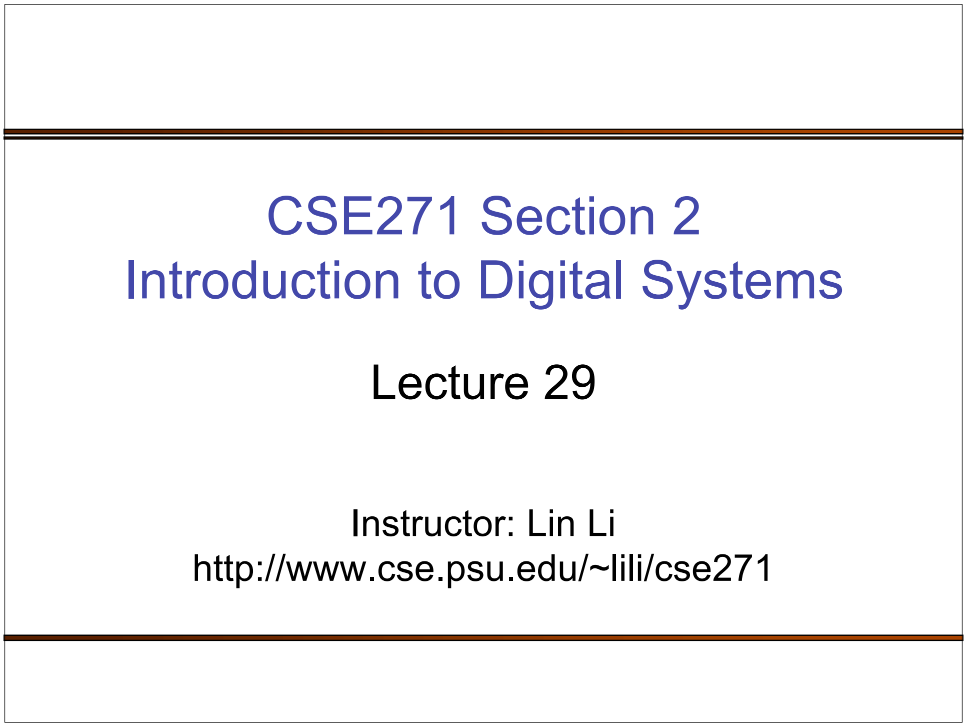
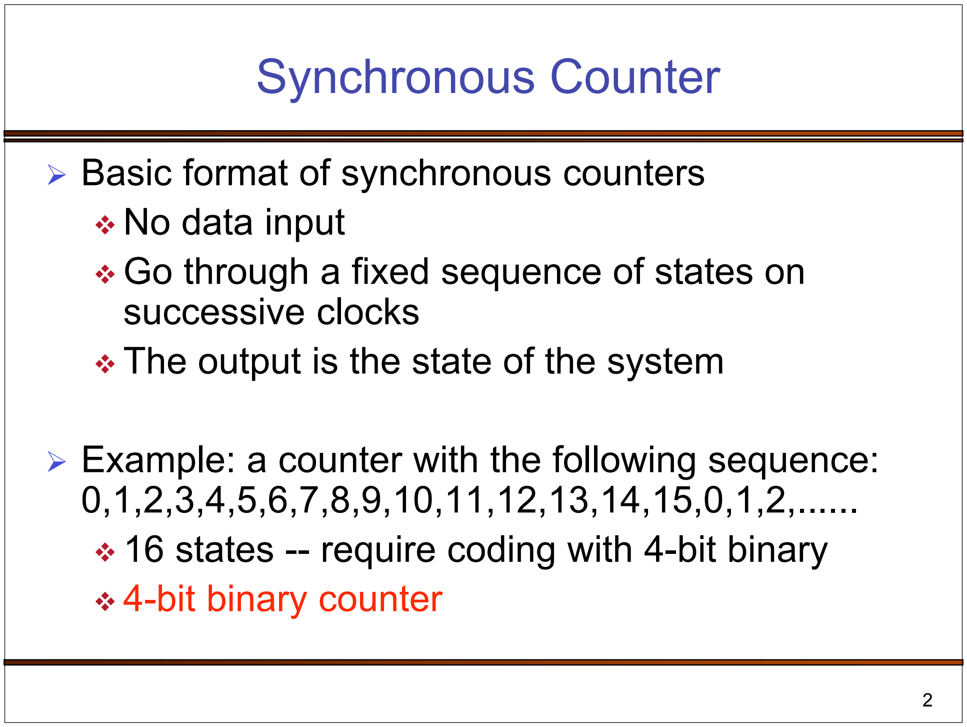


CSE271 Section 2
Introduction to Digital Systems

Lecture 29

Instructor: Lin Li
http://www.cse.psu.edu/~lili/cse271



Synchronous Counter

- Basic format of synchronous counters
  - No data input
  - Go through a fixed sequence of states on successive clocks
  - The output is the state of the system
- Example: a counter with the following sequence: 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,0,1,2,......
  - 16 states -- require coding with 4-bit binary
  - 4-bit binary counter

Synchronous Counter (Cont’d)

D	C	B	A	D*	C*	B*	A*
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	1	0	1	0
1	0	1	0	1	0	1	1
1	0	1	1	1	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	1
1	1	1	1	0	0	0	0

BA \ DC	00	01	11	10
00			1	1
01			1	1
11		1		1
10			1	1

D*

BA \ DC	00	01	11	10
00		1	1	
01		1	1	
11	1			1
10		1	1	

C*

$$\begin{aligned} D_D = D^* &= DC' + DB' + DA' + D'CBA \\ &= D(C' + B' + A') + D'CBA \\ &= D(CBA)' + D'CBA \\ &= D\oplus(CBA) \end{aligned}$$

$$\begin{aligned} D_C = C^* &= CB' + CA' + C'BA \\ &= C\oplus(BA) \end{aligned}$$

Synchronous Counter (Cont’d)

D	C	B	A	D*	C*	B*	A*
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	1	0	1	0
1	0	1	0	1	0	1	1
1	0	1	1	1	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	1
1	1	1	1	0	0	0	0

BA \ DC	00	01	11	10
00				
01	1	1	1	1
11				
10	1	1	1	1

B*

BA \ DC	00	01	11	10
00	1	1	1	1
01				
11				
10	1	1	1	1

A*

$$\begin{aligned} D_B = B^* &= B'A + BA' \\ &= B\oplus A \end{aligned}$$

$$D_A = A^* = A'$$

Synchronous Counter (Cont'd)

BA \ DC	00	01	11	10
00			1	1
01			1	1
11		1		1
10			1	1

D*

BA \ C	0	1
00		
01		
11		1
10		

J_D

BA \ C	0	1
00	1	1
01	1	1
11	1	
10	1	1

K_D'

BA \ C	0	1
00		
01		
11		1
10		

K_D

Split D* based on the value of D.

$J_D=K_D=CBA$

BA \ DC	00	01	11	10
00		1	1	
01		1	1	
11	1			1
10		1	1	

C*

BA \ D	0	1
00		
01		
11	1	1
10		

J_C

BA \ D	0	1
00	1	1
01	1	1
11		
10	1	1

K_C'

BA \ D	0	1
00		
01		
11	1	1
10		

K_C

Split C* based on the value of C.

$J_C=K_C=BA$

Synchronous Counter (Cont'd)

BA \ DC	00	01	11	10
00				
01	1	1	1	1
11				
10	1	1	1	1

B*

A \ DC	00	01	11	10
0				
1	1	1	1	1

J_B

A \ DC	00	01	11	10
0	1	1	1	1
1				

K_B'

$J_B=K_B=A$

BA \ DC	00	01	11	10
00	1	1	1	1
01				
11				
10	1	1	1	1

A*

B \ DC	00	01	11	10
0	1	1	1	1
1	1	1	1	1

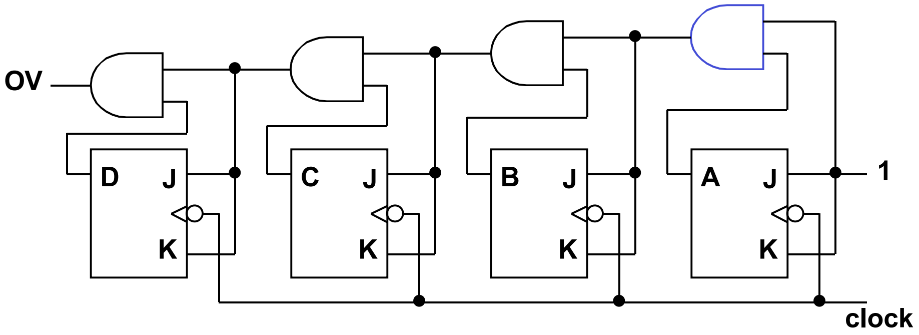
J_A

B \ DC	00	01	11	10
0				
1				

K_A'

$J_A=K_A=1$

Synchronous Counter (Cont'd)



J_D=K_D=CBA
J_C=K_C=BA
J_B=K_B=A
J_A=K_A=1

The blue AND gate is not necessary.
OV is 1 when the counter is in state 15 (1111).
Extend the design to 5 or more flip flops:
J_E=K_E=DCBA
J_F=K_F=EDCBA ......

Up/down Counter

x	C	B	A	C*	B*	A*
0	0	0	0	0	0	1
0	0	0	1	0	1	0
0	0	1	0	0	1	1
0	0	1	1	1	0	0
0	1	0	0	1	0	1
0	1	0	1	1	1	0
0	1	1	0	1	1	1
0	1	1	1	0	0	0
1	0	0	0	1	1	1
1	0	0	1	0	0	0
1	0	1	0	0	0	1
1	0	1	1	0	1	0
1	1	0	0	0	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	1
1	1	1	1	1	1	0

A counter that can count in either direction, depending upon a control input, labeled as x.

When x=0, the counter has following sequence: 0,1,2,3,4,5,6,7,0,1......

When x=1, the counter has following sequence: 7,6,5,4,3,2,1,0,7,6,......

Up/down Counter (Cont'd)

BA \ DC	00	01	11	10
00		1		1
01		1	1	
11	1		1	
10		1	1	

C*

BA \ DC	00	01	11	10
00			1	1
01	1	1		
11			1	1
10	1	1		

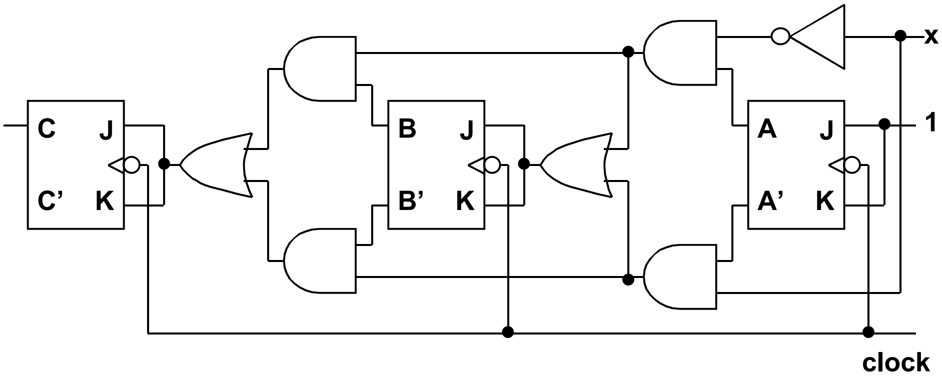
B*

BA \ DC	00	01	11	10
00	1	1	1	1
01				
11				
10	1	1	1	1

A*

$J_C=K_C=x'BA + xB'A'$
$J_B=K_B=x'A+xA'$
$J_A=K_A=1$

Up/down Counter (Cont'd)



$J_C=K_C=x'BA + xB'A'$
$J_B=K_B=x'A+xA'$
$J_A=K_A=1$

Extend the design to 4 or more flip flops:
$J_D=K_D=x'CBA+xC'B'A'$
$J_E=K_E=x'DCBA+xD'C'B'A'$ ......

Decimal Counter

D	C	B	A	D*	C*	B*	A*
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	0	0	0	0
1	0	1	0	X	X	X	X
1	0	1	1	X	X	X	X
1	1	0	0	X	X	X	X
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X

A counter that goes through the sequence: 0,1,2,3,4,5,6,7,8,9,0,1......

Decimal Counter (Cont’d)

BA \ DC	00	01	11	10
00			X	1
01			X	
11		1	X	X
10			X	X

D*

BA \ DC	00	01	11	10
00		1	X	
01		1	X	
11	1		X	X
10		1	X	X

C*

BA \ DC	00	01	11	10
00			X	
01	1	1	X	
11			X	X
10	1	1	X	X

B*

BA \ DC	00	01	11	10
00	1	1	X	1
01			X	
11			X	X
10	1	1	X	X

A*

$J_D = CBA$

$K_D = A$

$J_C = K_C = BA$

$J_B = D'A$

$K_B = A$

$J_A = K_A = 1$

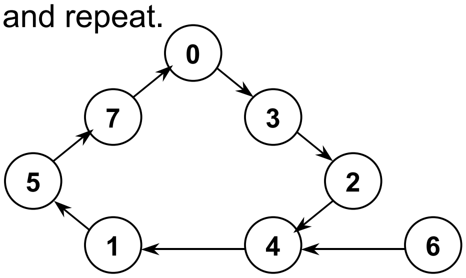
Initial State of Counters

- We do not know in what state each flip flop will be initially, when we turn the system on.
  - Often, all we care about is that once the system is turned on, it goes through the desired sequence after one or two clocks.
  - If we care about the initial state,
    - Use some combination of clears and presets to get the system into the proper initial state.

Counter with Unused States

A counter that goes through the following sequence that is not in numeric order: 0,3,2,4,1,5,7,and repeat.

C	B	A	C*	B*	A*
0	0	0	0	1	1
0	0	1	1	0	1
0	1	0	1	0	0
0	1	1	0	1	0
1	0	0	0	0	1
1	0	1	1	1	1
1	1	0	X	X	X
1	1	1	0	0	0



D_C = BA'+ B'A
D_B = C'B'A'+ C'BA + CB'A
D_A = B'

If CBA=110, $D_C D_B D_A$= 100.
After the design, there are no any don't cares.
With different designs, the next state of the state "110" is different.

Cycling/Saturating Counter

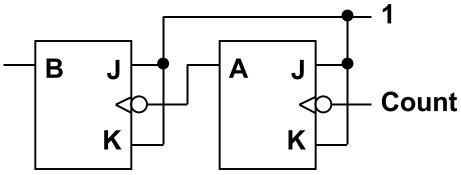
A 2-bit up/down, cycling/saturating counter with two control inputs x and y:

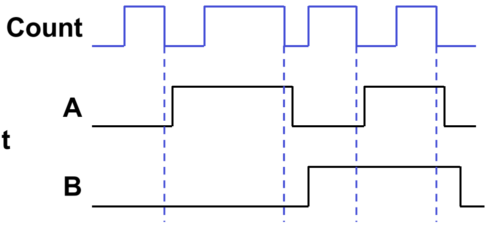
If x=0, it counters up;
If x=1, it counters down;
If y=0, it cycles -- 0,1,2,3,0,1,...... or 3,2,1,0,3,2,......
If y=1, it saturates -- 0,1,2,3,3,3,...... or 3,2,1,0,0,0,......

	A*B*			
AB	**xy = 00**	**xy = 01**	**xy = 10**	**xy = 11**
00	**01**	**01**	**11**	**00**
01	**10**	**10**	**00**	**00**
10	**11**	**11**	**01**	**01**
11	**00**	**11**	**10**	**10**

Asynchronous Counter

- **Asynchronous counter**
  - **Counters without a clock input.**
  - **Each flip flop is triggered by the transition of the previous one.**
  - **Advantage -- the simplicity of the hardware.**
  - **Disadvantage -- longer delay.**
    - **Flip flop B changes later than flip flop A.**
- **Count signal may be irregular.**

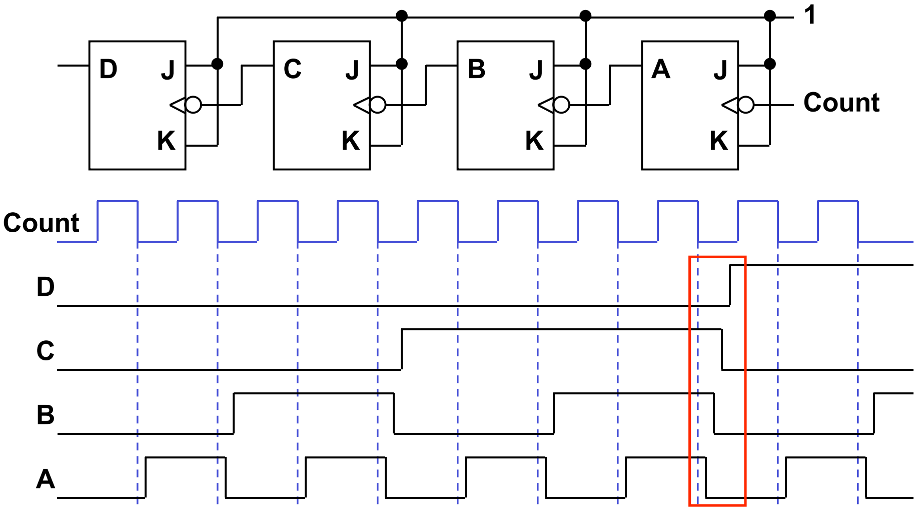




2-bit asynchronous counter

go through the sequence 00,01,10,11, and repeat.

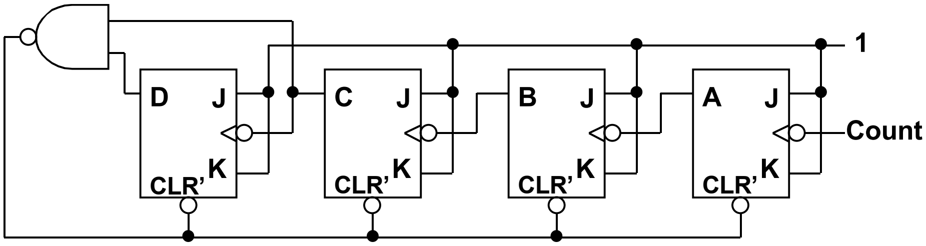
4-bit Asynchronous Counter



One unit of delay through each flip flop.

Asynchronous Counter with Reset

Design an asynchronous base-12 counter with sequence 0,1,2,3,4,5,6,7,8,9,10,11,0,1,......



Use a 4-bit binary counter and reset it when it reaches 12. Note that it remains in state 12 for a short time, due to the delay of flip flops.