Logic Design and Computer Organization

IS234AT

Unit - II

Dr. Premananda B.S.

Modules

I. Synchronous Sequential Logic
II. Registers, Counters, and Memory
III. A Top-Level View of Computer Function and Interconnection, Cache Memory
IV. Input/Output and Computer Arithmetic
V. Instruction Sets Characteristics and Functions, Processor Structure and Function, and Parallel Processing

Books

- **Digital Design with an Introduction to the Verilog HDL, M. Morris Mano, Michael D. Ciletti, 5th Edition, 2013, Pearson.**

Registers, Counters, and Memory

- Registers, Registers with Parallel load
- Shift Registers, Serial Addition, Universal Shift Register
- Binary Ripple Counters, BCD Ripple Counters
- Synchronous Counters: Binary Counters, BCD Counters
- Other Counters: Ring and Johnson Counter
- Design Examples
- Random Access Memory
- Memory Decoding
- Read Only Memory
- Semiconductor Main Memory Organization

Registers

- A clocked sequential circuit consists of a group of flip-flops and combinational gates.
- Circuits that include flip-flops are classified by the function they perform rather than by the name of the sequential circuit.
- 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 consists of a group of *n* flip-flops capable of storing *n* bits of binary information.
- 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.

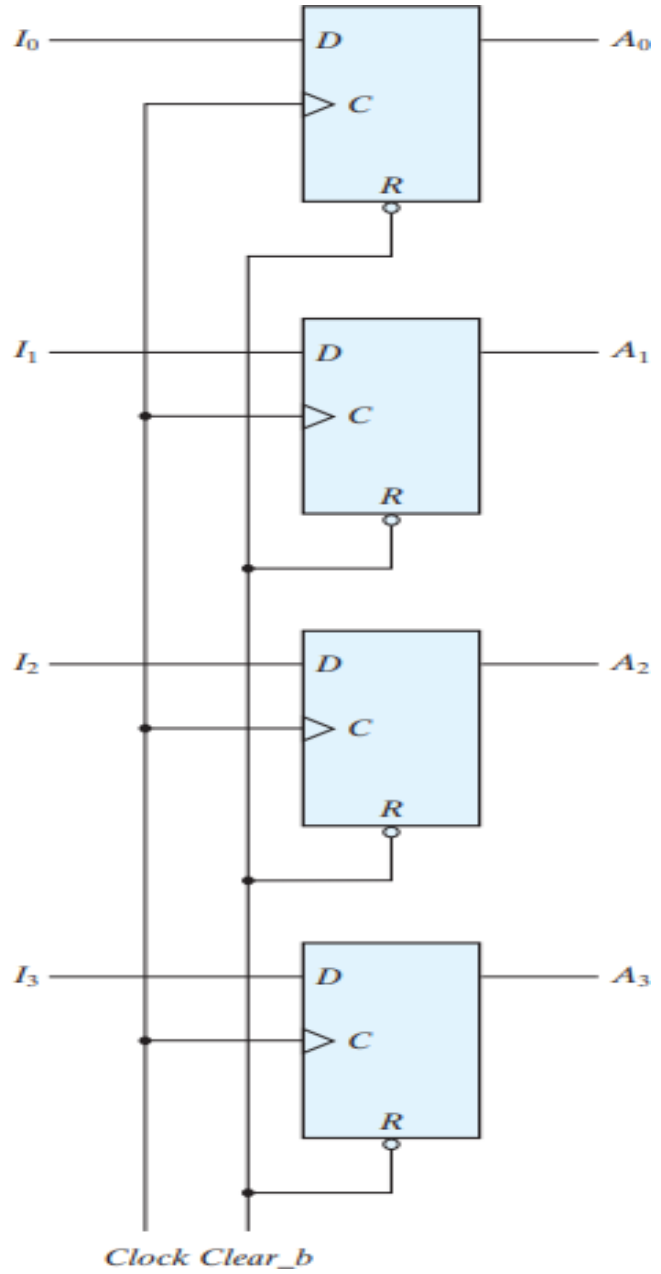
Counter

- A *counter* is essentially 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.
- Although counters are a special type of register, it is common to differentiate them by giving them a different name.

- 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.

Four-bit Registers

- A register constructed with four D-type flip-flops to form a four-bit data storage register as shown.
- 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.
- Outputs can be sampled at any time to obtain the binary information stored in the register.
- The Clear_b input is useful for clearing the register to all 0's before its clocked operation.
- The R inputs must be maintained at logic 1 (de-asserted) during normal clocked operation.

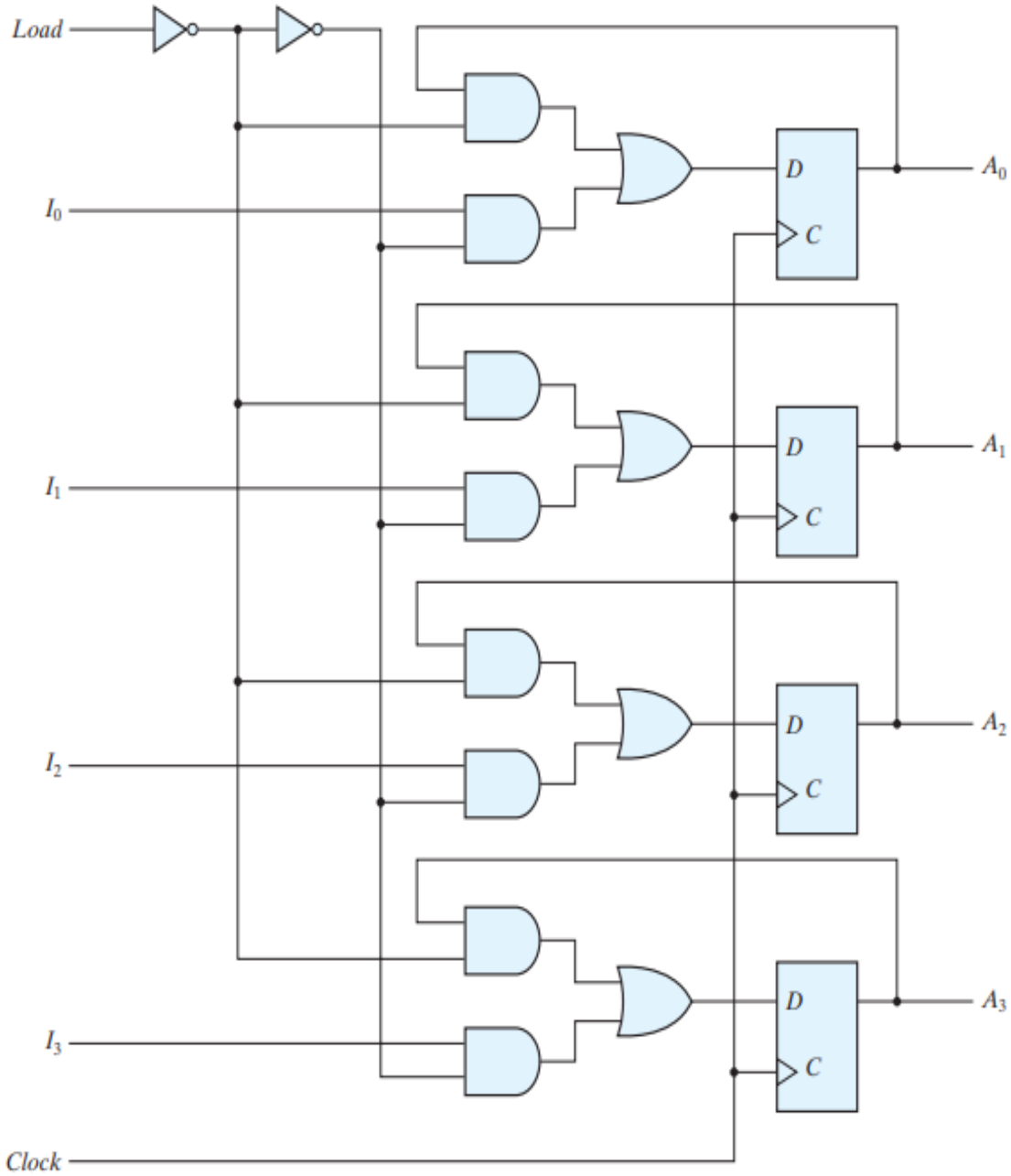


Register with Parallel Load

- The transfer of new information into a register is referred to as loading or updating the register.
- Parallel load: All bits of the register are loaded simultaneously with a common clock pulse.
- If the contents of the register must be left unchanged:
  i. inputs must be held constant or
  ii. clock must be inhibited from the circuit
- First case, the data bus driving the register would be unavailable for other traffic.
- Second case, the clock can be inhibited from reaching the register by controlling the clock input signal with an enabling gate.
- Inserting gates into the clock path is ill-advised because it means that logic is performed with clock pulses.
- Advisable to control the operation of the register with the D inputs, rather than controlling the clock in the C inputs of the flip-flops.

Register with Parallel Load

- A four-bit data-storage register with a load control input that is directed through gates and into the D inputs of the flip-flops is shown in Figure.
- Additional gates implement a two-channel multiplexer (mux) whose output drives the input to the register with either the data bus or the output of the register.
- The load input to the register determines the action to be taken with each clock pulse.
- When the load input is 1, the data at the four external inputs are transferred into the register with the next positive edge of the clock.
- When the load input is 0, the outputs of the flip-flops are connected to their respective inputs.
- The feedback connection from output to input is necessary because a D flip-flop does not have a "no change" condition.

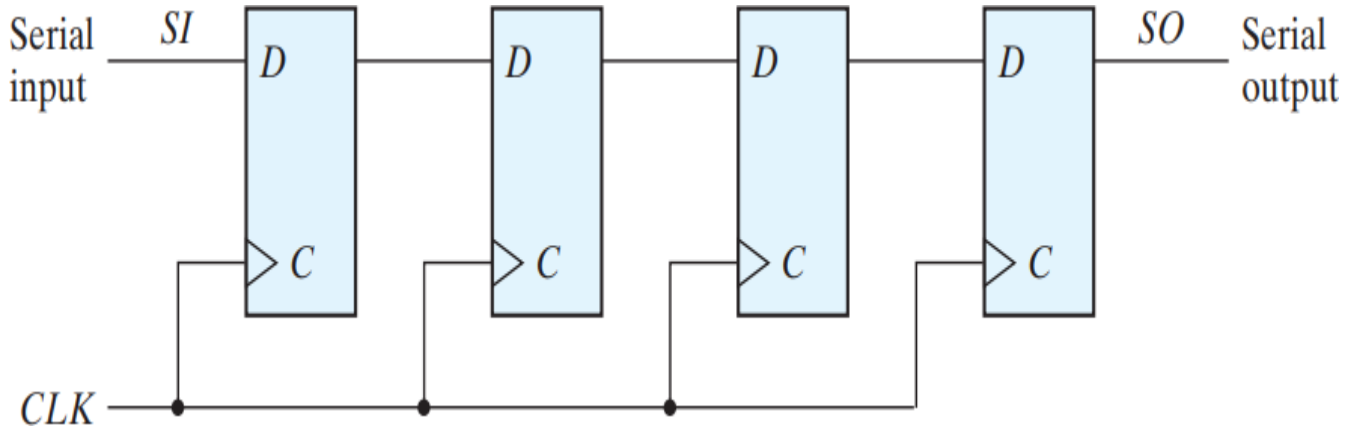


Registers, Counters, and Memory

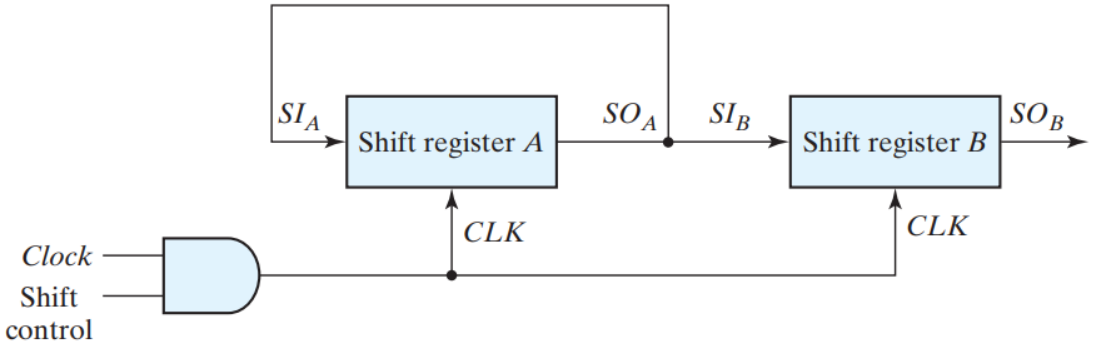
- Registers, Registers with Parallel load
- **Shift Registers, Serial Addition, Universal Shift Register**
- Binary Ripple Counters, BCD Ripple Counters
- Synchronous Counters: Binary Counters, BCD Counters
- Other Counters: Ring and Johnson Counter
- Random Access Memory
- Memory Decoding
- Read Only Memory
- Semiconductor Main Memory Organization

Shift Registers

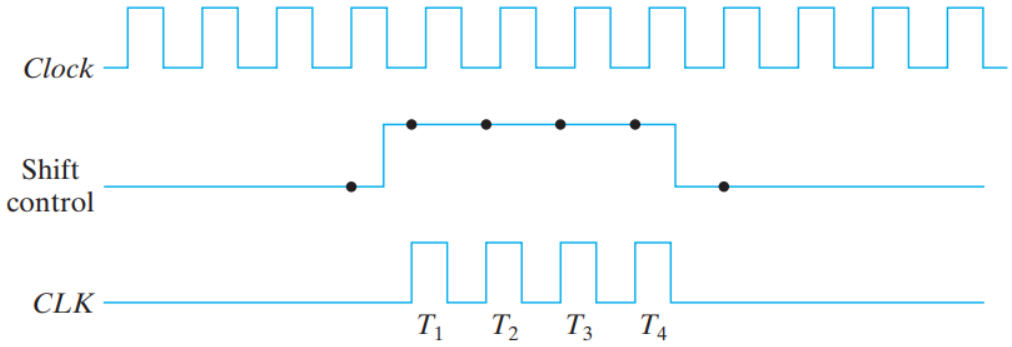
- A register capable of shifting the binary information held in each cell to its neighboring cell, in a selected direction.
- The logical configuration of a shift register consists of a chain of flip-flops in cascade, with the output of one flip-flop connected to the input of the next flip-flop.
- All flip-flops receive common clock pulses, which activate the shift of data from one stage to the next
- The unidirectional (left-to-right) four-bit shift register is shown.



Serial Transfer



(a) Block diagram



(b) Timing diagram

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

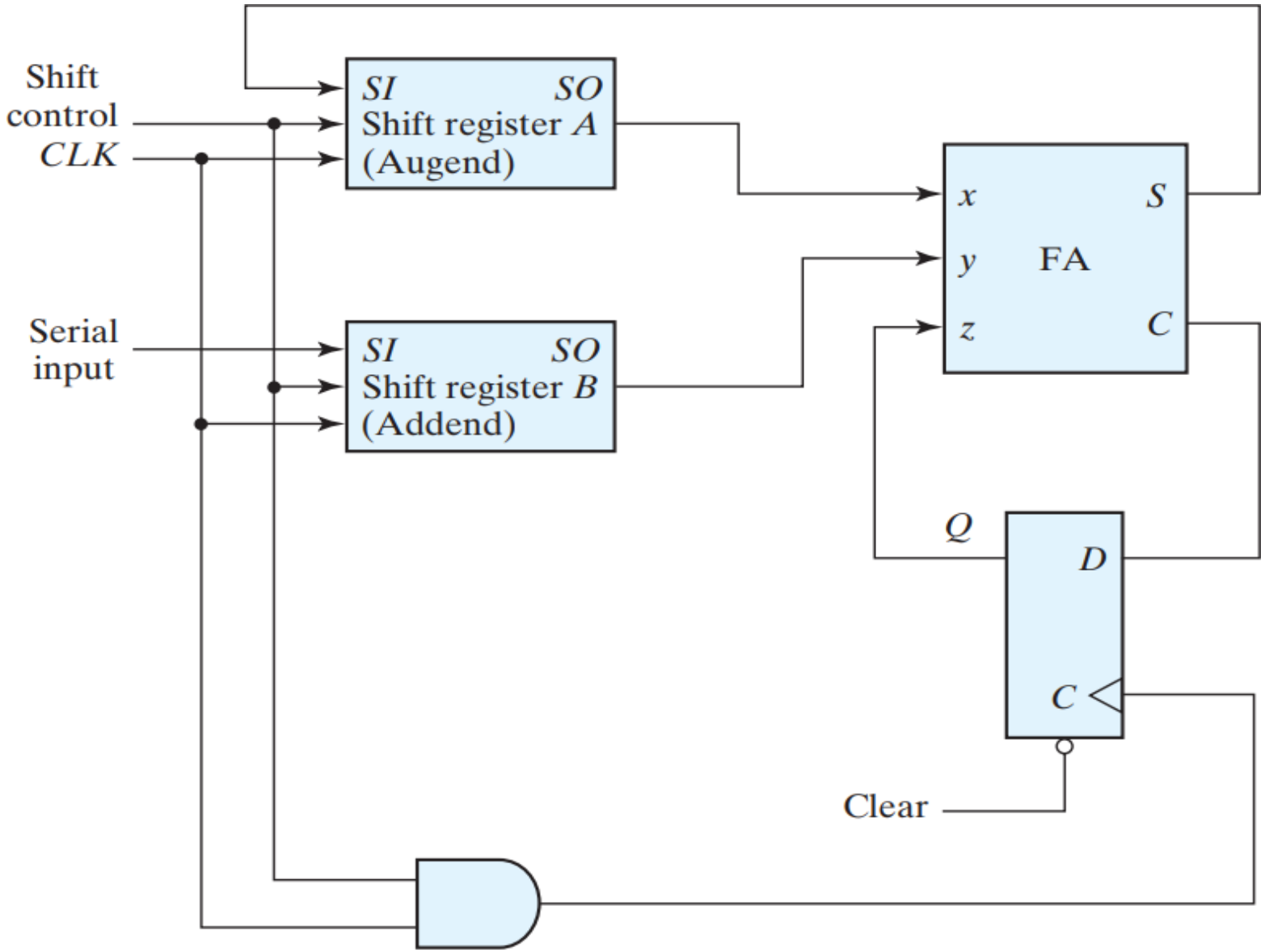
- The serial output (SO) of register A is connected to the serial input (SI) of register B.
- To prevent the loss of information stored in the source register, the information in register A is made to circulate by connecting the serial output to its serial input.
- The shift control input determines when and how many times the registers are shifted. For illustration here, this is done with an AND gate that allows clock pulses to pass into the CLK terminals only when the shift control is active.
- The shift control signal is synchronized with the clock and changes value just after the negative edge of the 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.
- The fourth pulse changes the shift control to 0, and the shift registers are disabled.
- Example

Serial Addition

- Operations in digital computers are performed in parallel because of the faster mode of operation.
- Serial operations are slower because a datapath operation takes several clock cycles, but serial operations have the advantage of requiring fewer hardware components.
- The two binary numbers to be added serially are stored in two shift registers.
- Starting with the least significant pair of bits, the circuit adds one pair at a time through a single full-adder (FA) circuit.
- The carry-out of the full adder is transferred to a D flip-flop, the output of which is then used as the carry input for the next pair of significant bits.
- The sum bit from the S output of the full adder could be transferred into a third shift register.
- Serial Adder design using D/JK Flip-flops is discussed next.

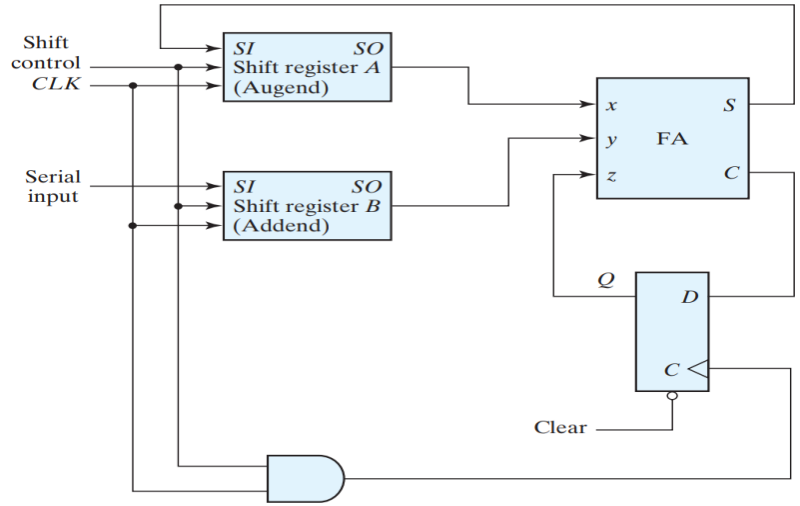
A. Serial Adder using D Flip-flop

- Shift the sum into register A while the bits of register A are shifted out, it is possible to use one register for storing both the augend and the sum bits.
- The serial input of register B can be used to transfer a new binary number while the addend bits are shifted out during the addition.
- Initially, register A holds augend, register B holds addend, and the carry flip-flop is cleared to 0.
- The outputs (SO) of A and B provide a pair of significant bits for the full adder at x and y.
- Operation…



Serial Adder using D Flip-flop

- The state table that specifies the sequential circuit is shown.
- The present state of Q is the present value of carry.
- The present carry in Q is added together with inputs *x* and *y* to produce the sum bit in output S.
- 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.



State Table for Serial Adder

Present State	Inputs		Next State	Output
Q	x	y	Q	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

B. Serial Adder using JK Flip-flop

- 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.

State Table for Serial Adder

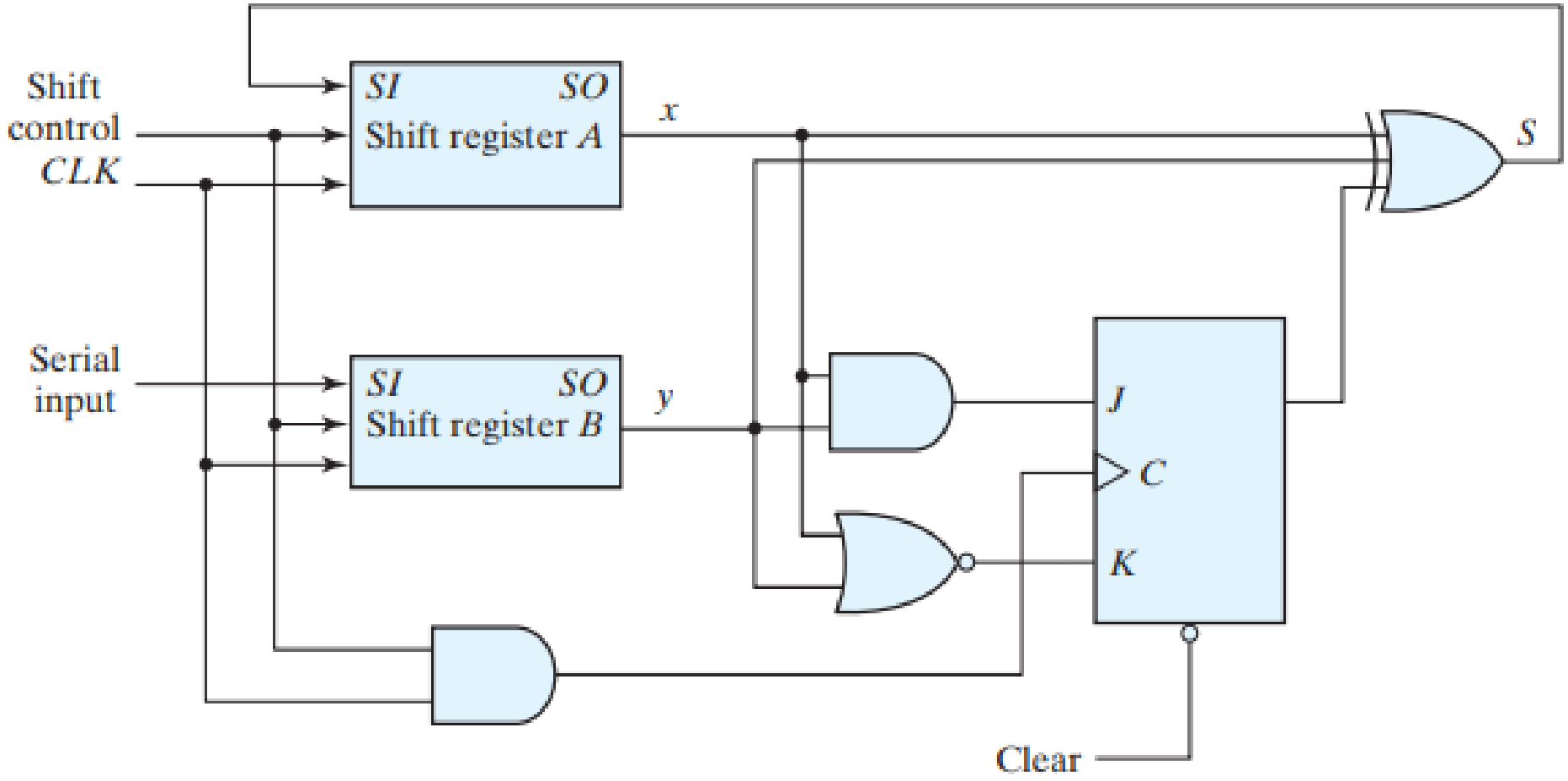
Present State	Inputs		Next State	Output	Flip-Flop Inputs	
Q	x	y	Q	S	J_Q	K_Q
0	0	0	0	0	0	X
0	0	1	0	1	0	X
0	1	0	0	1	0	X
0	1	1	1	0	1	X
1	0	0	0	1	X	1
1	0	1	1	0	X	0
1	1	0	1	0	X	0
1	1	1	1	1	X	0

- Solving using K-maps.
- The flip-flop input equations and the output equation are.

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

Serial Adder using JK Flip-flop

- 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.
- Logic diagram is as shown.

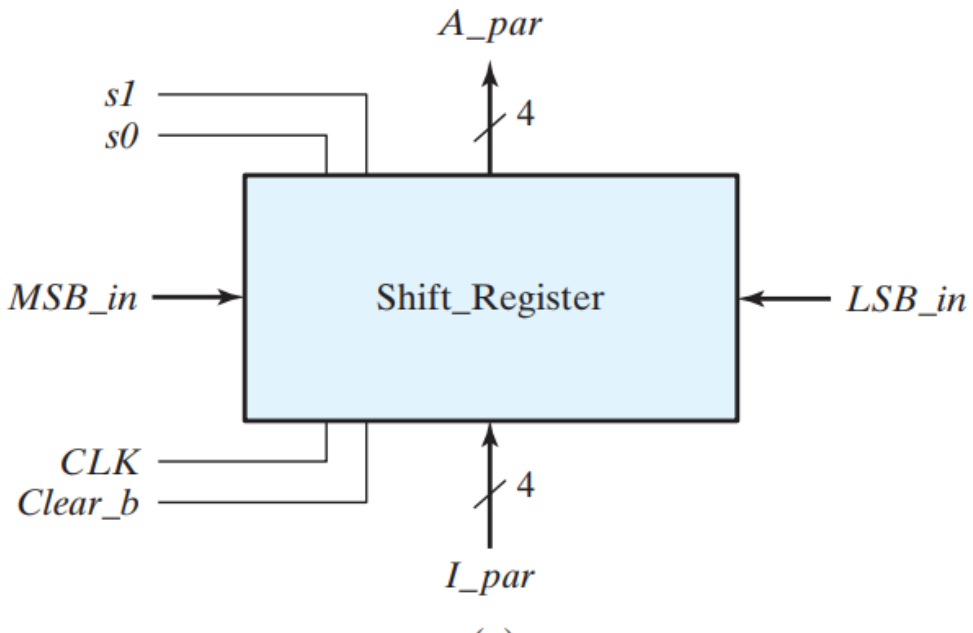


Universal Shift Register

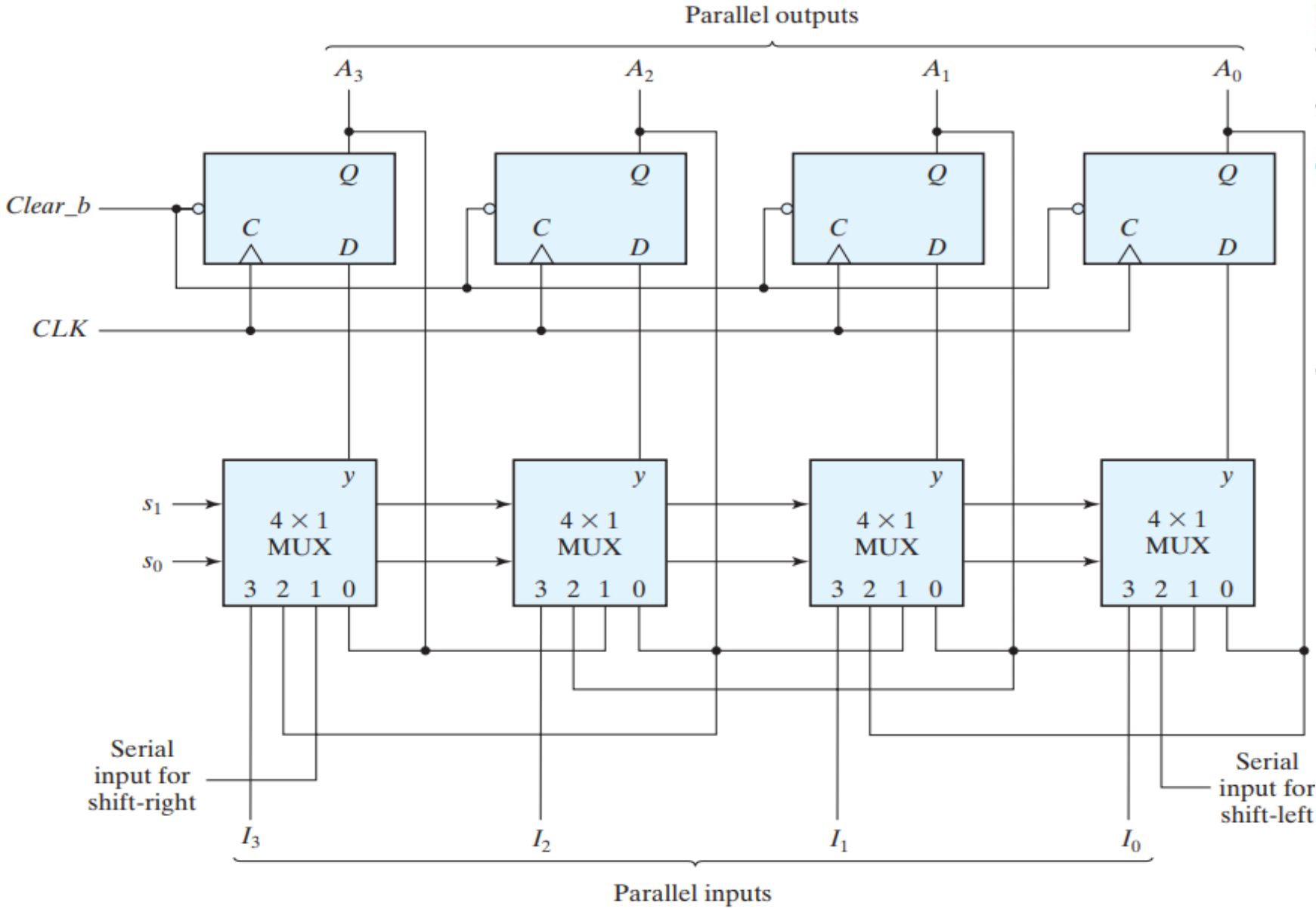
- The most general shift register has the following capabilities:
  1. A *clear* control to clear the register to 0.
  2. A *clock* input to synchronize the operations.
  3. A *shift-right* control to enable the shift-right operation and the serial input and output lines associated with the shift-right.
  4. A *shift-left* control to enable the shift-left operation and the serial input and output lines associated with the shift-left.
  5. A *parallel-load* control to enable a parallel transfer and the n input lines associated with the parallel transfer.
  6. *n* parallel output lines.
  7. A control state that leaves the information in the register unchanged in response to the clock. Other shift registers may have only some of the preceding functions, with at least one shift operation.

Universal Shift Register

- Some shift registers provide the necessary input and output terminals for parallel transfer.
- They may also have both shift-right and shift-left capabilities.
- A register capable of shifting in one direction only is a unidirectional shift register.
- One that can shift in both directions is a bidirectional shift register.
- If the register has both shifts and parallel-load capabilities, it is referred to as a universal shift register (USR).
- The block diagram symbol of 4-bit USR is shown.



Four-bit Universal Shift Register



Function Table for the Register

Mode Control		
s_1	s_0	Register Operation
0	0	No change
0	1	Shift right
1	0	Shift left
1	1	Parallel load

Shift Register

- Shift registers are often used to interface digital systems situated remotely from each other.
- For example, suppose it is necessary to transmit an n-bit quantity between two points.
- If the distance is far, it will be expensive to use n lines to transmit the n bits in parallel.
- It will be economical to use a single line and transmit the information serially, one bit at a time.
- The transmitter accepts the n-bit data in parallel into a shift register and then transmits the data serially along the common line.
- The receiver accepts the data serially into a shift register.
- When all n bits are received, they can be taken from the outputs of the register in parallel.
- Thus, the transmitter performs a parallel-to-serial conversion of data and the receiver does a serial-to-parallel conversion.

Shift Registers-Examples

- Chip Study:
  - IC-74LS95: A 4-bit shift register with serial and parallel synchronous modes.
  - IC-74LS195: A 4-bit parallel access shift register with asynchronous master reset.

Registers, Counters, and Memory

- Registers, Registers with Parallel load
- Shift Registers, Serial Addition, Universal Shift Register
- **Binary Ripple Counters, BCD Ripple Counters**
- Synchronous Counters: Binary Counters, BCD Counters
- Other Counters: Ring and Johnson Counter
- Random Access Memory
- Memory Decoding
- Read Only Memory
- Semiconductor Main Memory Organization

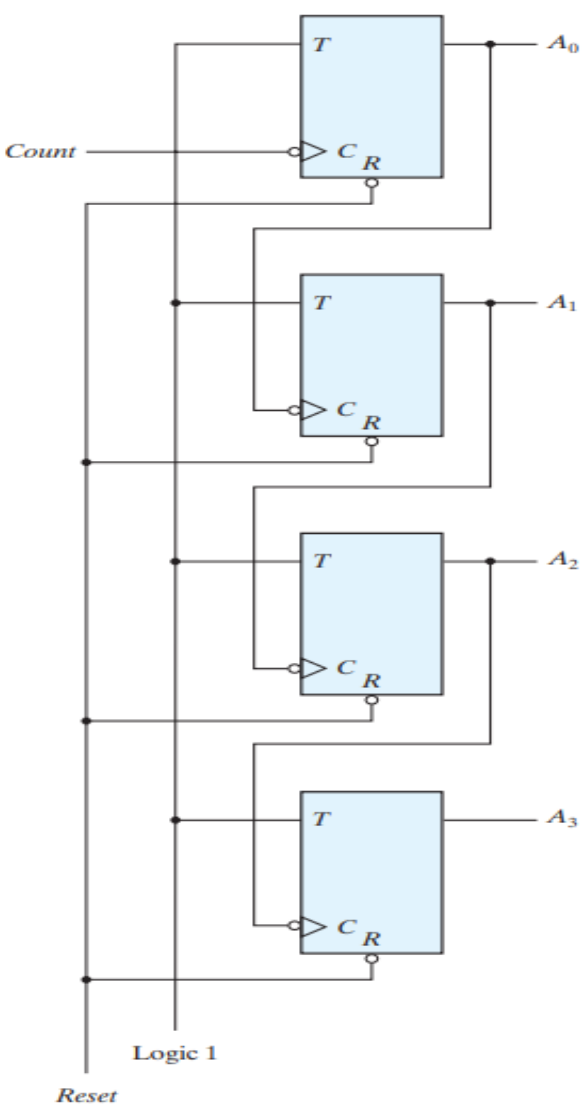
Ripple Counters

- Counters are a special type of registers.
- It is a register that goes through a predetermined sequence of binary states.
- A counter that follows the binary number sequence is called a binary counter.
- An *n*-bit binary counter consists of *n* flip-flops and can count in binary from 0 to $2^n - 1$.
- Counters are available in two categories: ripple counters and synchronous counters.
- In a ripple counter, a flip-flop output transition serves as a source for triggering other flip-flops. In other words, the C input of some or all flip-flops is triggered, not by the common clock pulses, but rather by the transition that occurs in other flip-flop outputs.
- In a synchronous counter, the C inputs of all flip-flops receive the common clock.

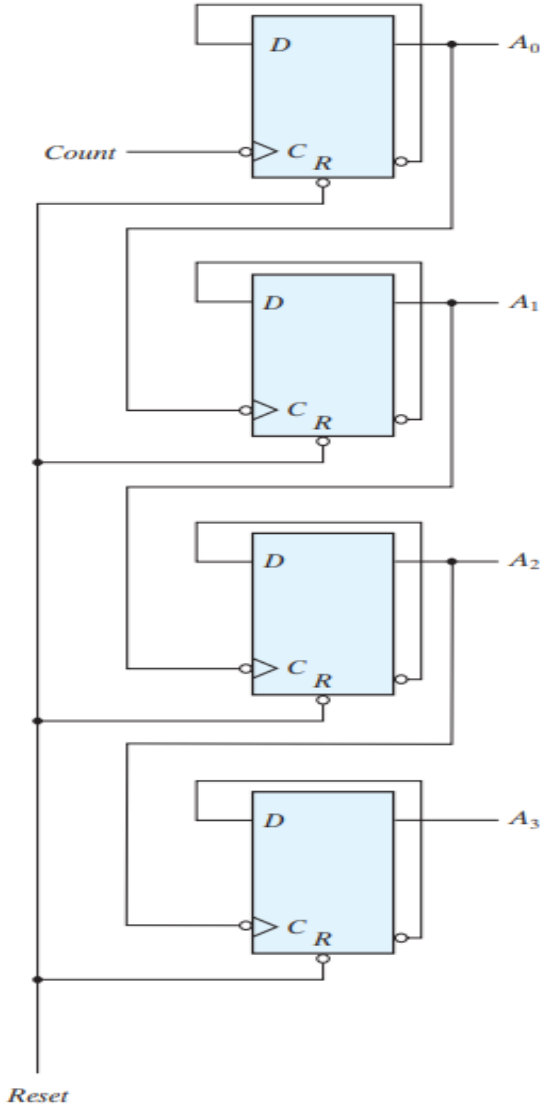
Binary Ripple Counters-Four-bit

- A binary ripple counter consists of a series connection of complementing flip-flops, with the output of each flip-flop connected to the C input of the next higher-order flip-flop.
- The flip-flop holding the least significant bit receives the incoming count pulses.
- A complementing flip-flop can be obtained from a JK flip-flop with the J and K inputs tied together or from a T flip-flop.
- A third possibility is to use a D flip-flop with the complement output connected to the D input.
- The D input is always the complement of the present state, and the next clock pulse will cause the flip-flop to complement.
- The counter is constructed with complementing flip-flops of the T type in part (a) and the D type in part (b) shown next.

Binary Ripple Counters-Four-bit



(a) With *T* flip-flops



(b) With *D* flip-flops

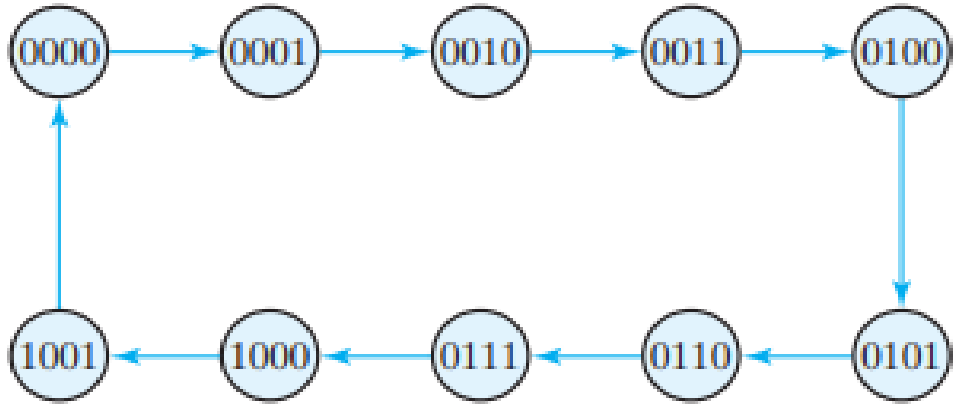
Binary Count Sequence

A_3	A_2	A_1	A_0
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0

- Every time that A0 goes from 1 to 0, it complements A1.
- Every time that A1 goes from 1 to 0, it complements A2.
- Every time that A2 goes from 1 to 0, it complements A3, and so on for any other higher-order bits of a ripple counter.

BCD Ripple Counter

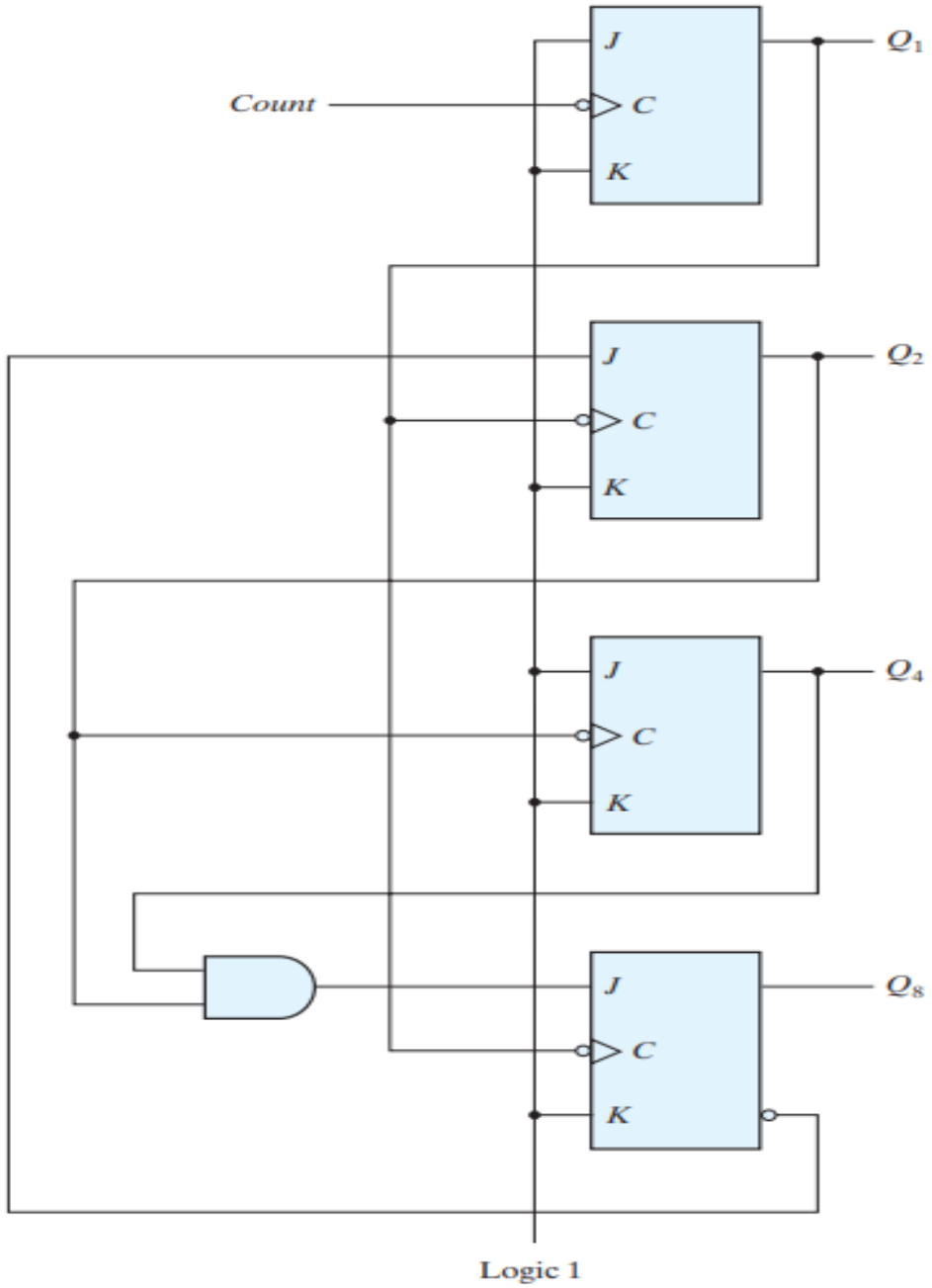
- A decimal counter follows a sequence of 10 states and returns to 0 after the count of 9.
- Such a counter must have at least four flip-flops to represent each decimal digit since a decimal digit is represented by a binary code with at least four bits.
- The sequence of states in a decimal counter is dictated by the binary code used to represent a decimal digit.
- If Binary Coded Decimal (BCD) is used, the sequence of states is as shown in a state diagram.
- A decimal counter is similar to a binary counter, except that the state after 1001 (the code for decimal digit 9) is 0000 (the code for decimal digit 0).
- The BCD counter shown next is a decade counter since it counts from 0 to 9.



State diagram of BCD Counter

BCD Ripple Counter

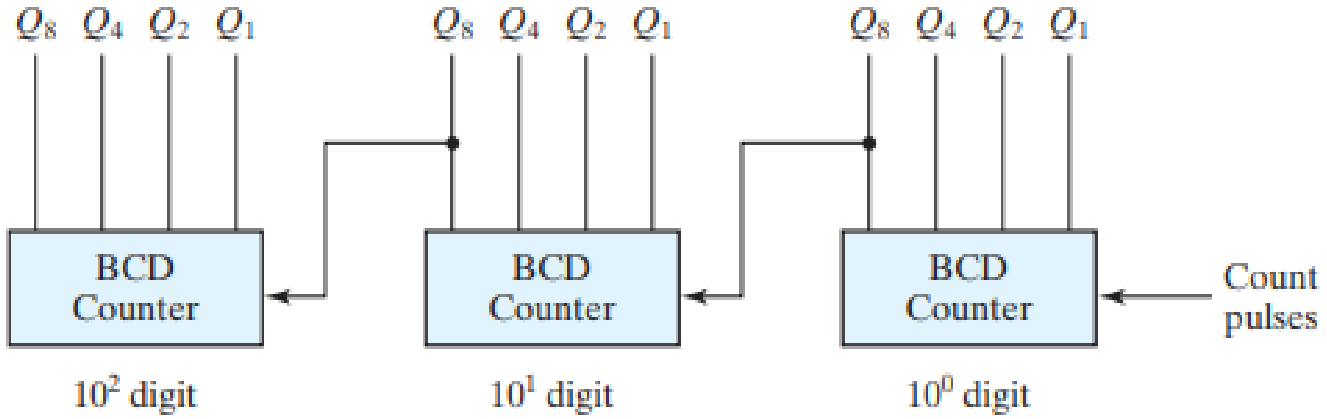
- Outputs are designated by Q, with a numeric subscript equal to the binary weight of corresponding bit in the BCD code.
- The output of Q1 is applied to the C inputs of both Q2 and Q8 and the output of Q2 is applied to the C input of Q4.
- The J and K inputs are connected either to a permanent 1 signal or to outputs of other flip-flops.
  - When the C input goes from 1 to 0, the flip-flop is set if J = 1, is cleared if K = 1, is complemented if J = K = 1, and is left unchanged if J = K = 0.
- Q1 changes state after each clock pulse.
- Q2 complements every time Q1 goes from 1 to 0, as long as Q8 = 0.
- When Q8 becomes 1, Q2 remains at 0.
- Q4 complements every time Q2 goes from 1 to 0.
- Q8 remains at 0 as long as Q2 or Q4 is 0.
- When both Q2 and Q4 become 1, Q8 complements when Q1 goes from 1 to 0.
- Q8 is cleared on the next transition of Q1.



Decade Counter

BCD Ripple Counter

- To count in decimal from 0 to 99, we need a two-decade counter.
- To count from 0 to 999, we need a three-decade counter.
- Multiple decade counters can be constructed by connecting BCD counters in a cascade, one for each decade.
- A three-decade counter is shown in Figure.
- The inputs to the second and third decades come from Q8 of the previous decade.
- When Q8 in one decade goes from 1 to 0, it triggers the count for the next higher-order decade while its decade goes from 9 to 0.



Block diagram of a three-decade decimal BCD counter

Registers, Counters, and Memory

- Registers, Registers with Parallel load
- Shift Registers, Serial Addition, Universal Shift Register
- Binary Ripple Counters, BCD Ripple Counters
- **Synchronous Counters: Binary Counters, BCD Counters**
- Other Counters: Ring and Johnson Counter
- Random Access Memory
- Memory Decoding
- Read Only Memory
- Semiconductor Main Memory Organization

Synchronous Counters

- Synchronous counters are different from ripple counters in that clock pulses are applied to the inputs of all flip-flops.
- A common clock triggers all flip-flops simultaneously, rather than one at a time in succession as in a ripple counter.
- The decision whether a flip-flop is to be complemented is determined from the values of the data inputs, such as *T* or *J* and *K* at the time of the clock edge.
  - If T = 0 or J = K = 0, the flip-flop does not change state.
  - If T = 1 or J = K = 1, the flip-flop complements.

A. Binary Counter

B. Up-Down Binary Counter

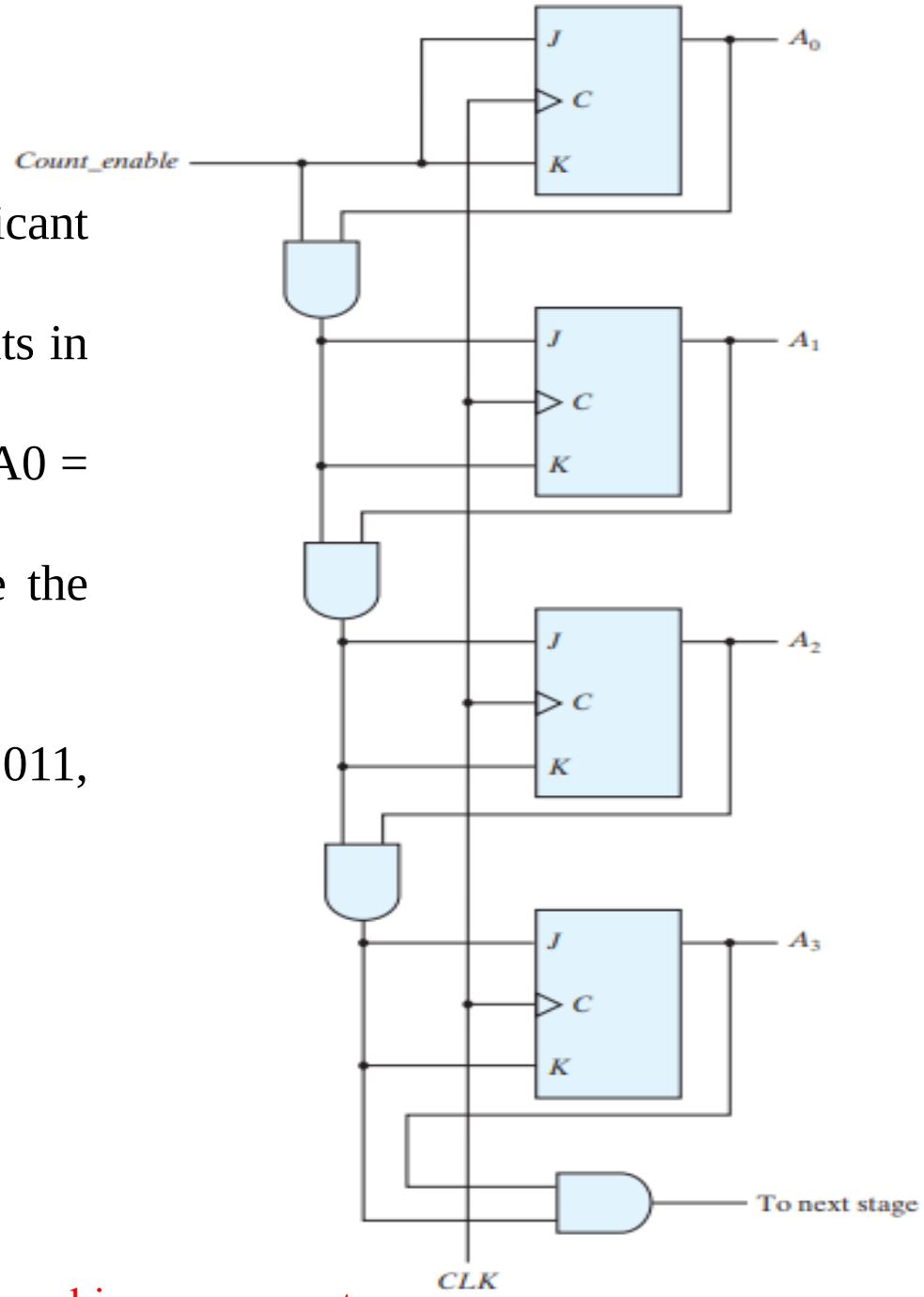
C. BCD Counter

D. Binary Counter with Parallel Load

A. Binary Counter

- In a synchronous binary counter, the flip-flop in the least significant position is complemented with every pulse.
- A flip-flop in any other position is complemented when all the bits in the lower significant positions are equal to 1.
- For example, if the present state of a four-bit counter is A3A2A1A0 = 0011, the next count is 0100.
- A0 is always complemented, and A1 is complemented because the present state of A0 = 1.
- A2 is complemented because the present state of A1A0 = 11.
- A3 is not complemented, because the present state of A2A1A0 = 011, which does not give an all-1's condition.

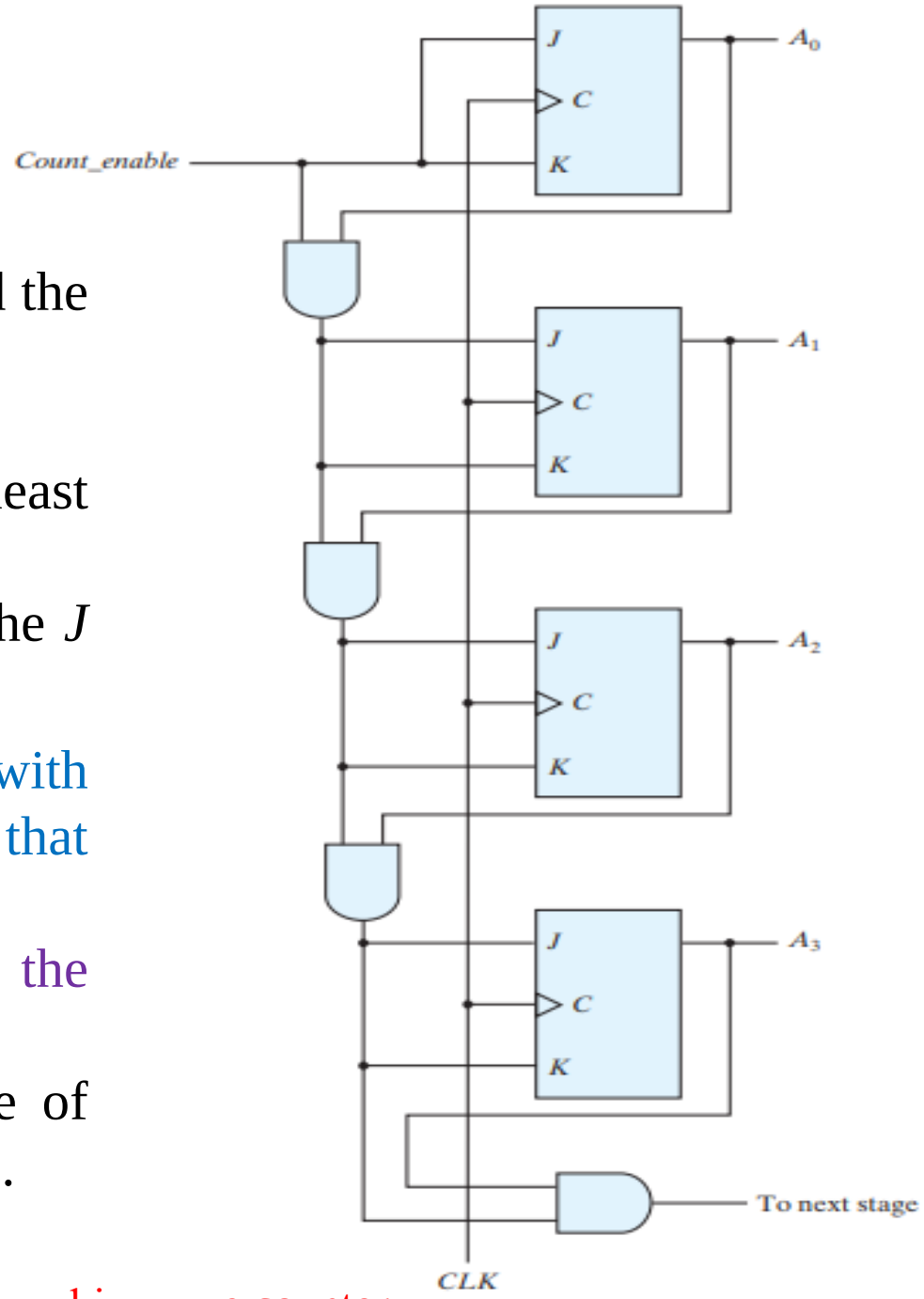
A_3	A_2	A_1	A_0
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0



Four-bit synchronous binary up counter

A. Binary Counter

- The counter is enabled by *Count_enable*.
- If the enable input is 0, all *J* and *K* inputs are equal to 0 and the clock does not change the state of the counter.
- The A0, has its *J* and *K* = 1 if the counter is enabled.
- The other *J* and *K* inputs are equal to 1 if all previous least significant stages are equal to 1 and the count is enabled.
- The chain of AND gates generates the required logic for the *J* and *K* inputs in each stage.
- The counter can be extended to any number of stages, with each stage having an additional flip-flop and an AND gate that gives an output of 1 if all previous flip-flop outputs are 1.
- The synchronous counter can be triggered with either the positive or the negative clock edge.
- The complementing flip-flops in a binary counter can be of either the *JK* type, the *T* type, or the *D* type with XOR gates.



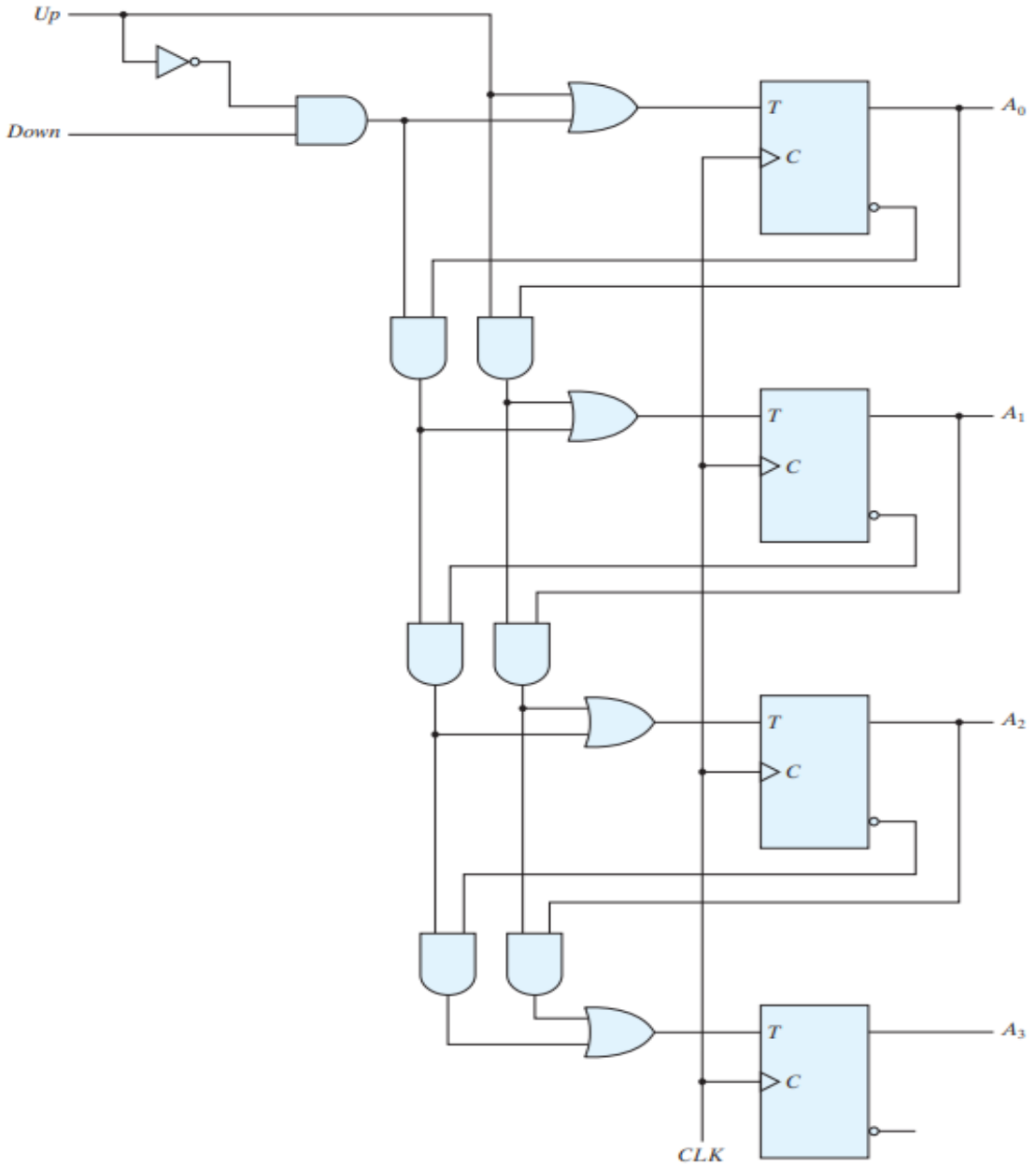
Four-bit synchronous binary up counter

A. Countdown Binary Counter

- A synchronous countdown binary counter goes through the binary states in reverse order, from 1111 down to 0000 and back to 1111 to repeat the count.
- A countdown binary counter can be constructed similar to Figure (Four-bit synchronous binary up counter), except that the inputs to the AND gates must come from the complemented outputs, instead of the normal outputs, of the previous flip-flops.
- The bit in the least significant position is complemented with each pulse.
- A bit in any other position is complemented if all LSBs are equal to 0.

B. Up-Down Binary Counter

- The two operations (up and down) can be combined in one circuit to form a counter capable of counting either *up* or *down*.
- The circuit of an up–down binary counter using T flip-flops is shown in Figure.
- It has an up-control and a down-control input.
- When the ***up*** is 1, the circuit counts up, since the T inputs receive their signals from the values of the previous normal outputs of the flip-flops.
- When the ***down*** is 1 and ***up*** is 0, the circuit counts down, since the complemented outputs of the previous flip-flops are applied to the T inputs.
- When the ***up*** and ***down*** inputs are both 0, the circuit does not change state and remains in the same count.
- When the ***up*** and ***down*** inputs are both 1, the circuit counts up.
- The set of conditions ensures that only one operation is performed at any given time.



Four-bit up-down binary counter

C. BCD Counter Design using *JK* Flip-flops

- A BCD counter counts in binary-coded decimal from 0000 to 1001 and back to 0000.
- The state table of a BCD counter is listed in Table along with the output y.
- An output y, which is equal to 1 when the present state is 1001.
- The y can enable the count of the next-higher significant decade while the same pulse switches the present decade from 1001 to 0000.
- The input conditions for the *T* flip-flops are obtained from the present- and next-state conditions.

State Table for BCD Counter

Present State				Next State				Output
Q_8	Q_4	Q_2	Q_1	Q_8	Q_4	Q_2	Q_1	y
0	0	0	0	0	0	0	1	0
0	0	0	1	0	0	1	0	0
0	0	1	0	0	0	1	1	0
0	0	1	1	0	1	0	0	0
0	1	0	0	0	1	0	1	0
0	1	0	1	0	1	1	0	0
0	1	1	0	0	1	1	1	0
0	1	1	1	1	0	0	0	0
1	0	0	0	1	0	0	1	0
1	0	0	1	0	0	0	0	1

C. BCD Counter Design using *JK* Flip-flops

- The state table of a BCD counter with Flip-flop inputs is listed in Table.
- The flip-flop input equations can be simplified using K-maps.
- The unused states for minterms **10** to **15** are taken as don't-care terms.

State Table for BCD Counter

Present State				Next State				Output	Flip-Flop Inputs			
Q_8	Q_4	Q_2	Q_1	Q_8	Q_4	Q_2	Q_1	y	TQ_8	TQ_4	TQ_2	TQ_1
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	1	0	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	0	1
0	0	1	1	0	1	0	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	0	1
0	1	0	1	0	1	1	0	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	0	1
0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	0	1
1	0	0	1	0	0	0	0	1	1	0	0	1

C. BCD Counter Design using *JK* Flip-flops

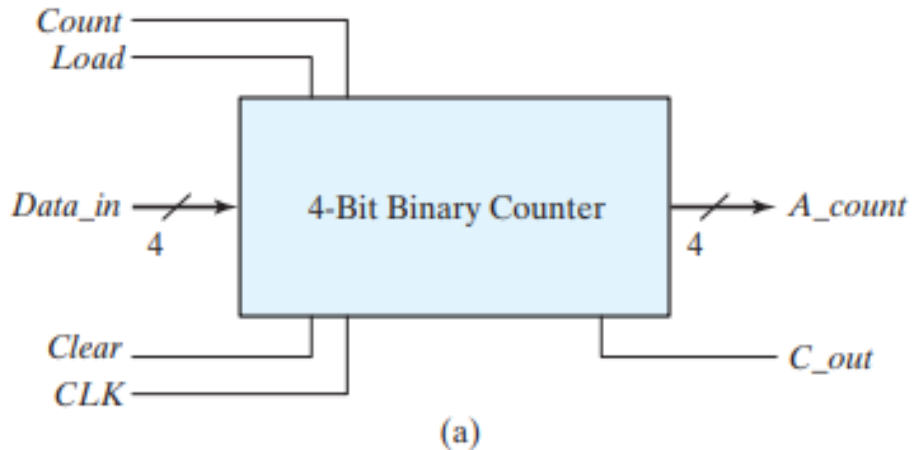
- The simplified functions using K-maps are:

$$T_{Q1} = 1$$
$$T_{Q2} = Q'_8 Q_1$$
$$T_{Q4} = Q_2 Q_1$$
$$T_{Q8} = Q_8 Q_1 + Q_4 Q_2 Q_1$$
$$y = Q_8 Q_1$$

- Draw logic diagram:
  - Requires four *T* flip-flops, five AND gates, and one OR gate.

D. Binary Counter with Parallel Load

- When the *load* is equal to 1, the input load control disables the count operation and causes a transfer of data from the four data inputs into the four flip-flops.
- If both control inputs (load and count) are 0, clock pulses do not change the state of the register.
- The carry output (C_out) is 1 if all the flip-flops are equal to 1 while the count input is enabled.
- This is the condition for complementing the flip-flop that holds the next significant bit.
- The carry output is useful for expanding the counter to more than four bits.
- The speed of the counter is increased when the carry is generated directly from the outputs of all four flip-flops because the delay to generate the carry bit is reduced.
- The four control inputs: *Clear, CLK, Load,* and *Count* - determine the next state.
- The operation of the counter is summarized in Table.

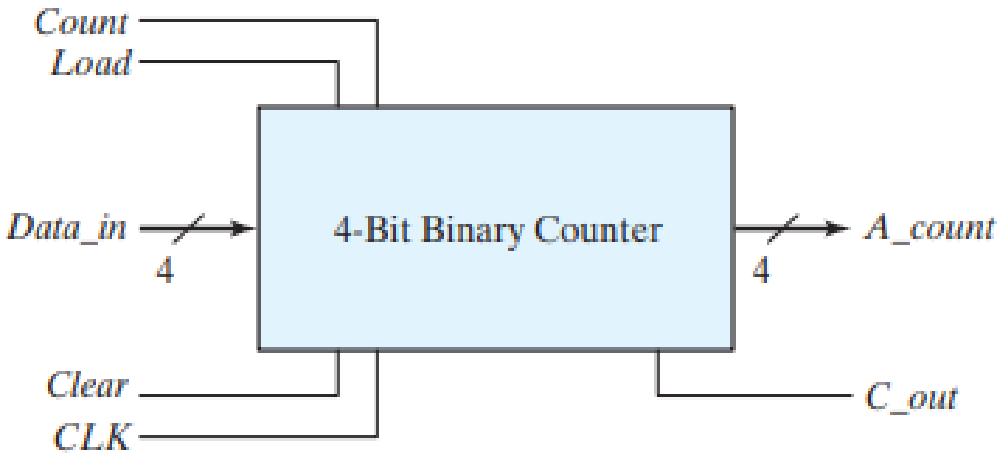


(a)

Function Table for the Counter

Clear	CLK	Load	Count	Function
0	X	X	X	Clear to 0
1	↑	1	X	Load inputs
1	↑	0	1	Count next binary state
1	↑	0	0	No change

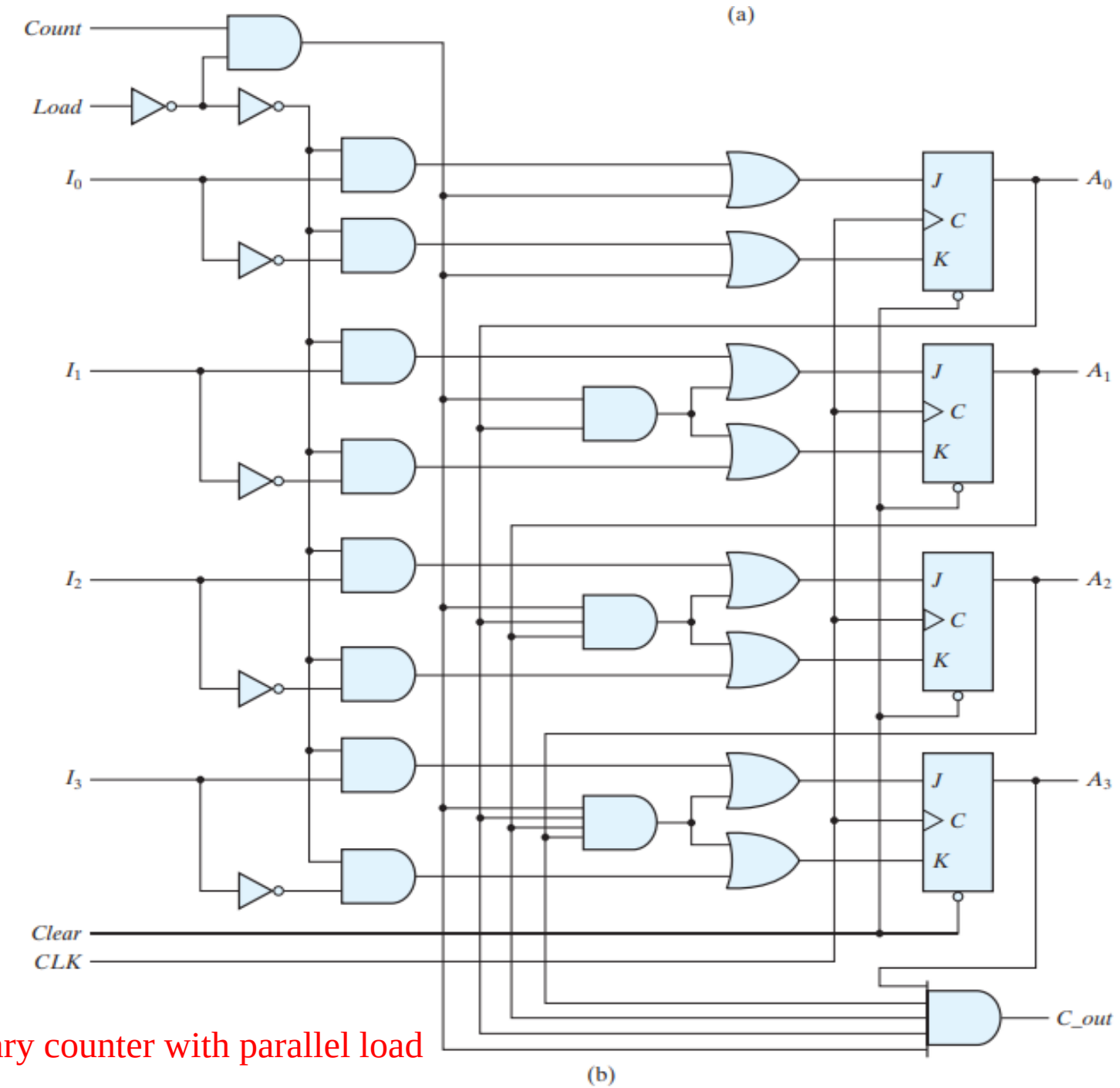
Binary Counter with Parallel Load



(a)

Function Table for the Counter

Clear	CLK	Load	Count	Function
0	X	X	X	Clear to 0
1	↑	1	X	Load inputs
1	↑	0	1	Count next binary state
1	↑	0	0	No change



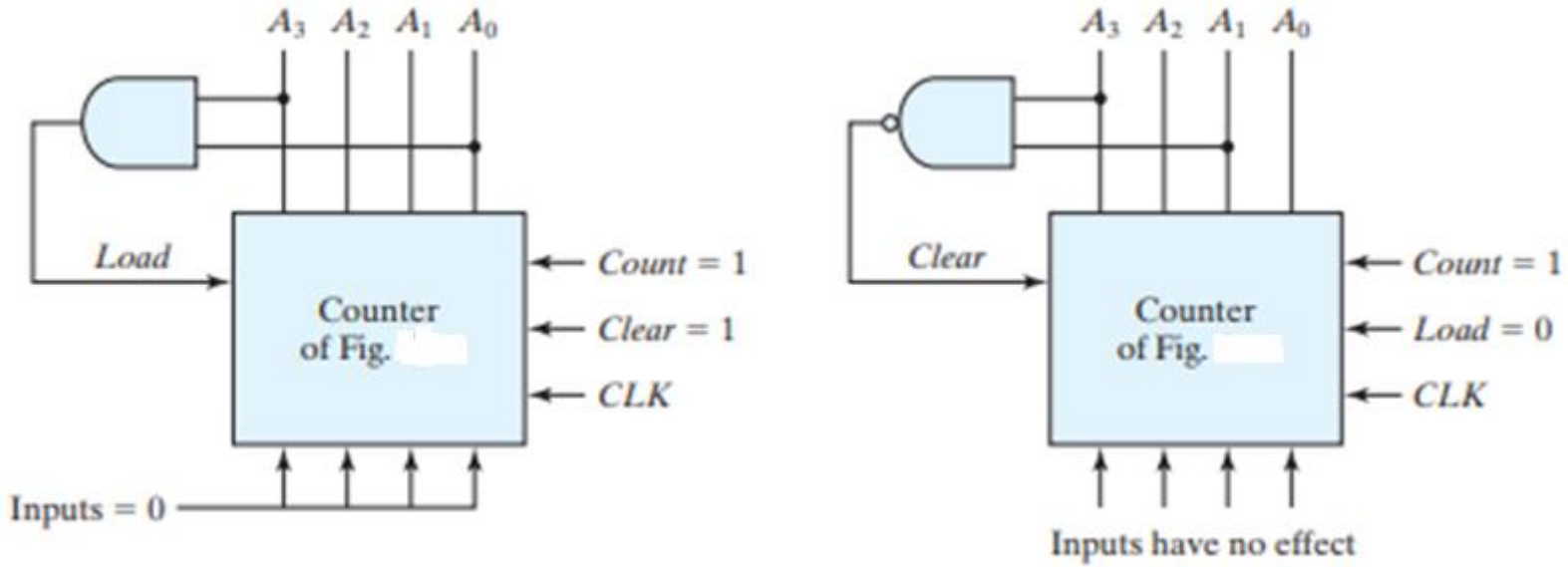
Four-bit binary counter with parallel load

BCD Counter using a Counter with Parallel load: Examples

- A counter with a parallel load can be used to generate any desired count sequence.
- The *Count control* is set to 1 to enable the count through the CLK input.
- The *Load control* inhibits the count and the clear operation is independent of other control inputs.
- The AND gate in Figure (a) shown next detects the occurrence of state 1001.
- The counter is initially cleared to 0, and then the Clear and Count inputs are set to 1, so the counter is active at all times.
- Until the AND gate output is 0, each positive-edge clock increments the counter by 1.
- When the output reaches the count of 1001, both A0 and A3 become 1, making the output of the AND gate equal to 1.
  - This condition activates the *Load* input; therefore, on the next clock edge, the register does not count but is loaded from its four inputs.
  - All four inputs are connected to logic 0, an all-0's (0000) value is loaded into the register following the count of 1001.
- The count is from 0000 through 1001 and back to 0000, as is required in a **BCD counter**.

BCD Counter using a Counter with Parallel load: Examples

- In Figure (b), the NAND gate detects the count of 1010, but as soon as this count occurs, the register is cleared.
- The count 1010 has no chance of staying on for any appreciable time because the register goes immediately to 0.
- A momentary spike occurs in output A0 as count goes from 1010 to 1011 & immediately to 0000.
- The spike may be undesirable, and for that reason, this configuration is not recommended.
- If the counter has a synchronous clear input, it is possible to clear the counter with the clock after an occurrence of the 1001 count.



(a) Using the load input

(b) Using the clear input

Registers, Counters, and Memory

- Registers, Registers with Parallel load
- Shift Registers, Serial Addition, Universal Shift Register
- Binary Ripple Counters, BCD Ripple Counters
- Synchronous Counters: Binary Counters, BCD Counters
- **Other Counters: Ring and Johnson Counter**
- Random Access Memory
- Memory Decoding
- Read Only Memory
- Semiconductor Main Memory Organization

Other Counters

- Timing signals that control the sequence of operations in a digital system can be generated by a shift register or by a counter with a decoder.
- Counters can be designed to generate any desired sequence of states.
- A divide-by-N (modulo-N) counter goes through a repeated sequence of N states.
- The sequence may follow the binary count or may be any other arbitrary sequence.
- Counters are used to generate timing signals to control the sequence of operations in a digital system.
- Counters can also be constructed using shift registers.

1. Counter with Unused States
2. Ring Counter
3. Johnson Counter

1. Counter with Unused States: Example

- The next state from an unused state can be determined from the analysis of the circuit after it is designed.
- **Example: For the counter table specified in state table, design using *JK* Flip-flops.**
  - The count has a repeated sequence of six states, with flip-flops B and C repeating the binary count 00, 01, and 10.
  - The flip-flop A alternating between 0 and 1 every three counts.
  - The count sequence of the counter is not straight binary, and two states, 011 and 111, are not included in the count.

State Table for Counter

Present State			Next State		
A	***B***	***C***	***A***	***B***	***C***
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	0	0	0

Counter with Unused States: Using *JK* Flip-flops

- Realization: The choice of *JK* flip-flops results in the flip-flop input conditions listed in the table.
- Inputs K_B and K_C have only 1's and X's in their columns, so these inputs are always equal to 1.
- The other flip-flop input equations obtained by using minterms 3 and 7 as don't-care conditions.

State Table for Counter

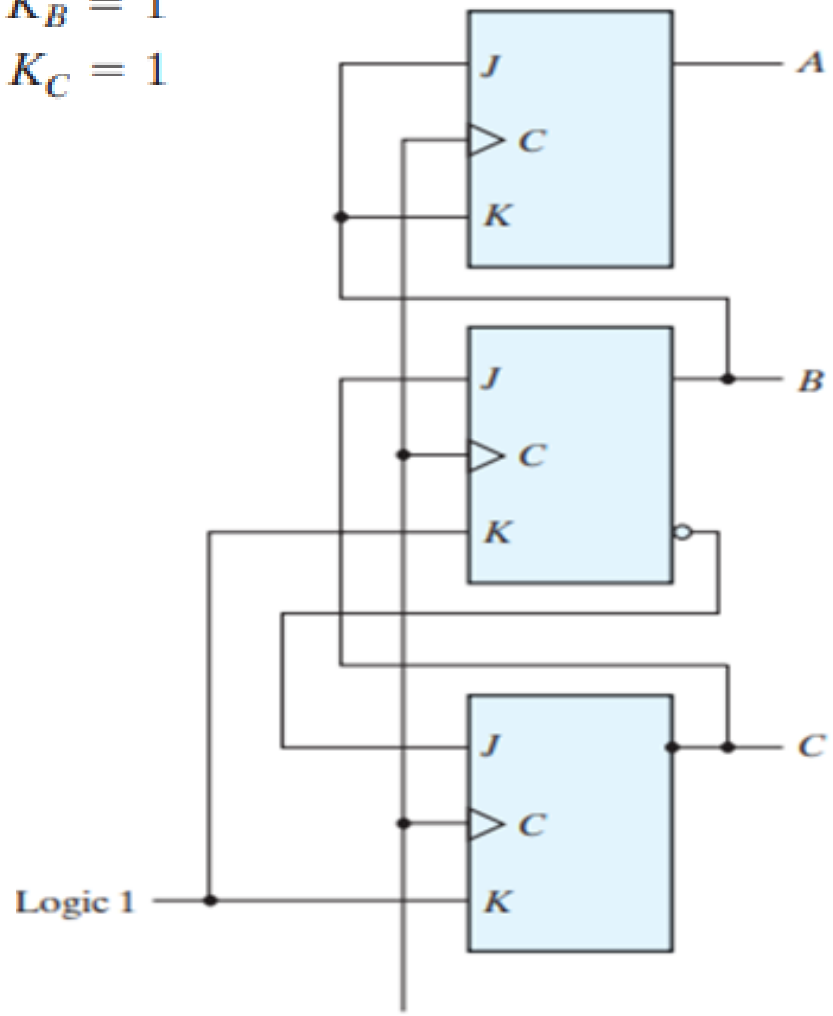
Present State			Next State			Flip-Flop Inputs					
A	*B*	*C*	*A*	*B*	*C*	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	1	0	0	1	X	X	1	0	X
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	1	1	0	X	0	1	X	X	1
1	1	0	0	0	0	X	1	X	1	0	X

- Solve using K-maps.

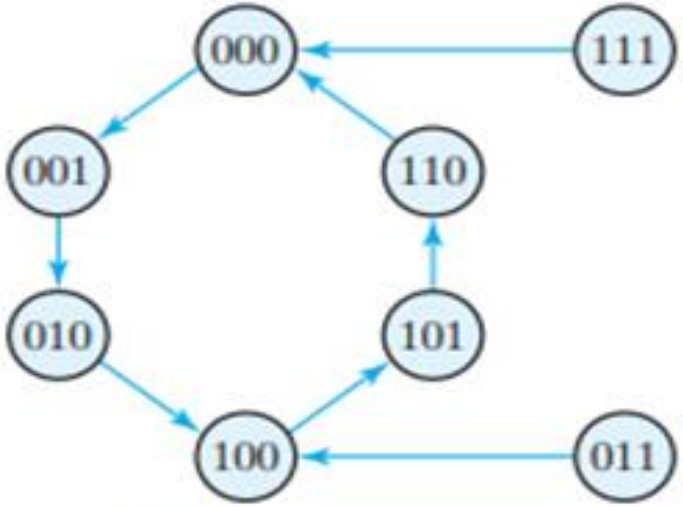
Counter with Unused States: Using *JK* Flip-flops

- The simplified equations are: $J_A = B \quad K_A = B$
  $J_B = C \quad K_B = 1$
  $J_C = B' \quad K_C = 1$

- The logic diagram of the counter is shown in Figure (a).
- The state diagram including the effect of the unused states is shown in Figure (b).
- If the circuit is in state 011 because of an error signal, the circuit goes to state 100 after the application of a clock pulse.
- When B = 1, the next clock edge complements A and clears C to 0.
- When C = 1, the next clock edge complements B.
- Similarly, evaluate the next state from present state 111 to be 000.



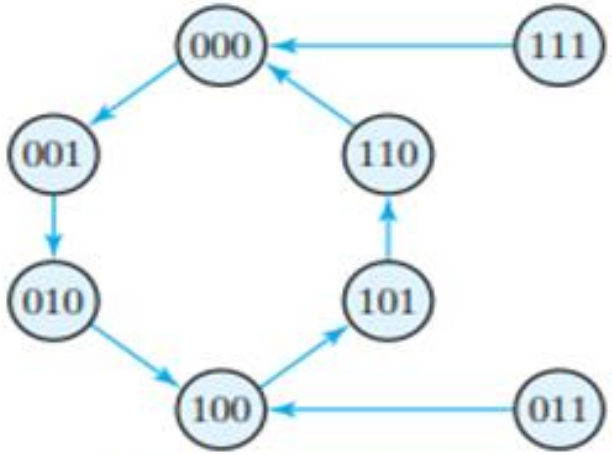
(a) Logic circuit diagram



(b) State transition diagram

Counter with Unused States

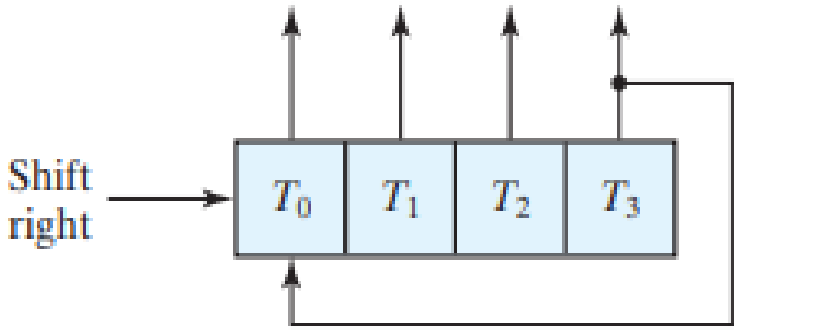
- The state diagram including the effect of the unused states is shown in Figure (b).
- If the circuit ever goes to one of the unused states because of outside interference, the next count pulse transfers it to one of the valid states and the circuit continues to count correctly, the counter is self-correcting.
- In a self-correcting counter, if the counter happens to be in one of the unused states, it eventually reaches the normal count sequence after one or more clock pulses.
- An alternative design could use additional logic to direct every unused state to a specific next state.



(b) State transition diagram

2. Ring Counter

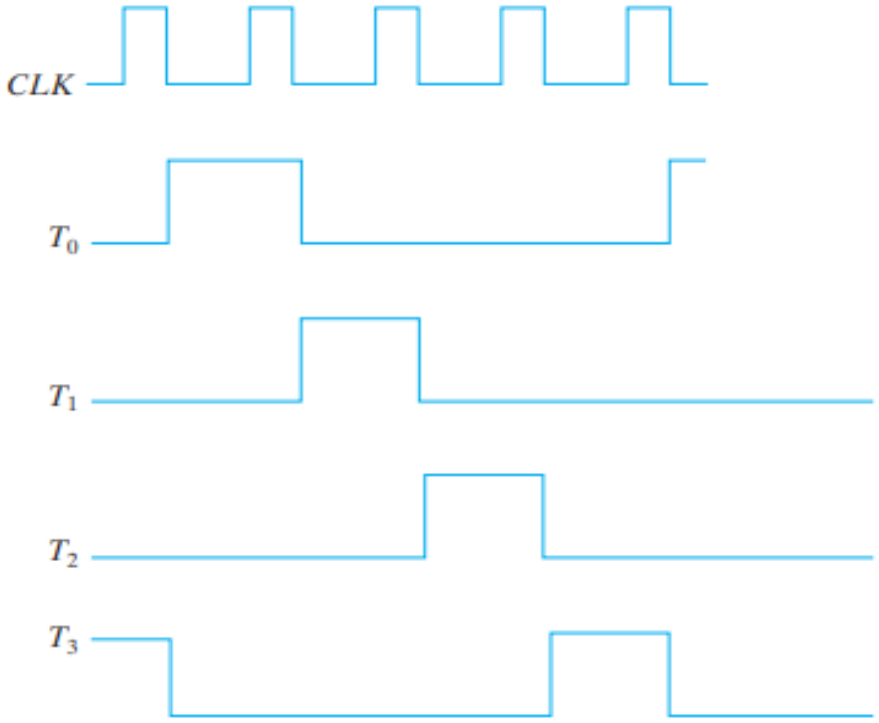
- Timing signals that control the sequence of operations in a digital system can be generated by a shift register or by a counter with a decoder.
- A *ring counter* is a circular shift register with only one flip-flop being set at any particular time; all others are cleared.
- One bit is shifted from one flip-flop to the next to produce the sequence of timing signals.
- Figure (a) shows a four-bit shift register connected as a ring counter.
- The initial value of the register is 1000 and requires Preset/Clear flip-flops.
- The single bit is shifted right with every clock pulse and circulates back from T_3 to T_0.



(a) Ring-counter (initial value = 1000)

Ring Counter

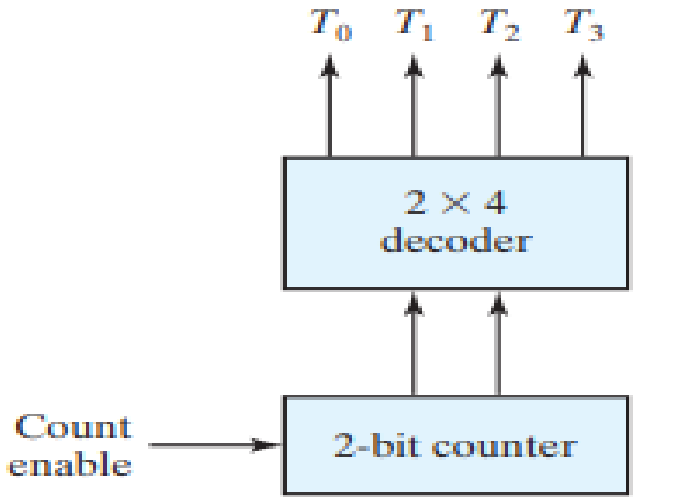
- Each flip-flop is in the 1 state once every four clock cycles and produces one of the four timing signals shown in Figure (b).
- Each output becomes a 1 after the negative-edge transition of a clock pulse and remains 1 during the next clock cycle.
- For an alternative design, the timing signals can be generated by a two-bit counter that goes through four distinct states.



(b) Sequence of four timing signals

Ring Counter

- The decoder shown in Figure decodes the four states of the counter and generates the required sequence of timing signals.
- To generate 2^n timing signals, we need either a:
  i. shift register with 2^n flip-flops or
  ii. n-bit binary counter together with an n-to-2^n-line decoder



(c) Counter and decoder

Ring Counter

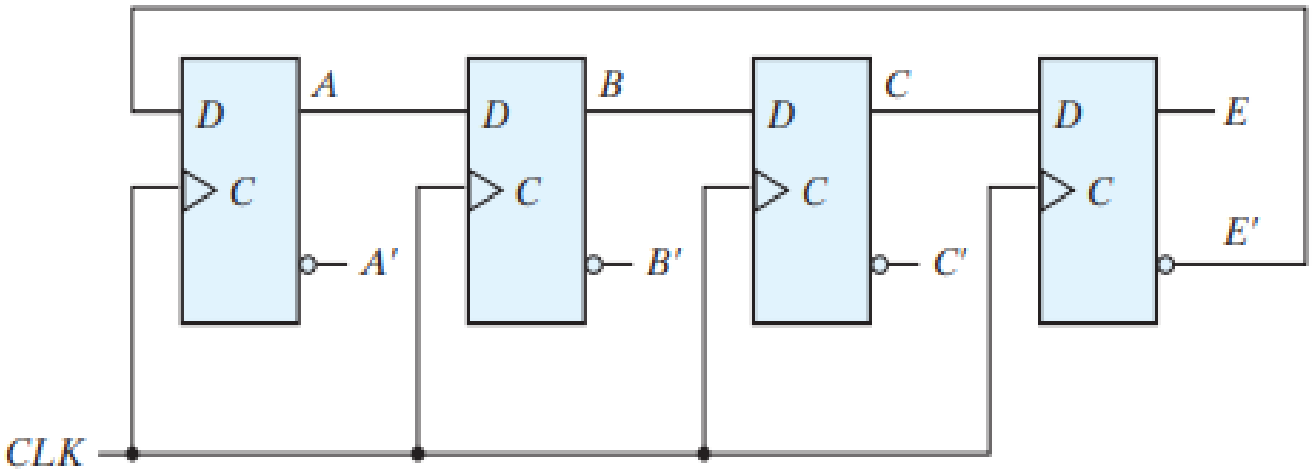
- To generate 2^n timing signals, we need either:
  i. shift register with 2^n flip-flops or
  ii. n-bit binary counter together with an n-to-2^n-line decoder
- For example, 16 timing signals can be generated with a:
  i. 16-bit shift register connected as a ring counter or
     - We need 16 flip-flops.
  ii. 4-bit binary counter and a 4-to-16-line decoder.
     - We need 4 flip-flops and 16 four-input AND gates for the decoder .
- It is also possible to generate the timing signals with a combination of a shift register and a decoder.
- The number of flip-flops is less than that in a ring counter, and the decoder requires only two-input gates, this combination is called a *Johnson counter*.

3. Johnson Counter

- A *k-bit ring counter* circulates a single bit among the flip-flops to provide *k* distinguishable states.
- The number of states can be doubled if the shift register is connected as a *switch-tail* ring counter.
- A *switch-tail* ring counter is a circular shift register with the complemented output of the last flip-flop connected to the input of the first flip-flop.
- A *k*-bit *switch-tail* ring counter will go through a sequence of 2*k* states.
- Starting from all 0's, each shift operation inserts 1's from the left until the register is filled with all 1's.
- In the next sequences, 0's are inserted from the left until the register is again filled with all 0's.
- A *Johnson counter* is a *k-bit switch-tail ring counter* with 2*k* decoding gates to provide outputs for 2*k* timing signals.

Johnson Counter

- A *switch-tail ring counter* is a circular shift register with the complemented output of the last flip-flop connected to the input of the first flip-flop.
- In Figure (a), the circular connection is made from the complemented output of the rightmost flip-flop to the input of the leftmost flip-flop.
- The register shifts its contents once to the right with every clock pulse, and at the same time, the complemented value of the E flip-flop is transferred into the A flip-flop.
- Starting from a cleared state, the switch-tail ring counter goes through a sequence of eight states, as listed in Figure (b).



(a) Four-stage switch-tail ring counter

Sequence number	Flip-flop outputs A	B	C	E	AND gate required for output
1	0	0	0	0	$A'E'$
2	1	0	0	0	AB'
3	1	1	0	0	BC'
4	1	1	1	0	CE'
5	1	1	1	1	AE
6	0	1	1	1	$A'B$
7	0	0	1	1	$B'C$
8	0	0	0	1	$C'E$

(b) Count sequence and required decoding

Johnson Counter

- The decoding gates are not shown in Fig. (a) but are specified in the last column of table.
- The eight AND gates listed in the table, when connected to the circuit, will complete the construction of the Johnson counter.
- Each gate is enabled during one particular state sequence, the outputs of the gates generate eight timing signals in succession.
- The decoding of a *k*-bit switch-tail ring counter to obtain 2*k* timing signals follows a regular pattern.
- The all-0's state is decoded by taking the complement of the two extreme flip-flop outputs.
- The all-1's state is decoded by taking the normal outputs of the two extreme flip-flops.
- All other states are decoded from an adjacent 1, 0, or 0, 1 pattern in the sequence.
- For example, sequence 7 has an adjacent 0, 1 pattern in flip-flops B and C.

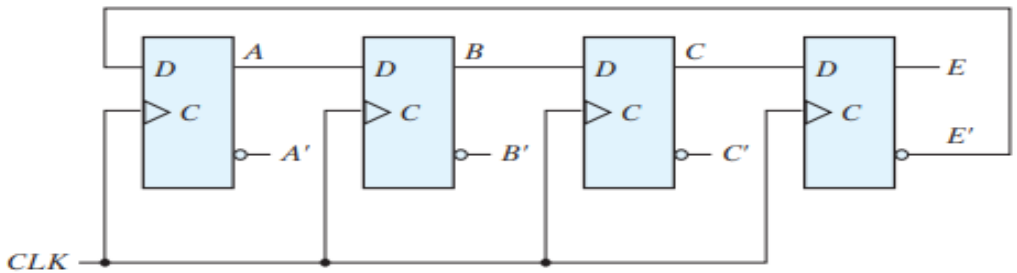
Johnson Counter

- **Disadvantage** of the circuit in Figure (a) is that if it finds itself in an unused state, it will persist in moving from one invalid state to another and never find its way to a valid state.
- This can be corrected by modifying the circuit to avoid this undesirable condition.
- One solution is to disconnect the output from flip-flop B that goes to the D input of flip-flop C and instead enable the input of flip-flop C by the function

$$D_C = (A + C)\cdot B$$

where D_C is the flip-flop input equation for the D input of flip-flop C.

- Johnson counters can be constructed for any number of timing sequences.
- The number of flip-flops needed is one-half the number of timing signals.
- The number of decoding gates is equal to the number of timing signals, and only two-input gates are needed.



(a) Four-stage switch-tail ring counter

Counters-Examples

- Chip Study:
  - IC-74LS90: A 4-bit ripple-type decade counter.
  - IC-74192: A 4-bit synchronous, presettable decade counter.
  - IC-74LS93: A 4-bit ripple binary counter.
  - IC-74193: A 4-bit synchronous, presettable binary counter.

Registers, Counters, and Memory

- Registers, Registers with Parallel load
- Shift Registers, Serial Addition, Universal Shift Register
- Binary Ripple Counters, BCD Ripple Counters
- Synchronous Counters: Binary Counters, BCD Counters
- Other Counters: Ring and Johnson Counter

- **Random Access Memory**
  - Memory Decoding
  - Read Only Memory
  - Semiconductor Main Memory Organization

Memory

- A memory unit is a device to which binary information is transferred for storage and from which information is retrieved when needed for processing.
- A memory unit is a collection of cells capable of storing a large quantity of binary information.
- A memory unit is a collection of storage cells, together with associated circuits needed to transfer information into and out of a device.
- The architecture of memory is such that information can be selectively retrieved from any of its internal locations.
- Two types of memories are used in digital systems:
  - Random-access memory (RAM)
  - Read-only memory (ROM)

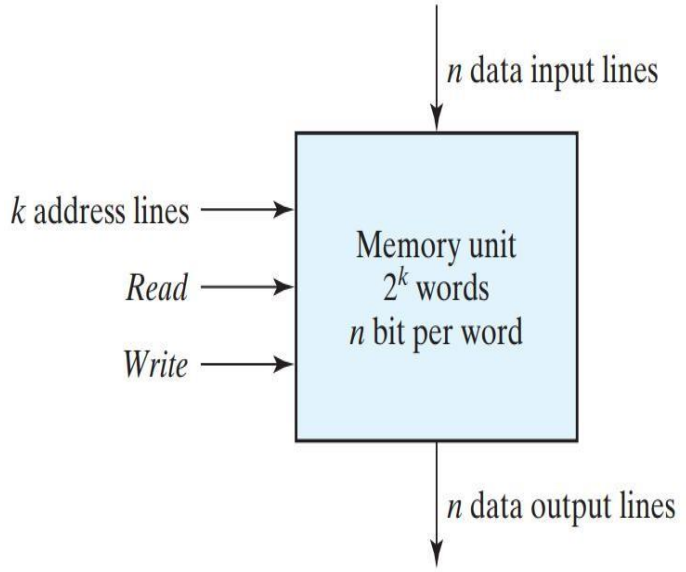
RAM and ROM

- **RAM** stores new information for later use.
- 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.
- RAM can perform both write and read operations.
- **ROM** is a programmable logic device (PLD).
- ROM can perform only the read operation, that information cannot be altered by writing.
- Other such units are the programmable logic array (PLA), programmable array logic (PAL), and the field-programmable gate array (FPGA).
- A PLD is an integrated circuit with internal logic gates connected through electronic paths that behave similarly to fuses.

(a) Conventional symbol (b) Array logic symbol

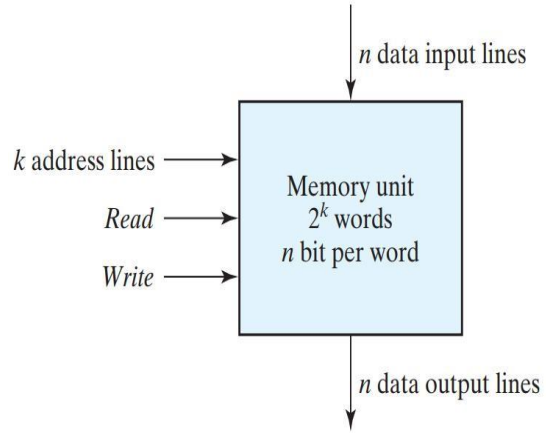
Random-Access Memory

- The time it takes to transfer information to or from any desired random location is always the same-hence the name random-access memory (RAM).
- The time required to retrieve information that is stored on magnetic tape depends on the location of the data.
- A memory unit stores binary information in groups of bits called words.
- A word in memory is an entity of bits that move in and out of storage as a unit.
- Communication between memory and its environment is achieved through data input and output lines, address selection lines, and control lines that specify the direction of transfer.
- A block diagram of a memory unit is shown in Figure.



Random-Access Memory

- The *n* data input lines provide the information to be stored in memory, and the *n* data output lines supply the information coming out of memory.
- The *k* address lines specify the particular word chosen among the many available.
- The two control inputs specify the direction of transfer desired:
  - Write input causes binary data to be transferred into the memory.
  - Read input causes binary data to be transferred out of memory.
- The memory unit is specified by the number of words it contains and the number of bits in each word.
- The address lines select one particular word.
- Each word in memory is assigned an identification number, called an address, starting from 0 up to $2^k - 1$, where k is the number of address lines.



Random-Access Memory

- The 1K x 16 memory of Figure has 10 bits in the address and 16 bits in each word.
- A 64K x 10 memory will have 16 bits in the address and each word will consist of 10 bits.
- The number of address bits needed in memory is dependent on the total number of words that can be stored in the memory and is independent of the number of bits in each word.
- The number of bits in the address is determined from the relationship $2^k \geq m$,
  where, m is the total number of words
  k is the number of address bits needed

Memory address		Memory content
Binary	Decimal	
0000000000	0	1011010101011101
0000000001	1	1010101110001001
0000000010	2	0000110101000110
⋮	⋮	⋮
1111111101	1021	1001110100010100
1111111110	1022	0000110100011110
1111111111	1023	1101111000100101

Contents of a 1024 X 16 memory

- 1K * 16 memory =Address(10 bits)+Word(16 bits)
- 64K * 10 =Address(16 bits)+Word(10 bits)

Write and Read Operations

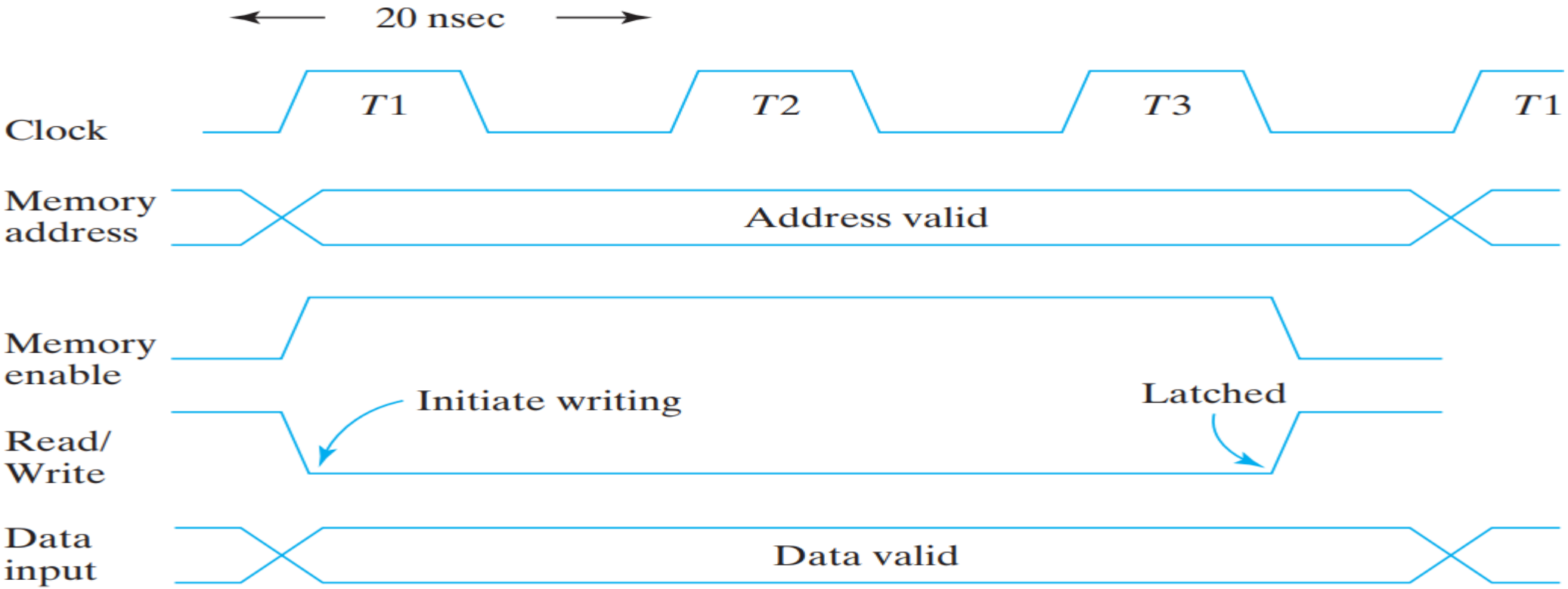
- The steps that must be taken to transfer a new word to be stored into memory are as follows: WRITE
  1. Apply the binary address of the desired word to the address lines.
  2. Apply the data bits that must be stored in memory to the data input lines.
  3. Activate the write input.
- The steps that must be taken to transfer a stored word out of memory are as follows: READ
  1. Apply the binary address of the desired word to the address lines.
  2. Activate the read input.

Control Inputs to Memory Chip

Memory Enable	Read/Write	Memory Operation
0	X	None
1	0	Write to selected word
1	1	Read from selected word

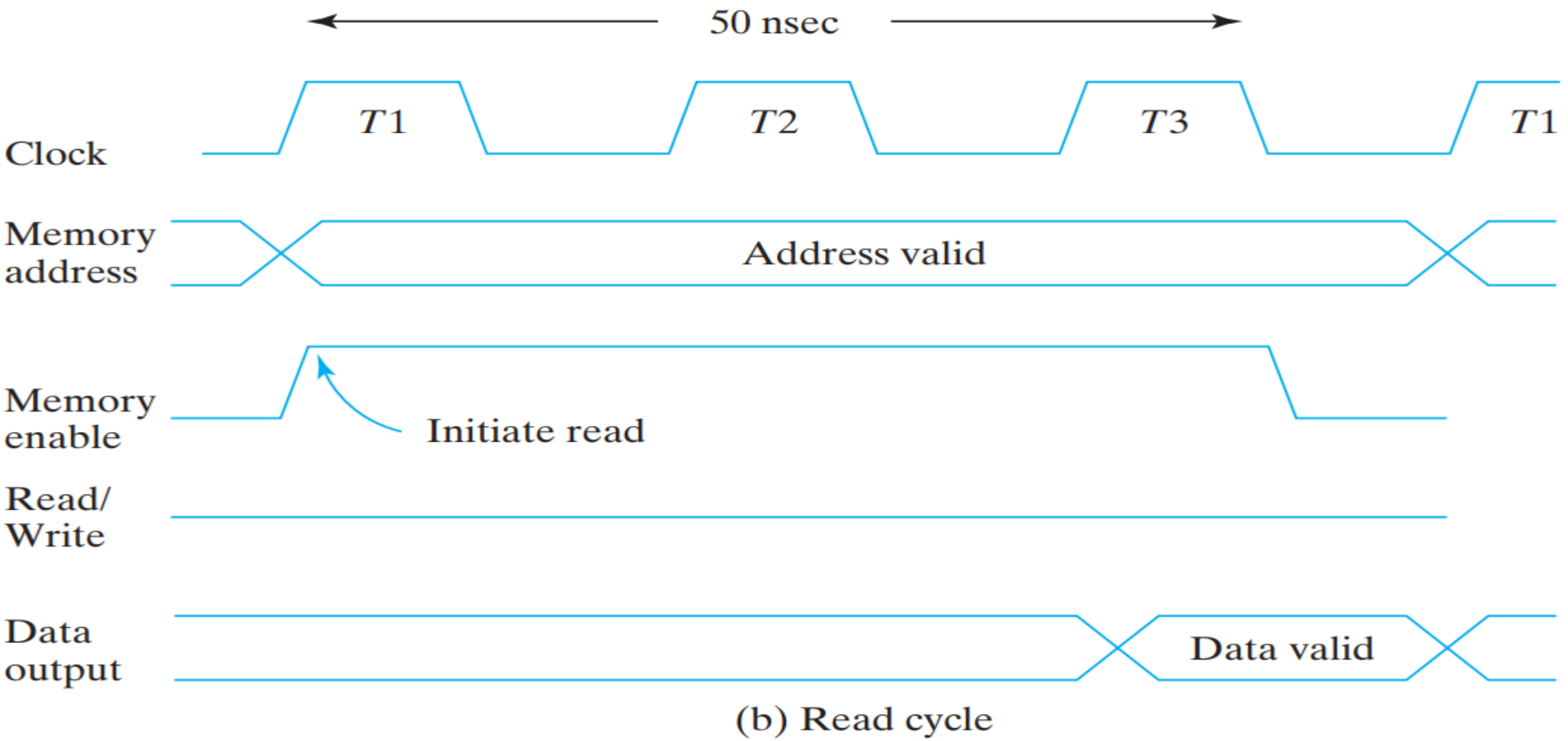
Timing Waveforms-WRITE

- The operation of the memory unit is controlled by an external device such as a CPU.
- The *access time* of memory is the time required to select a word and read it.
- The *cycle time* of memory is the time required to complete a write operation.



(a) Write cycle

Timing Waveforms-READ



(b) Read cycle

Types of Memories

- Sequential Access Vs Random Access
- Static RAM Vs Dynamic RAM
- Volatile Vs Non-Volatile

Sequential Access Memory Vs Random Access Memory

- In a random-access memory, the word locations may be thought of as being separated in space, each word occupying one particular location.
- The *access time* is always the same regardless of the particular location of the word.
- In a sequential-access memory, the information stored in some medium is not immediately accessible but is available only at certain intervals of time.
- A magnetic disk or tape unit.
- The time it takes to access a word depends on the position of the word with respect to the position of the read head; therefore, the *access time* is variable.
- Each memory location passes the read and write heads in turn, but information is read out only when the requested word has been reached.

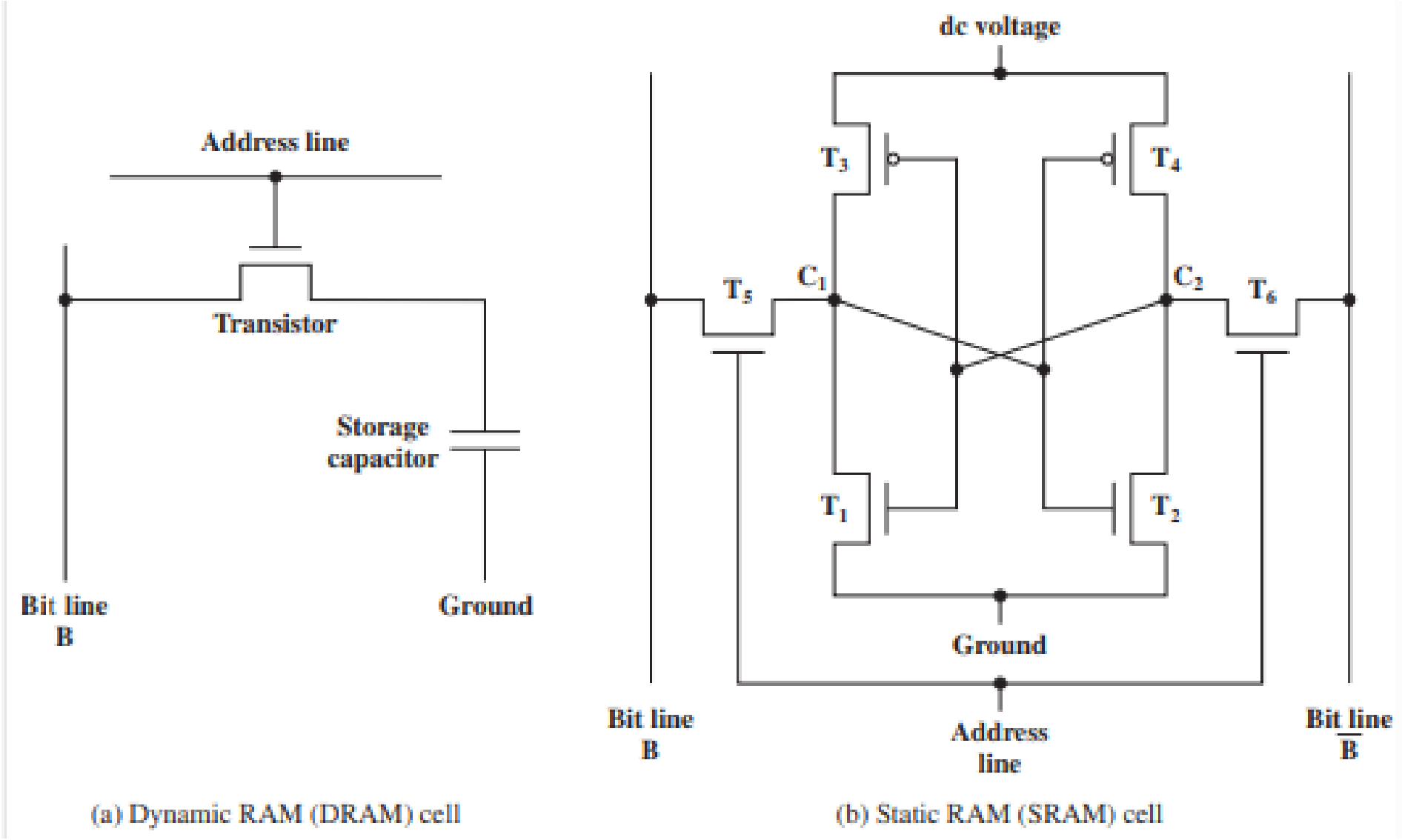
RAM (Static Vs Dynamic)

- Static RAM (SRAM) consists of internal latches that store binary information.
- The stored information remains valid as long as power is applied to the unit.
- SRAM is easier to use and has shorter read and write cycles.
- Dynamic RAM (DRAM) stores the binary information in the form of electric charges on capacitors provided inside the chip by MOS transistors.
- The stored charge on the capacitors tends to discharge with time, and the capacitors must be periodically recharged by *refreshing* the dynamic memory.
- Refreshing is done by cycling through the words every few milliseconds to restore the decaying charge.
- DRAM offers reduced power consumption and larger storage capacity in a single memory chip.

SRAM Vs DRAM

- The SRAM memory cell typically contains six transistors.
- To build memories with higher density, it is necessary to reduce the number of transistors in a cell.
- The DRAM cell contains a single MOS transistor and a capacitor.
- The charge stored on the capacitor discharges with time and the memory cells must be periodically recharged by refreshing the memory.
- DRAMs typically have four times the density of SRAMs.
- The cost per bit of DRAM storage is three to four times less than that of SRAM storage.
- A further cost savings is realized because of the lower power requirement of DRAM cells.
- DRAM chips are available in capacities from 64K to 256M bits.
- Most DRAMs have a 1-bit word size, so several chips have to be combined to produce a larger word size.
- Because of their large capacity, the address decoding of DRAMs is arranged in a two-dimensional array, and larger memories often have multiple arrays.

SRAM Vs DRAM



(a) Dynamic RAM (DRAM) cell

(b) Static RAM (SRAM) cell

SRAM Vs DRAM

DRAM

- They use capacitors to **store data**.
- They are cheap to manufacture.
- They are usually used as the **main memory**.
- They consume more power than **SRAMs**.
- They are slow compared to **SRAMs**.
- They need to refresh in order to **retain the data stored**.
- They have a **short data lifetime**.

SRAM

- They use transistors to **store data**.
- They are expensive to manufacture.
- They are usually used as **cache memory.**
- They consume less power than DRAMs.
- They are significantly **faster than DRAMs**.
- They occupy more space than DRAMs.

Volatile Vs Nonvolatile

- Memory units that lose stored information when power is turned off are said to be ***volatile***.
- CMOS integrated circuit RAMs, both static and dynamic, are volatile since the binary cells need external power to maintain the stored information.
- In contrast, a ***nonvolatile*** memory, such as a magnetic disk, retains its stored information after the removal of power.
- Nonvolatile memory retains information because the data stored on magnetic components are represented by direction of magnetization, which is retained after power is turned off.
- A nonvolatile memory enables digital computers to store programs that will be needed again after the computer is turned on.
- ROM is another nonvolatile memory.
- Programs and data that cannot be altered are stored in ROM, while other large programs are maintained on magnetic disks.

Registers, Counters, and Memory

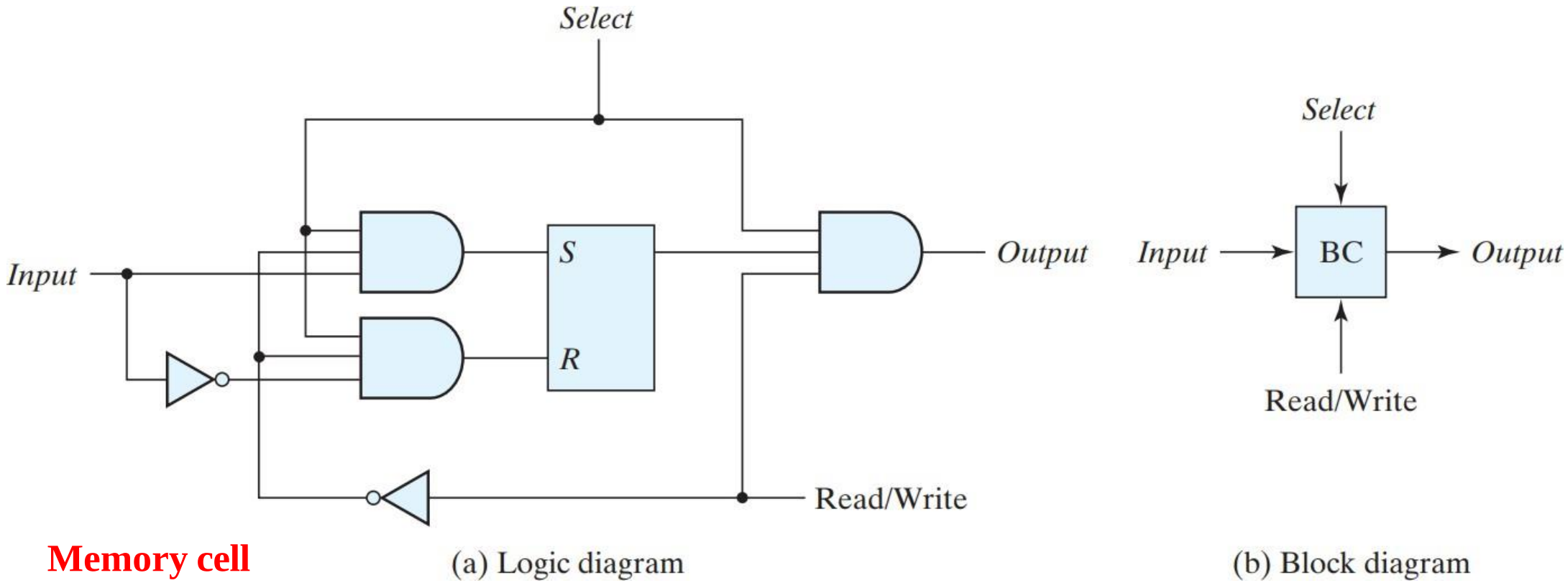
- Registers, Registers with Parallel load
- Shift Registers, Serial Addition, Universal Shift Register
- Binary Ripple Counters, BCD Ripple Counters
- Synchronous Counters: Binary Counters, BCD Counters
- Other Counters: Ring and Johnson Counter
- Random Access Memory

- **Memory Decoding**

- Read Only Memory
- Semiconductor Main Memory Organization

Memory Decoding-Internal Construction

- The internal construction of a RAM of *m* words and *n* bits per word consists of *m* x *n* binary storage cells and associated decoding circuits for selecting individual words.
- The equivalent logic of a binary cell that stores one bit of information is shown in Figure.
- The storage part of the cell is modeled by an SR latch with associated gates to form a D latch.



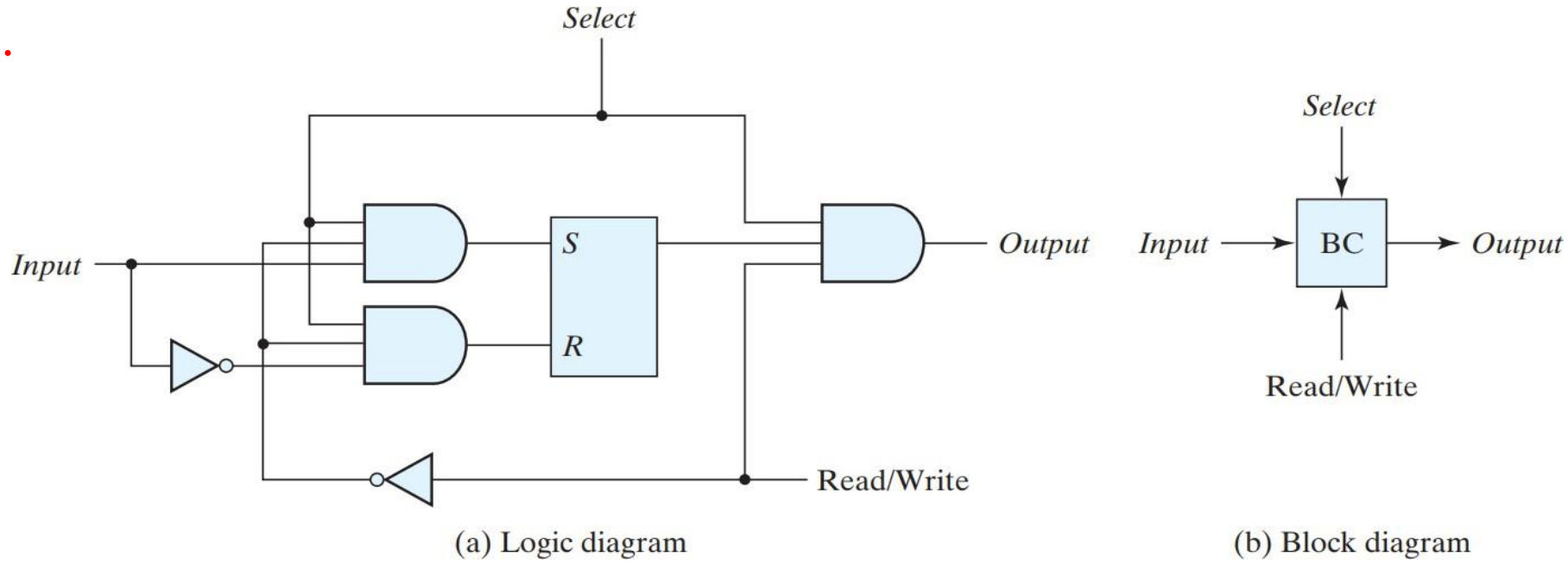
Memory cell

(a) Logic diagram

(b) Block diagram

Memory Decoding-Internal Construction

- The binary cell stores **one bit** in its internal latch and must be very small.
- The select input enables the cell to read or write, and the read/write input determines the operation of the cell when it is selected.
- If read/write input = 1 provides the *read* operation by forming a path from the latch to the output terminal.
- If read/write input = 0 provides the *write* operation by forming a path from the input terminal to the latch.
- Working…



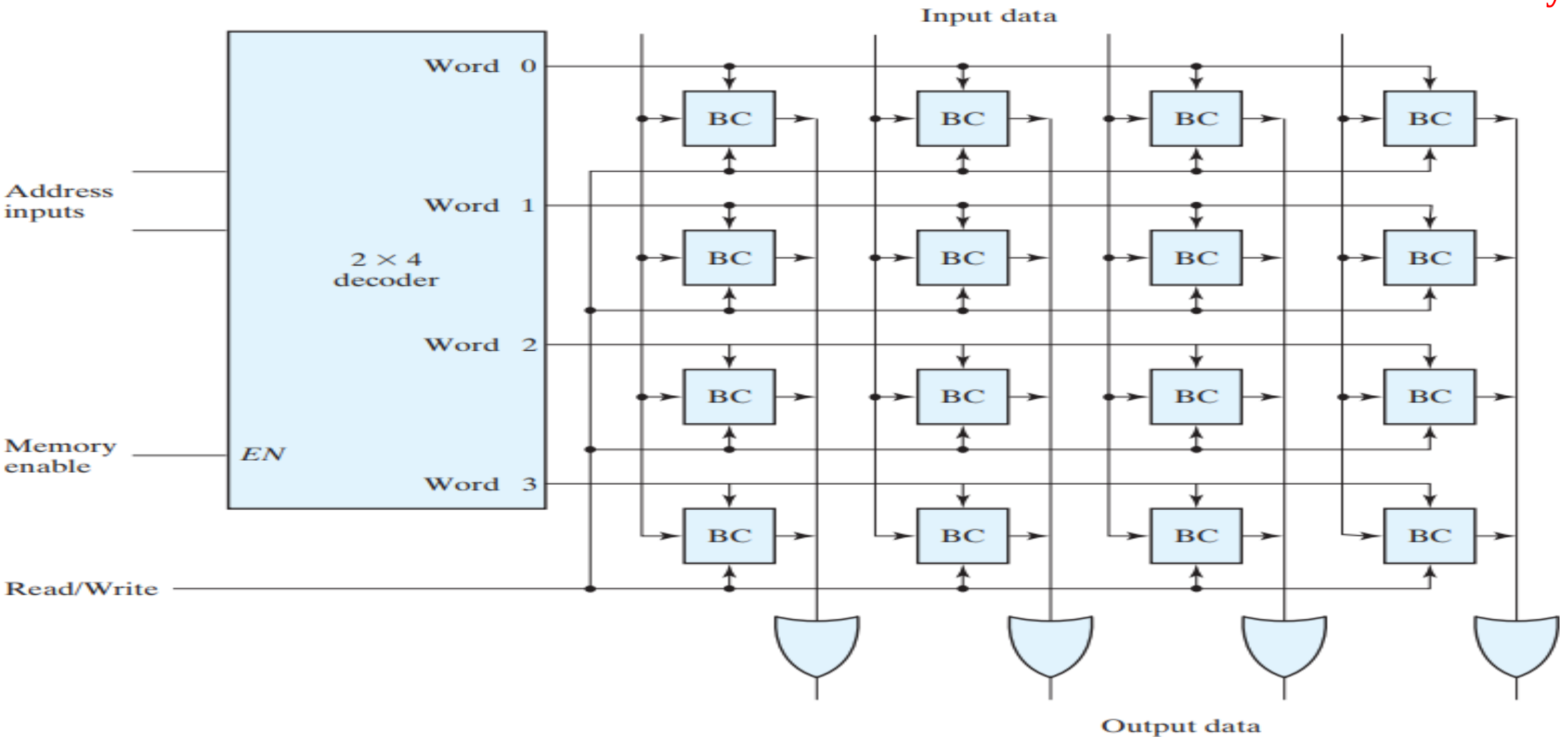
(a) Logic diagram

(b) Block diagram

Logic Construction of a 4 X 4 RAM

- RAM consists of four words of four bits each and has a total of 16 binary cells.

BC: Binary cell



Logic Construction of a 4 X 4 RAM

- A memory with four words needs two address lines.
- The two address inputs go through a 2 x 4 decoder to select one of the four words.
- The decoder is enabled with the memory-enable input.
- When the memory enable is 0, all outputs of the decoder are 0 and none of the memory words are selected.
- With the memory select at 1, one of the four words is selected, dictated by the value in the two address lines.
- Once a word has been selected, the read/write input determines the operation.
- During the read operation, the four bits of the selected word go through **OR** gates to the output terminals.
- During the write operation, the data available in the input lines are transferred into the four binary cells of the selected word.
- The binary cells that are not selected are disabled, and their previous binary values remain unchanged.

Logic Construction of a 4 X 4 RAM

- When the memory select input that goes into the decoder is equal to 0, none of the words are selected and the contents of all cells remain unchanged regardless of the value of the read/write input.
- Commercial RAMs may have a capacity of thousands of words, and each word may range from 1 to 64 bits.
- The logical construction of a large-capacity memory would be a direct extension of the configuration shown.
- A memory with 2^k words of n bits per word requires k address lines that go into a k x 2^k decoder.
- Each one of the decoder outputs selects one word of n bits for reading or writing.

Coincident Decoding

- A decoder with k inputs and 2^k outputs requires 2^k AND gates with k inputs per gate.
- The total number of gates and the number of inputs per gate can be reduced by employing two decoders in a two-dimensional selection scheme.
- The basic idea in two-dimensional (2D) decoding is to arrange the memory cells in an array that is as close as possible to a square.
- In this configuration, two k /2-input decoders are used instead of one k-input decoder.
- One decoder performs the row selection and the other the column selection in a two-dimensional matrix configuration.

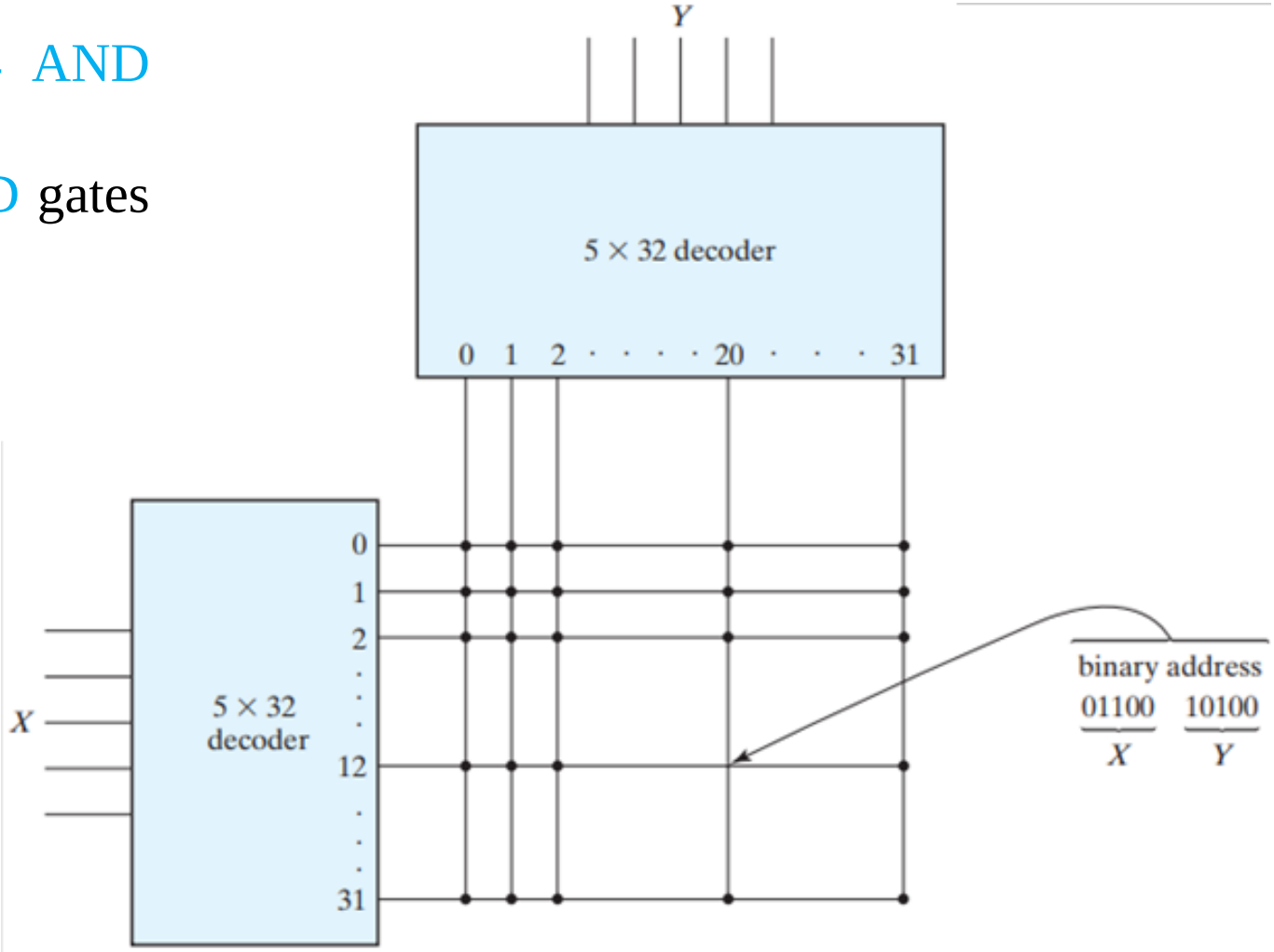
Two-dimensional Decoding Structure for a 1K-word memory

- The two-dimensional selection pattern is demonstrated in Figure for a 1K-word memory.
- Instead of using a single 10 x 1024 decoder, we use two 5 x 32 decoders.

- With the single decoder, we need 1024 AND gates with 10 inputs in each.
- In the two-decoder case, we need 64 AND gates with 5 inputs in each.

Example: Consider a word whose address is **404**.

- The 10-bit binary equivalent of 404 is 01100 10100.
- X = 01100 (12) and Y = 10100 (20).
- The n-bit word that is selected lies in the X decoder output number 12 and Y decoder output number 20.



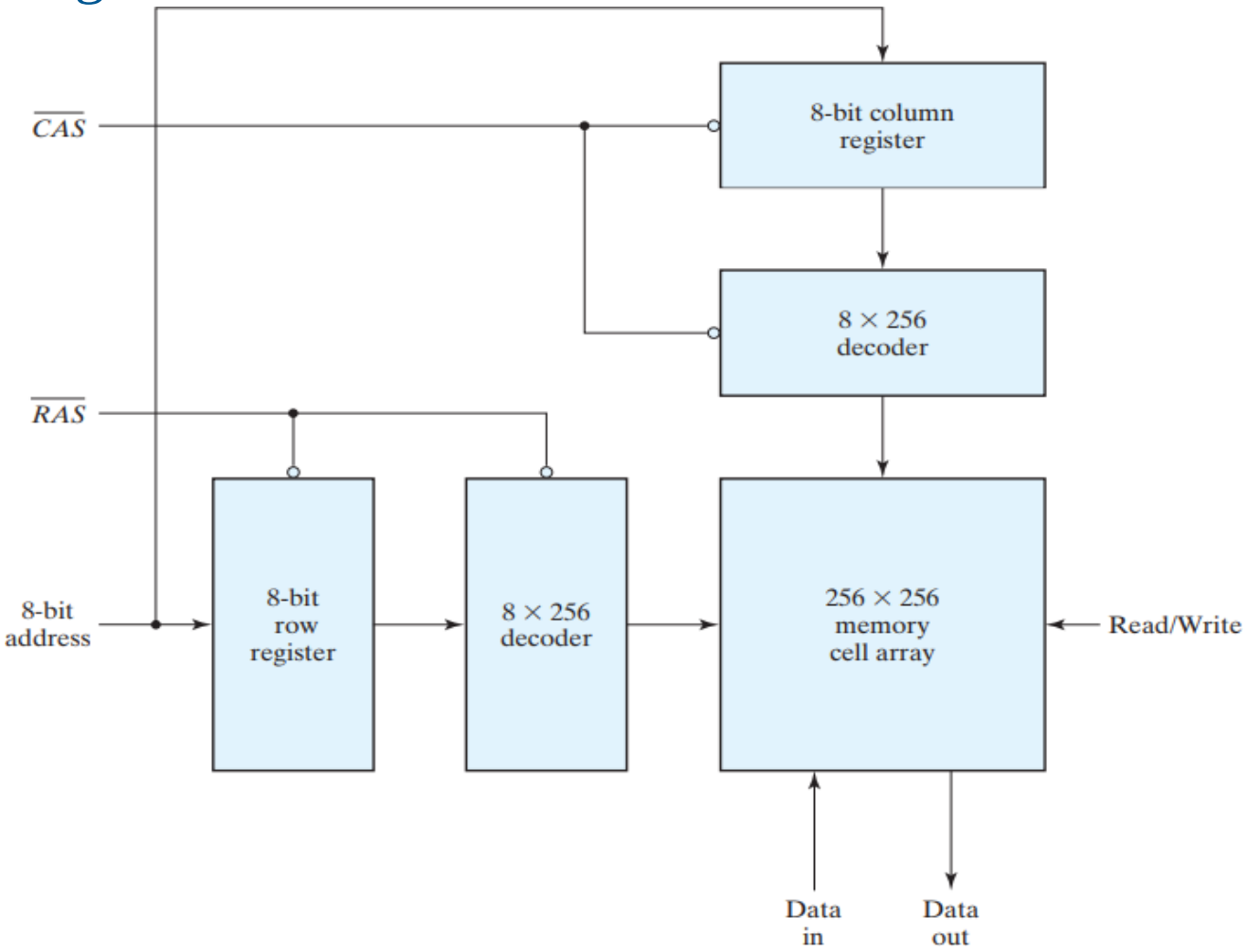
Address Multiplexing

- To reduce the number of pins in the IC package, designers utilize address multiplexing whereby one set of address input pins accommodates the address components.
- In a two-dimensional array, the address is applied in two parts at different times, with the row address first and the column address second.
- Since the same set of pins is used for both parts of the address, the size of the package is decreased significantly.
- We will use a 64K-word memory to illustrate the address-multiplexing idea.
- A diagram of the decoding configuration is shown in Figure next.
- The memory consists of a two-dimensional array of cells arranged into 256 rows by 256 columns, for a total of $2^8 \times 2^8 = 2^{16} = 64K$ words.
- There is a single data input line, a single data output line, and a read/write control, as well as an eight-bit address input and two address strobes, the latter included for enabling the row and column address into their respective registers.

Address Multiplexing

- The row address strobe (RAS) enables the eight-bit row register, and the column address strobe (CAS) enables the eight-bit column register.

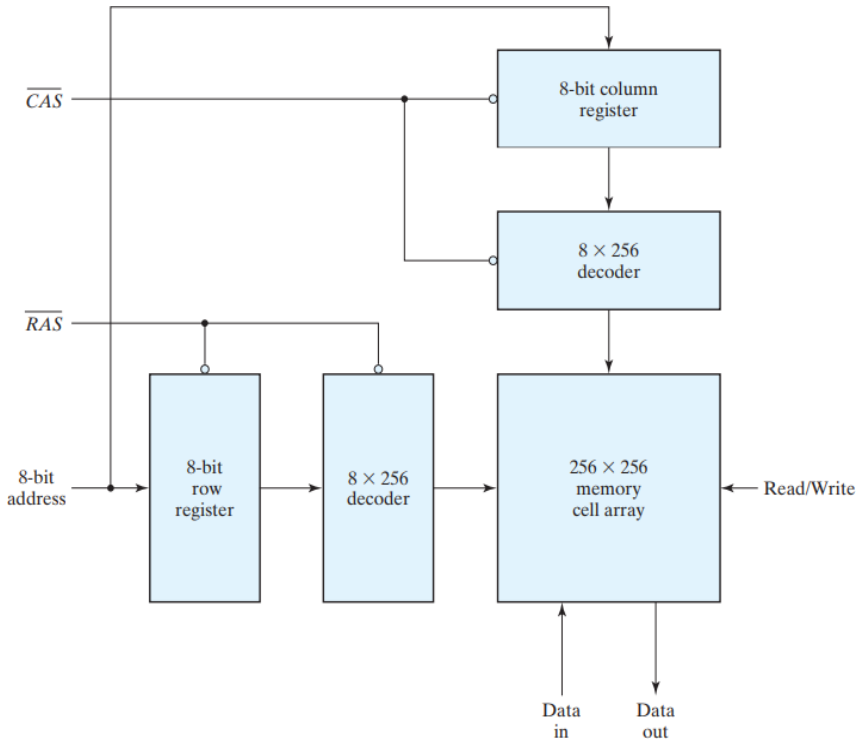
- The 16-bit address is applied to the DRAM in two steps using RAS and CAS.
- Initially, both strobes are in the 1 state.
- The 8-bit row address is applied to the address inputs and RAS is changed to 0.
  - This loads the row address into the row address register.



Address multiplexing for a 64K DRAM

Address Multiplexing

- RAS also enables the row decoder so that it can decode the row address and select one row of the array.
- After a time equivalent to the settling time of the row selection, RAS goes back to the 1 level.
- The 8-bit column address is then applied to the address inputs, and CAS is driven to the 0 state.
- This transfers the column address into the column register and enables the column decoder.
- The two parts of the address are in their respective registers, the decoders have decoded them to select the one cell corresponding to the row and column address, and a read or write operation can be performed on that cell.
- CAS must go back to the 1 level before initiating another memory operation.



Address multiplexing for a 64K DRAM

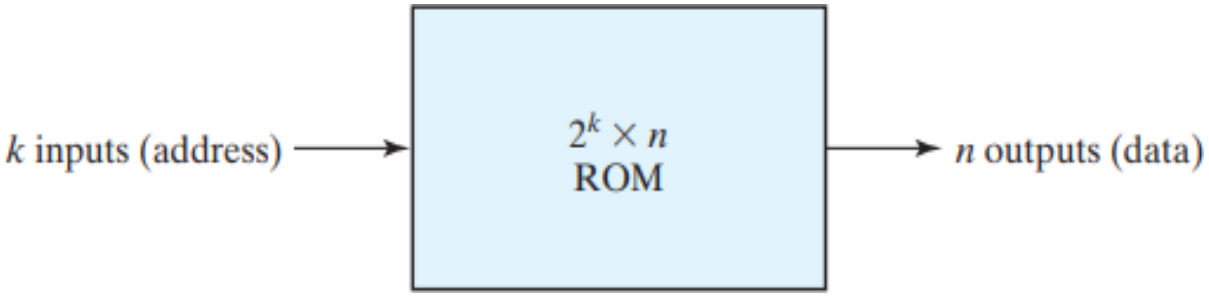
Registers, Counters, and Memory

- Registers, Registers with Parallel load
- Shift Registers, Serial Addition, Universal Shift Register
- Binary Ripple Counters, BCD Ripple Counters
- Synchronous Counters: Binary Counters, BCD Counters
- Other Counters: Ring and Johnson Counter
- Random Access Memory
- Memory Decoding

- **Read Only Memory**
- **Semiconductor Main Memory Organization**

Read-only Memory

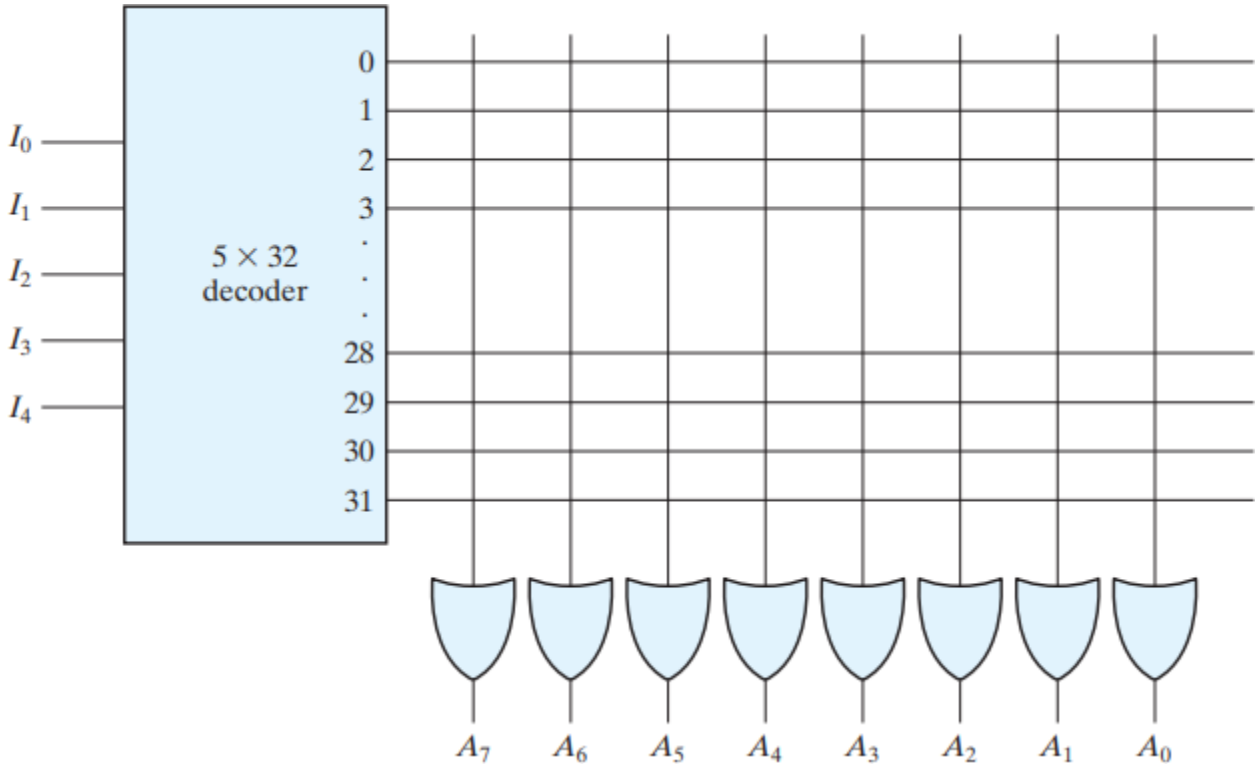
- A read-only memory (ROM) is a nonvolatile memory device in which permanent binary information is stored.
- The binary information must be specified by the designer and is then embedded in the unit to form the required interconnection pattern.
- Once the pattern is established, it stays within the unit even when power is turned off and on again.
- The number of words in a ROM is determined from the fact that *k* address input lines are needed to specify 2^k words.
- ROM does not have data inputs, because it does not have a write operation.



A block diagram of a ROM consisting of k inputs and n outputs

Internal Logic of a 32 X 8 ROM

- The unit consists of 32 words of 8 bits each.
- The five inputs are decoded into 32 distinct outputs using a 5 X 32 decoder.
- Each output of the decoder represents a memory address.
- The 32 outputs of the decoder are connected to each of the eight OR gates.
- Each OR gate must be considered as having 32 inputs.
- A 2^k x n ROM will have an internal k x 2^k decoder and n **OR** gates.
- Each **OR** gate has 2^k inputs, which are connected to each of the outputs of the decoder.
- The programmable intersection between two lines is sometimes called a crosspoint.

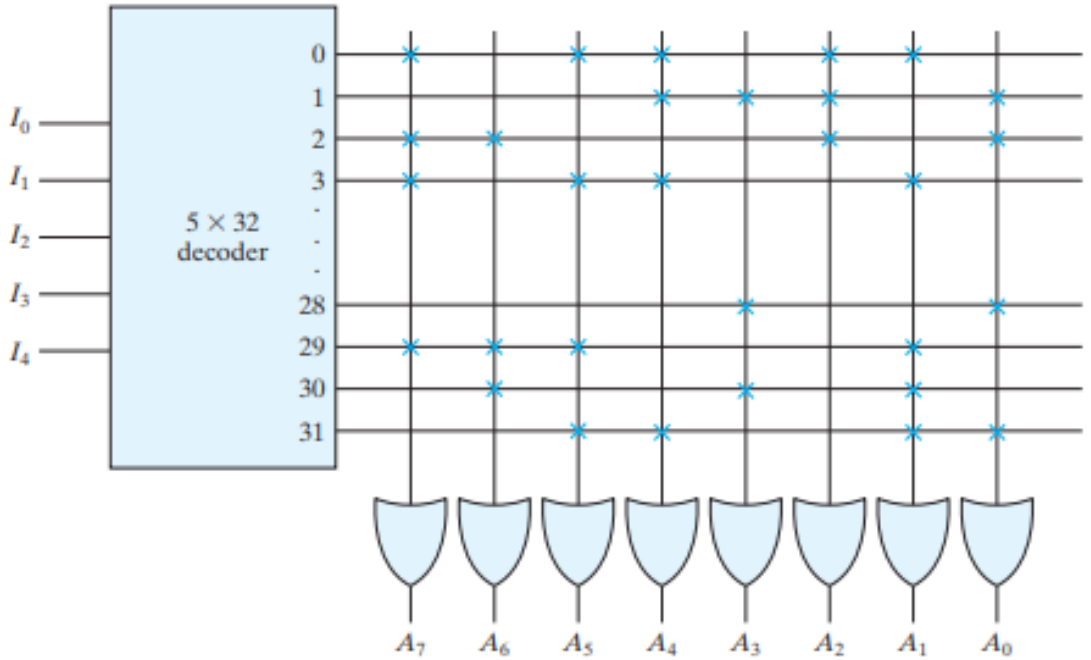


32 X 8 ROM: Programming the ROM

- The content of a 32 X 8 ROM may be specified with a truth table similar to Table.
- The truth table shows the five inputs under which are listed all 32 addresses.
- Each address stores a word of 8 bits, which is listed in the outputs columns.
- The complete table must include the list of all 32 words.
- Programming the ROM according to the truth table results in the configuration shown in Figure.
- Every 0 listed in the truth table specifies the absence of a connection, and every 1 listed specifies a path that is obtained by a connection.
- The table shows only the first four and the last four words in the ROM.

ROM Truth Table (Partial)

Inputs					Outputs							
I_4	I_3	I_2	I_1	I_0	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
0	0	0	0	0	1	0	1	1	0	1	1	0
0	0	0	0	1	0	0	0	1	1	1	0	1
0	0	0	1	0	1	1	0	0	0	1	0	1
0	0	0	1	1	1	0	1	1	0	0	1	0
		⋮							⋮			
1	1	1	0	0	0	0	0	0	1	0	0	1
1	1	1	0	1	1	1	1	0	0	0	1	0
1	1	1	1	0	0	1	0	0	1	0	1	0
1	1	1	1	1	0	0	1	1	0	0	1	1



Programming the ROM

Combinational Circuit Implementation

- Design a combinational circuit using a ROM. The circuit accepts a three-bit number and outputs a binary number equal to the square of the input number.
- **Steps:**
  1. Derive the truth table of the combinational circuit.
  2. Find ROM size.
  3. Specify the information needed for programming the ROM.

Combinational Circuit Implementation

Design a combinational circuit using a ROM. The circuit accepts a three-bit number and outputs a binary number equal to the square of the input number.

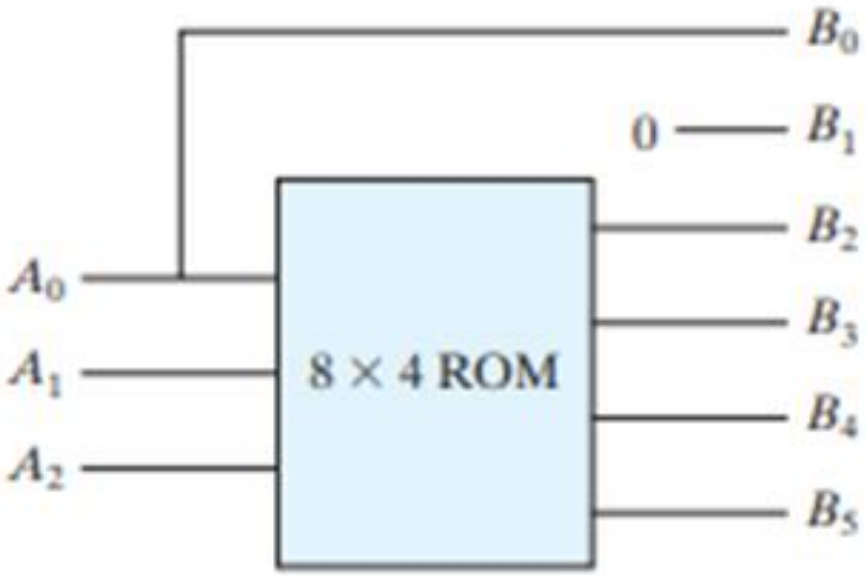
1. Truth table of the combinational circuit.

Inputs				Outputs					
A_2	A_1	A_0	Decimal	B_5	B_4	B_3	B_2	B_1	B_0
0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0	1
0	1	0	4	0	0	0	1	0	0
0	1	1	9	0	0	1	0	0	1
1	0	0	16	0	1	0	0	0	0
1	0	1	25	0	1	1	0	0	1
1	1	0	36	1	0	0	1	0	0
1	1	1	49	1	1	0	0	0	1

Combinational Circuit Implementation

Design a combinational circuit using a ROM. The circuit accepts a three-bit number and outputs a binary number equal to the square of the input number.

2. ROM size (three inputs specify eight words) is 8 x 4.
3. The information needed for programming the ROM is listed in ROM truth table.



(a) Block diagram

A_2	A_1	A_0	B_5	B_4	B_3	B_2
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	1
0	1	1	0	0	1	0
1	0	0	0	1	0	0
1	0	1	0	1	1	0
1	1	0	1	0	0	1
1	1	1	1	1	0	0

(b) ROM truth table

Types of Read-only Memory

- The required paths in a ROM may be programmed in four different ways:
  1. Mask programming ROM
  2. Programmable read-only memory (PROM)
  3. Erasable PROM (EPROM)
  4. Electrically erasable PROM (EEPROM or E^2PROM)

Memory Type	Category	Erasure	Write Mechanism	Volatility
Random-access memory (RAM)	Read-write memory	Electrically, byte-level	Electrically	Volatile
Read-only memory (ROM)	Read-only memory	Not possible	Masks	Nonvolatile
Programmable ROM (PROM)			Electrically	
Erasable PROM (EPROM)	Read-mostly memory	UV light, chip-level		
Electrically Erasable PROM (EEPROM)		Electrically, byte-level		
Flash memory		Electrically, block-level		

Types of Read-only Memory

1. Mask programming ROM

- Done by the semiconductor company during the fabrication process of the unit.
- This procedure is costly because the vendor charges the customer a special fee for custom masking the particular ROM.
- Mask programming is economical only if a large quantity of the same ROM configuration is to be ordered.

Types of Read-only Memory

2. Programmable read-only memory, or PROM

- For small quantities, it is more economical to use.
- PROM units contain all the fuses intact, giving all 1's in the bits of the stored words.
- The fuses in the PROM are blown by the application of a high-voltage pulse to the device through a special pin.
- A blown fuse defines a binary 0 state and an intact fuse gives a binary 1 state.
- This procedure allows the user to program the PROM in the laboratory to achieve the desired relationship between input addresses and stored words.

Types of Read-only Memory

3. Erasable PROM, or EPROM

- The hardware procedure for programming ROMs or PROMs is irreversible, and once programmed, cannot be altered.
- EPROM, which can be restructured to the initial state even though it has been programmed previously.
- When the EPROM is placed under a special ultraviolet light for a given length of time, the shortwave radiation discharges the internal floating gates that serve as the programmed connections.
- After erasure, the EPROM returns to its initial state and can be reprogrammed to a new set of values ROM.

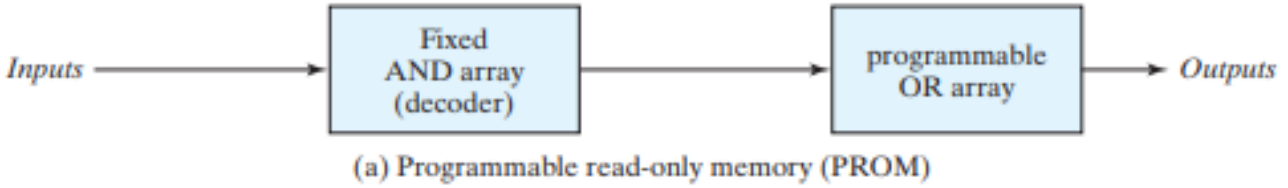
Types of Read-only Memory

4. Electrically Erasable PROM, or EEPROM

- The device is similar to EPROM, except that the previously programmed connections can be erased with an electrical signal instead of ultraviolet light.
- The advantage is that the device can be erased without removing it from its socket.
- Flash memory devices are similar to EEPROMs but have additional built-in circuitry to selectively program and erase the device in-circuit.
- Application in cell phones, digital cameras, set-top boxes, digital TV, telecommunications, nonvolatile data storage, and microcontrollers.
- Low power consumption makes them an attractive storage medium for laptop and notebook computers.
- Flash memories incorporate additional circuitry, allowing simultaneous erasing of blocks of memory, for example, of size 16 to 64 K bytes.
- Flash memories are subject to fatigue, typically having about 10^5 block-erase cycles.

Combinational PLDs

- The PROM is a combinational programmable logic device (PLD)-an integrated circuit with programmable gates divided into an AND array and an OR array to provide an AND–OR sum-of-product (SOP) implementation.
- Three are three major types of combinational PLDs, differing in the placement of the programmable connections in the AND-OR array:
  - PROM
  - PAL
  - PLA



(a) Programmable read-only memory (PROM)

(b) Programmable array logic (PAL)

(c) Programmable logic array (PLA)

Combinational PLDs

1. The **PROM** has a fixed AND array constructed as a decoder and a programmable OR array.
- The programmable OR gates implement the Boolean functions in sum-of-minterms form.
2. The **PAL** has a programmable AND array and a fixed OR array.
- The AND gates are programmed to provide the product terms for the Boolean functions, which are logically summed in each OR gate.
3. The flexible PLD is the **PLA**, in which both the AND and OR arrays can be programmed.
- The product terms in the AND array may be shared by any OR gate to provide the required SOPs implementation.

(a) Programmable read-only memory (PROM)

(b) Programmable array logic (PAL)

(c) Programmable logic array (PLA)

Semiconductor Main Memory Organization

- Main memory was often referred to as core.
- Today, the use of semiconductor chips for main memory is almost universal.
- The basic element of semiconductor memory is the memory cell.
- Although a variety of electronic technologies are used, all semiconductor memory cells share certain properties:
  - They exhibit two stable (or semistable) states, which can be used to represent binary 1 and 0.
  - They are capable of being written into (at least once), to set the state.
  - They are capable of being read to sense the state.

Semiconductor Main Memory Organization

The operation of a memory cell

- The cell has three functional terminals capable of carrying an electrical signal.
- The select terminal selects a memory cell for a read or write operation.
- The control terminal indicates read or write.
- For writing, the other terminal provides an electrical signal that sets the state of the cell to 1 or 0.
- For reading, that terminal is used for the output of the cell's state.

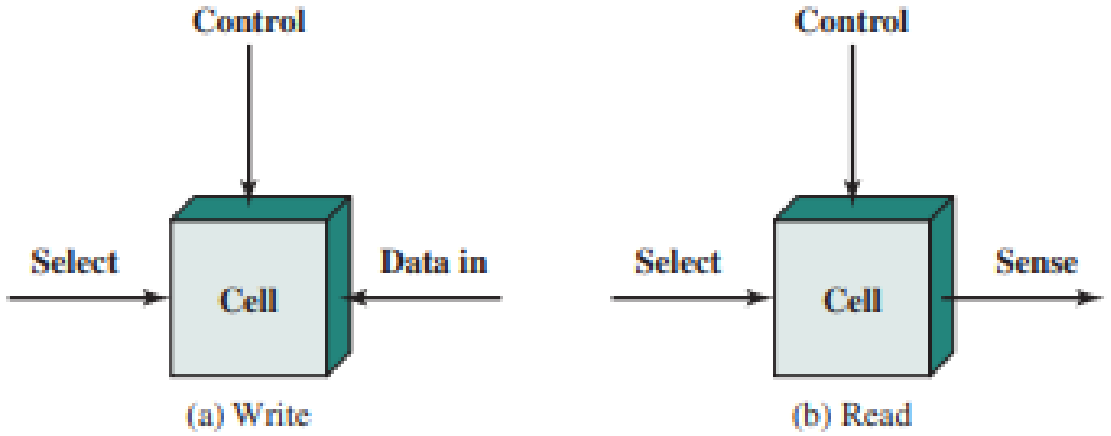


Figure Memory Cell Operation

Tutorials

Thank You