Digital Circuits

ECS 371

Dr. Prapun Suksompong

prapun@siit.tu.ac.th

Lecture 22

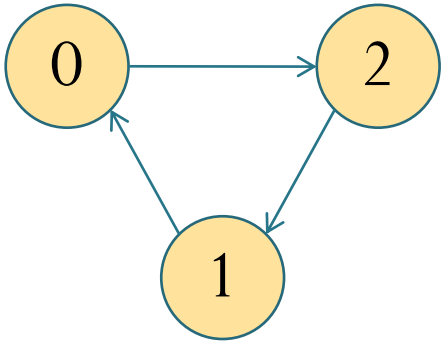
Office Hours:
BKD 3601-7
Monday 9:00-10:30, 1:30-3:30
Tuesday 10:30-11:30

ECS371.PRAPUN.COM

Announcement

- Reading Assignment:
  - Chapter 7: 7-1, 7-2, 7-4
  - Chapter 8: 8-1, 8-2, 8-4, 8-5
  - Chapter 9: 9-1 to 9-5
- HW8 is posted on the web
  - Due date: Next Wednesday (Sep 9)

Example: Arbitrary Counting Sequence



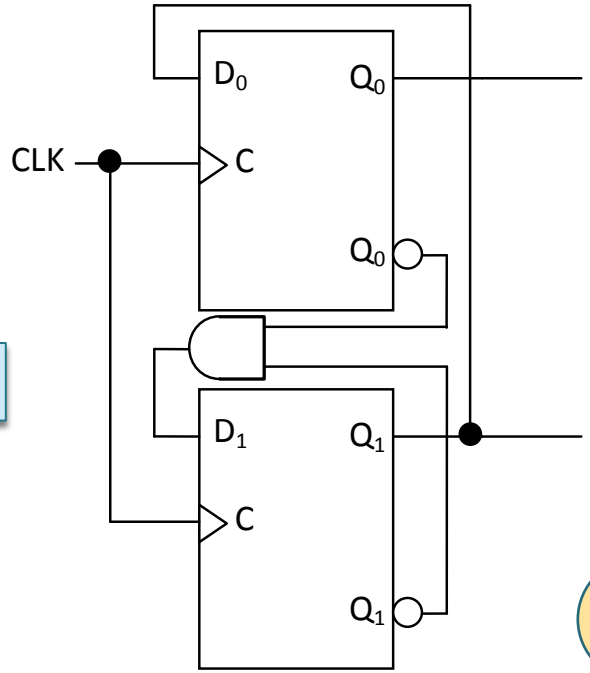
Method 1: Use D FFs

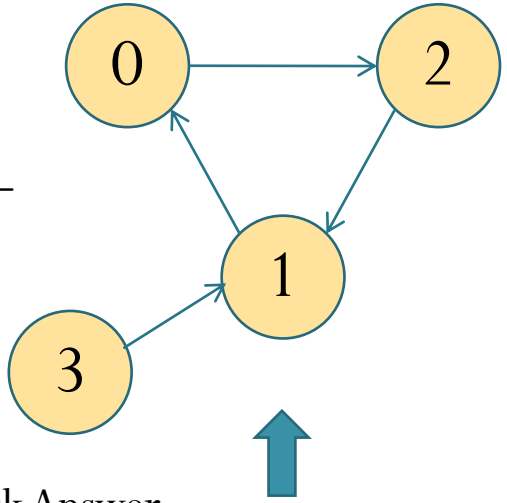
Current State		Next State		FF_1	FF_0
Q_1	Q_0	Q_1	Q_0	D_1	D_0
0	0	1	0	1	0
0	1	0	0	0	0
1	0	0	1	0	1
1	1	X	X	X	X

K-maps

$$D_1 = \overline{Q_1} \cdot \overline{Q_0}$$

$$D_0 = Q_1$$

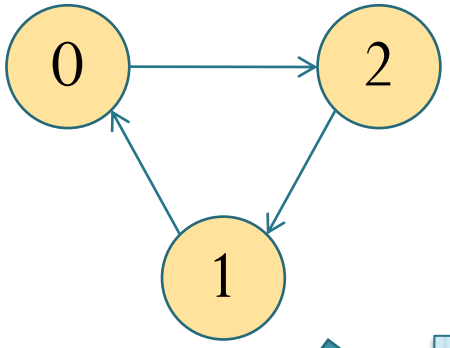




Check Answer

Current State		FF_1	FF_0	Next State	
Q_1	Q_0	$D_1 = \overline{Q_1} \cdot \overline{Q_0}$	$D_0 = Q_1$	Q_1	Q_0
0	0	1	0	1	0
0	1	0	0	0	0
1	0	0	1	0	1
1	1	0	1	0	1

Example: Arbitrary Counting Sequence

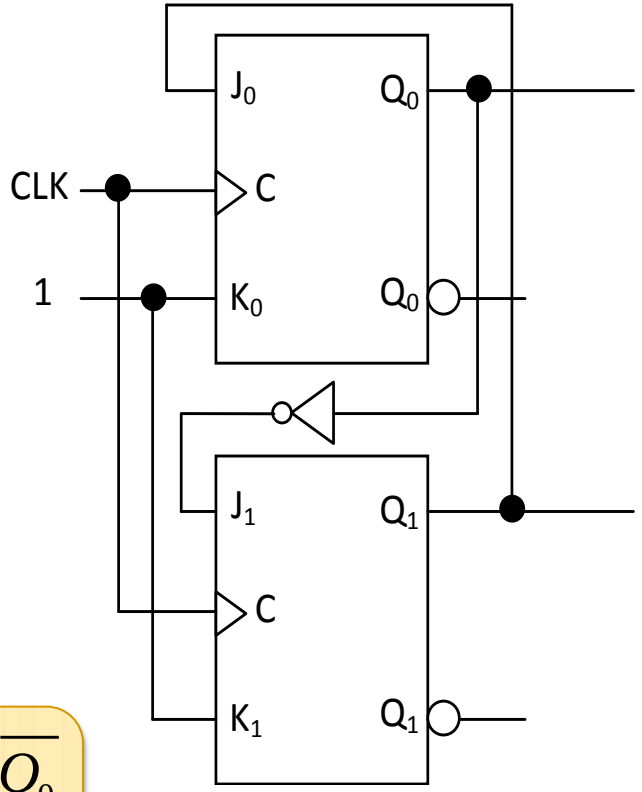


Method 2: Use J-K FFs

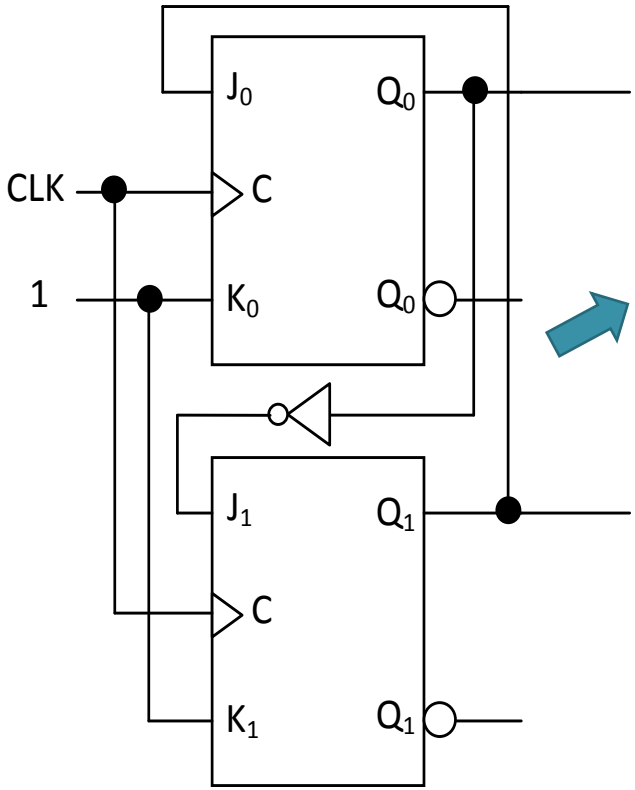
Current State		Next State		FF_1		FF_2	
Q_1	Q_0	Q_1	Q_0	J_1	K_1	J_0	K_0
0	0	1	0	1	X	0	X
0	1	0	0	0	X	X	1
1	0	0	1	X	1	1	X
1	1	X	X	X	X	X	X

K-maps

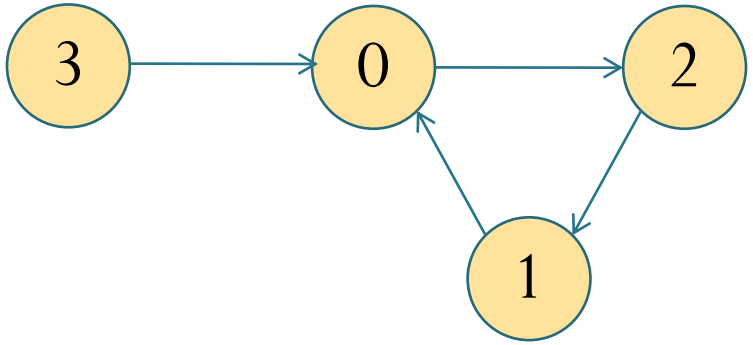
$$J_1 = \overline{Q_0}$$
$$K_1 = 1$$
$$J_0 = Q_1$$
$$K_0 = 1$$



Example: Check Answer

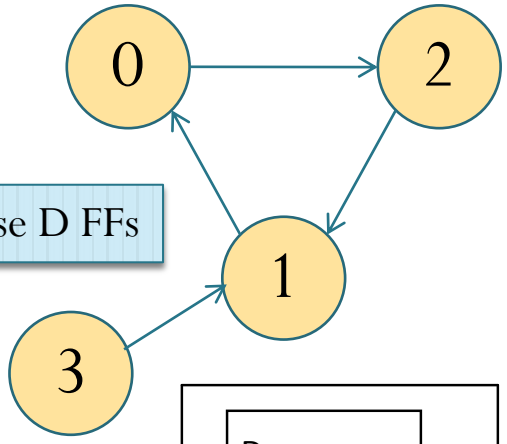


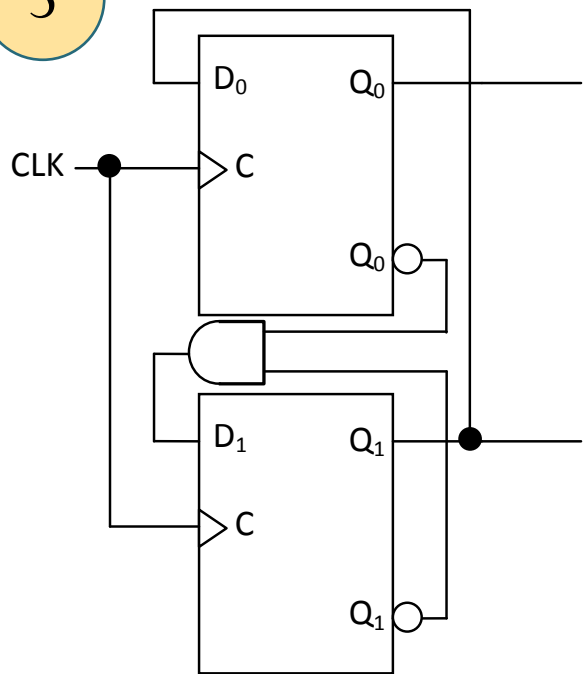
Current State		FF_1		FF_2		Next State	
Q_1	Q_0	$J_1 = \overline{Q_0}$	$K_1 = 1$	$J_0 = Q_1$	$K_0 = 1$	Q_1	Q_0
0	0	1	1	0	1	1	0
0	1	0	1	0	1	0	0
1	0	1	1	1	1	0	1
1	1	0	1	1	1	0	0



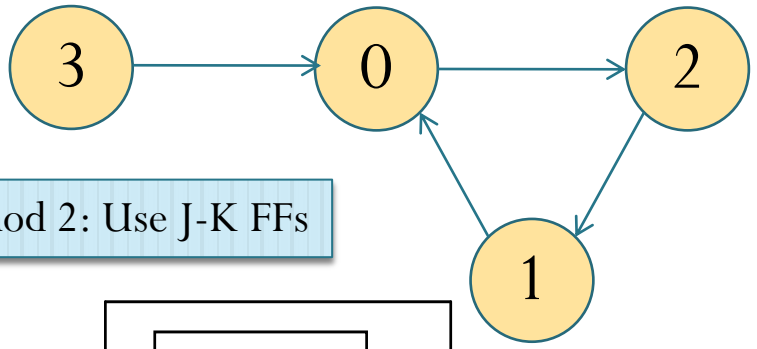
Example: Comparison

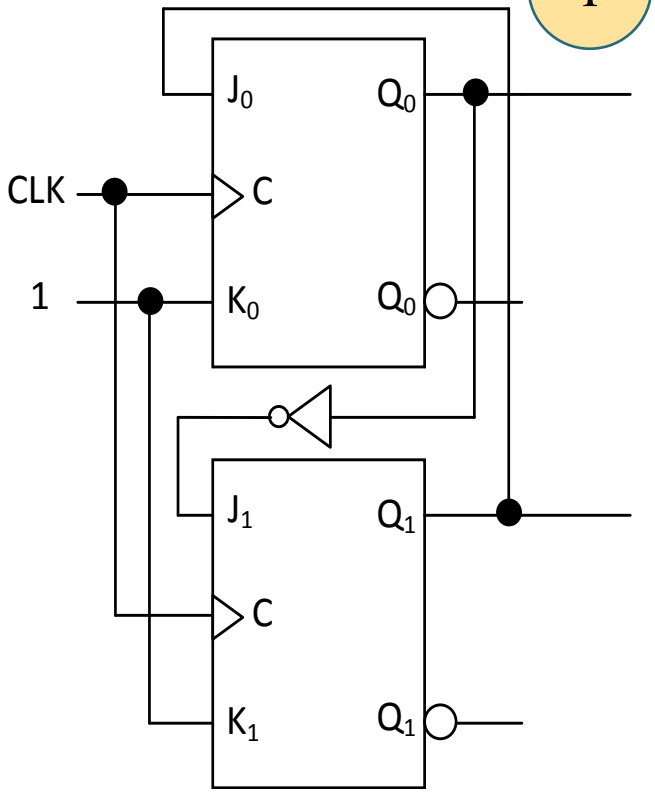
Method 1: Use D FFs





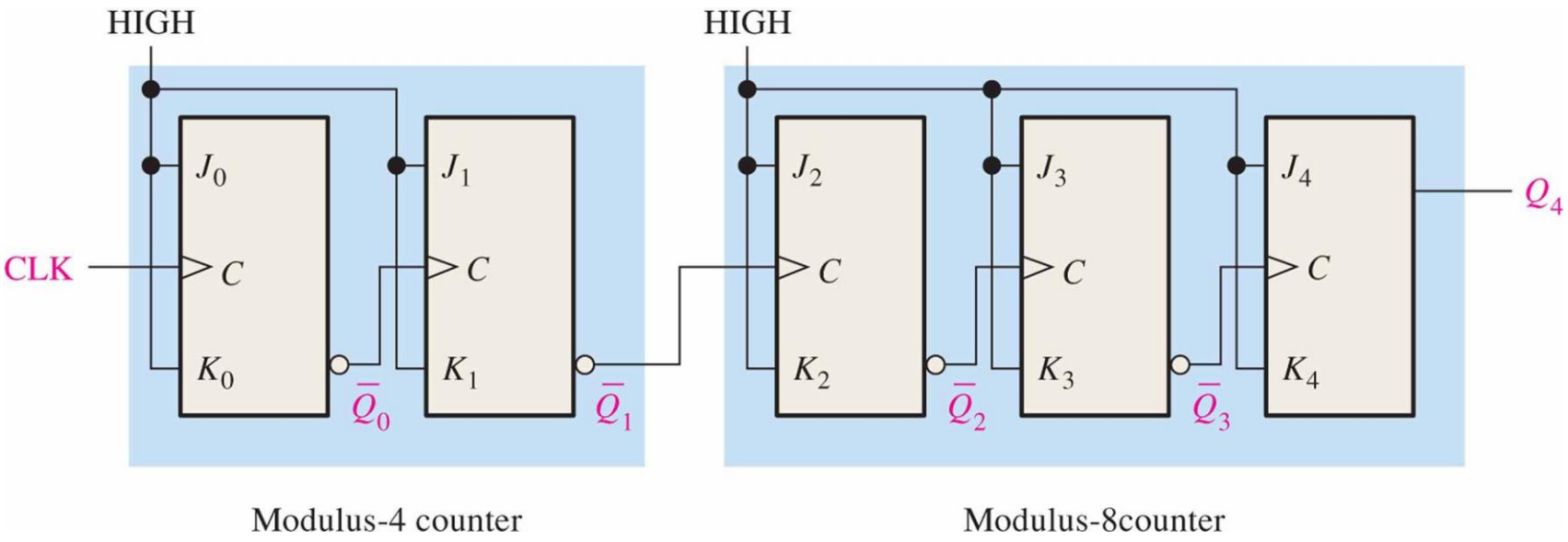
Method 2: Use J-K FFs





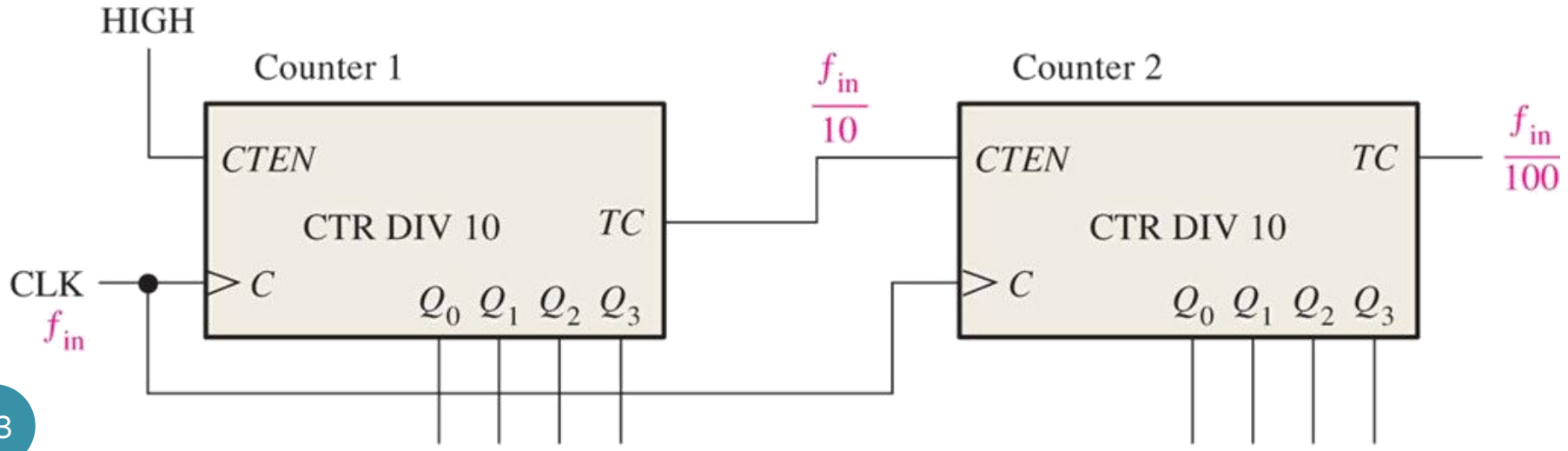
Cascaded Counters

- Cascading is a method of achieving higher-modulus counters.
- In essence, **cascading** means that the last-stage output of one counter drives the input of the next counter.
- Asynchronous Cascading:



Synchronous Cascading

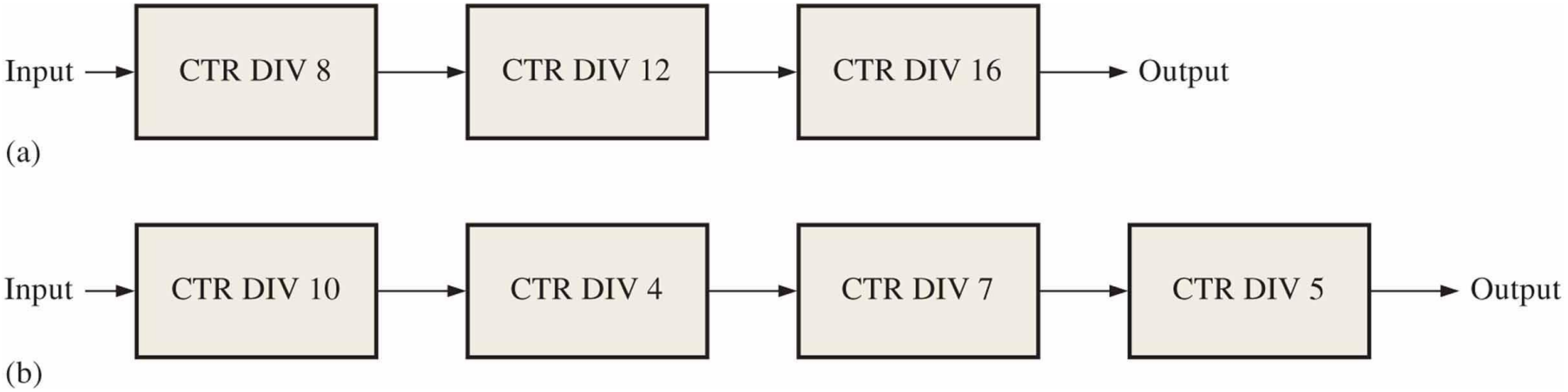
- Use
  1. count enable (CTEN)
     - On some devices, CTEN is labeled as G
  2. terminal count (TC)
     - Same as ripple clock output (RCO) on some IC counters
- The next counter is enabled only when the TC of the previous stage is reached.



Example:

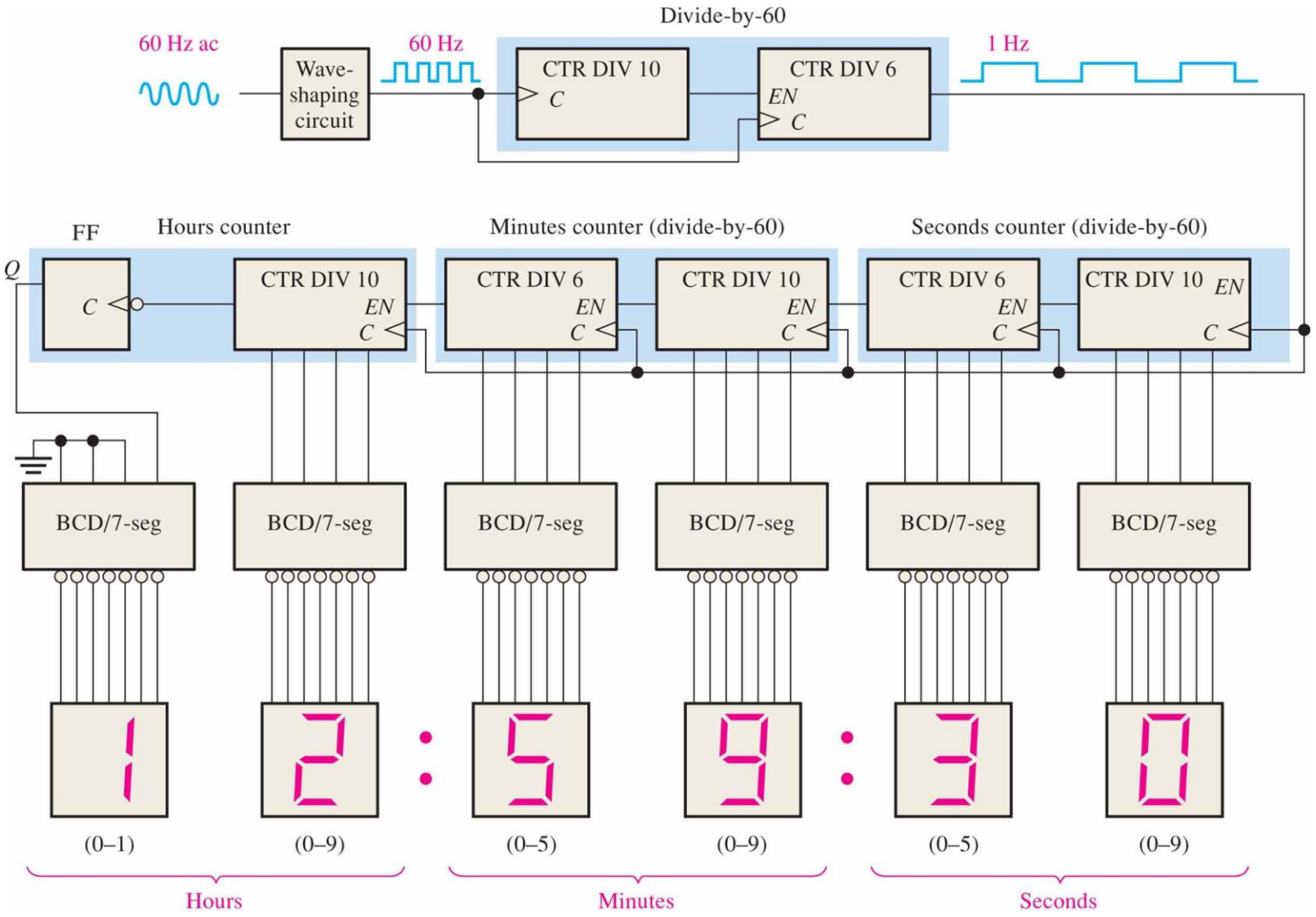
Determine the overall modulus of the following cascaded counters:

The over all modulus is $8 \times 12 \times 6 = 1536$



The over all modulus is

Application: Digital Clock



Register

- Flip-flops are used in a variety of application circuits, the most common among these being
  - counting circuits and
  - frequency division and
  - data storage and transfer (data movement) circuits.
- Counters and registers
  - Belong to the category of MSI sequential logic circuits.
  - Similar architecture: both comprise a cascaded arrangement of more than one FFs.
  - Both constitute very important building blocks of sequential logic

Register

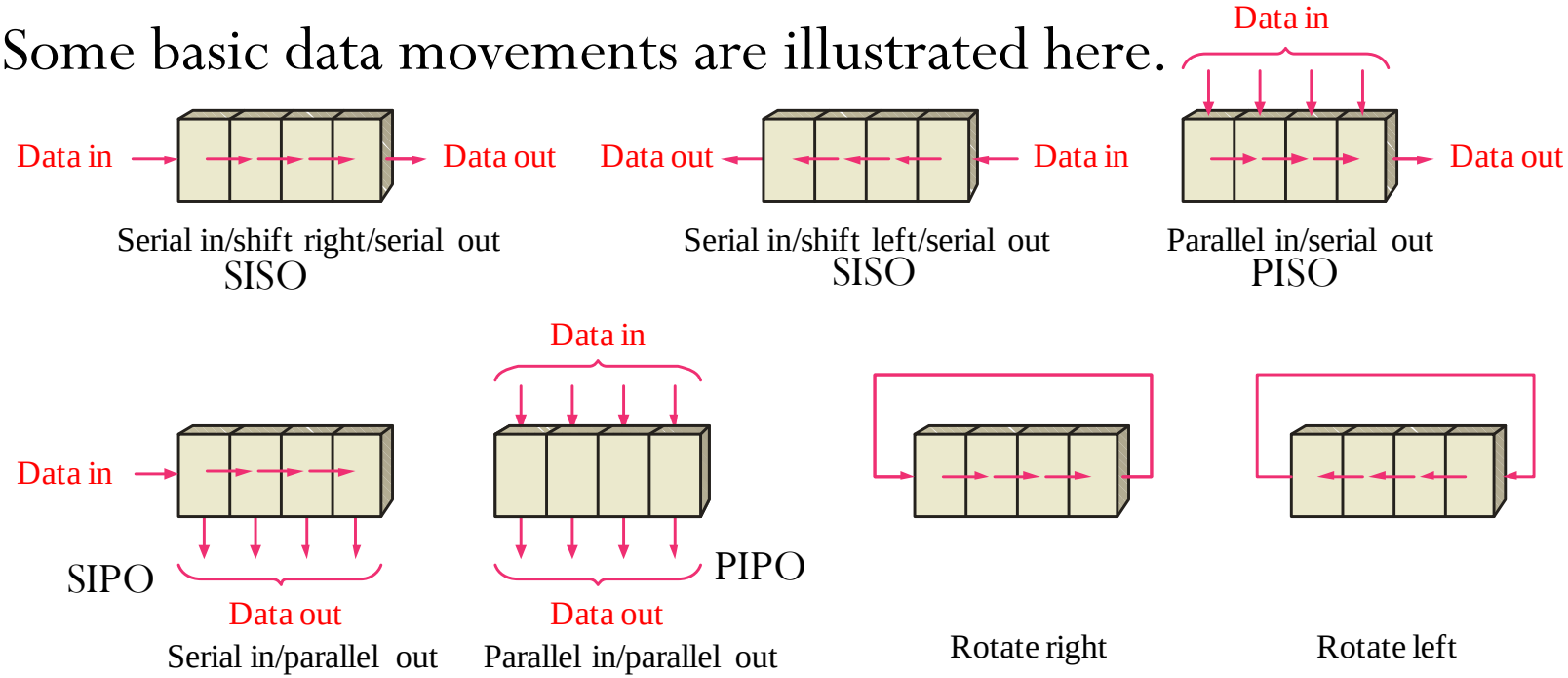
- A register, unlike a counter, has no specified sequence of states, except in certain very specialized applications.
- Application
  - Counters are mainly used in counting applications, where they either measure the time interval between two unknown time instants or measure the frequency of a given signal.
  - Registers are primarily used for the *temporary* storage of data present at the output of a digital circuit before they are fed to another digital circuit.

Register and Shift Register

- The **storage capacity** of a register is the total number of bits (1s and 0s) of digital data it can retain.
  - Since each FF can store one bit of data, the storage capacity of the shift register equals the number of FFs used.
- The **shift capability** of a register permits the movement of data from stage to stage within the register or into or out of the register upon application of clock pulses.

Shift Register

- A shift register is an arrangement of flip-flops with important applications in storage and movement of data.
- Some basic data movements are illustrated here.

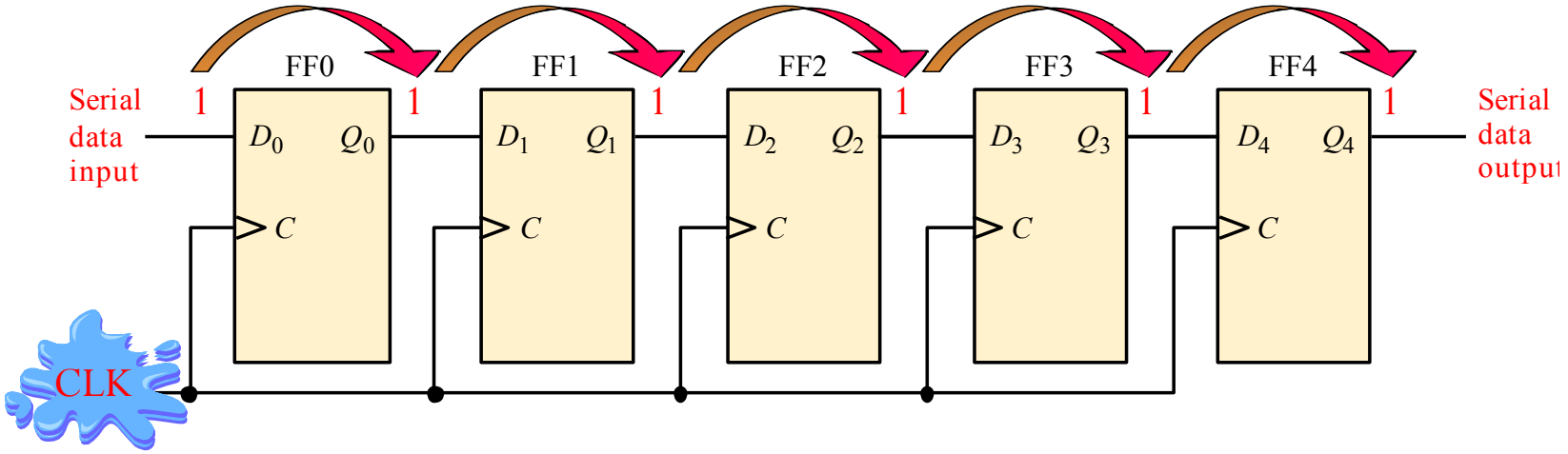


- The basic building block in all shift registers is the FF, mainly a D-FF.

Although in many of the commercial shift register ICs their internal circuit diagram might indicate the use of S-R FFs, a careful examination will reveal that these S-R FFs have been wired as D FFs only.

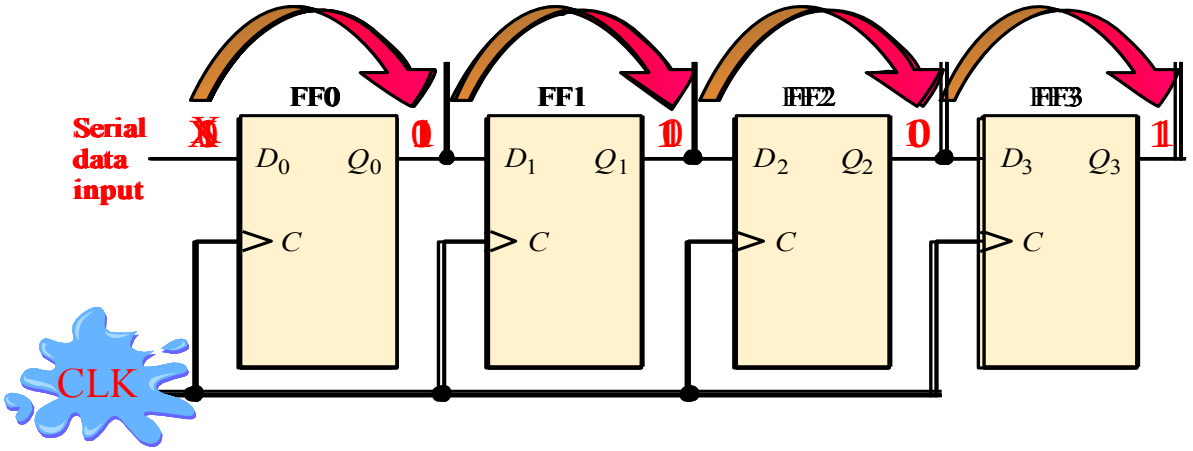
Serial-in/Serial out Shift Register

- Accept data serially: one bit at a time on a single line.
- Each clock pulse will move an input bit to the next FF. For example, a 1 is shown as it moves across.
- Five-bit serial-in serial-out register.



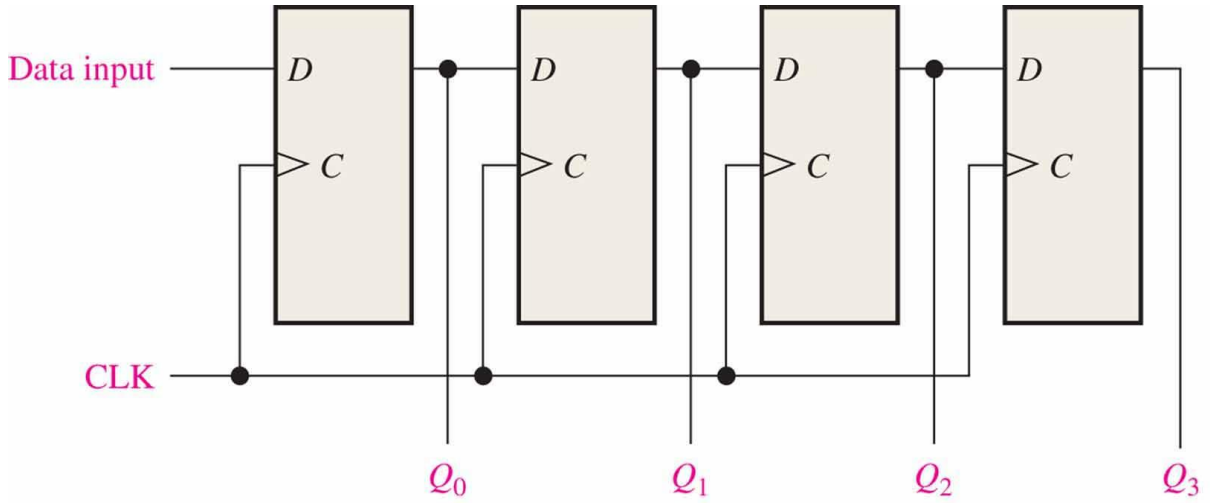
Basic Application: SIPO

- An application of shift registers is conversion of serial data to parallel form.
- For example, assume the binary number 1101 is loaded sequentially, one bit at each clock pulse.

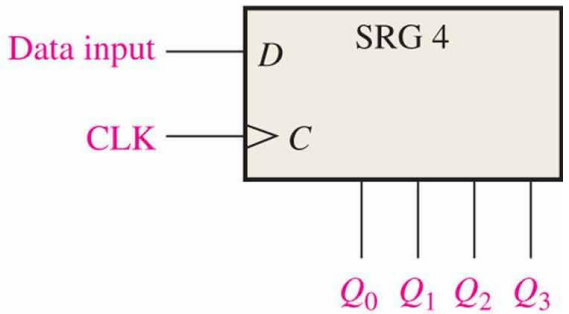


- After 4 clock pulses, the data is available at the parallel output.
  - They can be stored for any length of time as long as the FFs have dc power.

SIPO Shift Register

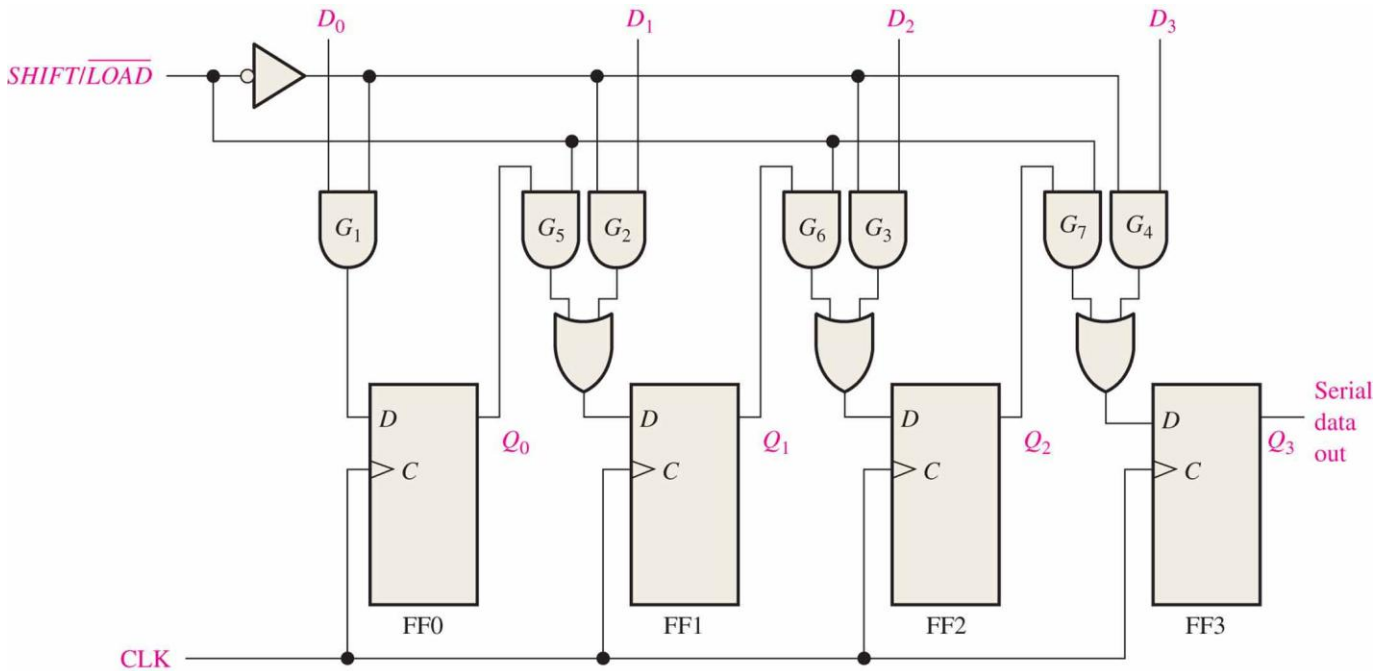


(a)

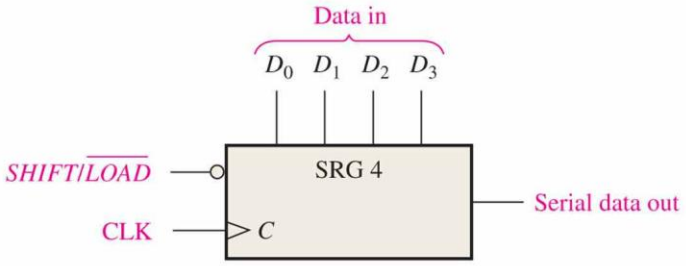


(b)

PISO Shift Register

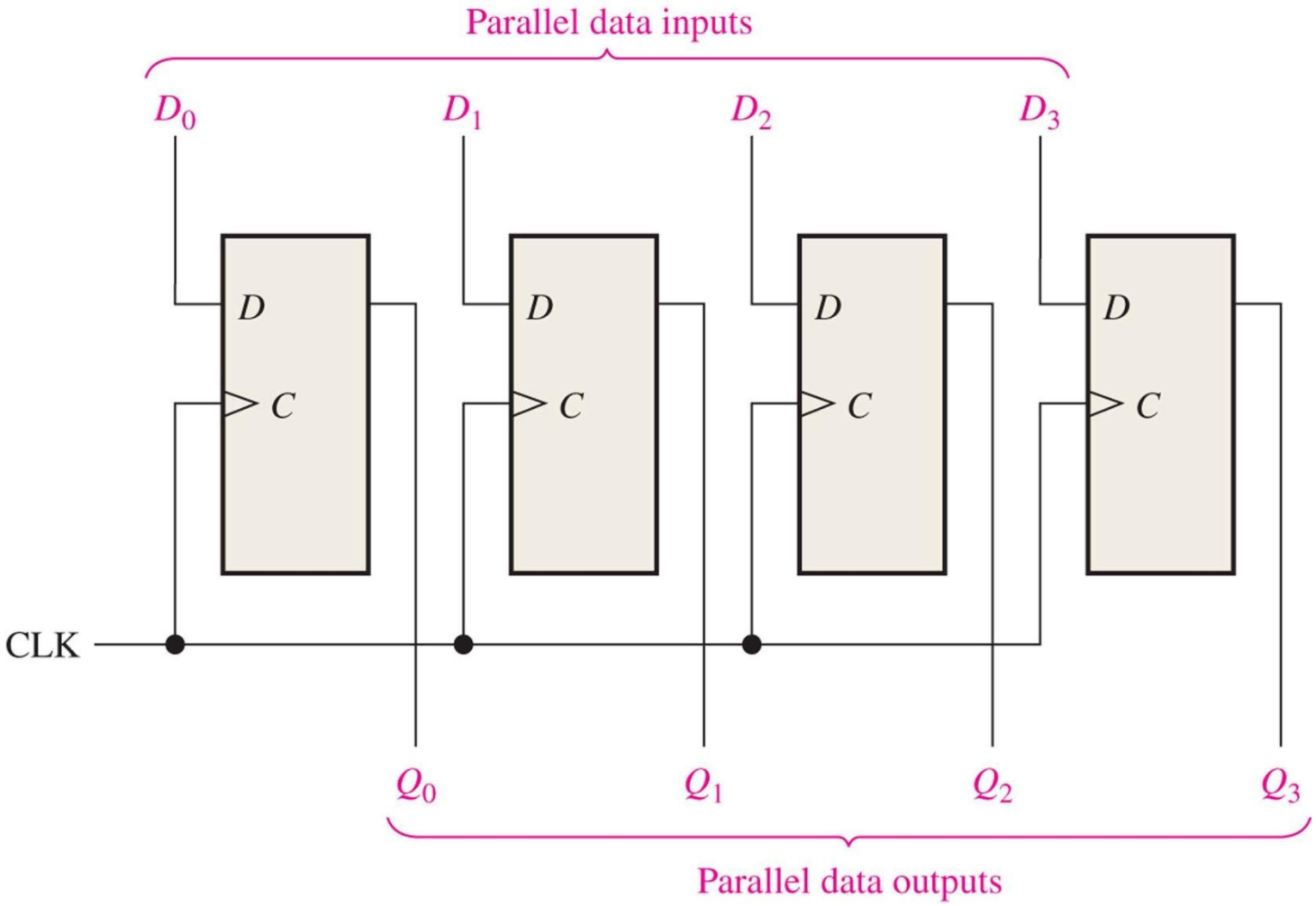


(a) Logic diagram



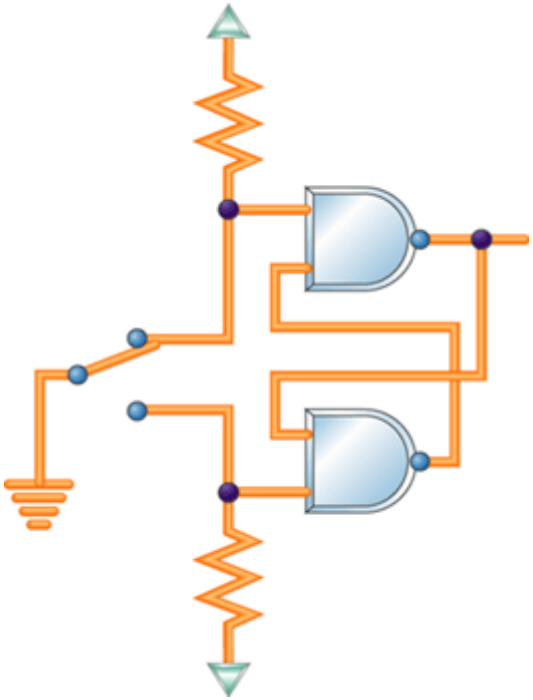
(b) Logic symbol

PIPO Shift Register

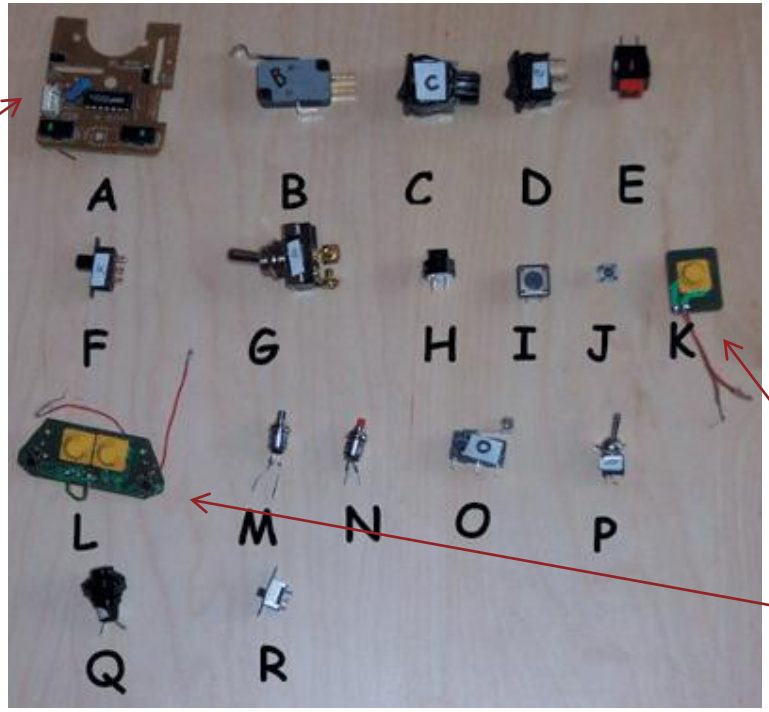


Latch as a Contact-Bounce Eliminator

- When the contacts of any mechanical switch bang together they rebound a bit before settling, causing bounce.
- Debouncing is the process of removing the bounces.



Mouse button from an ancient Compaq computer

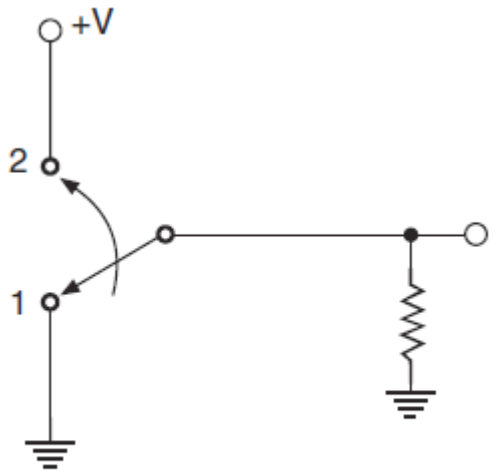


The trigger switches from an old cheap game-playing joystick.

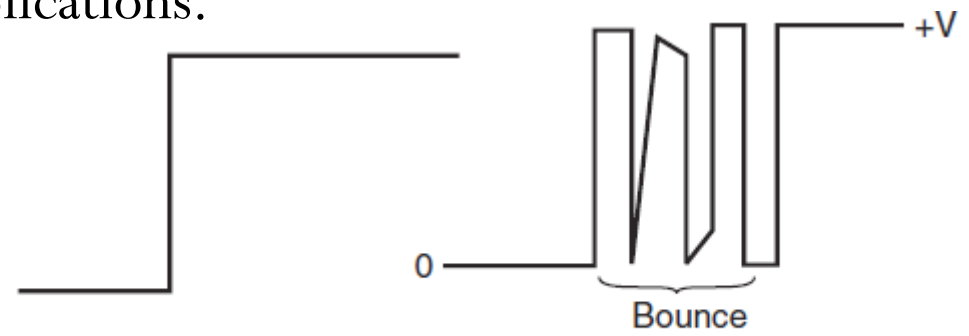
http://www.embedded.com/columns/breakpoint/18400810?_requestid=100290

Switch Bounce Phenomenon

- Mechanical switch cannot be used as such to produce a clean voltage transition.
- The output makes several transitions between 0 and +V volts for a few milliseconds owing to contact bounce before it finally settles at +V volts.
- Although this random behavior lasts only for a few milliseconds, it is unacceptable for many digital circuit applications.

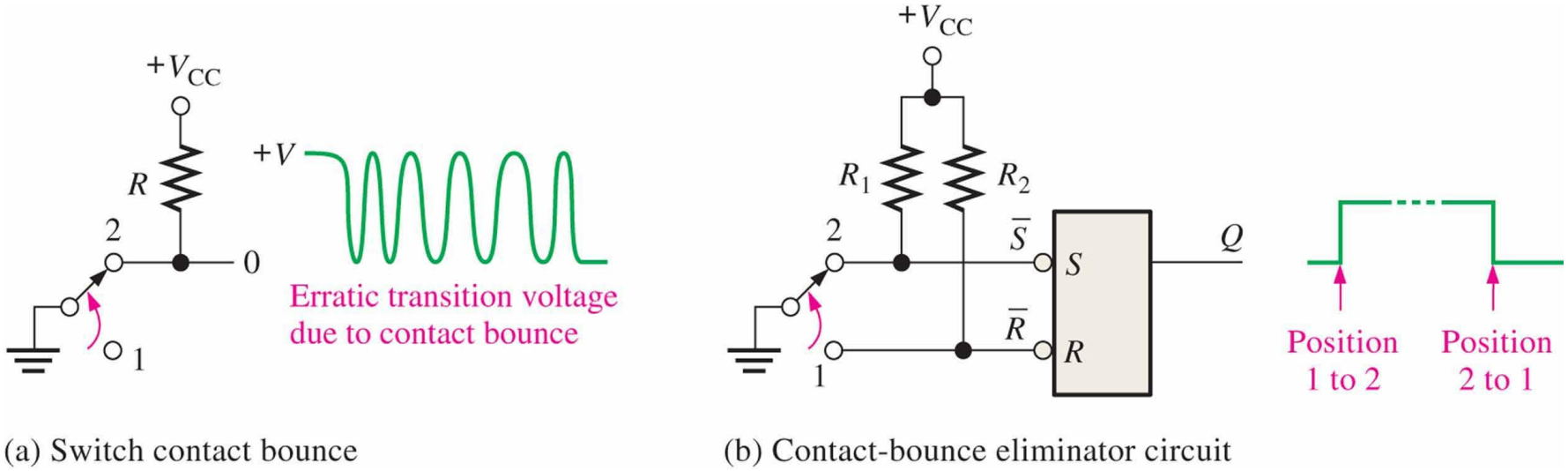


(a)



(b) (c)

Contact-Bounce Eliminator (2)



(a) Switch contact bounce

(b) Contact-bounce eliminator circuit

- Key Idea: The S-R latch is in "hold" mode for each bounce and hence the final output does not change.