Intro to Registers
Shift register and its use for Serial addition

Sparsh Mittal

What is a register?

A register is a group of flip-flops, each one of which shares a common clock and is capable of storing one bit of information.

An n-bit register has n flip-flops.

In addition to the flip-flops, a register may have combinational gates that perform certain data-processing tasks.

In its broadest definition, a register consists of a group of flip-flops together with gates that affect their operation. The flip-flops hold the binary information, and the gates determine how the information is transferred into the register.

Counter: a register that goes through a predetermined sequence of binary states (using gates).

Counters are a special type of register

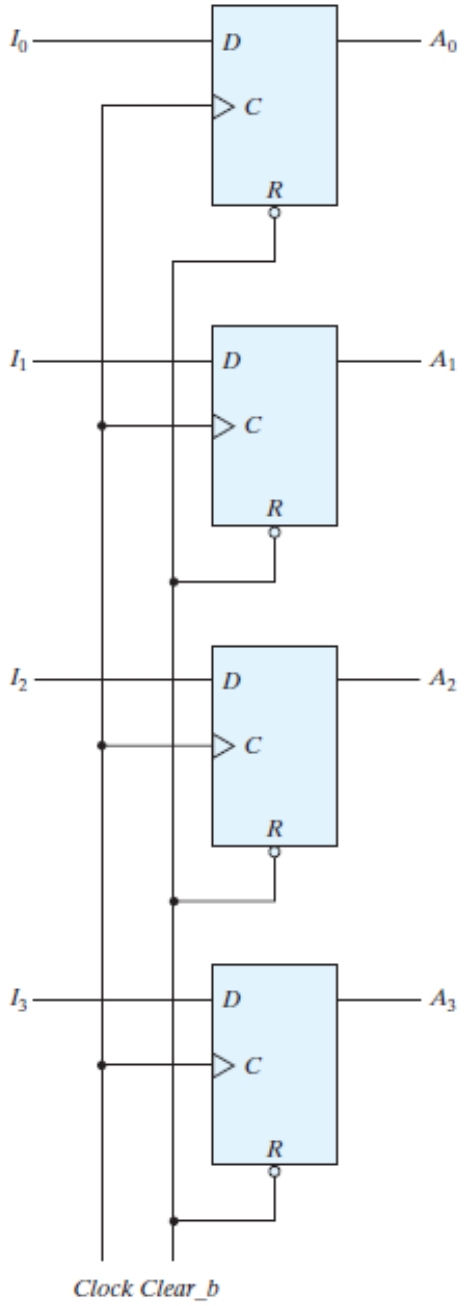
4-bit register using D flip-flop

The common clock input triggers all flip-flops on the positive edge of each pulse, and the binary data available at the four inputs are transferred into the register.

The value of (*I*3 etc) immediately before the clock edge determines value of (*A*3) after clock edge. The four outputs can be sampled anytime.

Clear_b input is useful for clearing the register to all 0's prior to its clocked operation.

R inputs must be maintained at logic 1 during normal clocked operation



Limitation of this circuit (1 of 2)

Synchronous digital systems have a master clock generator that supplies a continuous train of clock pulses.

The pulses are applied to all flip-flops and registers in the system.

A separate control signal must be used to decide which register operation will execute at each clock pulse.

In above circuit, all the bits are loaded in parallel with a common clock pulse.

Limitation of this circuit (2 of 2)

If register contents must be left unchanged

case 1. inputs must be held constant

case 2. or the clock must be inhibited from the circuit (using a disabling gate).

Problems with each of these cases.

case 1: data bus driving the register would be unavailable for other traffic.

case 2: Using a gate into the clock path is bad idea because we are performing logic with clock pulses. This produces uneven propagation delays between the master clock and the inputs of flip-flops. System may go out of synchronism

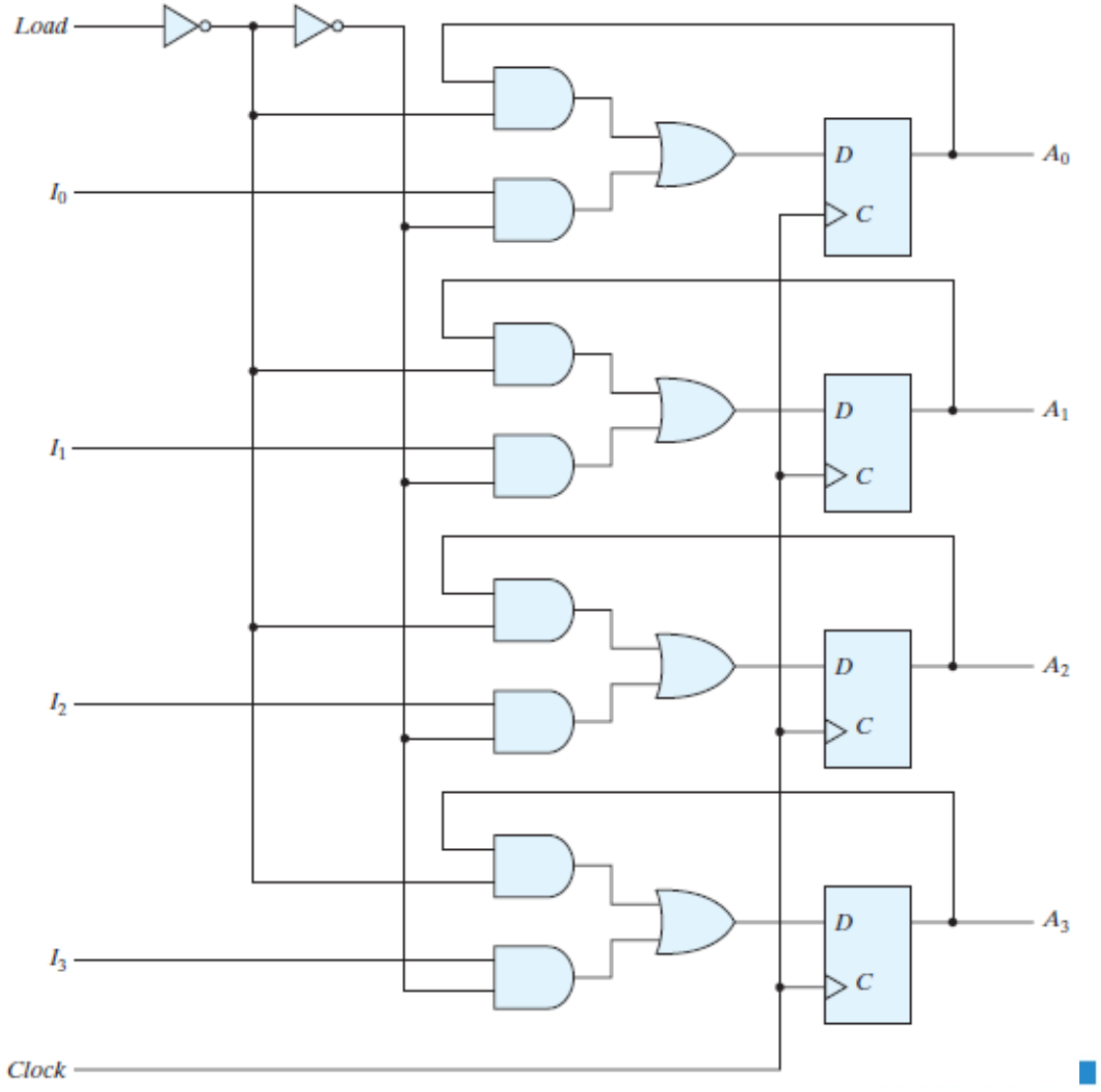
To fully synchronize the system, we must ensure that all clock pulses arrive at the same time anywhere in the system, so that all flip-flops trigger simultaneously.

Register with Parallel Load

Case 3: control register operation with D inputs, rather than clock.

The feedback connection from output to input is necessary because a *D* flip-flop does not have a “no change” condition.

The load input determines whether the next pulse will accept new information or leave the information in the register intact.



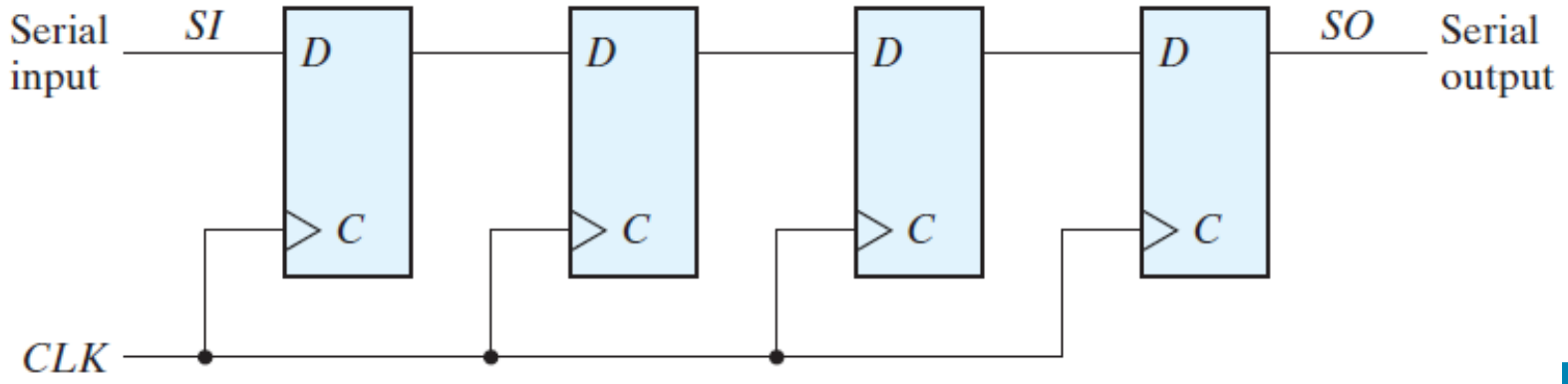
Shift registers

Shift register: a register capable of shifting data to its neighboring cell, in a selected direction.

All flip-flops receive common clock pulses, which activate data-shift from one stage to the next.

The simplest **uni-directional** shift register is one that uses only flip-flops is shown below.

The output of a given flip-flop is connected to the D input of the flip-flop at its right.



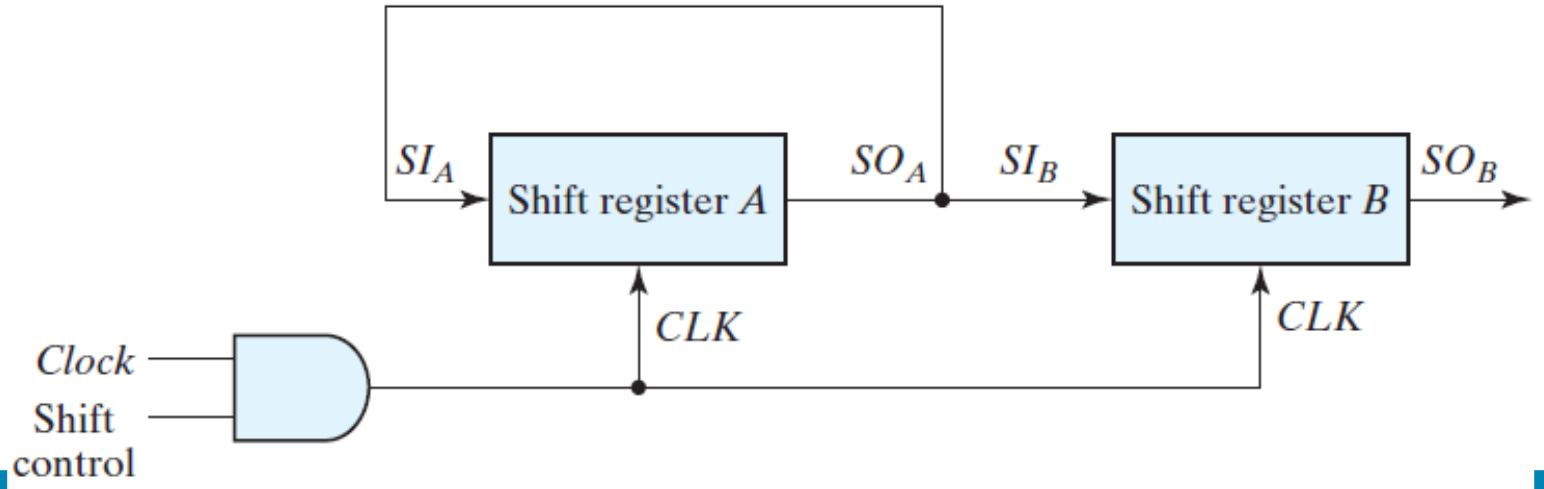
Serial Transfer

When information is transferred and manipulated one bit at a time

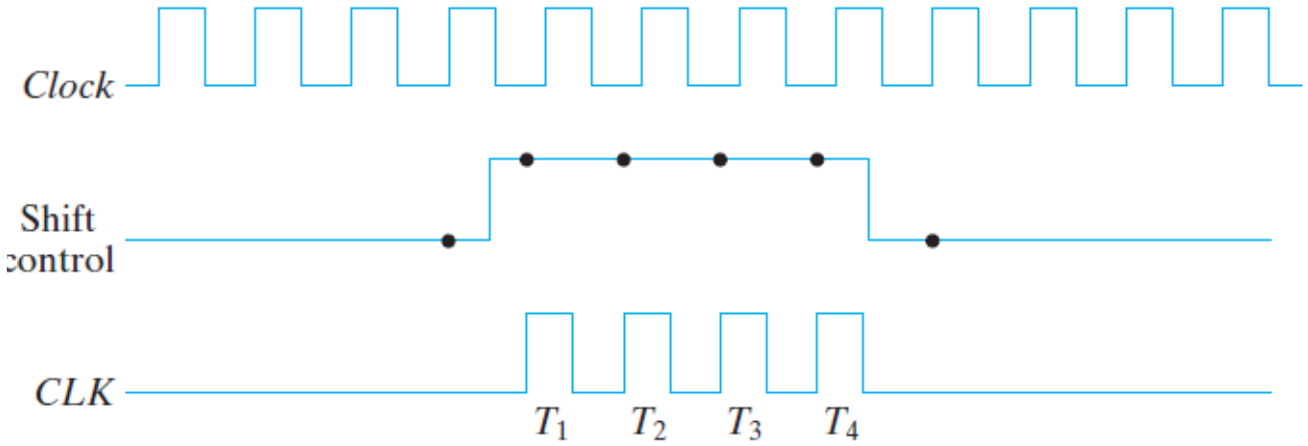
The serial output (SO) of register A is connected to serial input (SI) of register B.

To prevent the loss of information stored in the source register, information in register A is made to circulate by connecting serial output to its serial input.

The initial content of register B is shifted out through its serial output and is lost unless it is transferred to a third shift register.



Timing diagram



Suppose the shift registers have four bits each.

Then, shift registers must be enabled through the shift control signal, for a fixed time of four clock pulses to pass an entire word.

The shift control signal is synchronized with clock and changes value just after the negative edge of clock.

The next four clock pulses find the shift control signal in the active state, so the output of the AND gate connected to the CLK inputs produces four pulses: T1, T2, T3, and T4.

Each rising edge of the pulse causes a shift in both registers.

Example

Assume that the binary content of A before the shift is 1011 and that of B is 0010.

Serial-Transfer Example

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

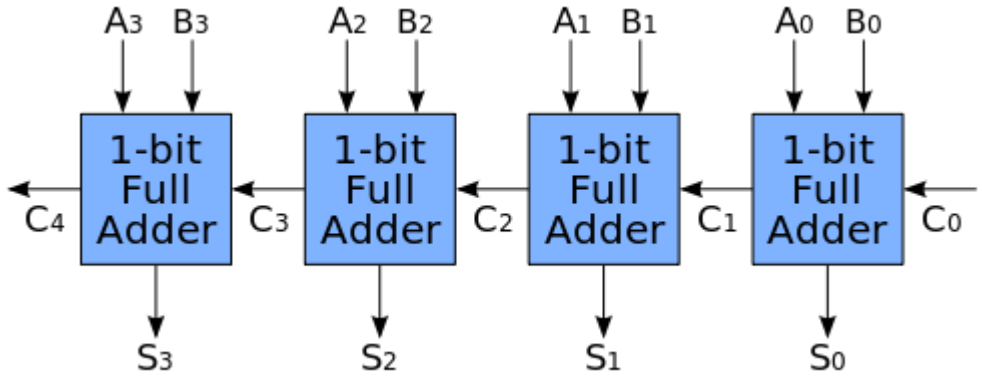
The contents of *A* are copied into *B*, so that the contents of *A* remain unchanged i.e., the contents of *A* are restored to their original value.

Using a shift register to perform serial addition

Multi-bit addition

For doing N-bit addition, we have 2 options:

1. **Parallel addition:** Use N full-adders (FAs)



2. **Serial addition:** Use only 1 FA, and keep shifting data to feed to this FA

Serial addition

Inputs are stored in shift registers.

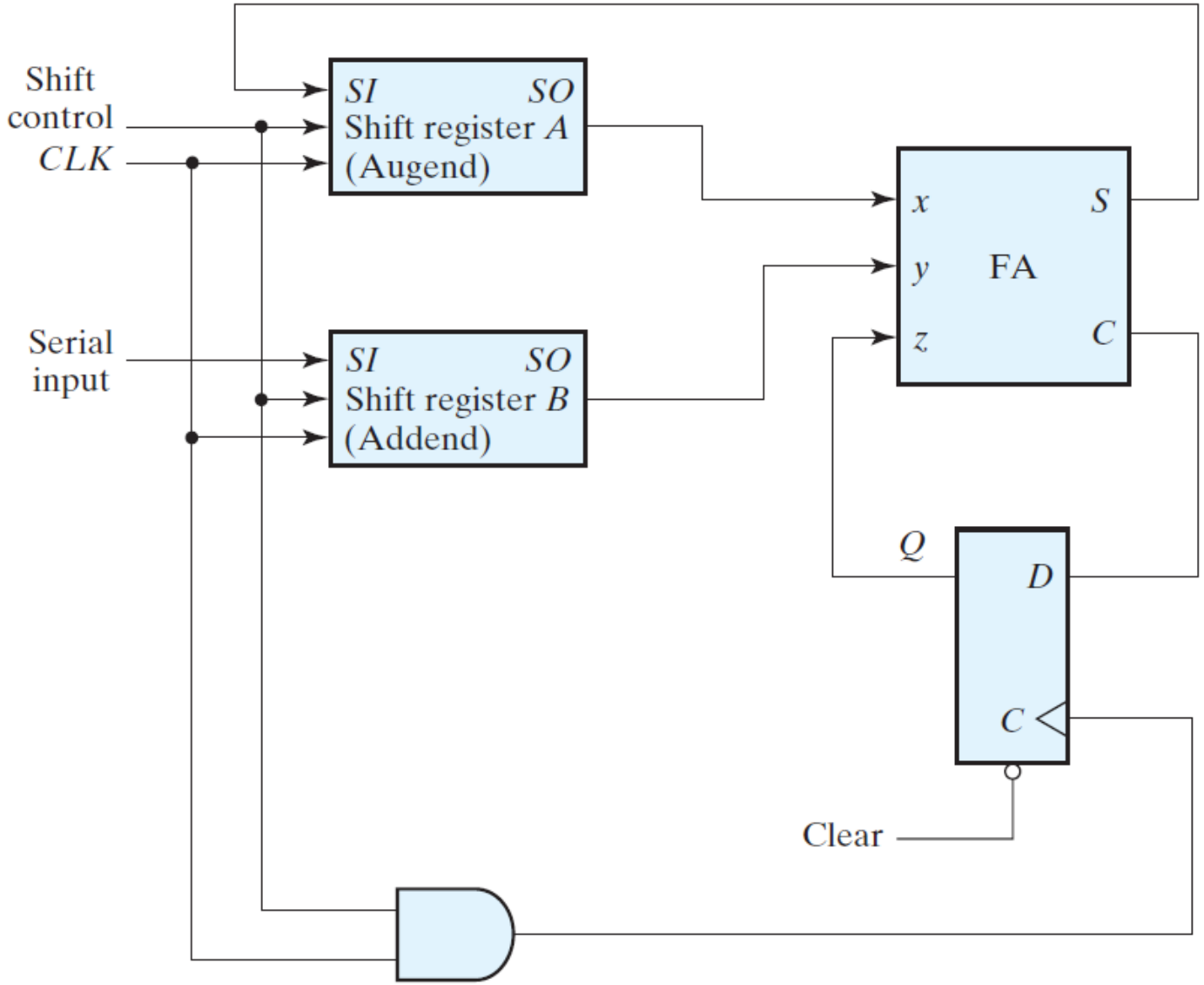
Beginning with LSB pair, we add one pair at a time through a single full-adder (FA).

CarryOut is saved in a D flip-flop, and this is used as the CarryIn for next pair of significant bits.

Sum bit could be transferred into a third shift register.

By shifting it into A while the bits of A are shifted out, it is possible to use one register for storing both the input and output value.

Serial input of register B can be used to transfer a new binary number while addend bits are shifted out during the addition.



Serial adder through state table.

We will now show: serial operations can be designed through sequential circuit design procedure.

We will design serial adder using a state table.

Two input numbers are stored in shift registers; their serial outputs are x and y.

Proposed sequential circuit will not include the shift registers, but they will be inserted later to show the complete circuit.

The circuit has two inputs, x and y, that provide a pair of significant bits, an output S that generates the sum bit, and flip-flop Q for storing the carry.

Serial adder using JK flip-flop

Flip-Flop Excitation Tables

Q(t)	Q(t = 1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

(a) *JK* Flip-Flop

State Table for Serial Adder

Present State	Inputs		Next State	Output	Flip-Flop Inputs	
Q	x	y	Q	S	J_Q	K_Q
0	0	0	0	0	0	X
0	0	1	0	1	0	X
0	1	0	0	1	0	X
0	1	1	1	0	1	X
1	0	0	0	1	X	1
1	0	1	1	0	X	0
1	1	0	1	0	X	0
1	1	1	1	1	X	0

Comments

The present state of Q is the present value of the carry.

The next state of Q is equal to the output carry.

The state table entries are identical to the entries in a full-adder truth table, except that the input carry is now the present state of Q and the output carry is now the next state of Q.

If a *D* flip-flop is used for *Q*, the circuit reduces to the one shown earlier.

Table 5.1
Flip-Flop Characteristic Tables

JK Flip-Flop

J	K	Q(t + 1)	
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	$Q'(t)$	Complement

Flip-Flop Excitation Tables

Q(t)	Q(t = 1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

(a) *JK* Flip-Flop

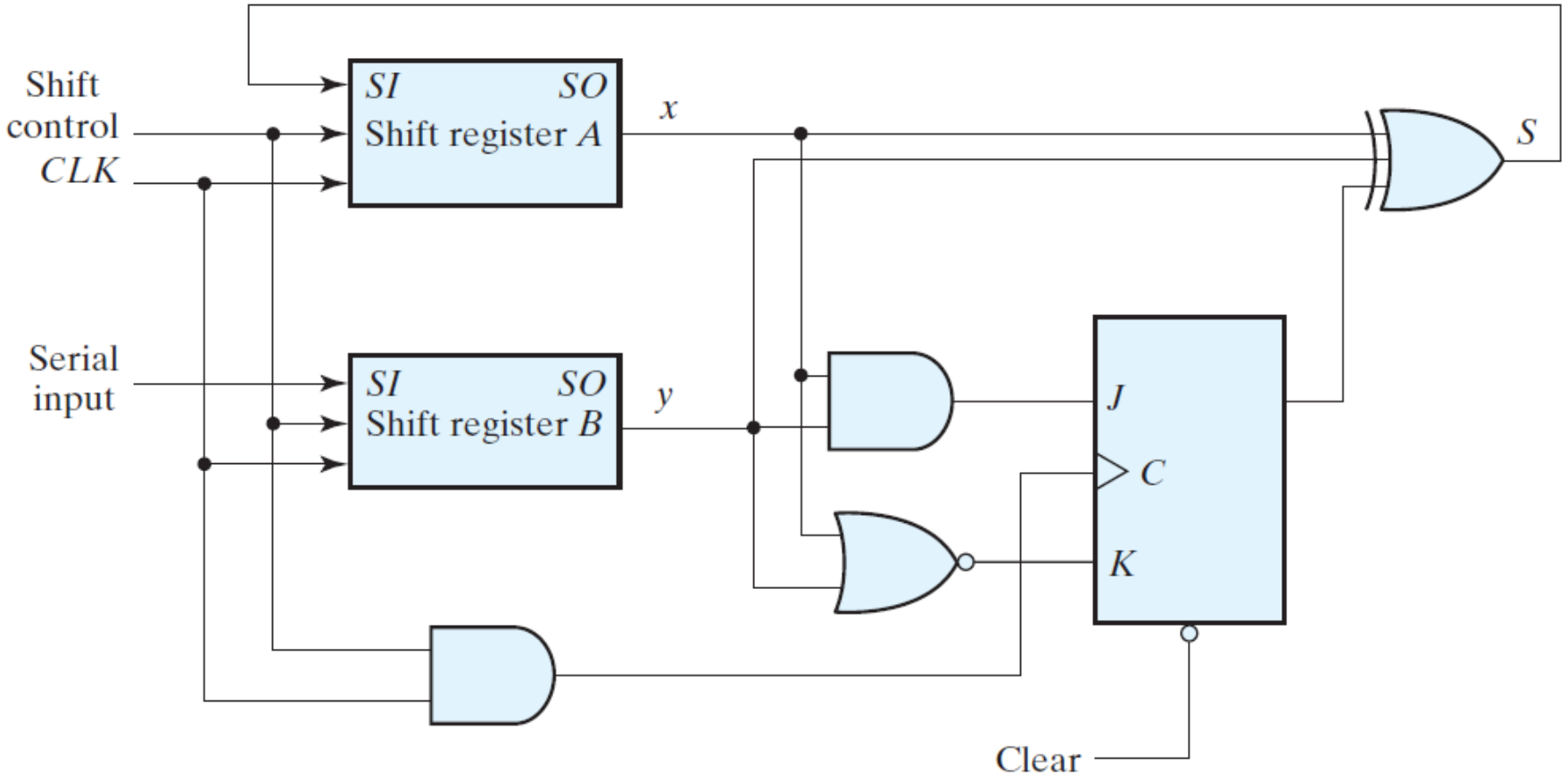
If a *JK* flipflop is used for Q, we need to determine values of inputs J and K by referring to the excitation table.

The two flip-flop input equations and the output equation can be simplified by means of maps to

$$J_Q = xy$$
$$K_Q = x'y' = (x + y)'$$
$$S = x \oplus y \oplus Q$$

Circuit dia

$$J_Q = xy$$
$$K_Q = x'y' = (x + y)'$$
$$S = x \oplus y \oplus Q$$



Serial adder using JK flip-flop

The two shift registers are included in the diagram to show the complete serial adder.

Output S is a function not only of x and y, but also of the present state of Q.

The next state of Q is a function of the present state of Q and of the values of x and y that come out of the serial outputs of the shift registers.