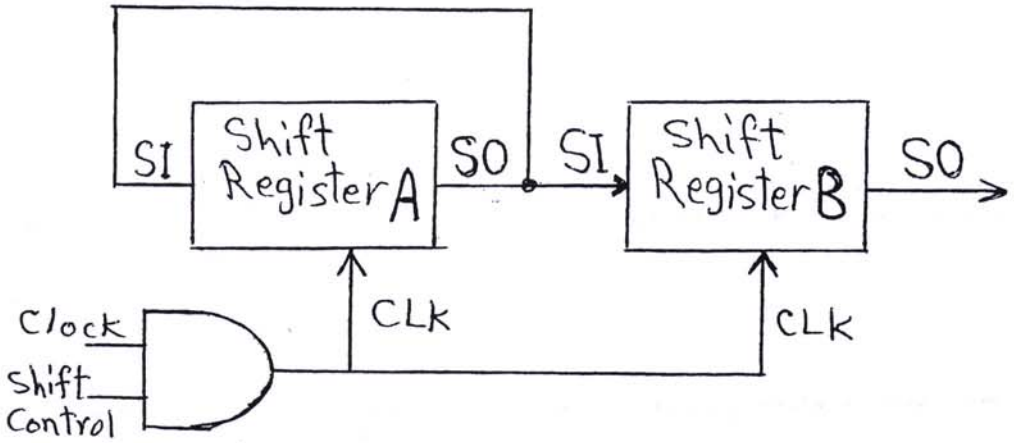
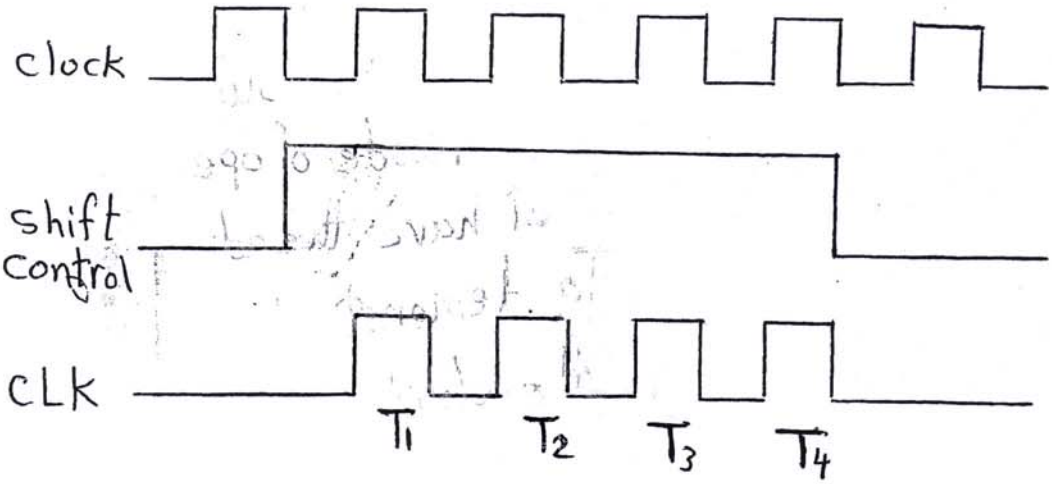
Serial Transfer

A digita system is said to operate in a serial mode when information is transferred and manipulated one bit at a time. This in contrast to parallel transfer where all the bits of the register are transferred at the same time.

Ex: Assume that the binary content of Register A before the shift is 1011 and that of B is 0010. What is the and how many clocks the serial transfer from A to B occurs.



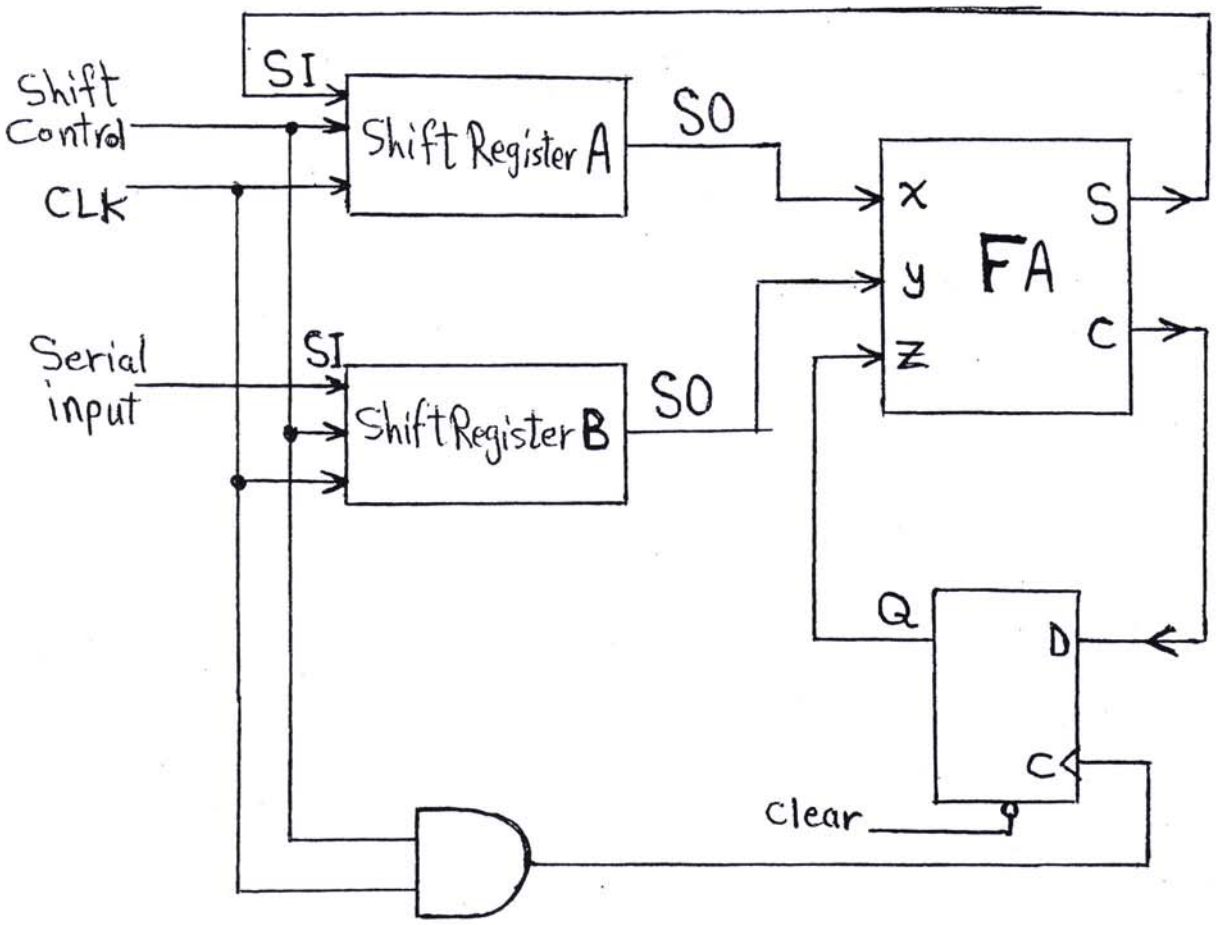


Soln: The serial output (SO) of register A is connected to the serial input (SI) of register B. To prevent the loss of information stored in the source register, the information in register A is made to circulate by connecting the serial output to its serial input. The serial transfer from A to B occurs in four steps.

Timing Pulse	Shift Register A	Shift Register B
Initial value	1 0 1 1	0 0 1 0
After T_1	1 1 0 1	1 0 0 1
After T_2	1 1 1 0	1 1 0 0
After T_3	0 1 1 1	0 1 1 0
After T_4	1 0 1 1	1 0 1 1

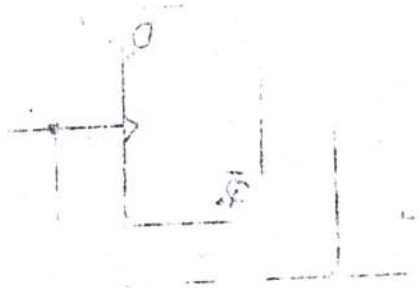
Serial Addition

Operations in digital computers are usually done in parallel because this is a faster mode of operation. Serial operations are slower, but have the advantage of requiring less equipment. To demonstrate the serial mode of operation, the derign of a serial adder is presented in the next page.



H.W: Redesign the serial adder with JK flip-flop for Q.

Counters



Chapter - Five
Registers

Introduction :

Registers are a type of sequential logic circuit closely related to digital counters. A register is a group of memory elements that work together as a unit. The simplest registers do nothing more than store a binary word (digital data); others modify the stored word by shifting its bits left or right or by performing other operations to be discussed in this chapter.

5.2 Parallel - In/Parallel - out (Buffer) Registers :

A buffer - register is the simplest kind of register; all it does is store a digital word. Fig.(5.1) shows a buffer register built with positive - going edge clocking D - flip-flops. The X - bits set up the flip-flops for loading. Therefore, when the first positive clock edge arrives, the stored word becomes $Q_3 Q_2 Q_1 Q_0 = X_3 X_2 X_1 X_0$, i.e. $Q = X$.

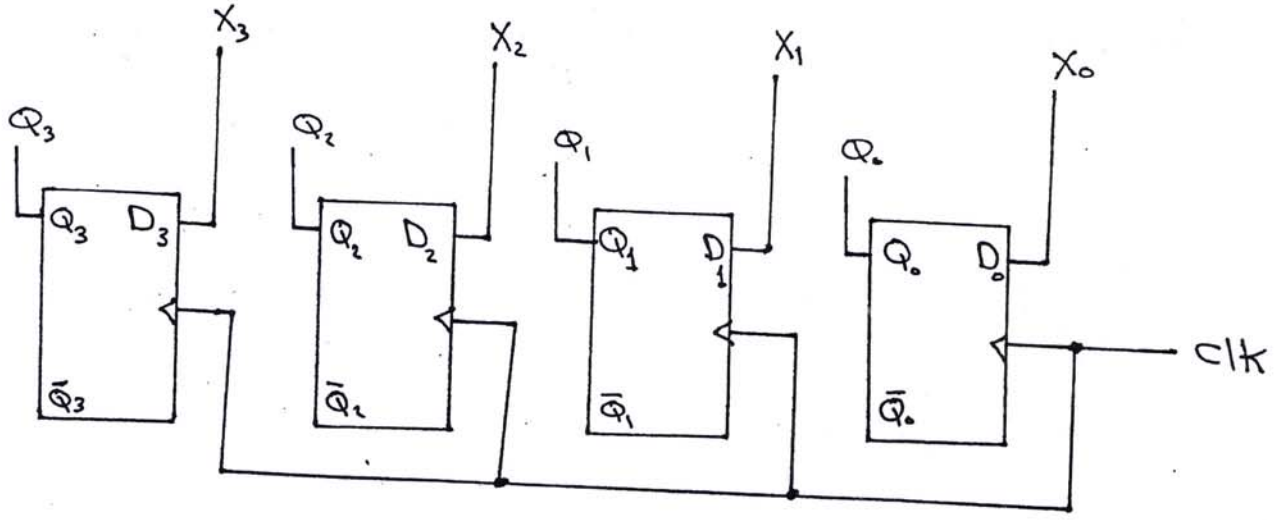


Fig. (5.1) : 4-bit Buffer Register

5.3 Shift Registers :

A shift register moves the stored bits left or right. This bit shifting is essential for certain arithmetic and logic operations used in microcomputers.

5.3.1 Shift-Left Register:

Fig. (5.4) is a shift-left register. As shown, "Din" sets up the right flip-flop, "Q_0" sets up the second flip-flop, "Q_1" the third, and so on. When the next positive clock edge strikes, therefore, the stored bits moves one position to the left.

As an example, here's what happens with "Din = 1" and "Q = 0000". All data inputs except the one on the right are "0s". The arrival of the first rising clock edge sets the right flip-flop, and the stored word becomes "Q = 0001".

This new word means "D_1" now equals "1", as well as "D_0=1". When the next positive clock edge hits, the "Q_1" flip-flop sets and the register contents become "Q = 0011". The third positive clock edge result in "Q = 0111", and the fourth positive clock edge gives "Q = 1111". Hereafter, the stored word is unchanged as long as "Din = 1".

Suppose "Din" is now changed to "0". Then, successive clock pulses produce these register contents:

Q = 1110
Q = 1100
Q = 1000
Q = 0000

As long as "D_{in} = 0", subsequent clock pulses have no fu-

ather effect. The above discussion is summarizes in the timing diagram shown in Fig. (5.5).

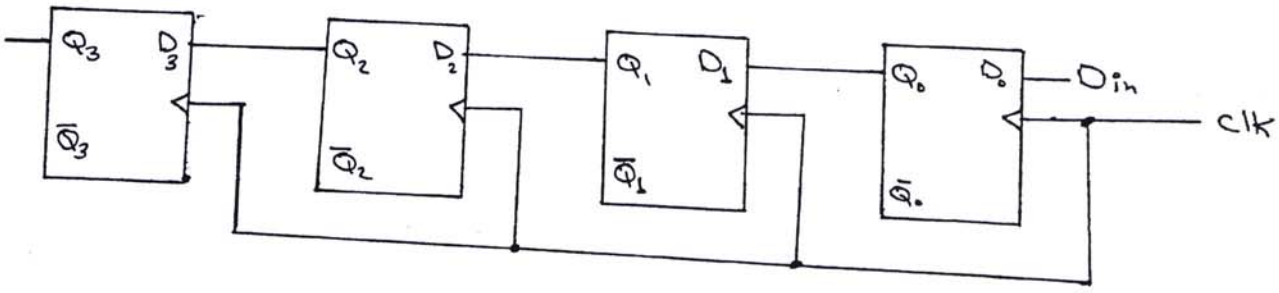


Fig. (5.4): Shift-left register

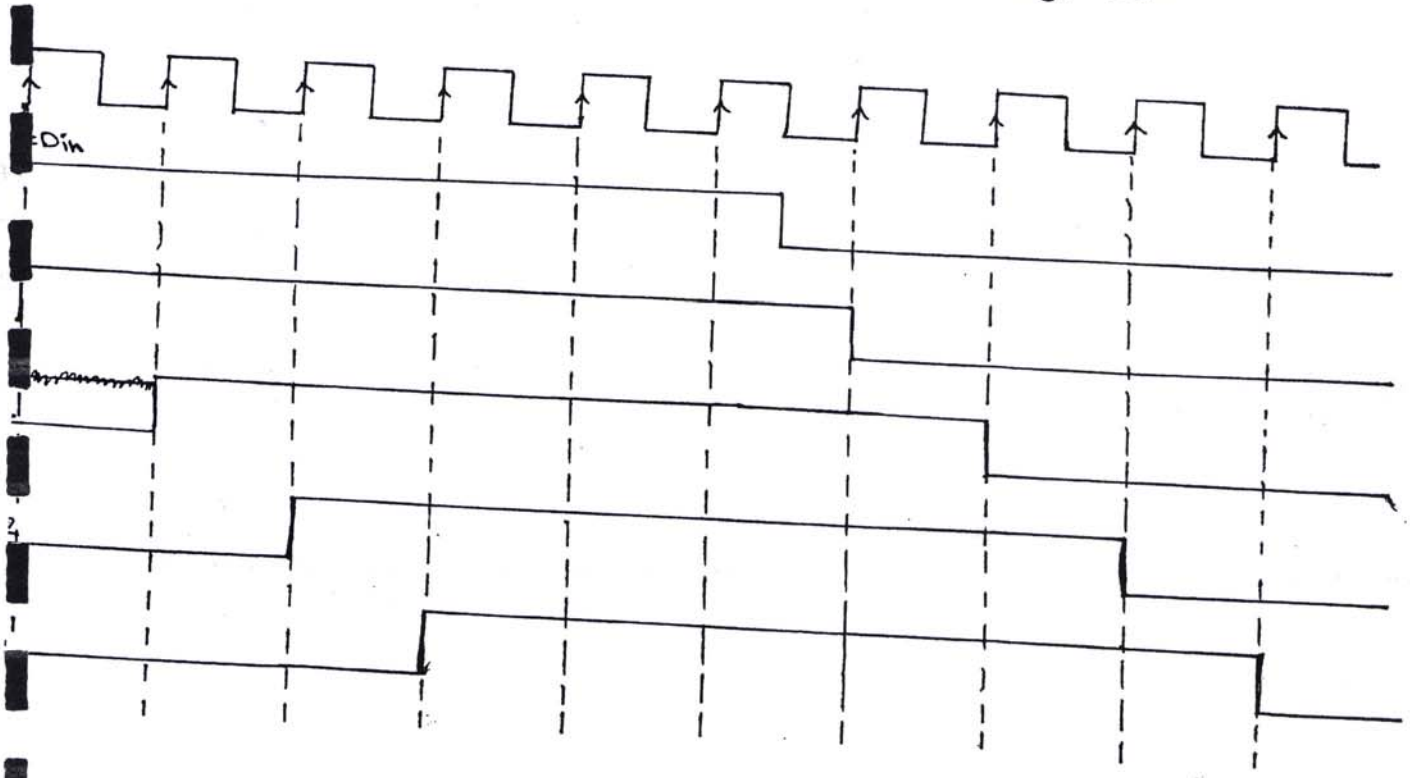


Fig. (5.5): Shift-left timing diagram

5.3.2 Shift-Right Registers:

Fig. (5.6) is a shift-right register. As shown, each "Q" output sets up the "D" input of the preceding flip-flop. When the rising clock edge arrives, the stored bits move one position to the right. Here's an example with "$D_{in}=1$" and "Q=0000'. All data inputs except the one on the left are "0s". The first positive clock edge sets the left flip-flop and the stored word becomes "Q=1000'. With the appearance of this word, "D_3" and "D_2" are "1"s. The second rising clock edge gives "Q=1100'. The third clock pulse gives "Q=1110", and the fourth clock pulse gives "Q=1111".

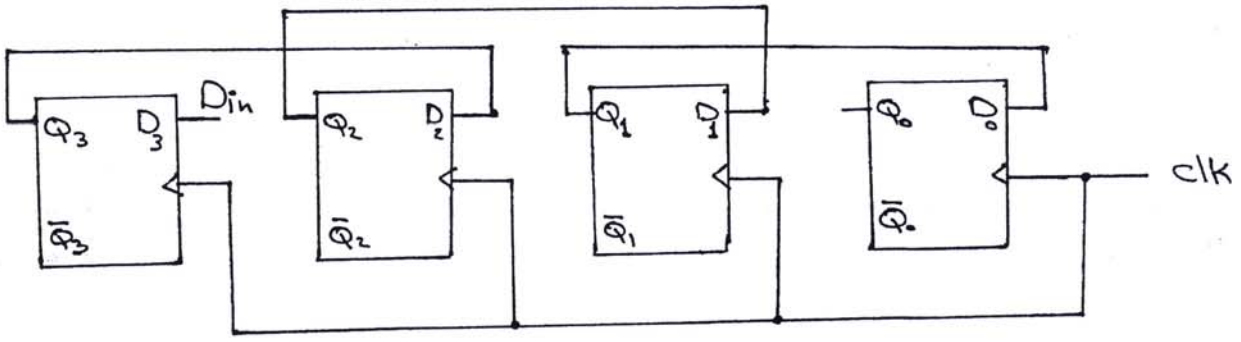


Fig.(5.6): Shift-right register

5.3.3 Controlled-Shift-Register:

A controlled-shift-register has control inputs that determine what it does on the next clock pulse.

Fig. (5.7) shows how the shift-left operation can be controlled. "SHL" is the control signal. When "SHL" is low, the inverted signal $\overline{SHL}$ is high. This forces each flip-flop output to feed back to its data input. Therefore, the data is

etained in each flip-flop as the positive clock pulses ar-rive. In this way, a digital word can be stored indefinitely.

When "SHL" goes high, "Din" sets up the right flip-flop, "Q_0" sets up the second flip-flop, "Q_1" the third flip-flop, and so on. In this mode the circuit acts as a shift-left register. Each positive clock edge shifts the stored bits one position to the left.

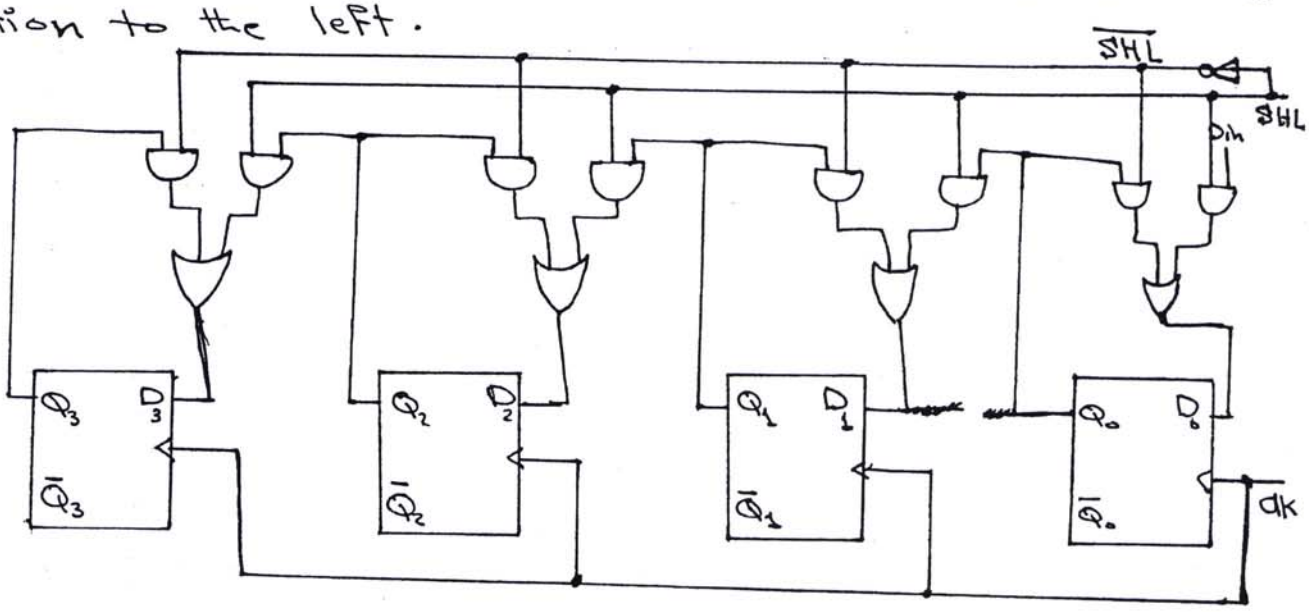


Fig.(5.7): Controlled shift register

H.W: Design a shift left/shift right controlled register.

H.W: Design a controlled register has the following properties:

① Shift to Left.
② Shift to right.
③ Not change state.

5.3.4 Serial-Loading:

Serial loading means storing a word in the shift register by entering 1-bit per clock pulse. To store a 4-bit word, we need four clock pulses. For instance, here's how to serially store the word = "X = 1010". With "SHL" high in Fig. (5.7), make "Din = 1" for the first clock pulse, "Din = 0" for the second clock pulse, "Din = 1" for the third clock pulse, and "Din = 0" for the fourth clock pulse. If the register is clear before the first clock pulse, the successive register contents look like this:

Q = 0001 (Din = 1 ; first clock pulse)
Q = 0010 (Din = 0 ; second = =)
Q = 0101 (Din = 1 ; third = =)
Q = 1010 (Din = 0 ; fourth = =)

In this way, dat is entered serially into the right end of the register and shifted left until all 4-bits have been stored. After the last bit is entered, "SHL" is taken low to freeze the register contents.

5.3.5 Serial-In / Serial-out Registers:

The serial-in/serial-out shift register accepts data serially that is; one bit at a time on a single line. It produces the stored information on its output also in serial form. Fig. (5.8) shows a 4-bit serial-in/serial-out register constructed by using D-type Flip-Flops.

مكتب الحدير
داخل كلية الهندسة

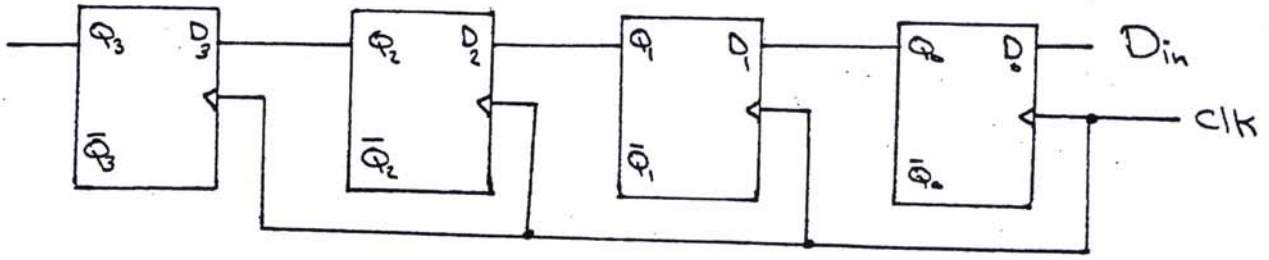


Fig. (5.8): Serial-in / Serial-out shift register

H.W.: Show the states of the 4-bit serial-in/ serial-out shift register for Din = 1011? The shift register is triggered at positive-going edge!

5.3.6 Parallel-In / Serial-out Shift Register:

In this type of shift register, the bits are entered simultaneously into their respective stages on parallel lines rather than one a bit-by-bit basis on one line as with serial data inputs. But the output information are produced in serial form. Fig. (5.9) illustrates a 4-bit parallel-in/ serial-out shift register.

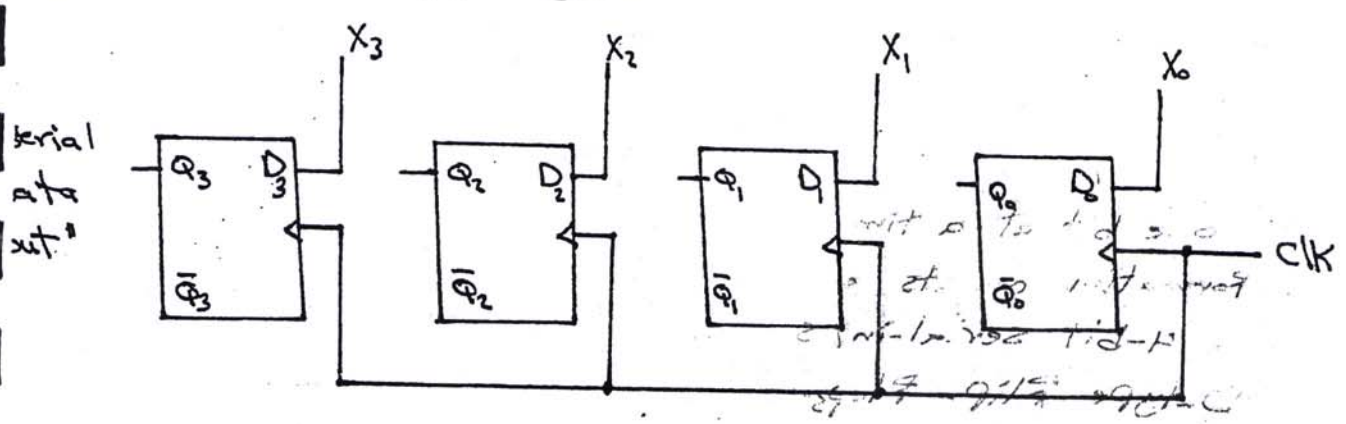
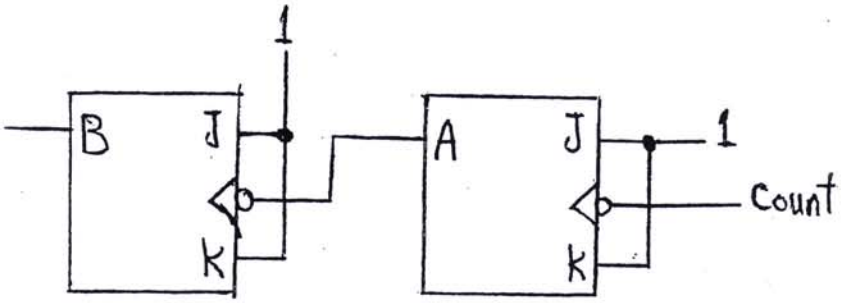


Fig. (5.9): Parall-in / Serial-out shift register

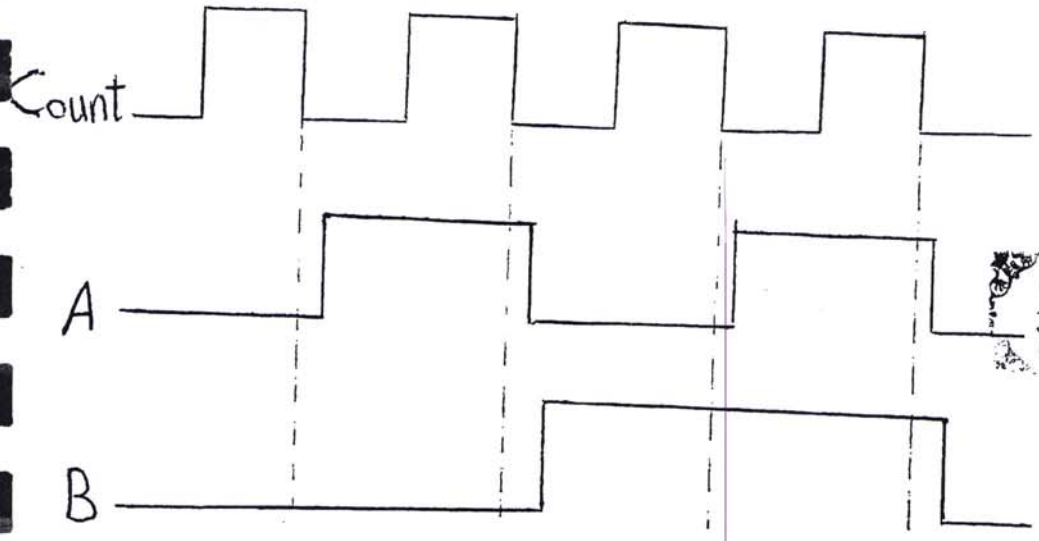
Design of Asynchronous Counters

Binary counters are sometimes designed with a clock input. They are constructed from same clocked flip flops (typically JK) as synchronous counters, but each flip flop is triggered by the transition of the previous one. Consider the circuit of below, with two flip flops.



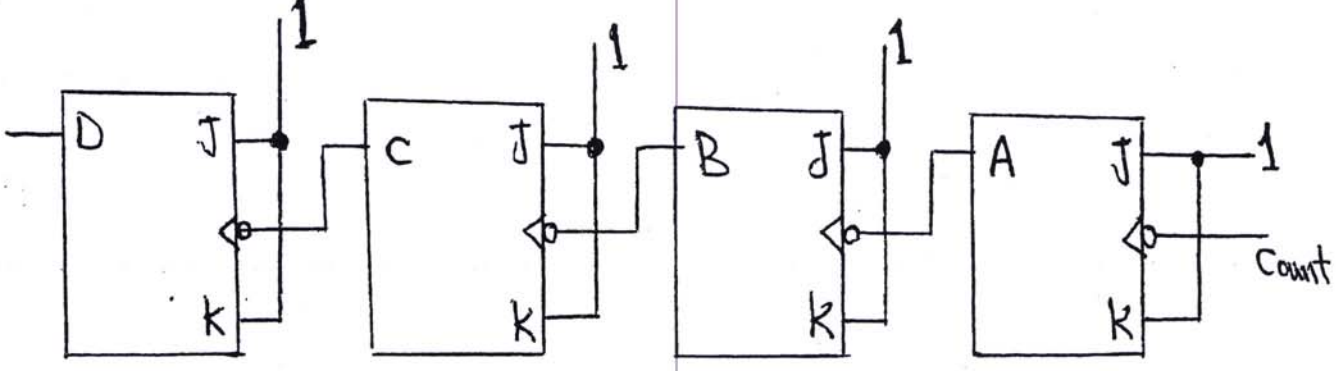
"A 2-bit asynchronous Counter"

When the Count signal goes from 1 to 0, flip flop A is triggered. If it started out at 0, it goes to 1. The 0 to 1 transition on the output of A, and thus on the clock input of B has no effect. When the next negative transition on Count occurs, A will go from 1 to 0 causing the clock input to B to do the same. Since J and K are 1, flip flop B will change states. Since there is a delay from the clock edge to the output change, flip flop B is clocked somewhat later than A and thus its output changes later.



Note that the flip flops (BA) go through the sequence 00, 01, 10, 11, and repeat. Thus, this is a 2-bit counter.

We can obtain a 4-bit counter by connecting four flip flops in the same fashion.



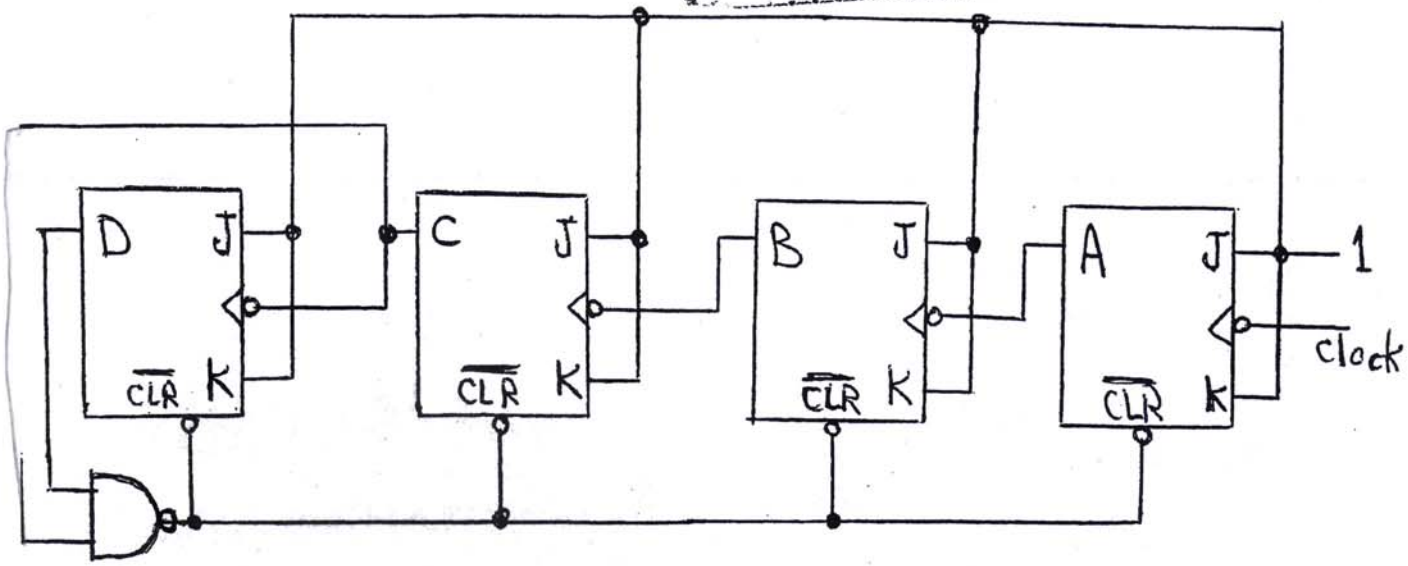
The advantage of the asynchronous counter is the simplicity of the hardware. There is no combinational logic required. The disadvantage is speed. The state of the system is not established until all of the flip flops have completed their transition, which, in this case, is four flip flop delays.

If the counter were larger or the clock faster, it might not reach its final state until after the next negative clock transition.

ex: Design an asynchronous base-12 counter using JK flip flops with active low clears and NAND gates.

soln

The easiest way to do this is to take the 4-bit binary counter and reset it when it reaches 12. Thus, the circuit below computes $(\overline{DC})$ and uses that to reset the counter.



The counter cycles 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, (12), 0