Digital Logic Systems – Registers, Counters and Memory - I

Assist. Prof. Özge ÖZTİMUR KARADAĞ

ALKÜ

- Previously:
  - Analysis and design of synchronous sequential circuit with flip-flops and combinational circuit on the feedback path.

- Starting this week:
  - Introduce sequential circuits based on their functions(purposes).

Registers

- a *group* of flip-flops, each one of *which* is *capable of* storing one *bit* of information.
- Flip-flop are binary units capable of storing one bit of information.
- A group of flip-flop makes-up a register.
- An n-bit register consists of n flip-flops and can store any n bit of information.
- Registers: consists of flip-flops and combinational circuits
- Flip-flops store data and gates control when and how the data will be transferred.

Counters

- A register that goes through a predetermined sequence of binary states.
- The gates in the counter are connected in such a way as to produce the prescribed sequence of states.

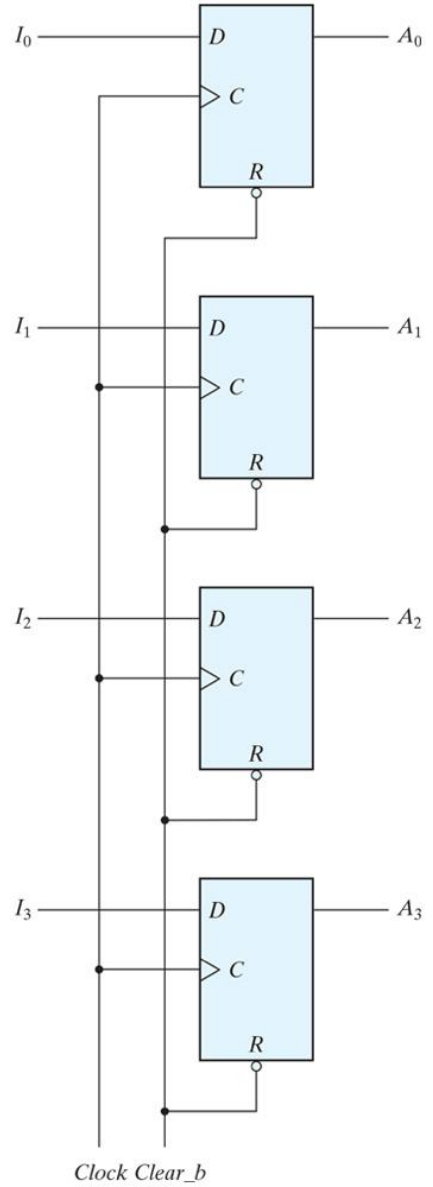
Memory Unit

- A memory unit is a device to which binary information is transferred for storage and from which information is retrieved when needed for processing.
- The process of storing new information into memory is referred to as a memory *write* operation.
- The process of transferring the stored information out of memory is referred to as a memory *read* operation.
- Two types of memories in digital systems:
  - Random Access Memory (RAM): can perform both read and write
  - Read Only Memory(ROM): can perform only read

- Registers, Counters and Memory Units → Design of Digital Systems
- Registers can be used to simplfy the design of sequential circuits.
- Counters are used to put the digital systems in order and generate variables to control the digital systems.
- Memory units store the program and data.

Registers

- 4 bit register
- The order of trigger in the registers is important.
- When CP=1 the input to D flip-flop is transferred to Q. When CP=0 the state of the flip-flop does not change, (Q is equal to the previous input value).
  - Flip-flop respond to clock level, ie. CP=1
  - Flip-flop respond to clock transition, ie. negative-to-positive



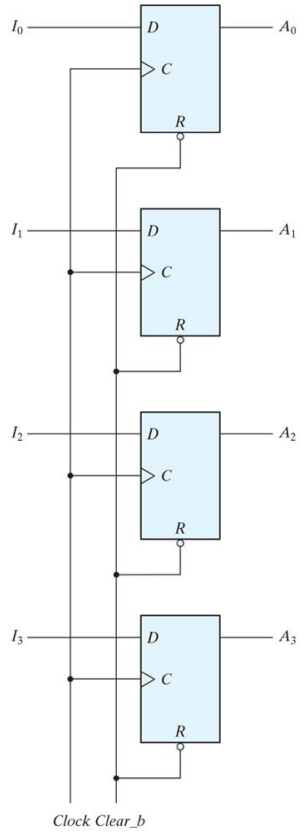
Registers

- Flip-flops responding to signal transitions:
  - Master-slave flip-flop
  - Edge-triggered flip-flop

- Flip-flops in the registers that we will deal with are all edge-triggered or master-slave flip-flops.

Parallel Load Register

- Transfer of new data to a registers is referred as loading of the register.
- If all the bits of the register are loaded simultenously with a clock pulse → parallel load.
- In this register, as clock pulse is provided with CP all four bits will be loaded in parallel.
  - CP can be used as a signal to control loading of register.
    - CP= 1 → register is loaded.
    - CP= 0 → register does not change.
  - This is a positive edge triggered register.

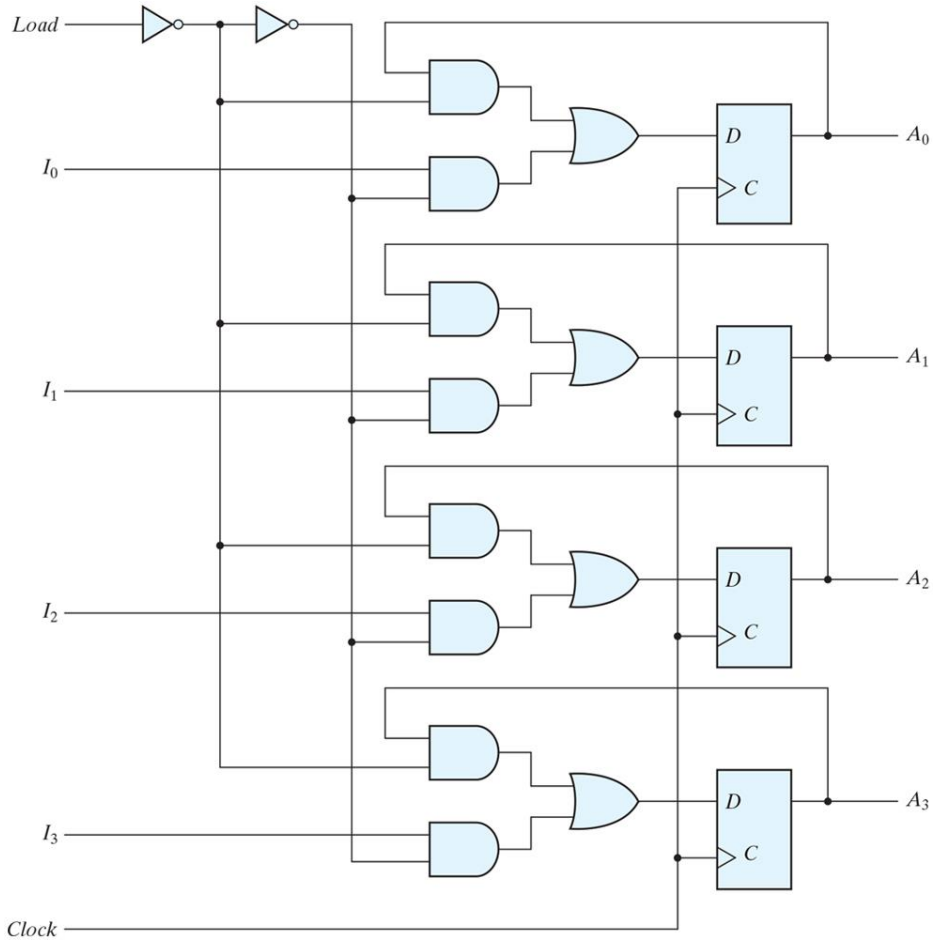


Parallel Load Register

- Load Control Input:
  - Digital systems usually have a clock generator which provides clock signals for the whole system.
  - In order to decide on which flip-flops will be loaded in which clock pulses, a control signal is ANDed with CP.
  - However, this causes delays due to gates.
  - All flip-flops should receive the clock pulses at the same time.
  - Clock pulses are directly provided to flip-flops and control is handled with flip-flop inputs.

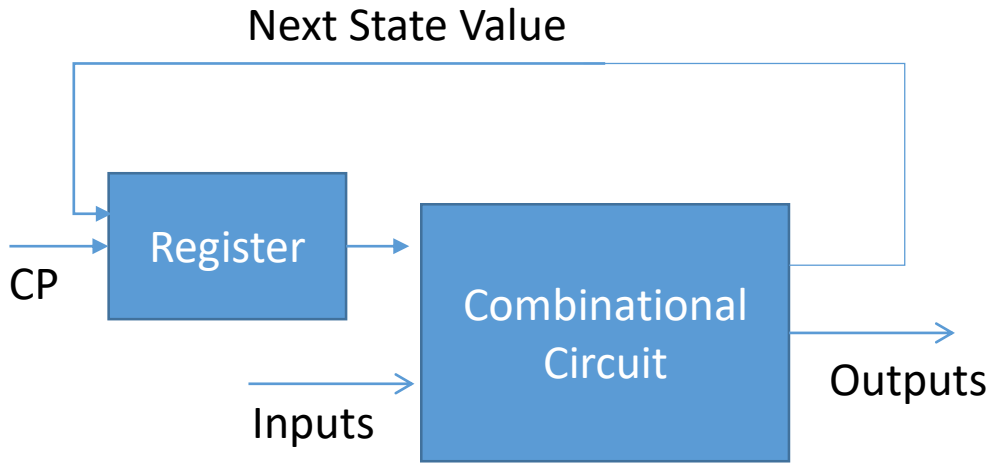
Ex: 4 bit Register with Parallel Load

- Gates before the flip-flops behave like a multiplexor. (choses one of the inputs)



A Sequential Logic Application with Register

- Registers can be found as MSI (medium scale integrated) elements and can be used as a part of a sequential circuit.
- Sequential Circuit Diagram



Ex:

- Design the sequential circuit whose state table is given.

Current State		Input	Next State		Output
A1	A2	x	A1	A2	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	0	0	1
1	0	0	1	0	0
1	0	1	0	1	0
1	1	0	1	1	0
1	1	1	0	0	1

Sum of minterms:

$$A_1(t+1) = \sum(4,6)$$

$$A_2(t+1) = \sum(1,2,5,6)$$

$$y(A_1, A_2, x) = \sum(3,7)$$

Ex..

$$A_1(t+1) = \sum(4,6)$$

	00	01	11	10
0				
1	1			1

$$A_1(t+1) = A_1 x'$$

$$A_2(t+1) = \sum(1,2,5,6)$$

	00	01	11	10
0		1		1
1		1		1

$$A_2(t+1) = A_2' x + A_2 x' = A_2 \oplus x$$

$$y(A_1, A_2, x) = \sum(3,7)$$

	00	01	11	10
0			1	
1			1	

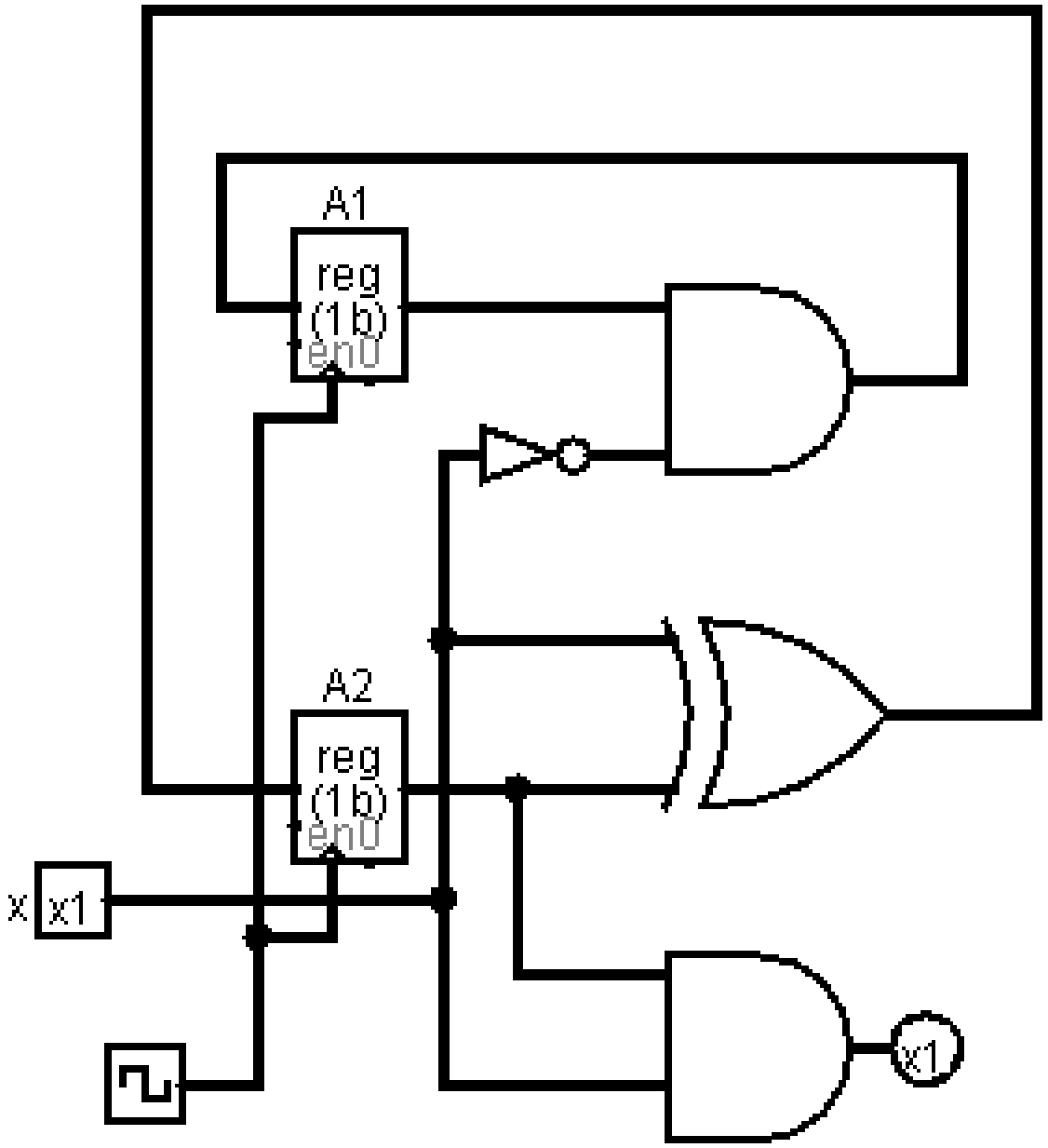
$$y = A_2 x$$

Ex.

$$A_1(t+1) = A_1 x'$$

$$A_2(t+1) = A_2' x + A_2 x' = A_2 \oplus x$$

$$y = A_2 x$$



Ex.

- Solve the previous example using a register and a ROM.
- What is a ROM?
  - Non-volatile
  - Not erasable

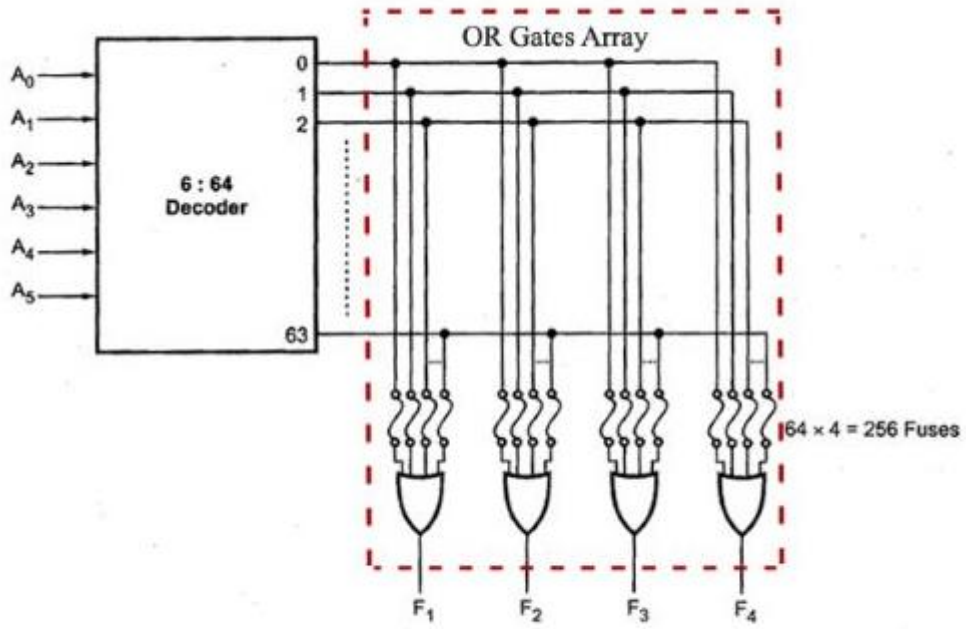
ROM (Read Only Memory)

- Memory Organization is referred as the number of and width of memory words:
  - 1024 one-bit word
  - 512 two-bit words
  - 256 four-bit words
  - 128 eight-bit words

  - Example: 128 x 8 ROM → 128 eight-bit words
    - 7 address bits
    - 8 data bits
    - 7 input, 8 output

ROM

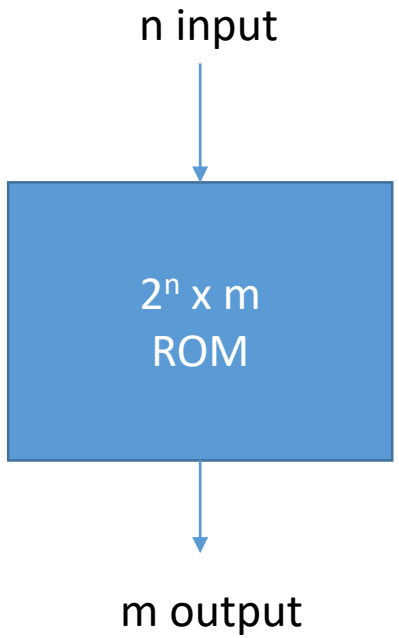
- Internal Structure
- Ex: 6x4 ROM



Internal construction of 64x4 ROM

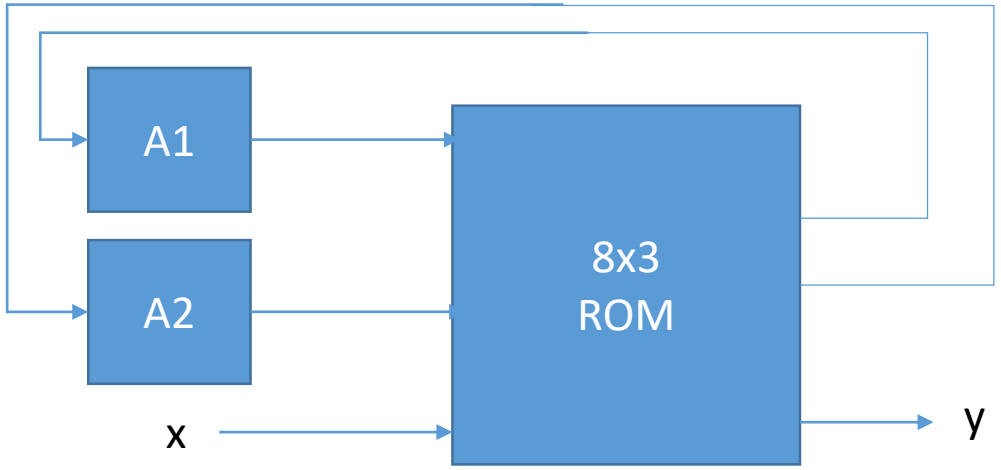
Block diagram?

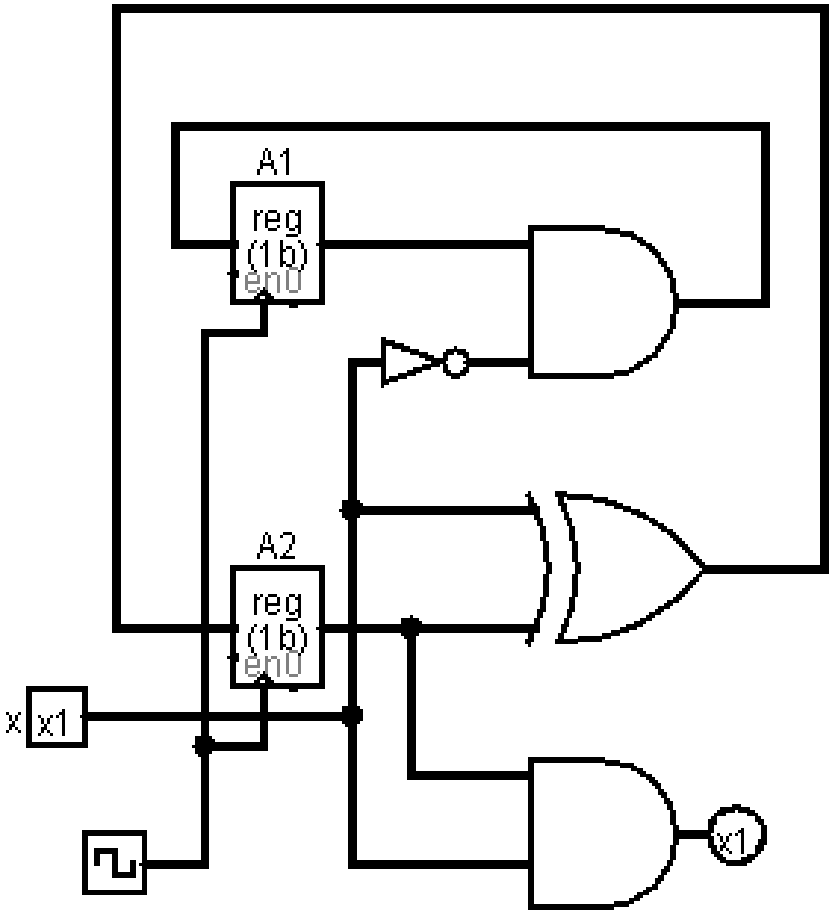
Number of inputs and outputs?



Ex.

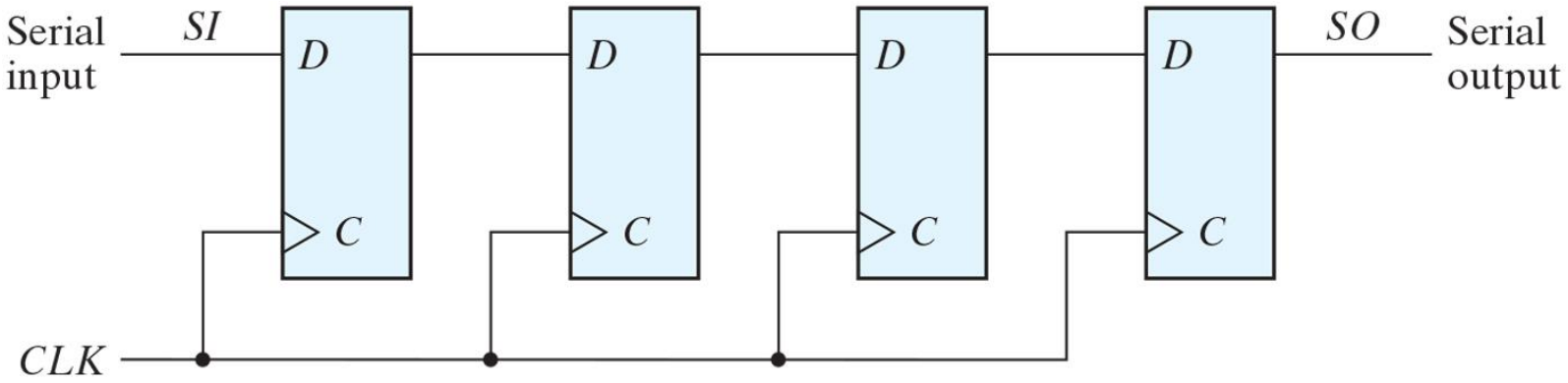
- Register – instead of flip-flops
- ROM – instead of combinational circuit
  - Number of input-output?
  - Number of ROM input: flip-flop + input
  - Number of ROM output: flip-flop + output





Shift Registers

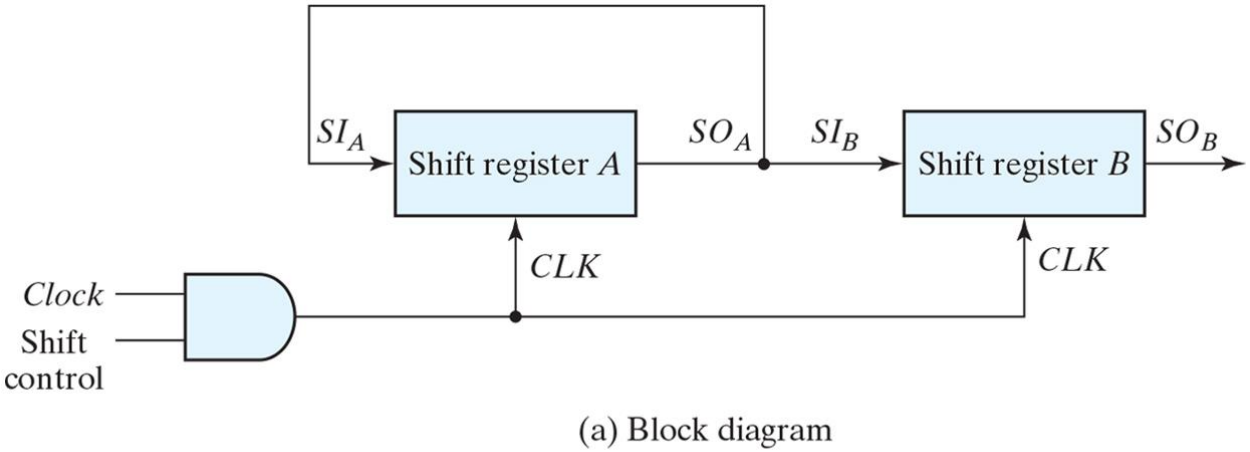
- A register capable of shifting the binary information held in each cell to its neighboring cell in a selected direction is called a *shift register.*
- 4 bit shift register:



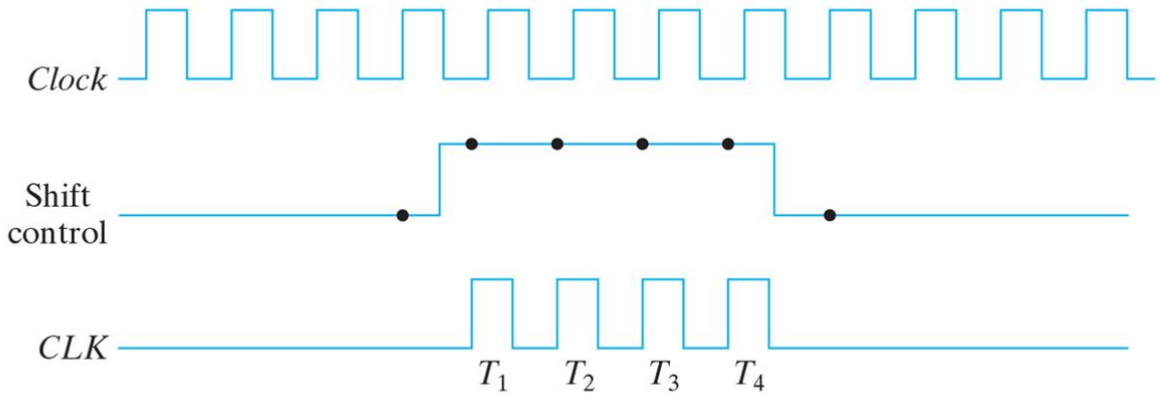
  - Each clock pulse causes a one bit right shift.
  - A bidirectional shift register can shift in both directions.

Shift Registers - Serial Transfer

- A content of one register can be transferred to another register by shifting bits.
  - Why is SO_A is connected to SI_A?
  - What is the AND used for?
  - Timing diagram for the 4 bit transfer operation:

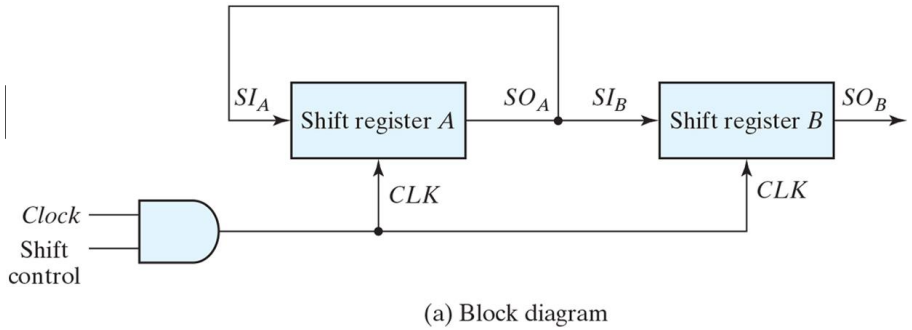


(a) Block diagram

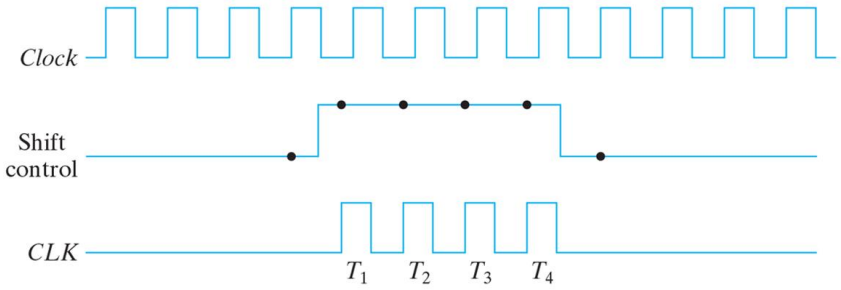


(b) Timing diagram

Shift Registers - Serial Transfe



(a) Block diagram



(b) Timing diagram

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

Shift Registers

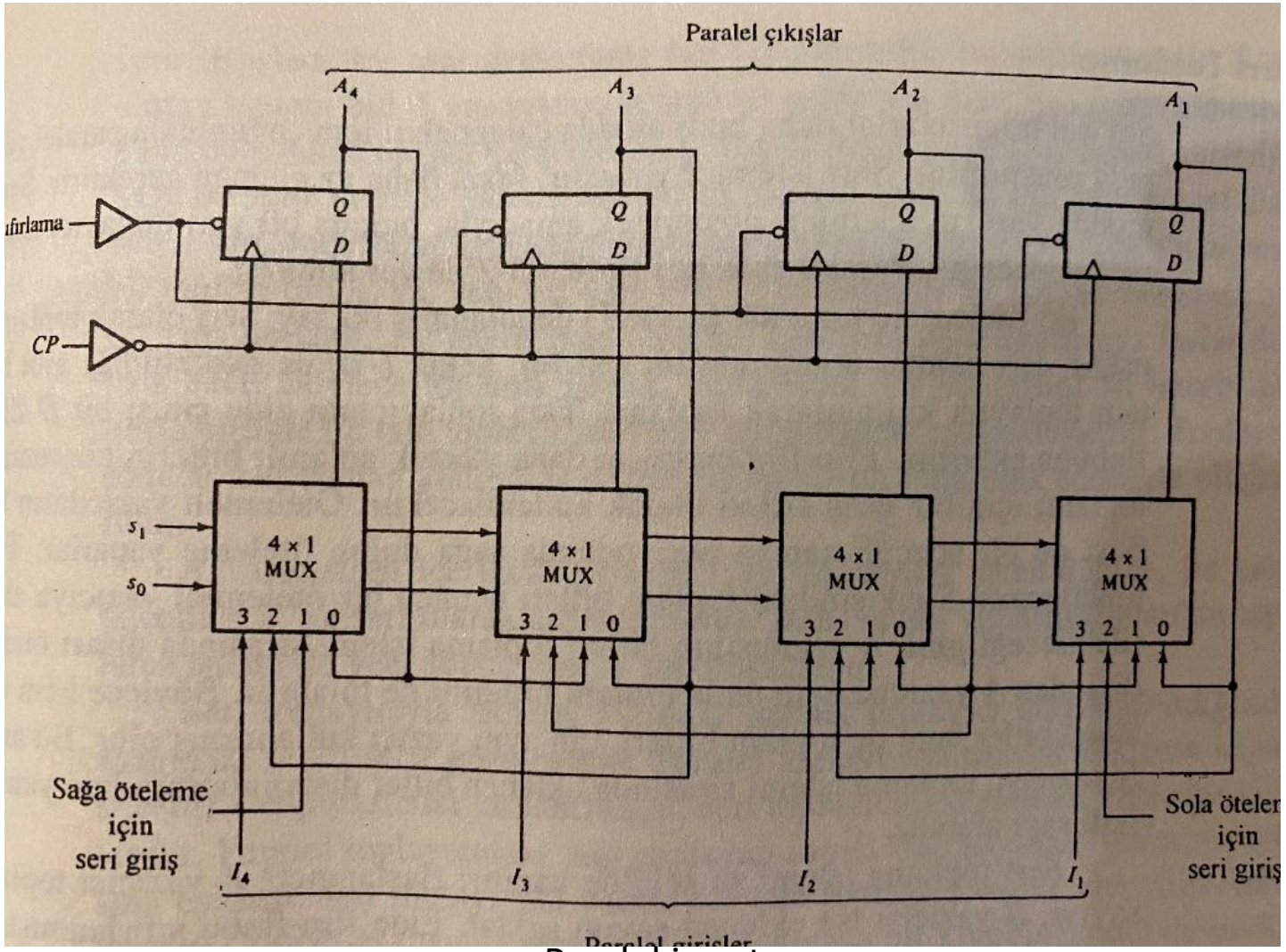
- In parallel mode, all bits are transferred in parallel with a clock pulse.
- In serial mode, register have a serial in and a serial out. At each clock pulse register is shifted one bit.
- Computers can run in combination of the two modes.
- Serial operations are slower but require less hardware.

Bidirectional Shift Register with Parallel Load

- Birectional Shift Register: can shift in both directions
- Unidirectional Shift Register: can shift in only one direction
- General properties of shift registers:
  - Reset control input to reset the register
  - CP input to synchronize all operations
  - Right shift control to enable right shift operation, serial input and serial out
  - Left shift control to enable left shift operation, serial input and serial out
  - Parallel load control to enable parallel transfer and n bit parallel input.
  - N bit parallel out
  - A control case to allow the register keep its state even if clock pulse is applied.

Bidirectional Shift Register with Parallel Load

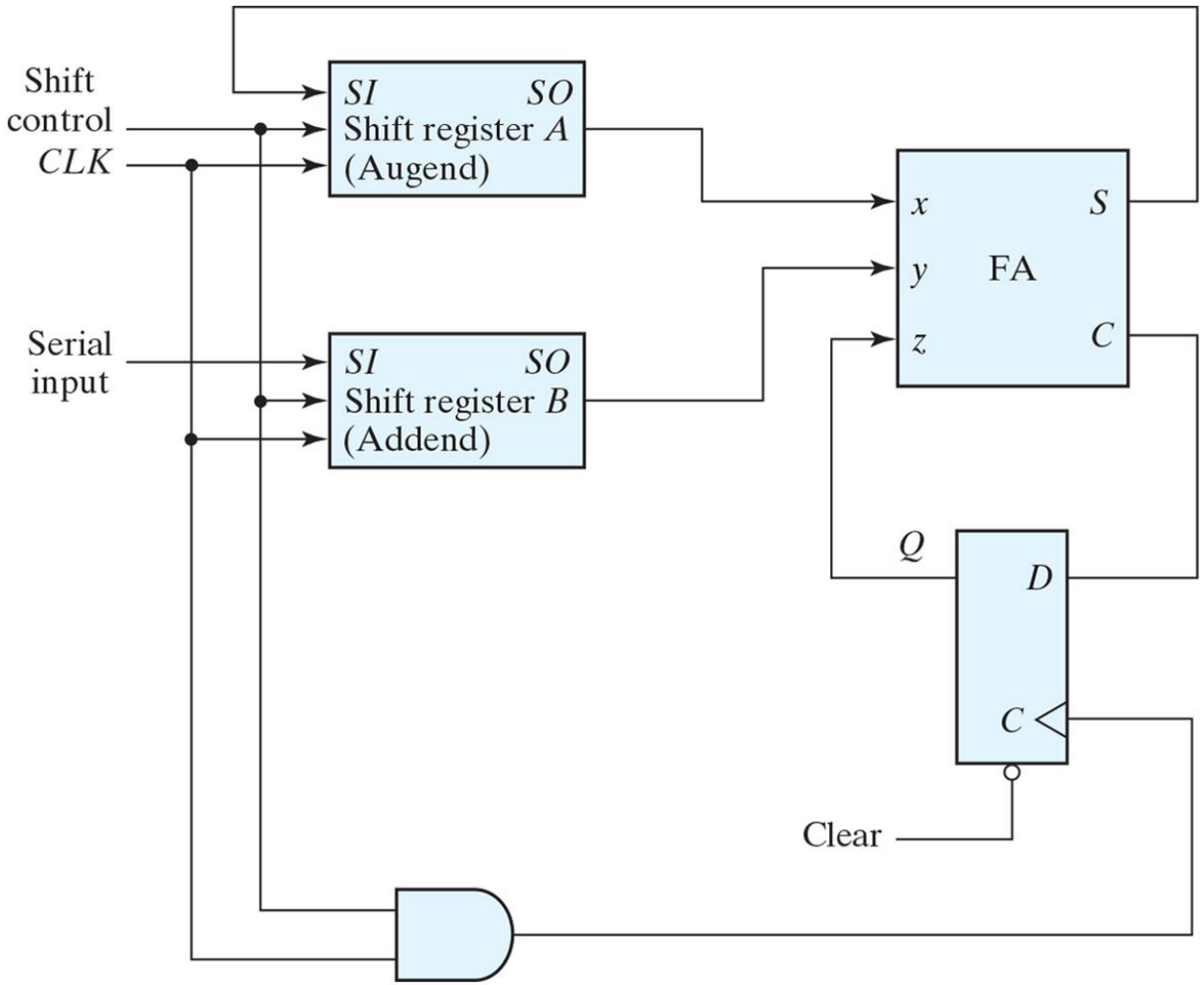
reset



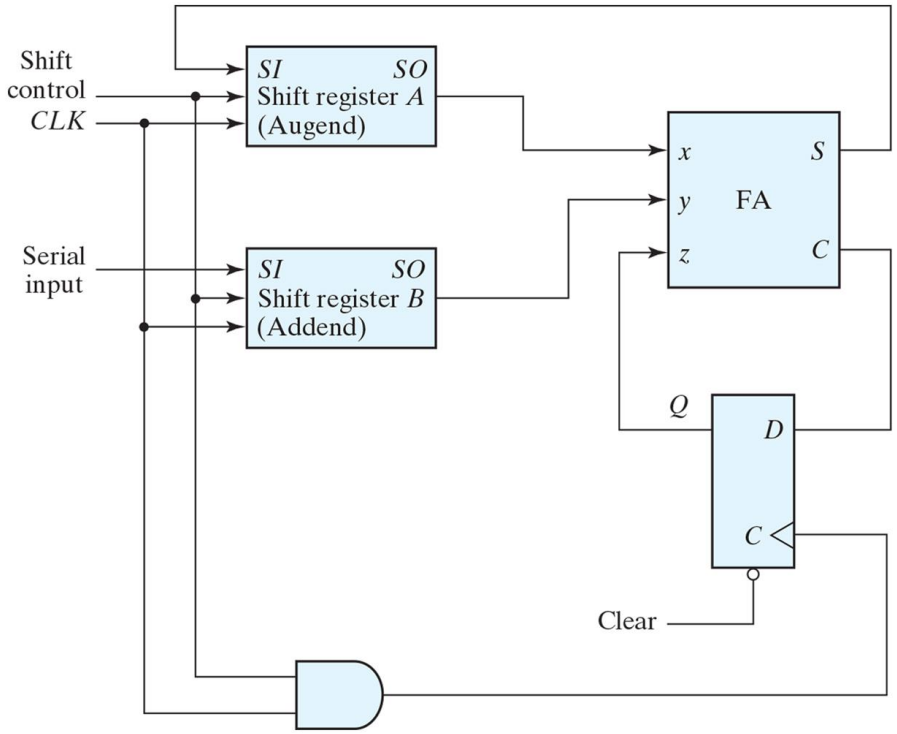
Paralel inputs

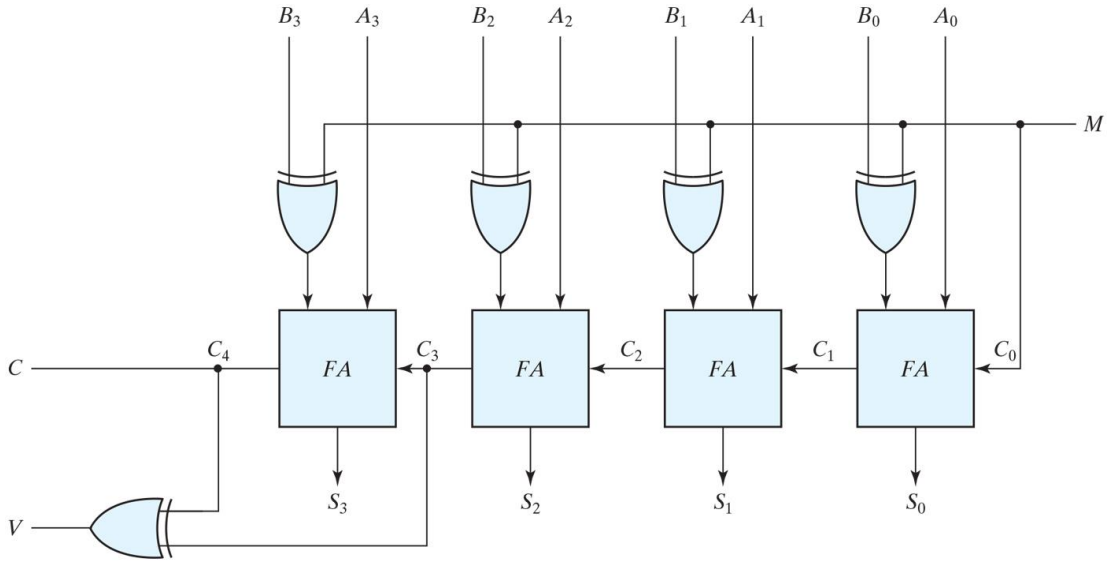
- s_1s_0 controls the mode of the register :
  - s_1s_0=00 → No change
  - s_1s_0=01 → Right shift
  - s_1s_0=10 → Left shift
  - s_1s_0=11 → Parallel load

Serial Adder



Serial Adder – Parallel Adder





Ex.

- Design a serial adder using sequential logic methods.
  - Two shift registers to keep the numbers to be added. Serial out of the registers are x and y
  - S output for summation of x and y
  - D flip-flop for carry.

  - State table?

Ex.

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

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

- State Table

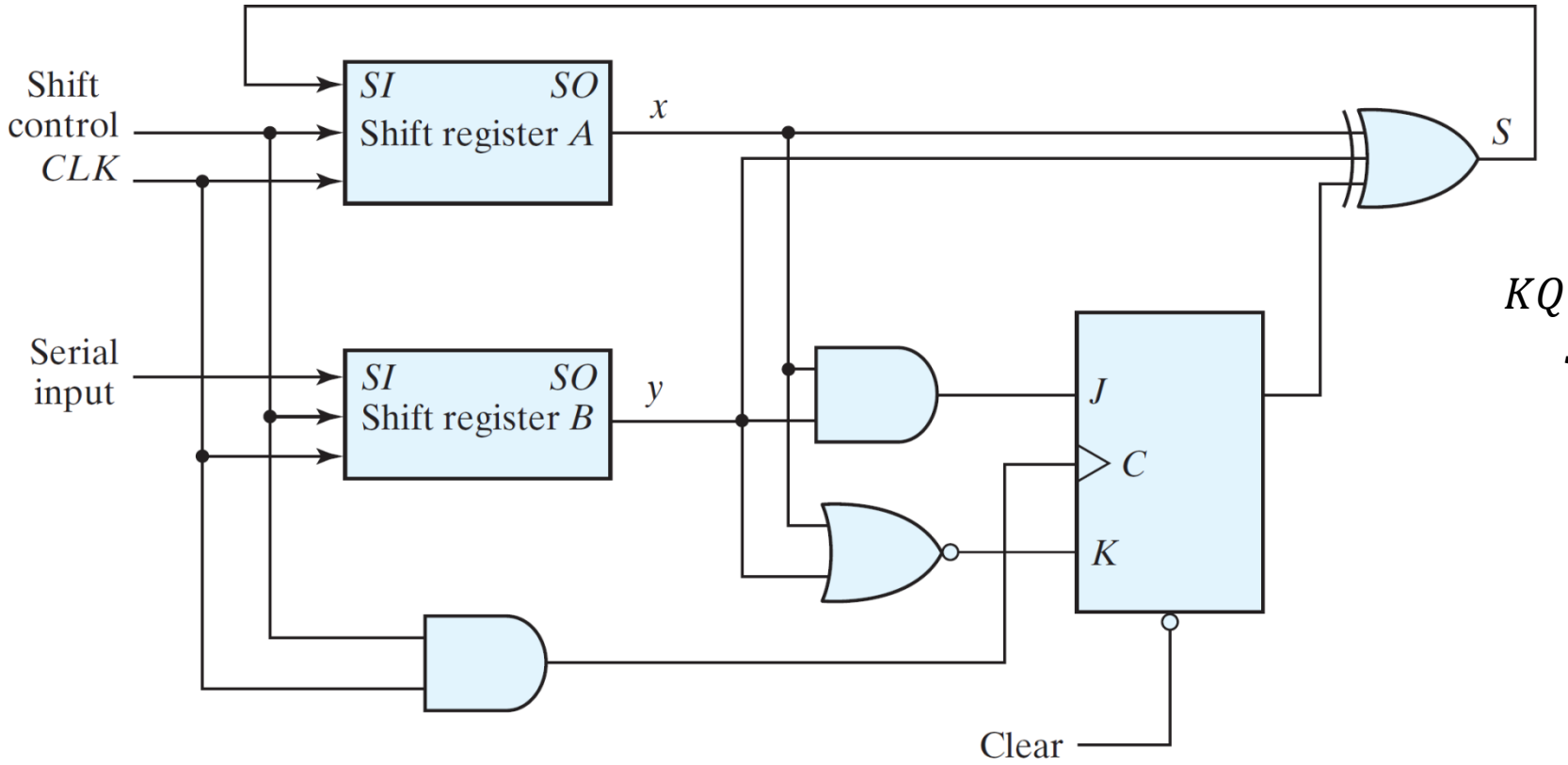
Present State	Inputs		Next State	Output	Flip-flop inputs	
Q	x	y	Q	S	JQ	KQ
0	0	0	0	0	1	X
0	0	1	0	1	0	X
0	1	0	0	1	0	X
0	1	1	1	0	1	X
1	0	0	0	1	X	1
1	0	1	1	0	X	0
1	1	0	1	0	X	0
1	1	1	1	1	X	0

$$JQ = xy$$
$$KQ = x'y' = (x + y)'$$
$$S = x \oplus y \oplus Q$$

Şimdiki Durum	Girişler		Sonraki Durum	Çıkış	Flip-flop girişleri	
Q	x	y	Q	S	JQ	KQ
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

Ex.

- Logic diagram of the serial adder: (second form)



$$JQ = xy$$
$$KQ = x'y' = (x + y)'$$
$$S = x \oplus y \oplus Q$$

References

- Digital Design, Morris Mano, Second Addition