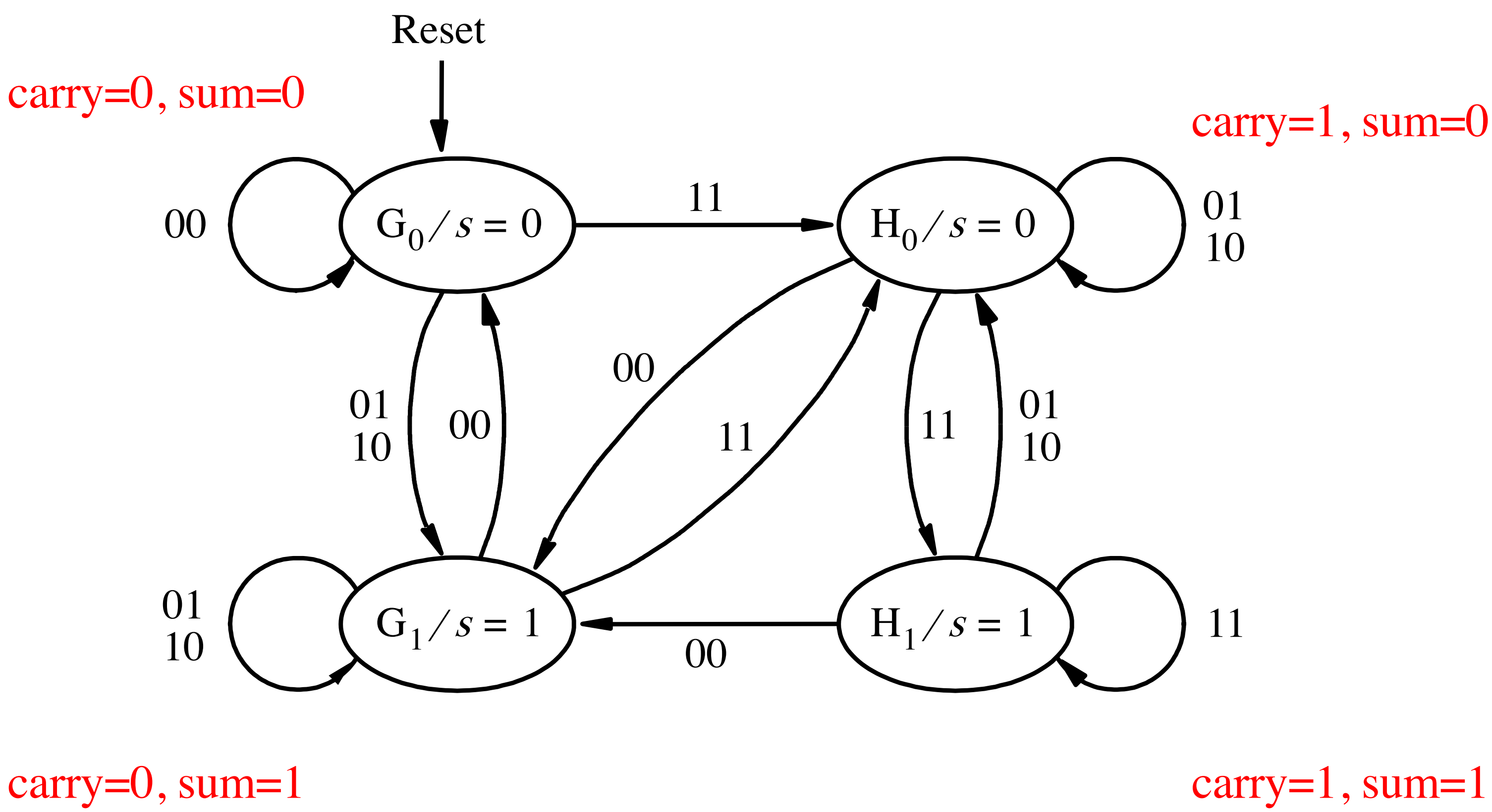
Moore Machine Implementation

State diagram for the Moore-type serial adder FSM



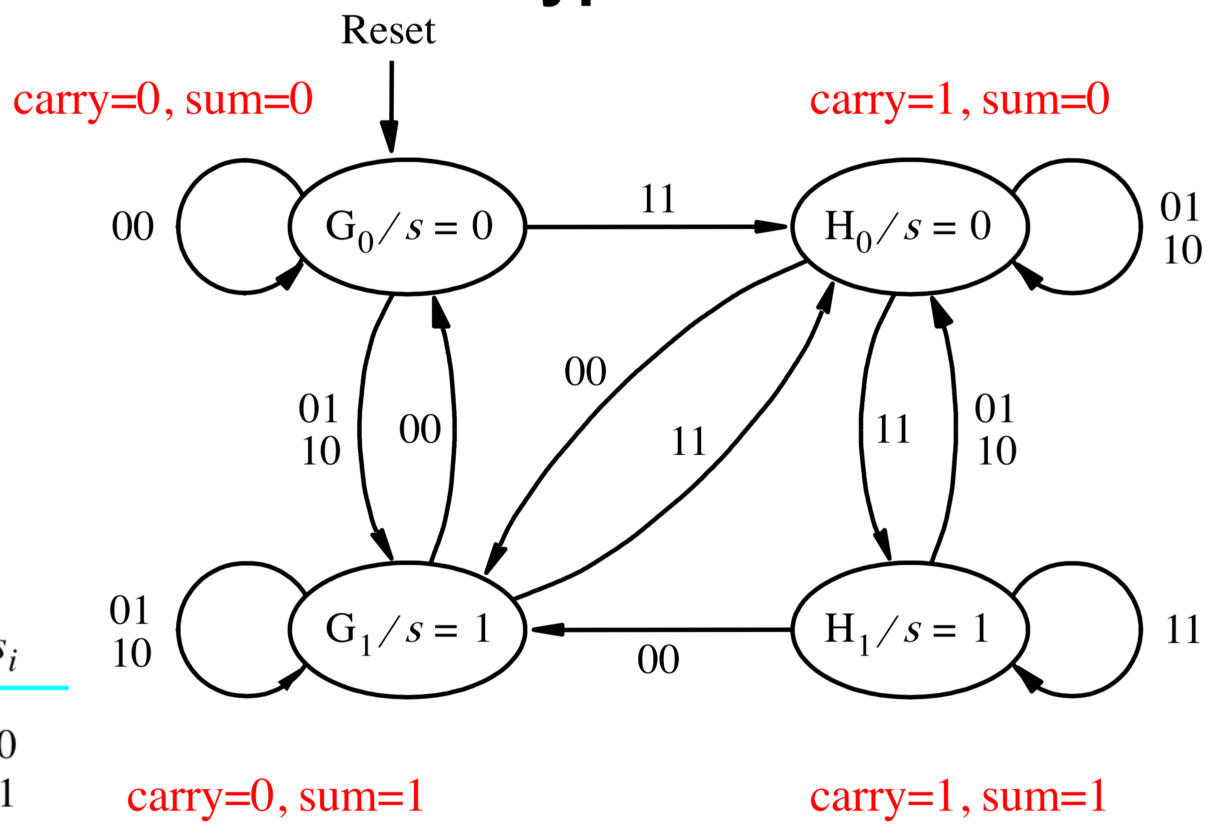
[Figure 6.44 from the textbook]

State diagram for the Moore-type serial adder FSM



[Figure 6.44 from the textbook]

State diagram for the Moore-type serial adder FSM

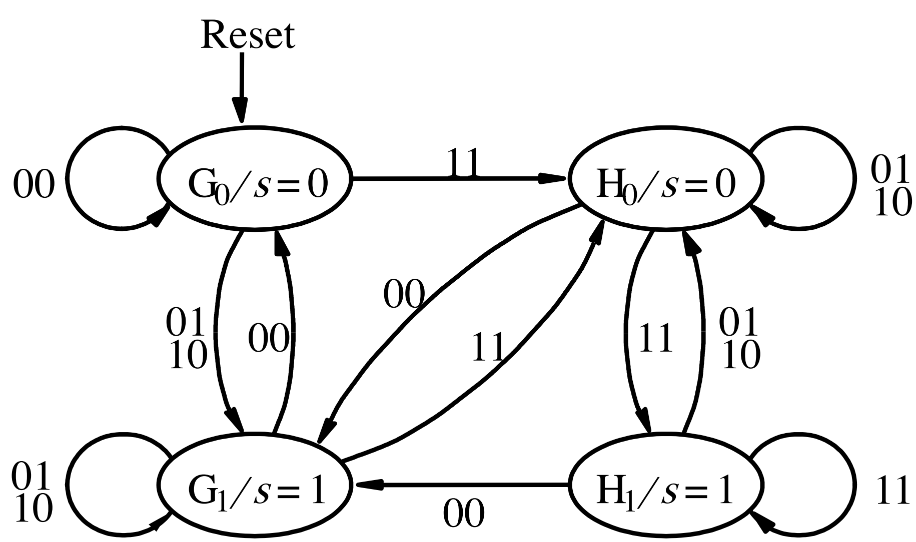


c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

[Figure 6.44 from the textbook]

State table for the Moore-type serial adder FSM

Present state	Next state				Output
	ab =00	01	10	11	*s*
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	Next state				Output
	ab =00	01	10	11	*s*
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	Next state				Output
	ab =00	01	10	11	s
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 $y_2 y_1$	Next state				Output s
	ab =00	01	10	11	
	$Y_2 Y_1$				
00					
01					
10					
11					

[Figure 6.45 & 6.46 from the textbook]

State table for the Moore-type serial adder FSM

Present state	Next state ab =00	01	10	11	Output s
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	Next state ab =00	01	10	11	Output s
	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

[Figure 6.45 & 6.46 from the textbook]

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

Present state $y_2 y_1$	Next state				Output s
	ab =00	01	10	11	
	$Y_2 Y_1$				
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

[Figure 6.46 from the textbook]

Deriving Y1, Y2, and s

Present state $y_2 y_1$	Next state				Output s
	ab =00	01	10	11	
	$Y_2 Y_1$				
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	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	Next state ab =00	01	10	11	Output s
	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	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	Next state ab =00	01	10	11	Output s
	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	Next state ab =00	01	10	11	Output s
	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

$a\,b$ \ y_2y_1	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	Next state ab =00	01	10	11	Output s
	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

$a\,b$ \ y_2y_1	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	Next state ab =00	01	10	11	Output s
	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

$a\,b$ \ y_2y_1	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	Next state Y_2, ab = 00	01	10	11	Output s
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	Next state ab =00	01	10	11	Output s
	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 $y_2 y_1$	Next state ab = 00	01	10	11	Output s
	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 \ $y_2 y_1$	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	Next state ab =00	01	10	11	Output s
	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	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	Next state ab =00	01	10	11	Output s
	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	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	Next state ab=00	01	10	11	Output s
	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	Next state ab=00	01	10	11	Output s
	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	1	
1	0	
1	1	

Deriving s

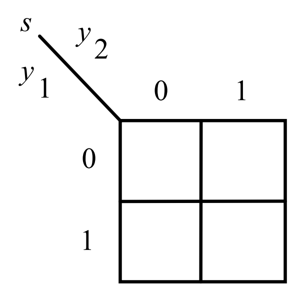
Present state $y_2 y_1$	Next state				Output s
	ab =00	01	10	11	
	$Y_2 Y_1$				
00	0 0	01	0 1	10	0
01	0 0	01	0 1	10	1
10	0 1	10	1 0	11	0
11	0 1	10	1 0	11	1

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

Deriving s

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

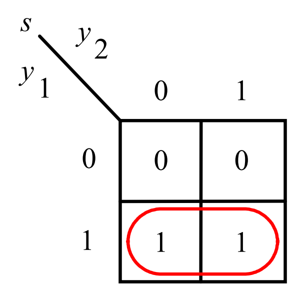
y_2	y_1	s
0	0	0
0	1	1
1	0	0
1	1	1



Deriving s

Present state y_2y_1	Next state ab =00	01	10	11	Output s
	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

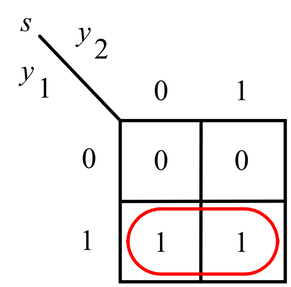
y_2	y_1	s
0	0	0
0	1	1
1	0	0
1	1	1



Deriving s

Present state $y_2 y_1$	Next state $Y_2 Y_1$, ab = 00	01	10	11	Output s
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



$s = y_1$

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

Present state $y_2 y_1$	Next state ab=00	01	10	11	Output s
	$Y_2 Y_1$				
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

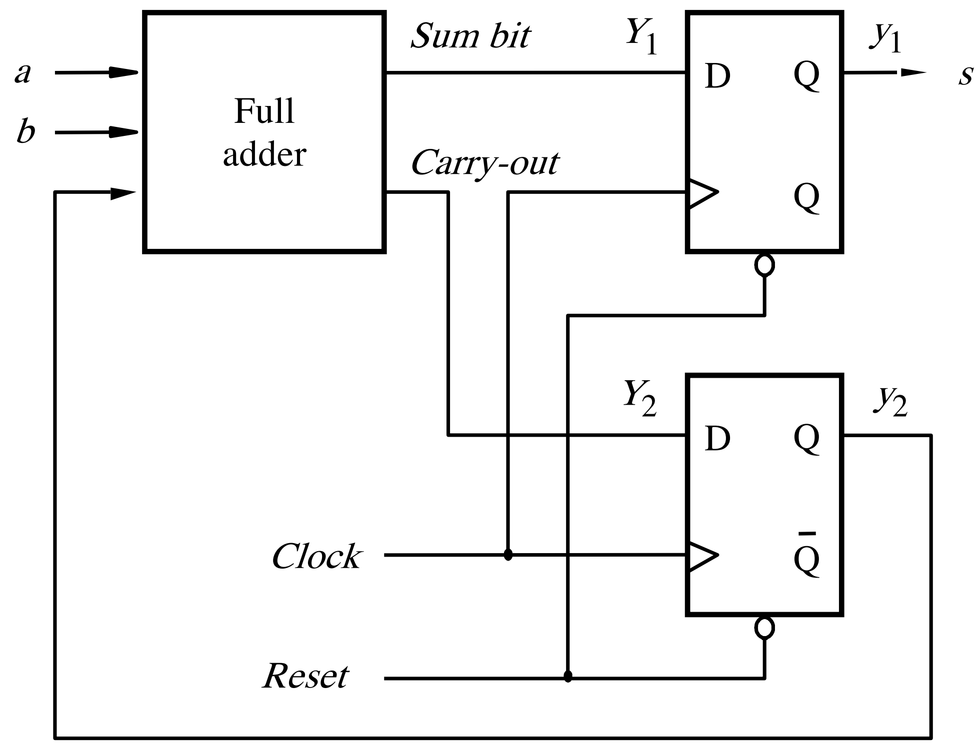
$$Y_1 = a \oplus b \oplus y_2$$

$$Y_2 = ab + ay_2 + by_2$$

$$s = y_1$$

[Figure 6.46 from the textbook]

Circuit for the Moore-type serial adder FSM



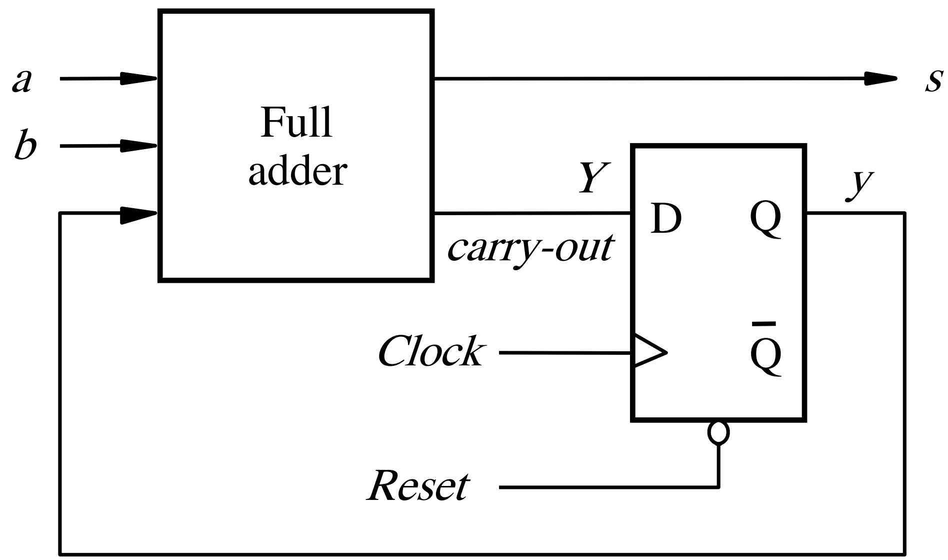
$Y_1 = a \oplus b \oplus y_2$ (sum bit from FA)

$Y_2 = ab + ay_2 + by_2$ (carry bit from FA)

$s = y_1$

[Figure 6.47 from the textbook]

Circuit for the Mealy-type serial adder FSM

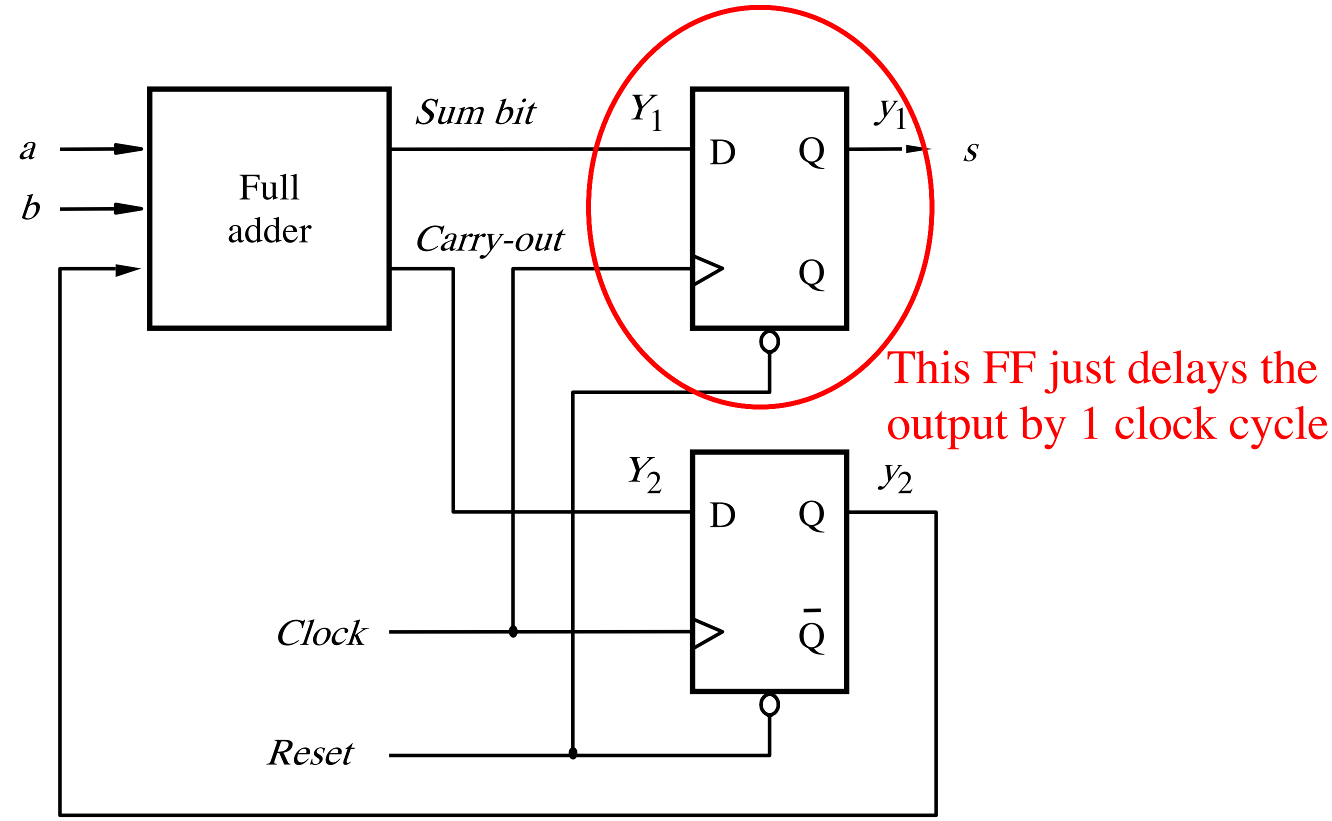


s = XOR(XOR(a, b), y) (sum bit from FA)

Y = ab + ay + by (carry bit from FA)

[Figure 6.43 from the textbook]

Circuit for the Moore-type serial adder FSM



$$Y_1 = a \oplus b \oplus y_2$$ (sum bit from FA)

$$Y_2 = ab + ay_2 + by_2$$ (carry bit from FA)

$$s = y_1$$

[Figure 6.47 from the textbook]

Arbiter Circuit

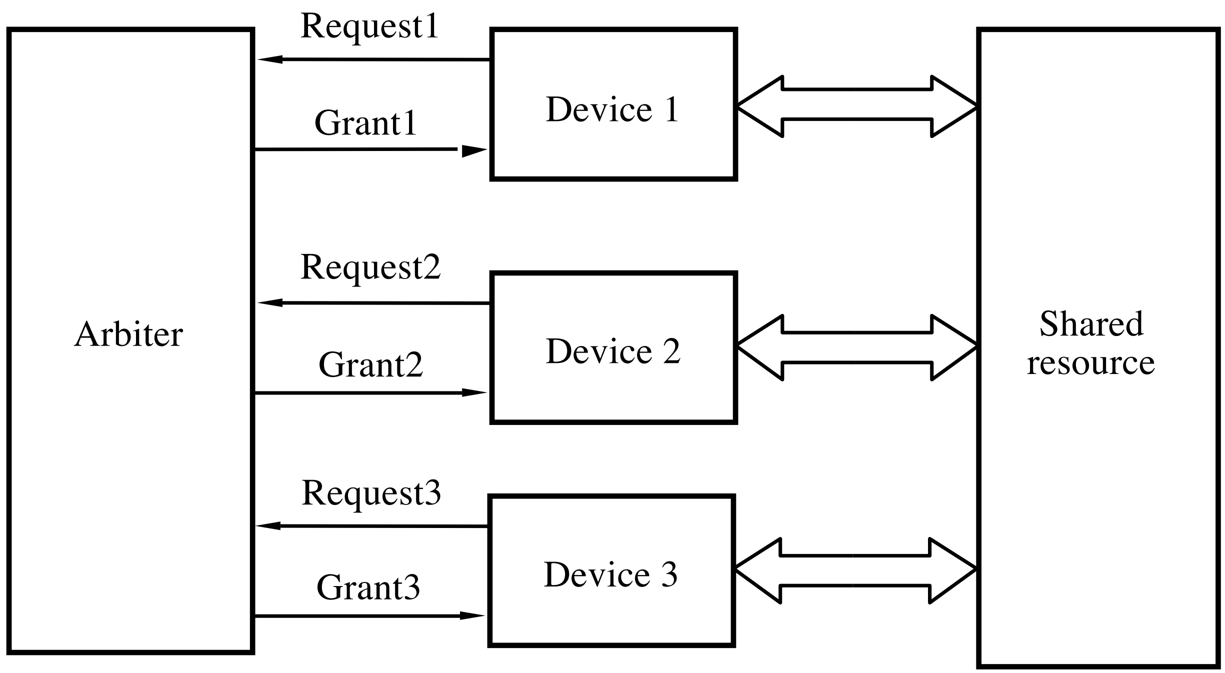
Goal

- Design a machine that controls access by several devices to a shared resource.
- The resource can be used by only one device at a time.
- Any changes can occur only on the positive edge of the clock signal.
- Each device provides one input to the FSM, which is called a request.
- The FSM produces one output for each device, which is called a grant.

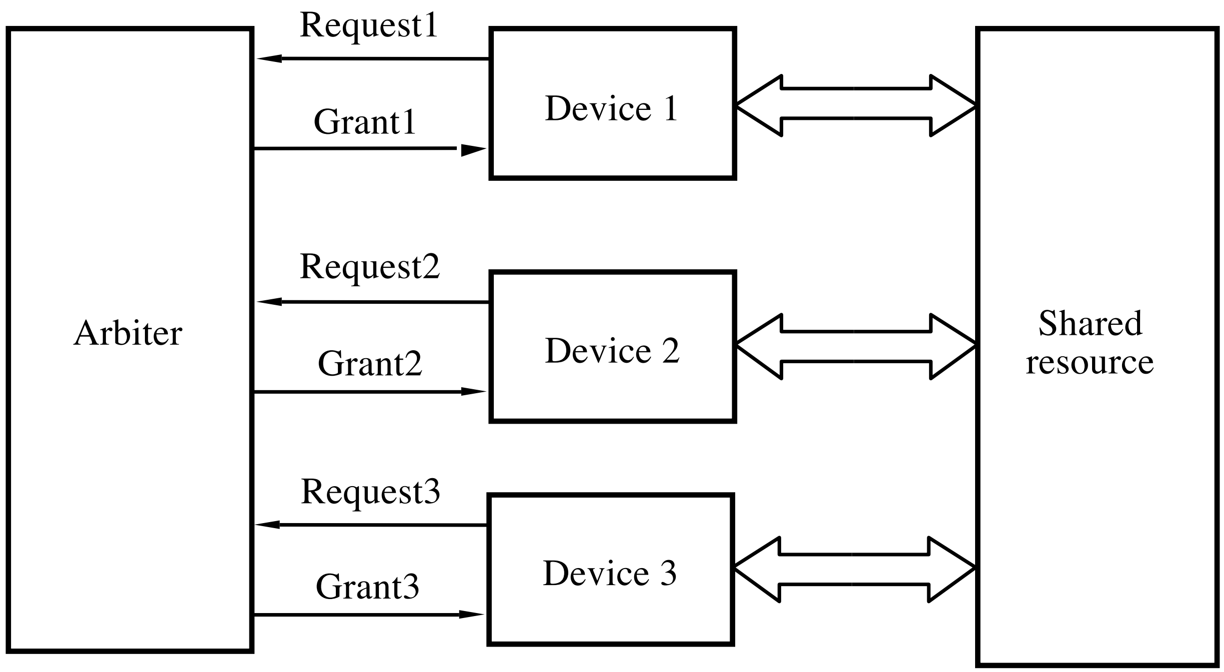
Goal

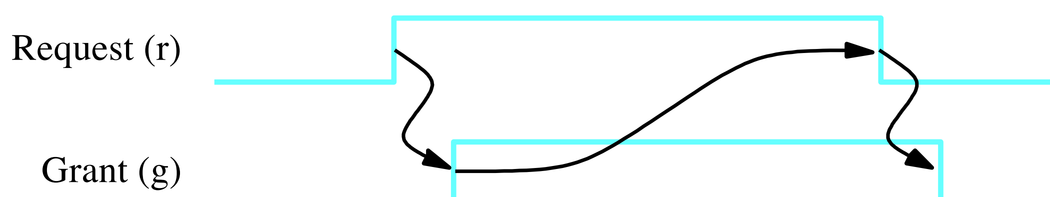
- The requests from the devices are prioritized.
- If two requests are active at the same time, then only the device with the highest priority will be given access to the shared resource.
- After a device is done with the shared resource, it must make its request signal equal to 0.
- If there are no outstanding requests, then the FSM stays in an Idle state.

Conceptual Diagram



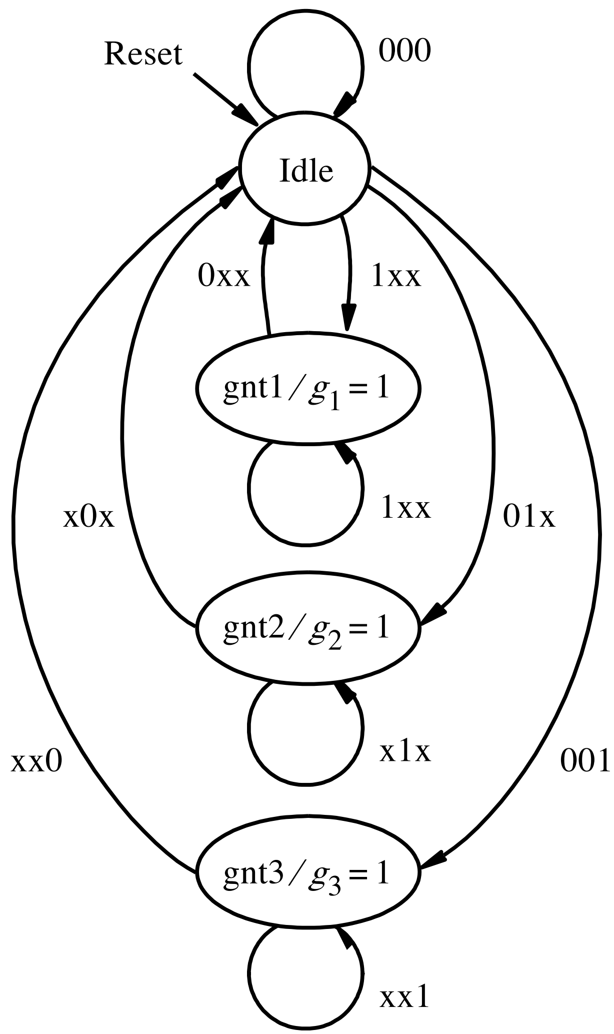
Conceptual Diagram





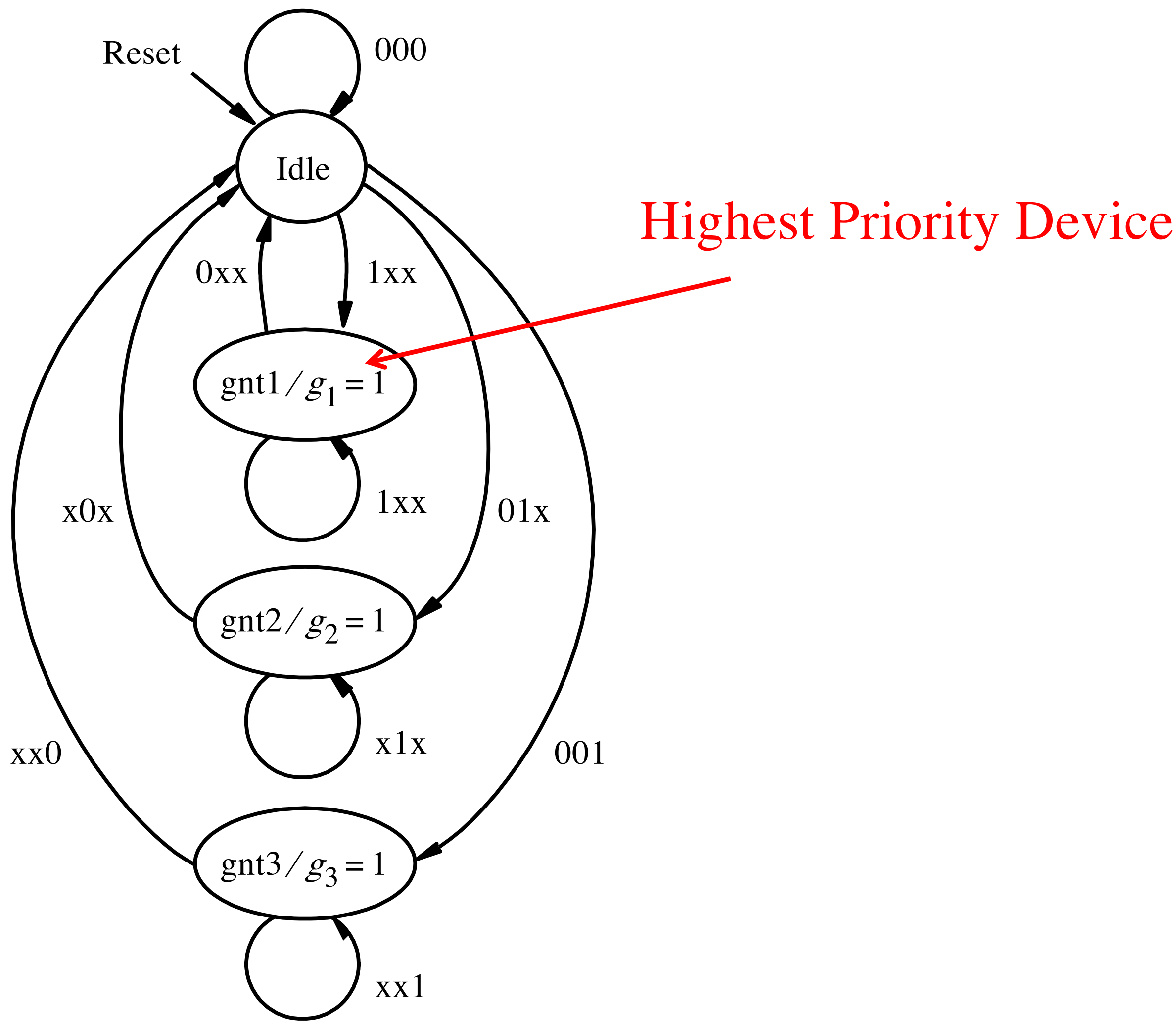
[Figure 9.20 from the textbook]

State diagram for the arbiter

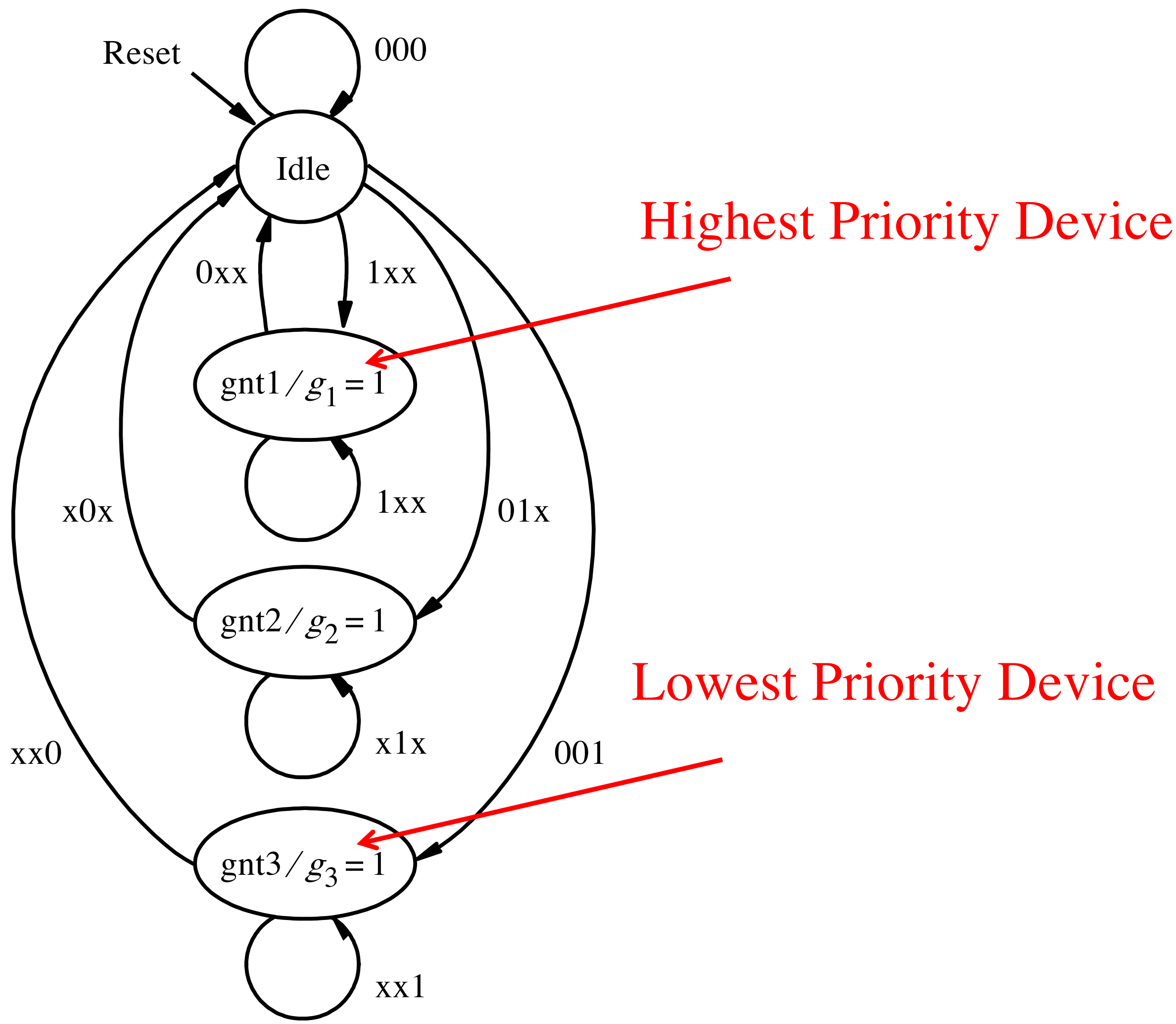


[Figure 6.72 from the textbook]

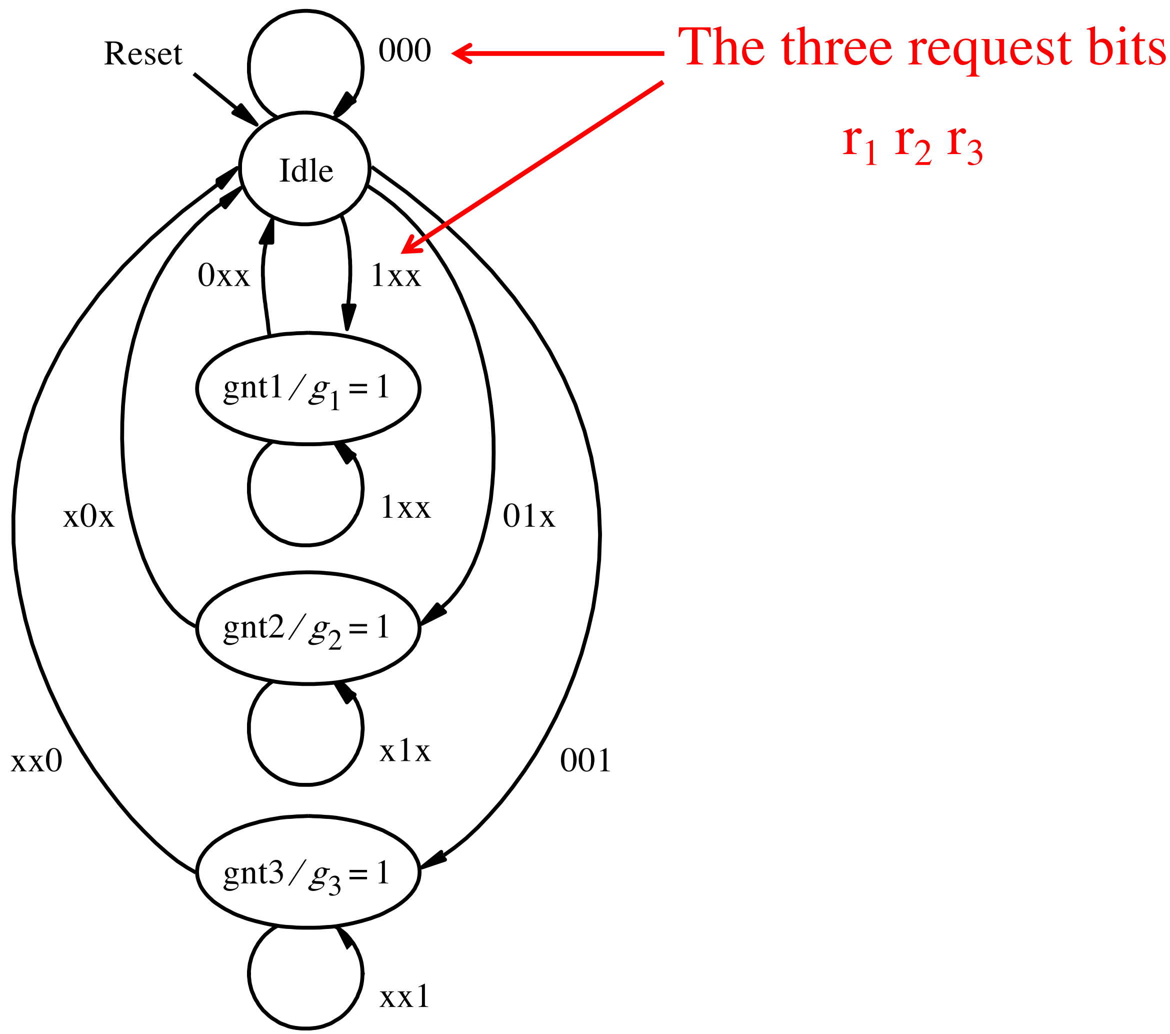
State diagram for the arbiter



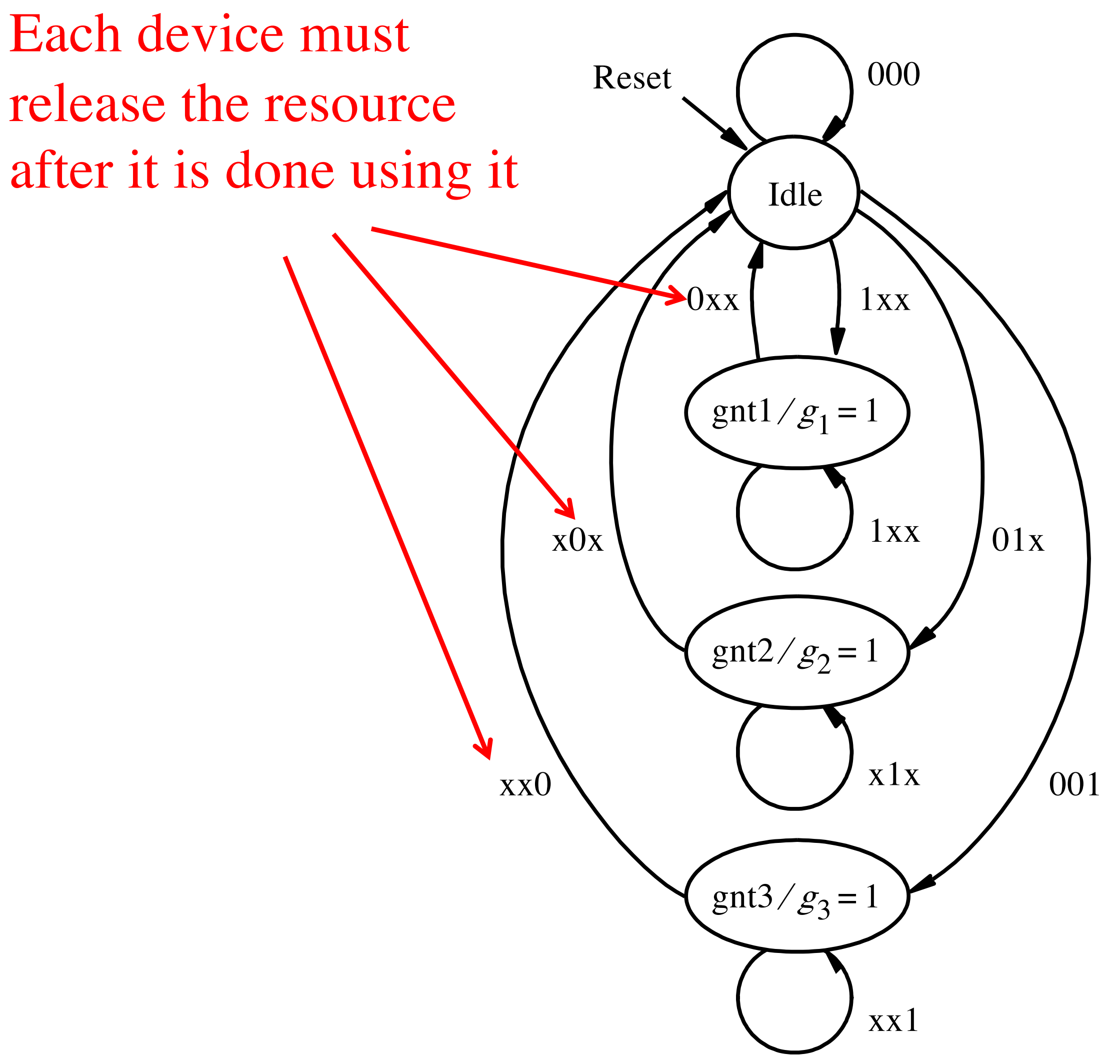
State diagram for the arbiter



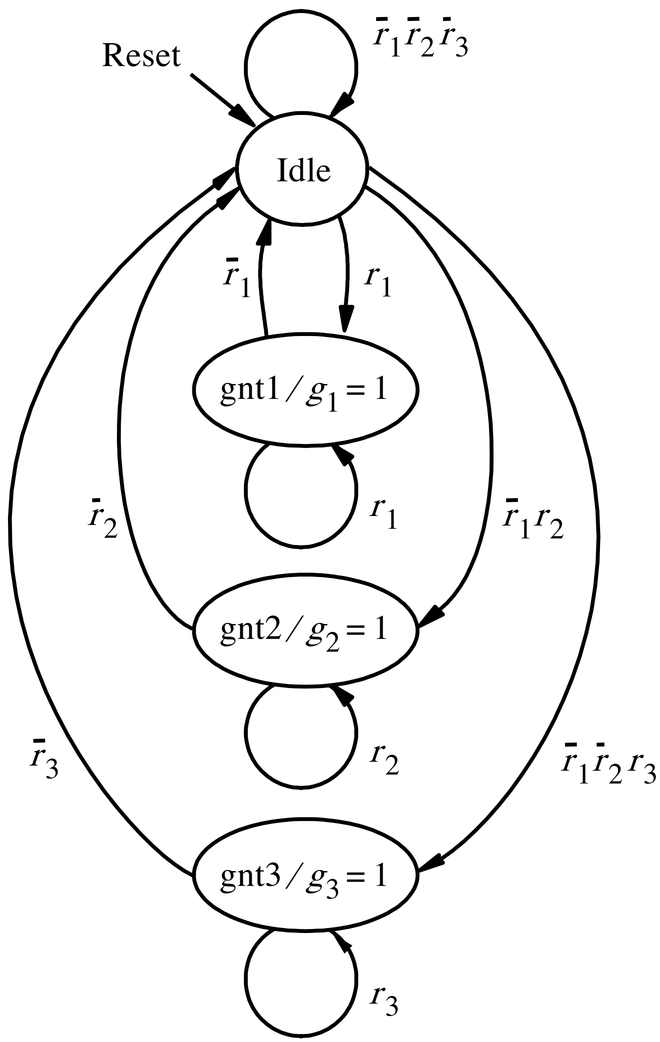
State diagram for the arbiter



State diagram for the arbiter



Alternative style of state diagram for the arbiter

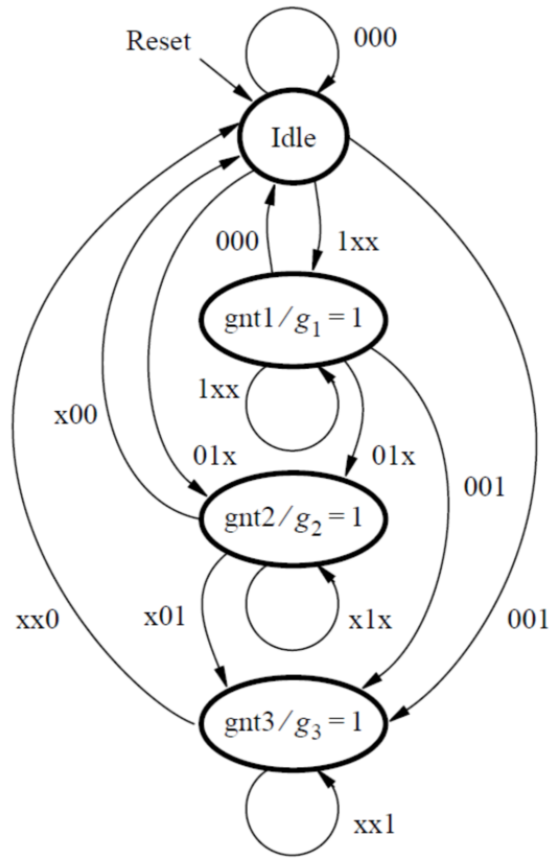


[Figure 6.73 from the textbook]

This design has one flaw:

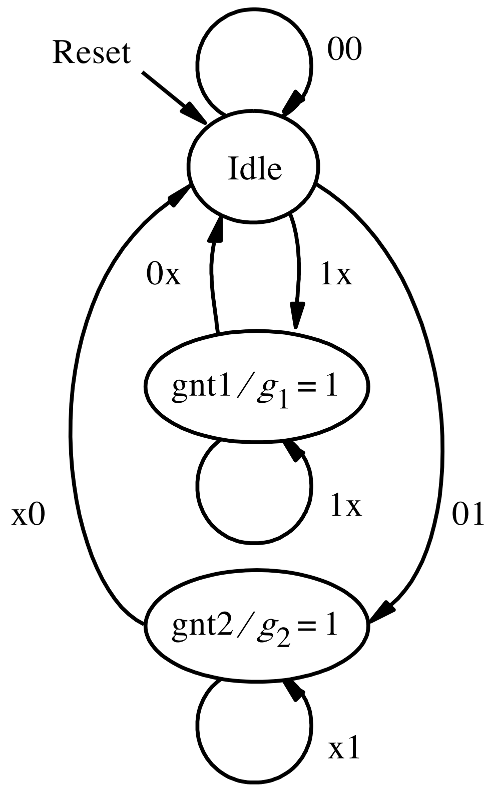
If device1 and device2 raise requests all the time, then device3 will never get serviced.

This state diagram solves this problem

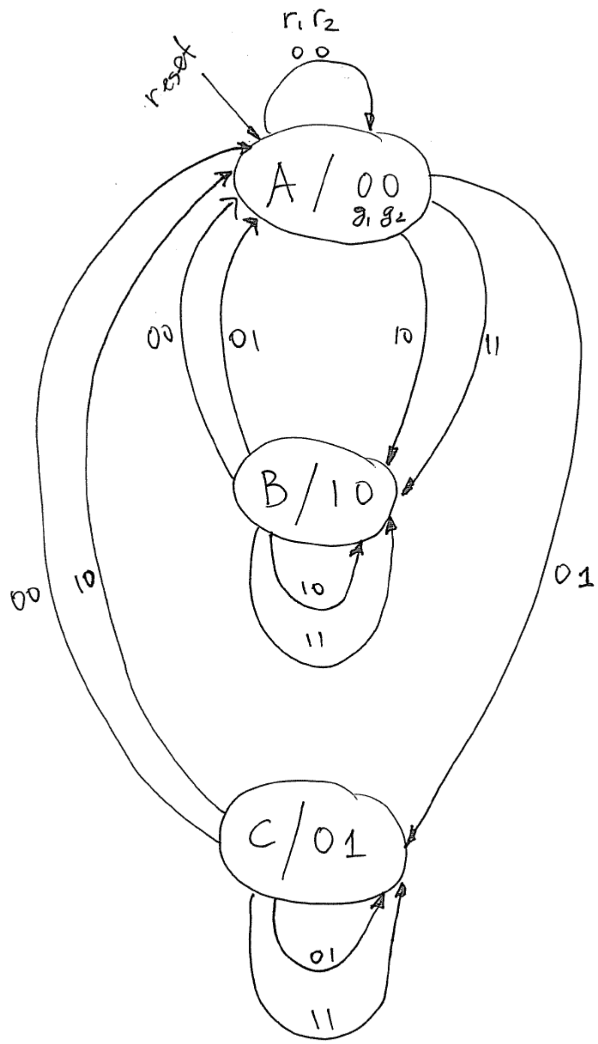


Let's look at a simpler example with only two devices that need to use the shared resource

State diagram for the simpler arbiter



State diagram for the arbiter circuit



State Table

	$r_1 r_2$ = 00	01	10	11	Output
A	A	C	B	B	0 0
B	A	A	B	B	1 0
C	A	C	A	C	0 1

State-Assigned Table

	Present state $y_2 y_1$	Next state $Y_2 Y_1$: $r_1 r_2$ = 00	01	10	11	Output $g_1 g_2$
A	00	00	10	01	01	00
B	01	00	00	01	01	10
C	10	00	10	00	10	01
	11	dd	dd	dd	dd	dd

Output Expressions

Output expressions

$$g_1 = Y_1$$

$$g_2 = Y_2$$

Next State Expressions

Y_2

$r_1 r_2$ \ $y_2 y_1$	00	01	11	10
00	0	0	d	0
01	1	0	d	1
11	0	0	d	1
10	0	0	d	0

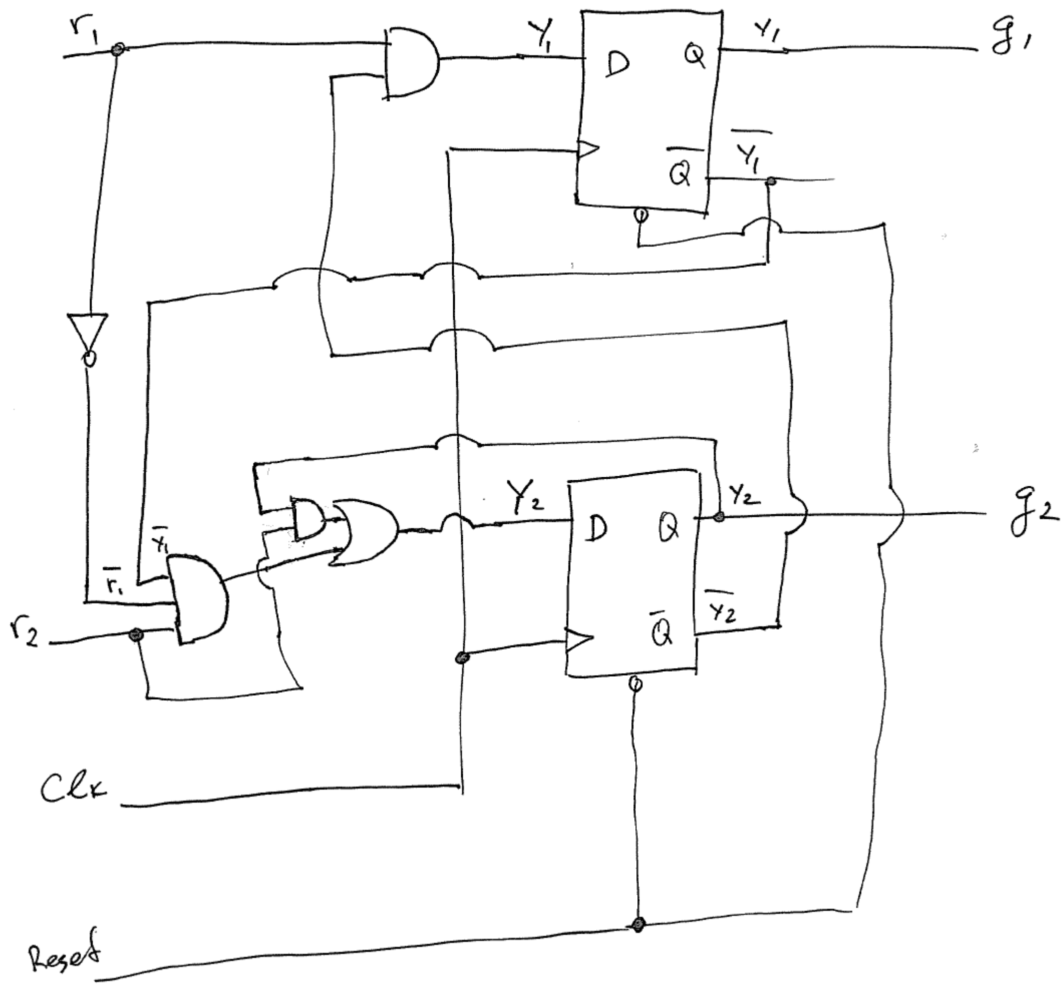
$$Y_2 = r_2 y_2 + \overline{r_1} r_2 \overline{y_1}$$

Y_1

$r_1 r_2$ \ $y_2 y_1$	00	01	11	10
00	0	0	d	0
01	0	0	d	0
11	1	1	d	0
10	1	1	d	0

$$Y_1 = r_1 \overline{y_2}$$

Circuit Diagram



Questions?

THE END