

# PDS0101

Introduction to Digital Systems

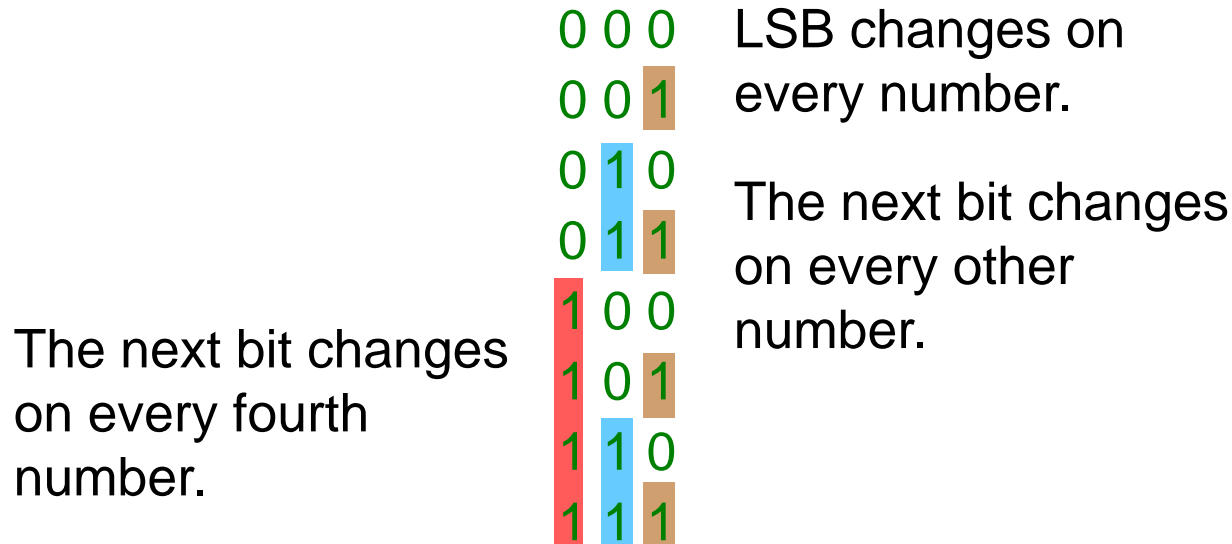
Counters I

# Lecture outcome

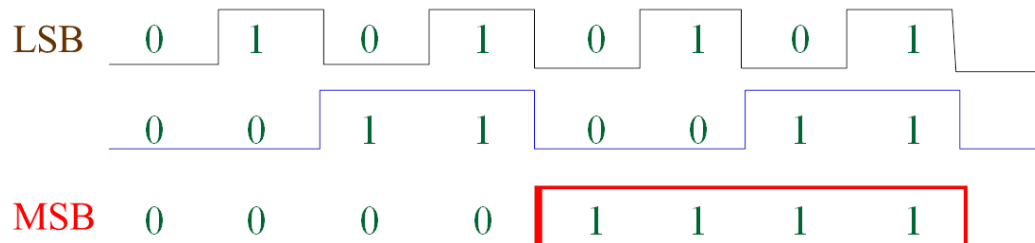
- ∞ By the end of today's lecture you should be able to
  - differentiate asynchronous and synchronous counters
  - analyse counter circuits and timing diagrams
  - explain how propagation delays affect counters
  - determine and modify the modulus of counters
  - recognize the difference between 4-bit and decade binary counters
  
- ∞ **NOTE:** contents in this set of slides are intentionally incomplete and content will be shown in class as examples of how they are derived

# Counting in binary

- As you know, the binary count sequence follows a familiar pattern of 0's and 1's to obtain the increasing count in bit form



- A counter can form the same pattern of 0's and 1's with logic levels. The first stage in the counter represents the least significant bit – notice that these waveforms follow the same pattern as counting in binary



# Counters

- ∞ Counters are important digital electronic circuits.
- ∞ They are *sequential logic* circuits because timing is obviously important and they need a memory characteristic.
- ∞ Digital counters have the following important characteristics:
  - Maximum number of count
  - Up-Down Count
  - Asynchronous or Synchronous Operation
  - Free-Running or Self-Stopping

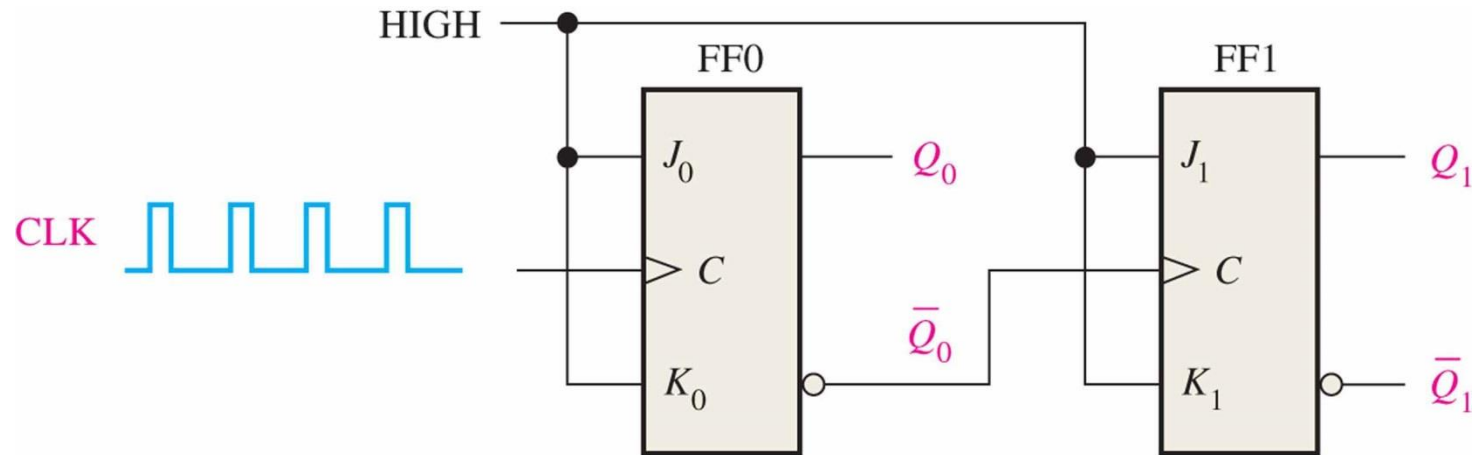
# Asynchronous counters

- ∞ Asynchronous counter are sometimes referred to as *ripple counters* because the effect of the input clock pulse is first “felt” by first flip-flop (FF0).
- ∞ This pulse cannot get to the second flip-flop (FF1) immediately because of the propagation delay through FF0.
- ∞ So the effect of an input clock pulse “ripples” through the counter, taking some time, due to propagation delays, to reach the last flip-flop.

*Typically, only the first FFs will receive clock pulse from the source (clock generator) whilst other FFs receive their own “clock pulse” from either Q or Q' of prior FFs*

# 2-bit asynchronous binary counter

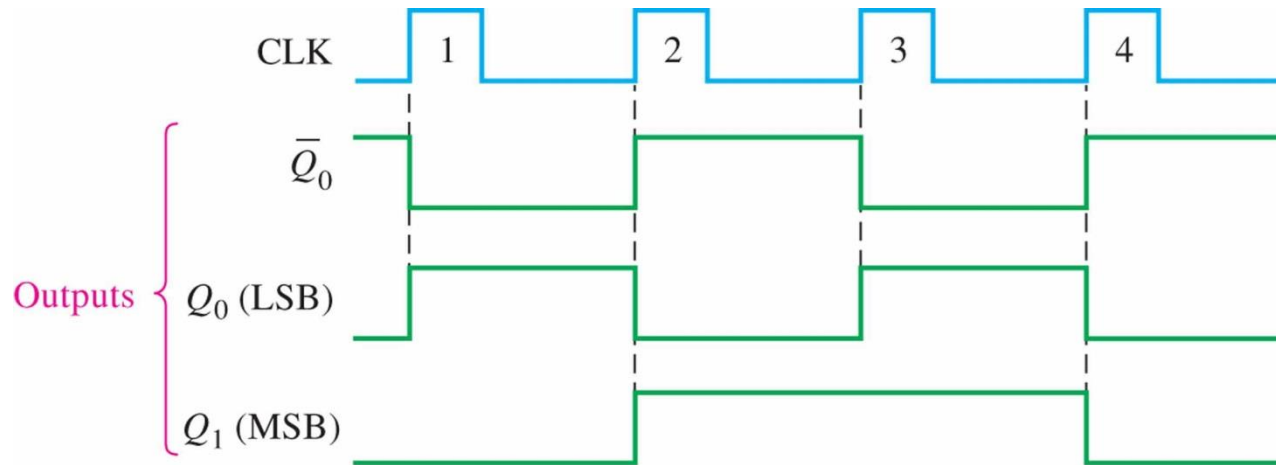
- ∞ The logic circuit below shows a 2-bit counter connected for asynchronous counter operation
- ∞ CLK is applied to the clock input, C, of only the first FF (FF0 or LSB) - FF1 is triggered by the  $Q'$  output of FF0



- ∞ FF0 changes state at the +ve going transition of the CLK whilst FF1 only changes when triggered by the +ve going transition of  $Q'$
- ∞ Because of inherent propagation delay of FFs, a transition of the CLK and transition of  $Q'$  will never occur simultaneously – thus the operation is asynchronous

# 2-bit async counter timing diagram

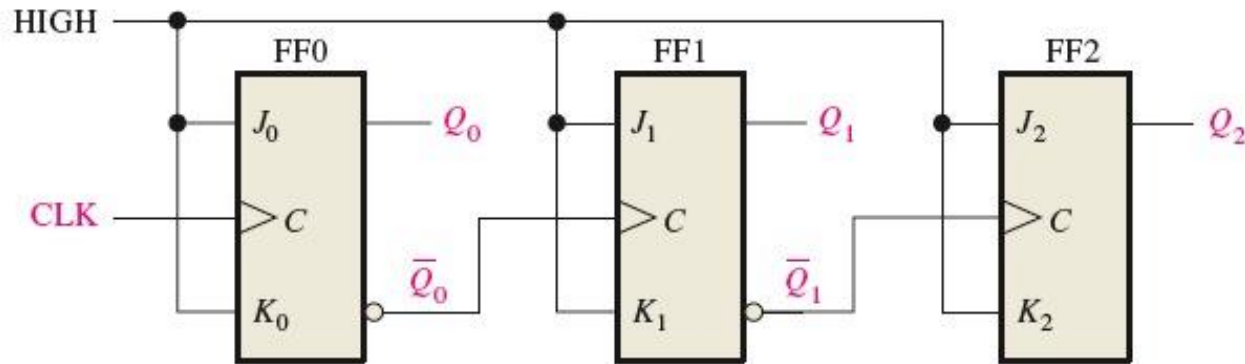
- ∞ The timing diagram below shows what happens in the counter when 4 clock pulses are applied to FF0 – both JKFF connected for toggle operation and Q initially LOW



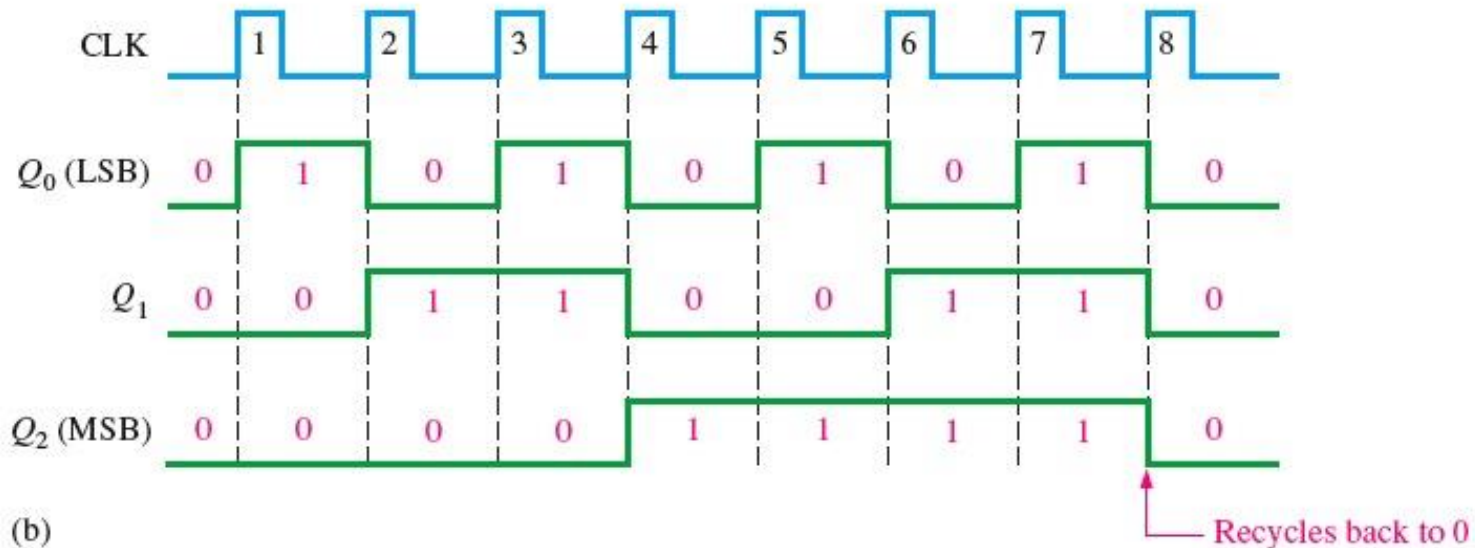
- ∞ CLK1 causes Q to go HIGH and Q' LOW – FF1 is not triggered
- ∞ CLK2 flips FF0 and Q = LOW and Q' = HIGH – FF1 triggers to HIGH
- ∞ CLK3 flips FF0 again and Q' = LOW – FF1 remains at HIGH
- ∞ CLK4 flips to set Q'=HIGH thus FF1 flips its output Q=LOW thus resetting the counter back to all LOW again → *recycled* back to original state

# 3-bit asynchronous binary counter

Similar to the 2-bit but now has 8 states before it recycles



(a)

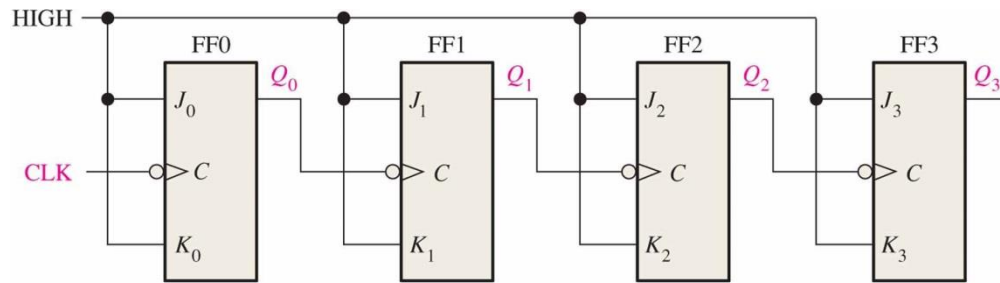


(b)

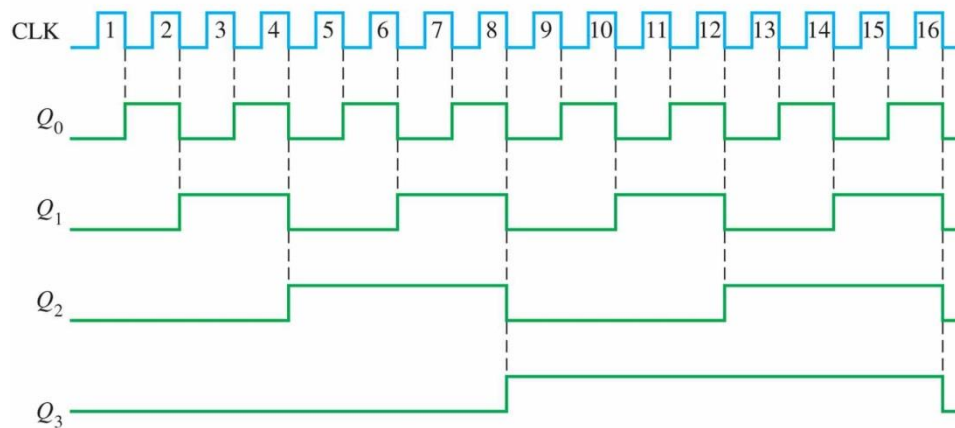


# Exercise

- Building on previous slide content, build the logic circuit and timing diagram for a 4-bit asynchronous binary counter
  - how many states will the counter hold before it resets? **16 states**
  - how many flip-flops are required? **4**



(a)

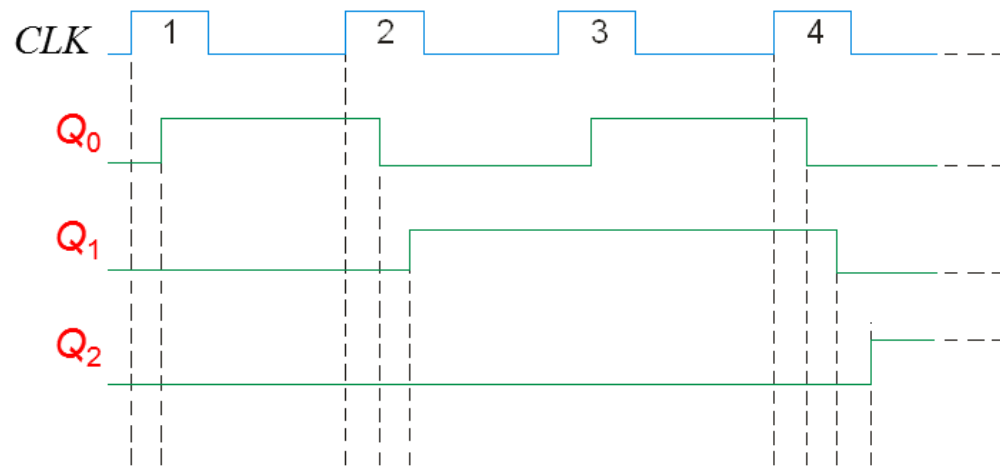


(b)

# Why call them ripple counters?

- Asynchronous counters are sometimes called **ripple** counters, because the stages do not all change together. For certain applications requiring high clock rates, this is a major disadvantage.
- For PDS0101, that is all you need to know

Notice how delays are cumulative as each stage in a counter is clocked later than the previous stage.



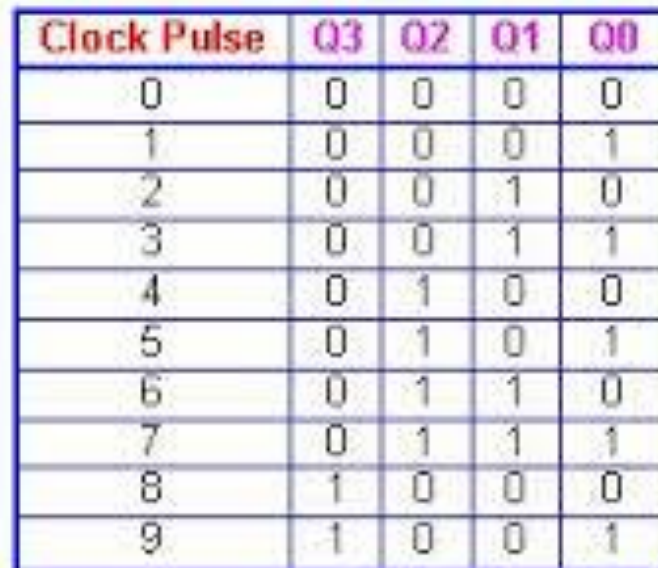
Q<sub>0</sub> is delayed by 1 propagation delay, Q<sub>1</sub> by 2 delays and Q<sub>2</sub> by 3 delays.

# Counter modulus

- ✧ The *modulus* of a counter is the number of unique states that the counter will sequence through before it recycles
- ✧ The usual modulus of a  $n$ -bit counter is to have  $2^n$  unique states in each cycle
- ✧ Counters can be designed to have a number of states in their sequence that is less than the maximum of  $2^n$
- ✧ To obtain a truncated sequence, it is necessary to force the counter to recycle before going through all of its possible states – an example is a asynchronous decade counter using a 4-bit counter as a basis

# Asynchronous decade counter

- ∞ A decade counter cycles through the values of one (1) decade
- ∞ This counter uses partial decoding to recycle the count sequence to zero after the 1001 state.
- ∞ The flip-flops are also trailing-edge triggered, so clocks are derived from the Q outputs.
- ∞ Other truncated sequences can be obtained using a similar technique to obtain other sequences.



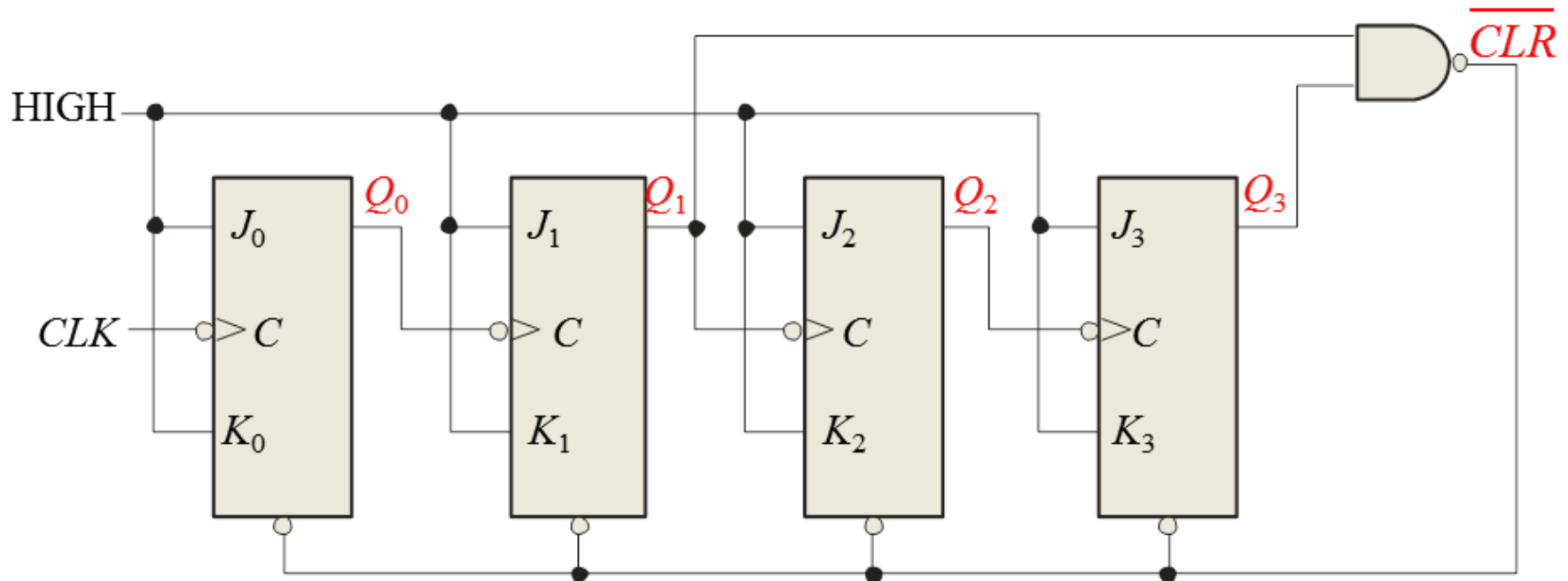
Clock Pulse	Q3	Q2	Q1	Q0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

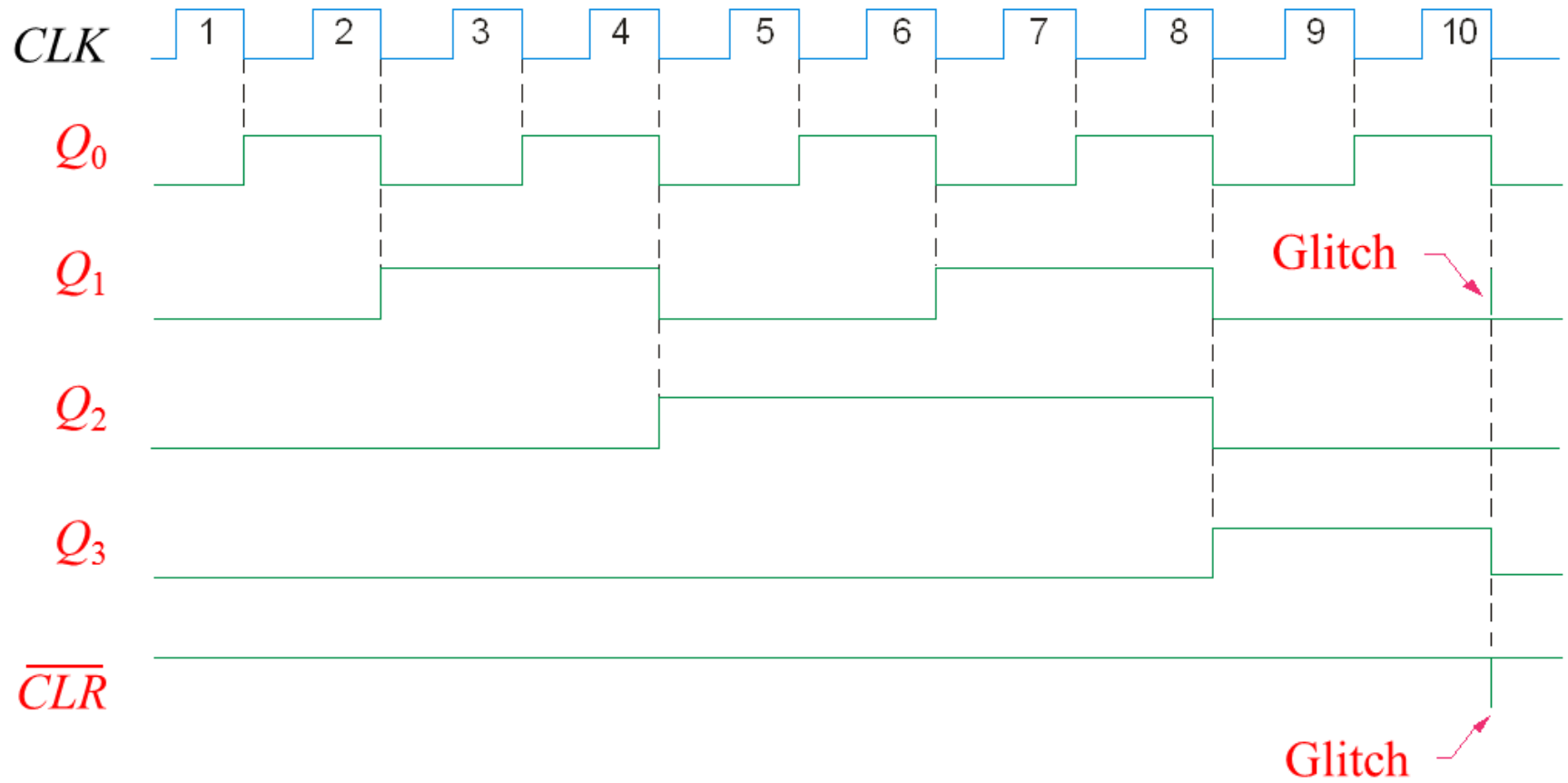
A normal 4-bit async counter recycles when it reaches 1111

CLK	Q3	Q2	Q1	Q0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

Building upon a 4-bit counter, the decade counter should recycle when it detects (decodes) 1010

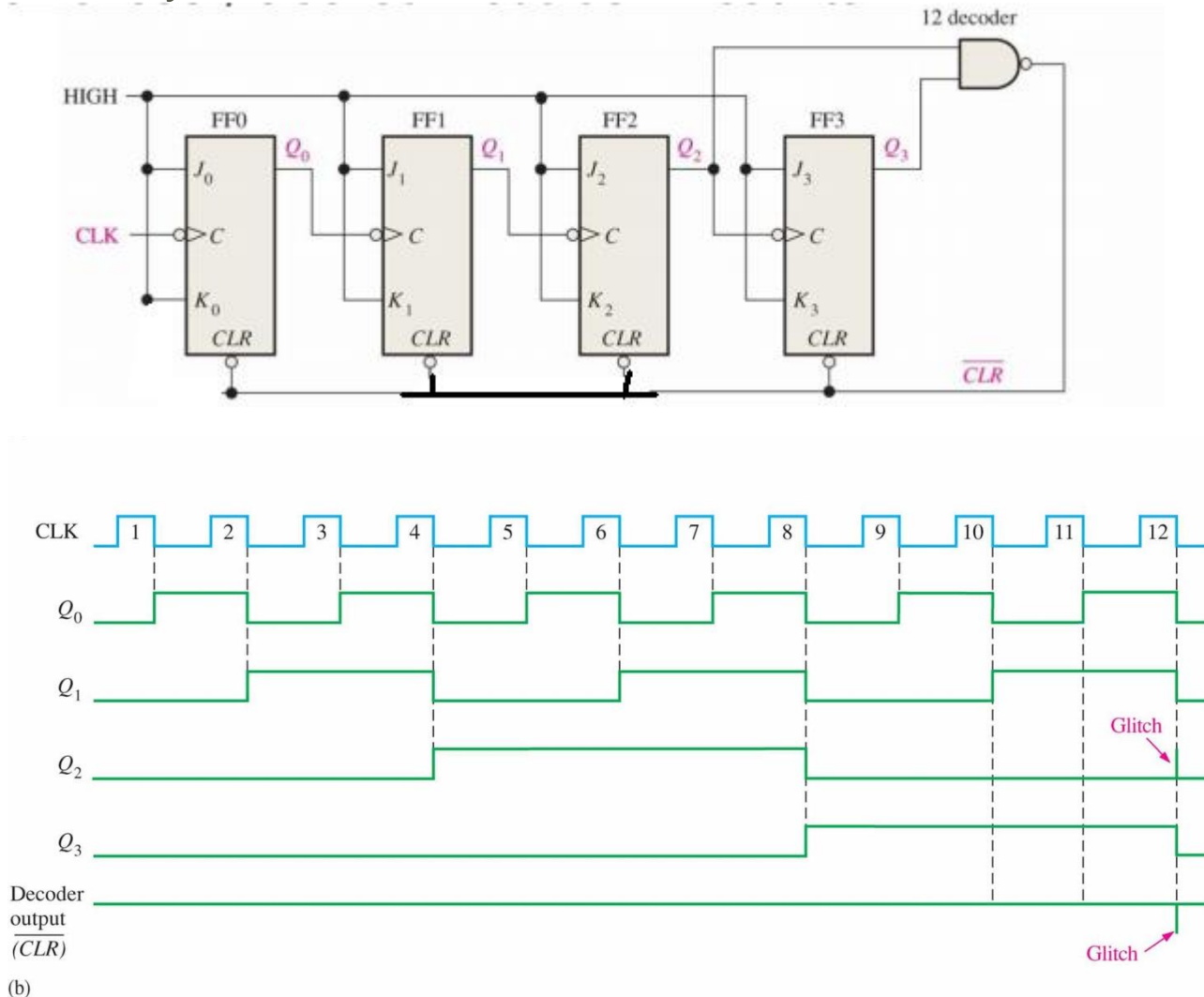
- One way to make the counter recycle after the count of nine (1001) is to decode count ten (1010) with a NAND gate and connect the output of this NAND gate to the clear (CLR) inputs of the flip-flops.
- The inputs of the NAND gate are from the Q output from FF1 and FF3 (from 1010 → FF3 FF2 FF1 FF0)





# Exercise

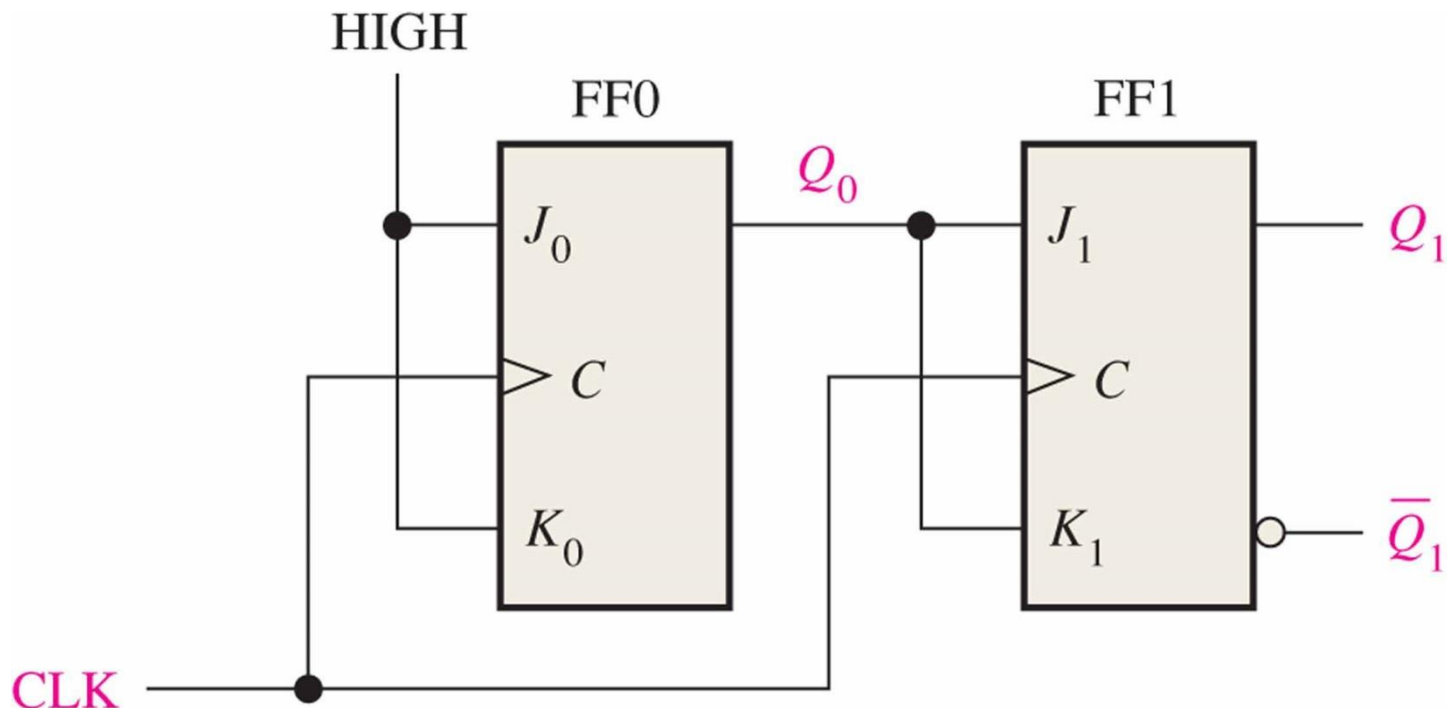
- ✎ Using the same concepts from the decade counter, construct an asynchronously clocked modulus-12 counter





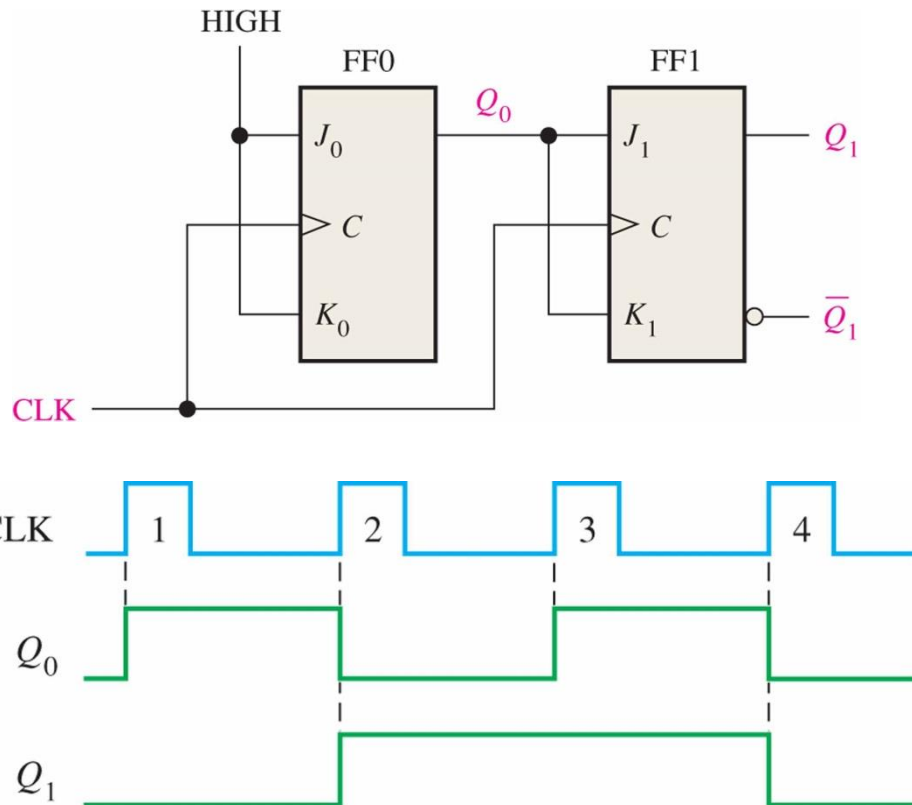
# Synchronous counters

- ✧ In a synchronous counter all flip-flops are clocked together with a common clock pulse.
- ✧ Synchronous counters overcome the disadvantage of accumulated propagation delays, but generally they require more circuitry to control states changes
- ✧ The diagram below shows a 2-bit synchronous binary counter

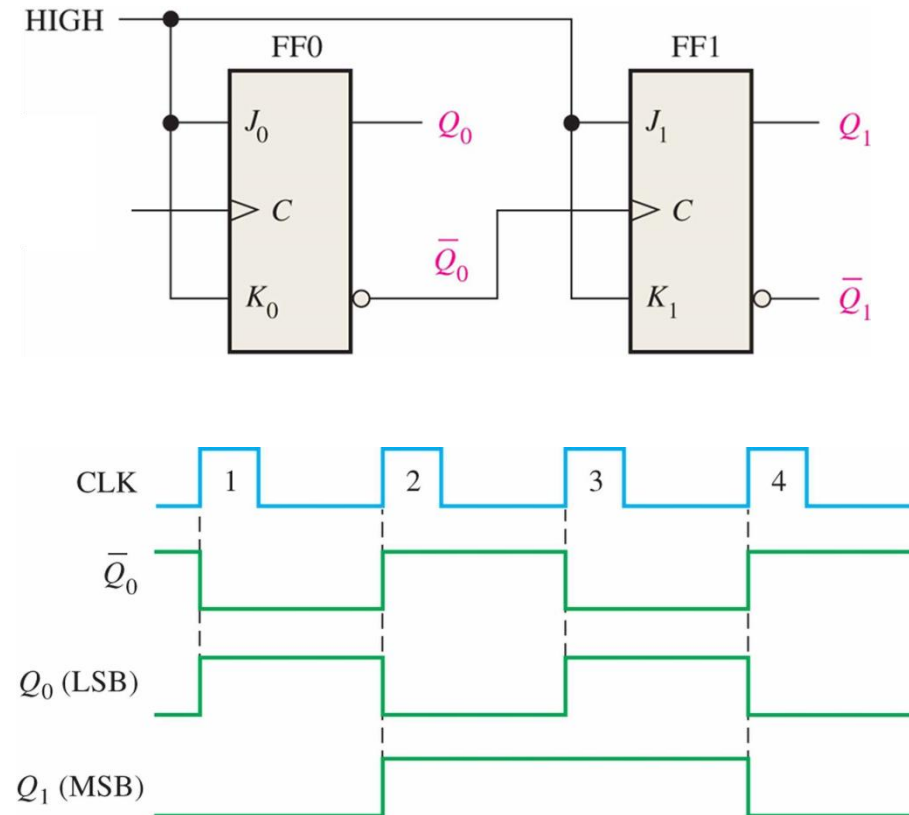


# 2-bit counter timing diagrams

## SYNCHRONOUS

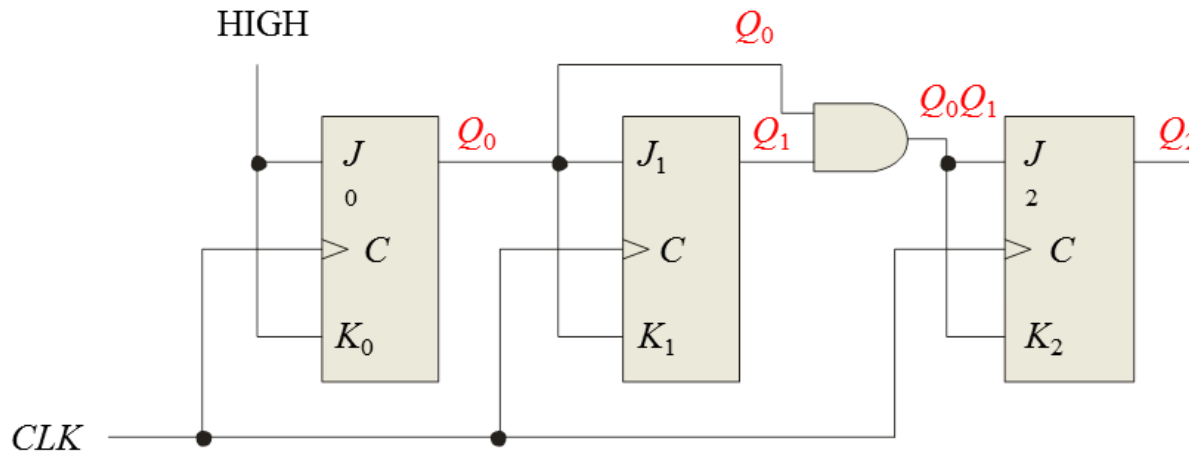


## ASYNCHRONOUS



# 3-bit synchronous counters

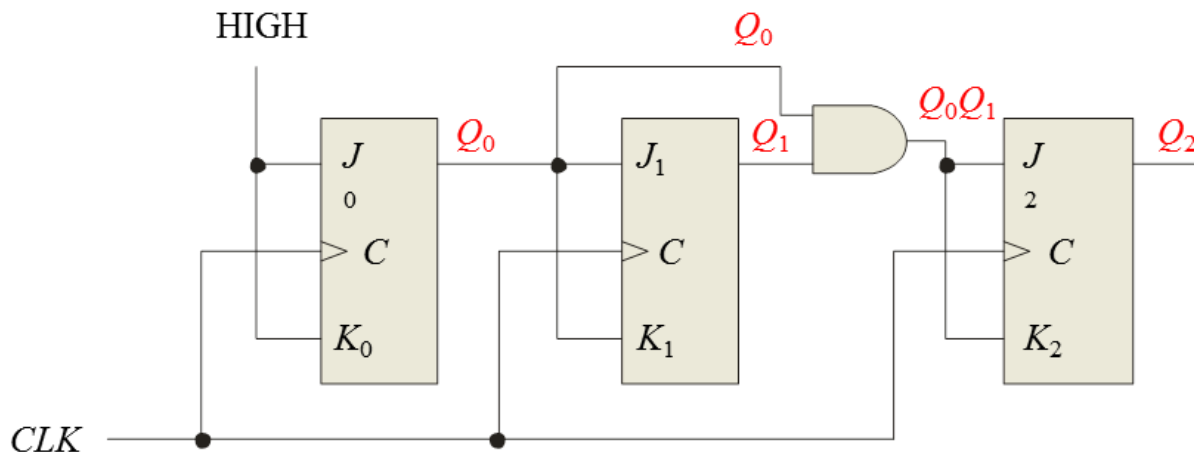
- ∞ The 3-bit synchronous counter circuit is shown below – note the additional gate between FF1 and FF2



- ∞ The counter above will perform the same number cycle iteration as per the 3-bit async counter shown previously – but its truth table is slightly more complicated

# 3-bit sync counter design

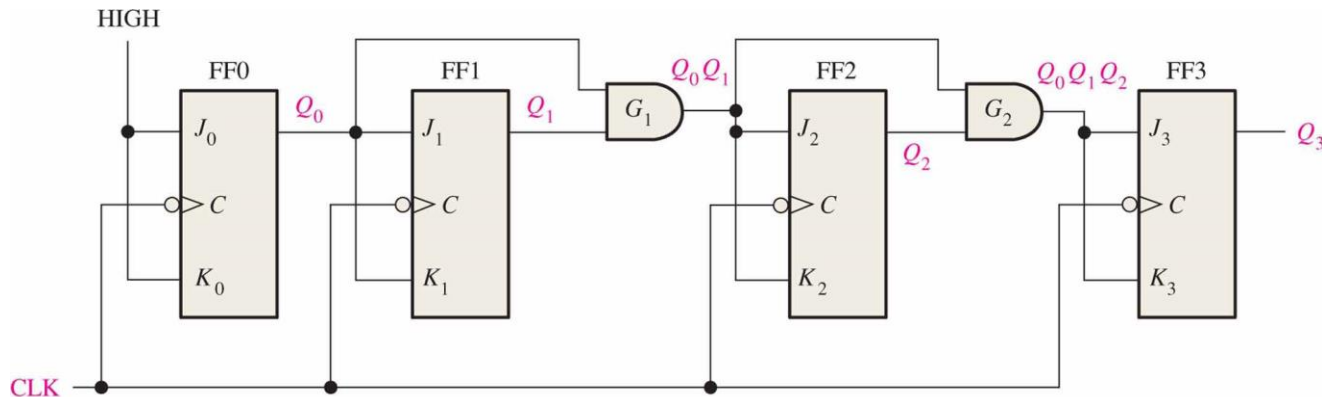
- Q0 alternates on every CLK input – so input to FF0 remains HIGH to result in toggle operation on each CLK pulse
- Q1 toggles each time  $Q_0 = 1$ , so the inputs to FF1 is sourced from the  $Q_0$  from FF0
- Q2 toggles each time  $Q_1=1$  and  $Q_0=1$ , so the input to FF2 is sourced from the ANDed  $Q_x$  outputs from FF0 and FF1



CLK	Q2	Q1	Q0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8 (REPEAT)	0	0	0

# Exercise

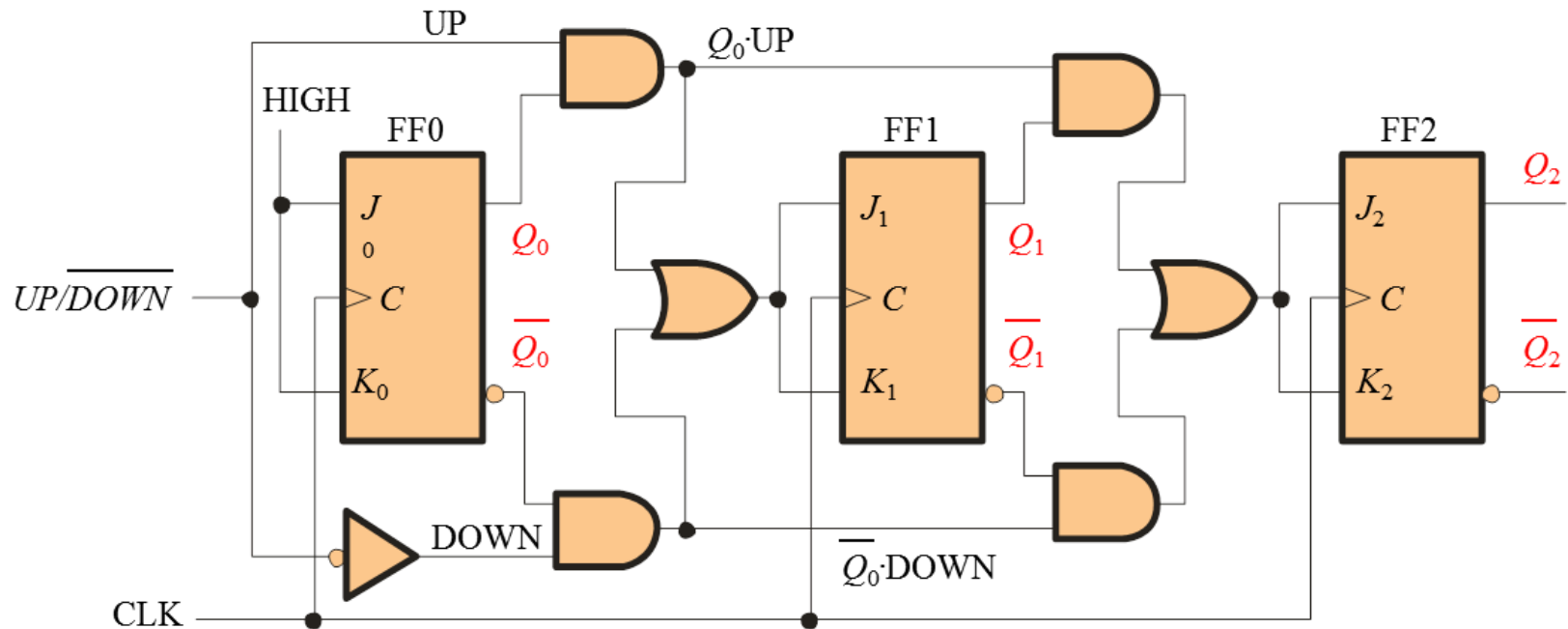
- ✎ Using similar principles from the 3-bit synchronous counter, design the 4-bit synchronous counter.



CLK PULSE	Q3	Q2	Q1	Q0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
16 REPEAT	0	0	0	0

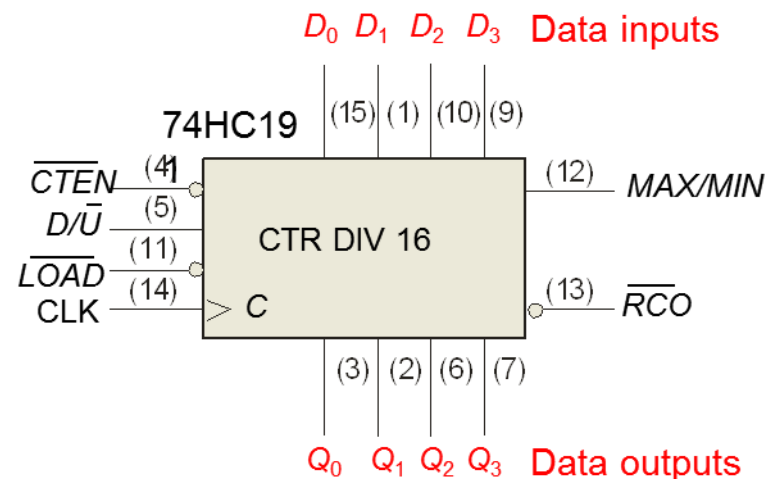
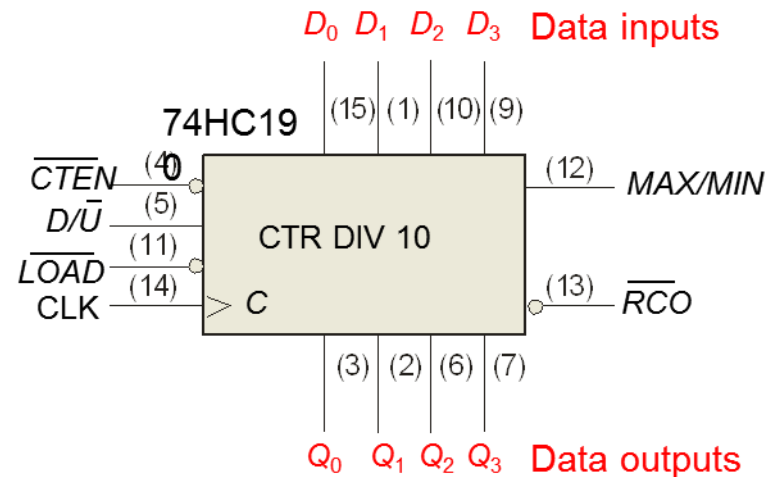
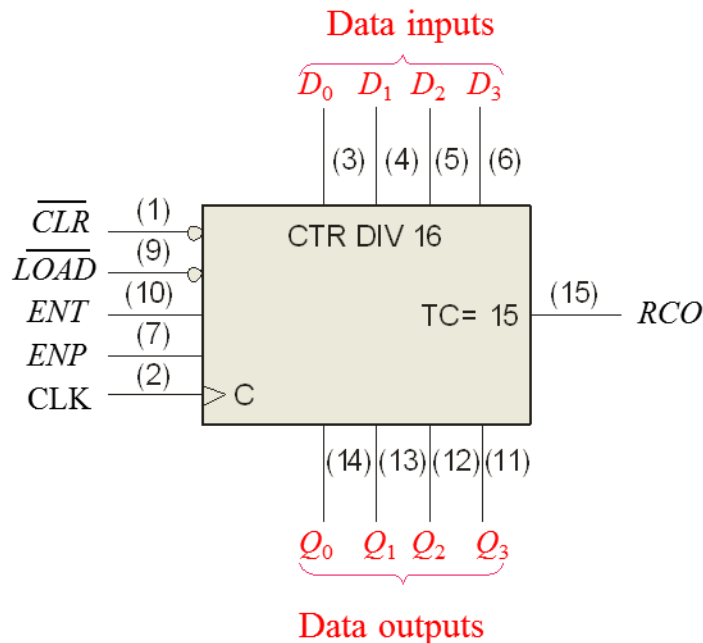
# Up/Down Synchronous Counters

- Up/Down counters are built similar to the counters you have seen earlier but the counter is capable of progressing in either direction depending on a direction control input
- The circuit below shows a 3-bit synchronous counter – now the outputs from each prior FF is ANDed together to enable the desired result



# Common counter ICs

- Many common counters are ready-made in the form of integrated circuits for use



# Summary

- ✧ Counters are logic circuits that cycle through a finite set of states
- ✧ Counters can be asynchronous or synchronous depending on whether all the bits in the counter react simultaneously to the CLK input or not
- ✧ Counters are built up of multiple J-K flip-flops that are connected in sequence – hence creating a sequential logic circuit