DLD & CO
Unit-3

Part-2

Registers and Counters

Syllabus

- Registers
- Shift Registers
- Counters

Introduction

- A clocked sequential circuit consists of a group of flip-flops and combinational gates.
- The flip-flops are essential because, in their absence, the circuit reduces to a purely combinational circuit.
- A circuit with flip-flops is considered a sequential circuit even in the absence of combinational gates.
- Circuits that include flip-flops are usually classified by the function they perform rather than by the name of the sequential circuit.
- Two such circuits are registers and counters.

Register

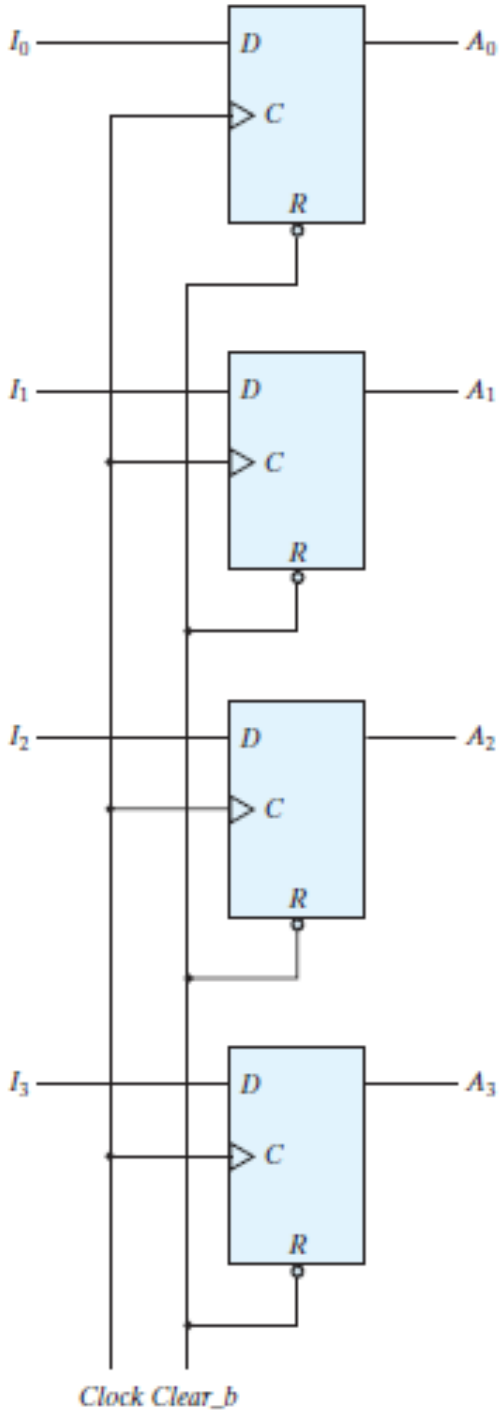
- A register is a group of flip-flops, each one of which shares a common clock and is capable of storing one bit of information.
- An n-bit register consists of a group of n flip-flops capable of storing n bits of binary information.
- In addition to the flip-flops, a register may have combinational gates that determine how the information is transferred into the register.

Counter

- Counters are a special type of registers.
- 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.

Registers

- The simplest register is one that consists of only flip-flops, without any gates.
- Figure shows register constructed with four D -type flip-flops to form a four-bit data storage register.
- 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.



Registers

- The input Clear_b goes to the active-low R (reset) input of all four flip-flops.
- When this input goes to 0, all flip-flops are reset asynchronously.
- The Clear_b input is useful for clearing the register to all 0's prior to its clocked operation.
- The R inputs must be maintained at logic 1 during normal clocked operation.
- Note:- Depending on the flip-flop, either Clear, Clear_b, reset, or reset_b can be used to indicate the transfer of the register to an all 0's state.

Register with Parallel Load

- The transfer of new information into a register is referred to as loading or updating the register.
- If all the bits of the register are loaded simultaneously with a common clock pulse, we say that the loading is done in parallel.
  - **Load control:** Determine when to load new information
- A clock edge applied to the C inputs of the register of above Fig.1 will load all four inputs in parallel.

Register with Parallel Load

Approaches to register with parallel load

1. Controlling the clock input signal with an enabling gate: uneven propagation delays between the master clock and the inputs of flip-flops
2. Controlling the D inputs: ensure that all clock pulses arrive at the same time anywhere in the system

- For synchronism, it is 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.
- 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 below Fig.

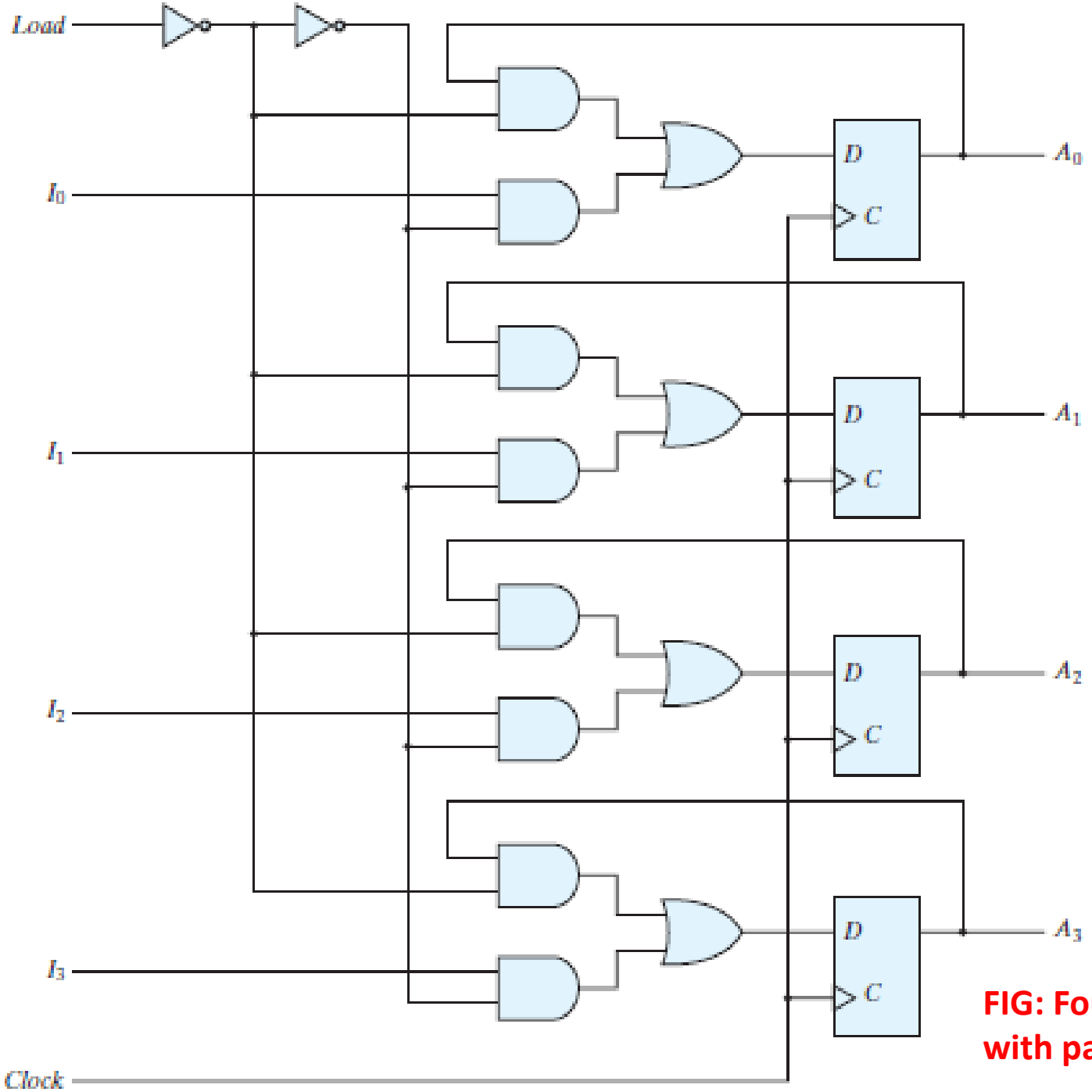


FIG: Four-bit register with parallel load

Register with Parallel Load

- 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.

Register with Parallel Load

- The feedback connection from output to input is necessary because a D flip-flop does not have a “no change” condition.
- The clock pulses are applied to the C inputs without interruption.

SHIFT REGISTERS

- A register capable of shifting its binary information in one or both direction is called a shift register.
- 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 simplest possible shift register is one that uses only flip-flops, as shown in Fig.

SHIFT REGISTERS

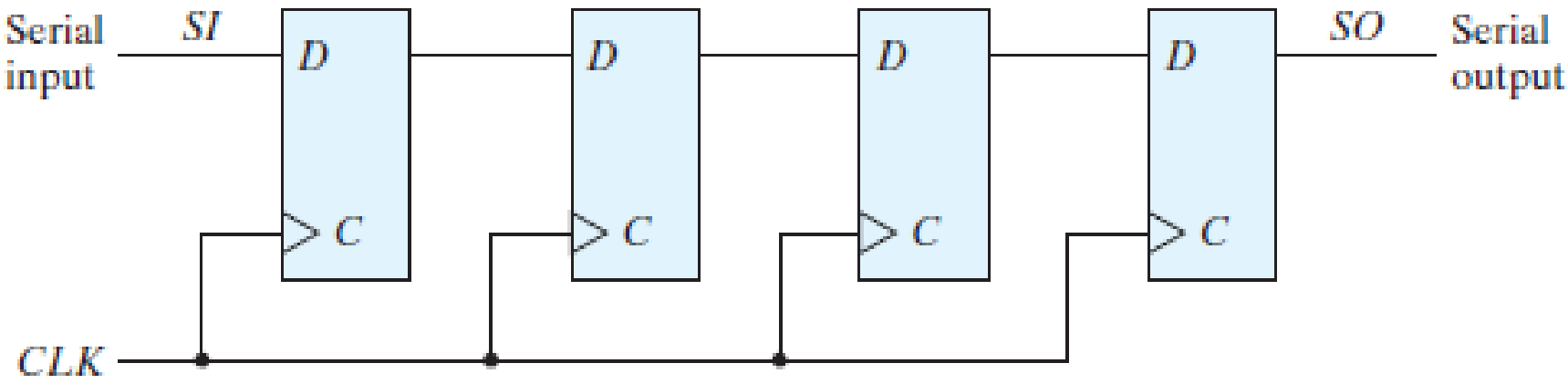


Fig: **Four-bit shift register**

- The output of a given flip-flop is connected to the D input of the flip-flop at its right.
- This shift register is unidirectional (left-to-right).

SHIFT REGISTERS

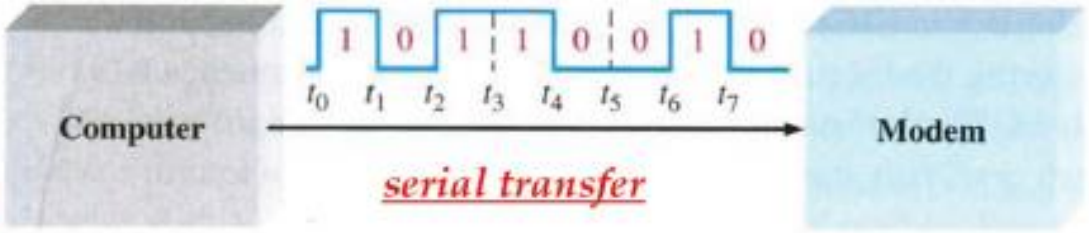
- Each clock pulse shifts the contents of the register one bit position to the right.
- The configuration does not support a left shift.
- The serial input determines what goes into the leftmost flip-flop during the shift.
- The serial output is taken from the output of the rightmost flip-flop.

- ***Shift control***: make the shift occur only with certain pulses
  - inhibiting the clock
  - control through the D inputs (shown later)

- Serial transfer vs. parallel transfer

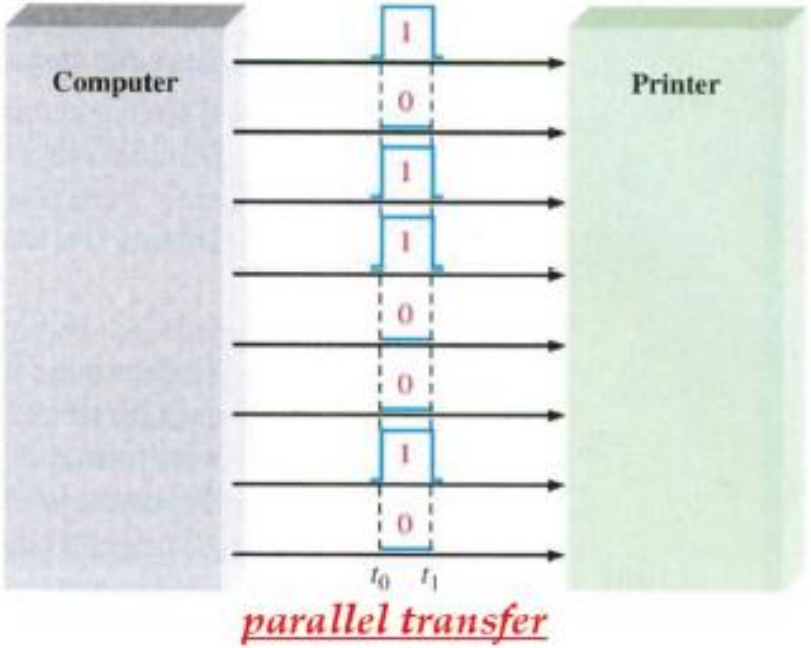
One bit a time

Need more time, low complexity



All bits at the same time

Transfer faster, higher complexity



Serial Transfer

- A digital system is said to operate in serial mode when information is transferred and manipulated one bit at a time.
- Information is transferred one bit at a time by shifting the bits out of the source register and into the destination register.
- In parallel transfer all the bits of the register are transferred at the same time.
- The serial transfer of information from register A to register B is done with shift registers, as shown in the below Fig.

Serial Transfer

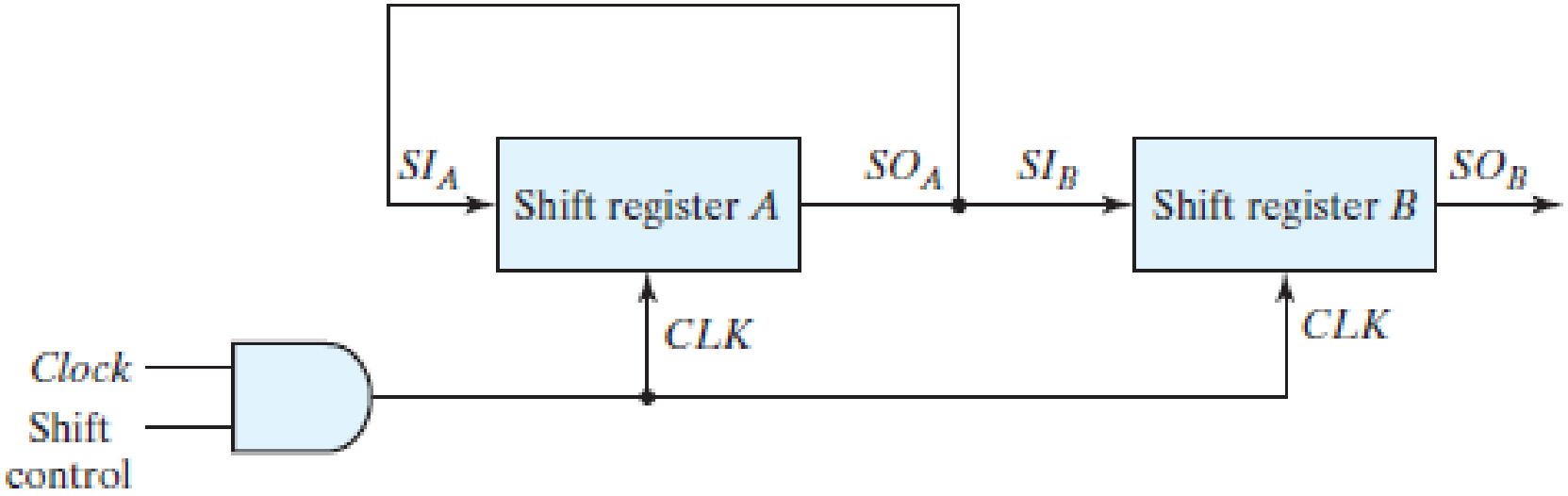


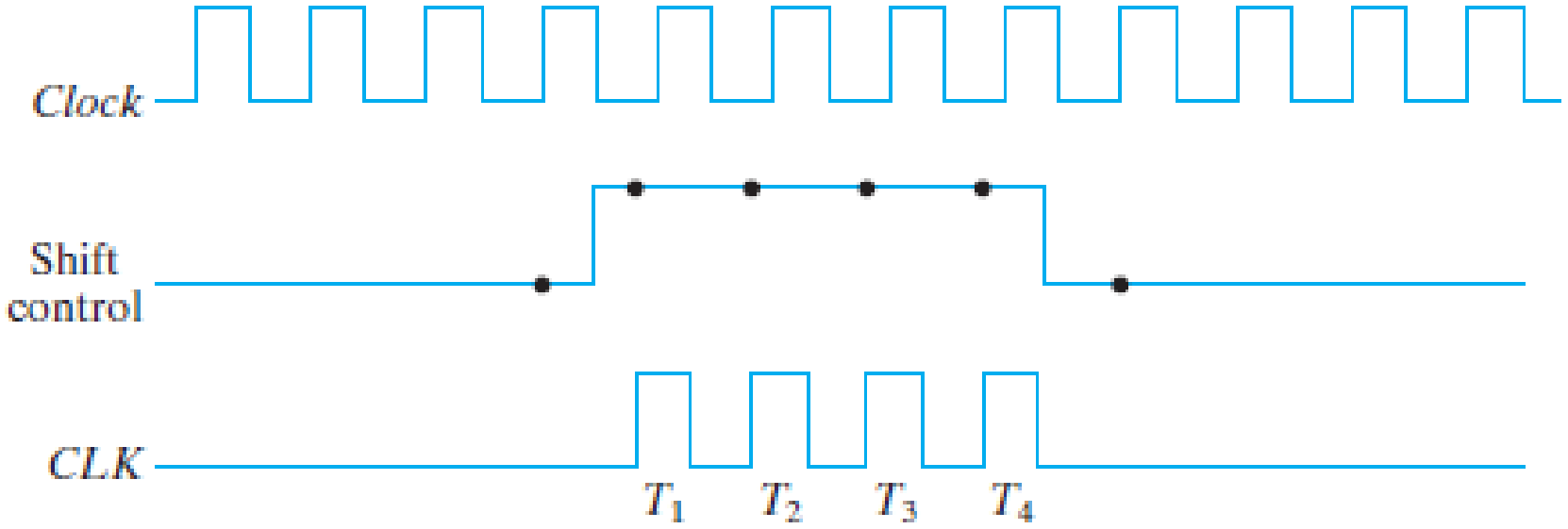
Fig: serial transfer of information from register A to register B is done with shift registers

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

Serial Transfer

- 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 initial content of register B is shifted out through its serial output and is lost unless it is transferred to a third shift register.
- The shift control input determines when and how many times the registers are shifted.
- This is done with an AND gate that allows clock pulses to pass into the CLK terminals only when the shift control is active.

Serial Transfer



(b) Timing diagram

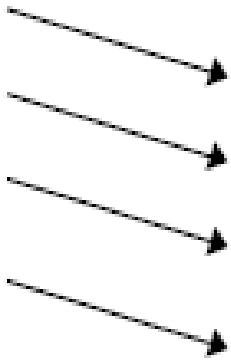
- The shift control signal is synchronized with the clock and changes value just after the negative edge of the clock.

Serial Transfer

- 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.
- Assume that the binary content of A before the shift is 1011 and that of B is 0010.
- The serial transfer from A to B occurs in four steps, as shown in below Table.

Serial Transfer Example

Timing Pulse	Shift Register A	Shift Register B
Initial value	1 0 1 1	0 0 1 0
After T1	1 1 0 1	1 0 0 1
After T2	1 1 1 0	1 1 0 0
After T3	0 1 1 1	0 1 1 0
After T4	1 0 1 1	1 0 1 1

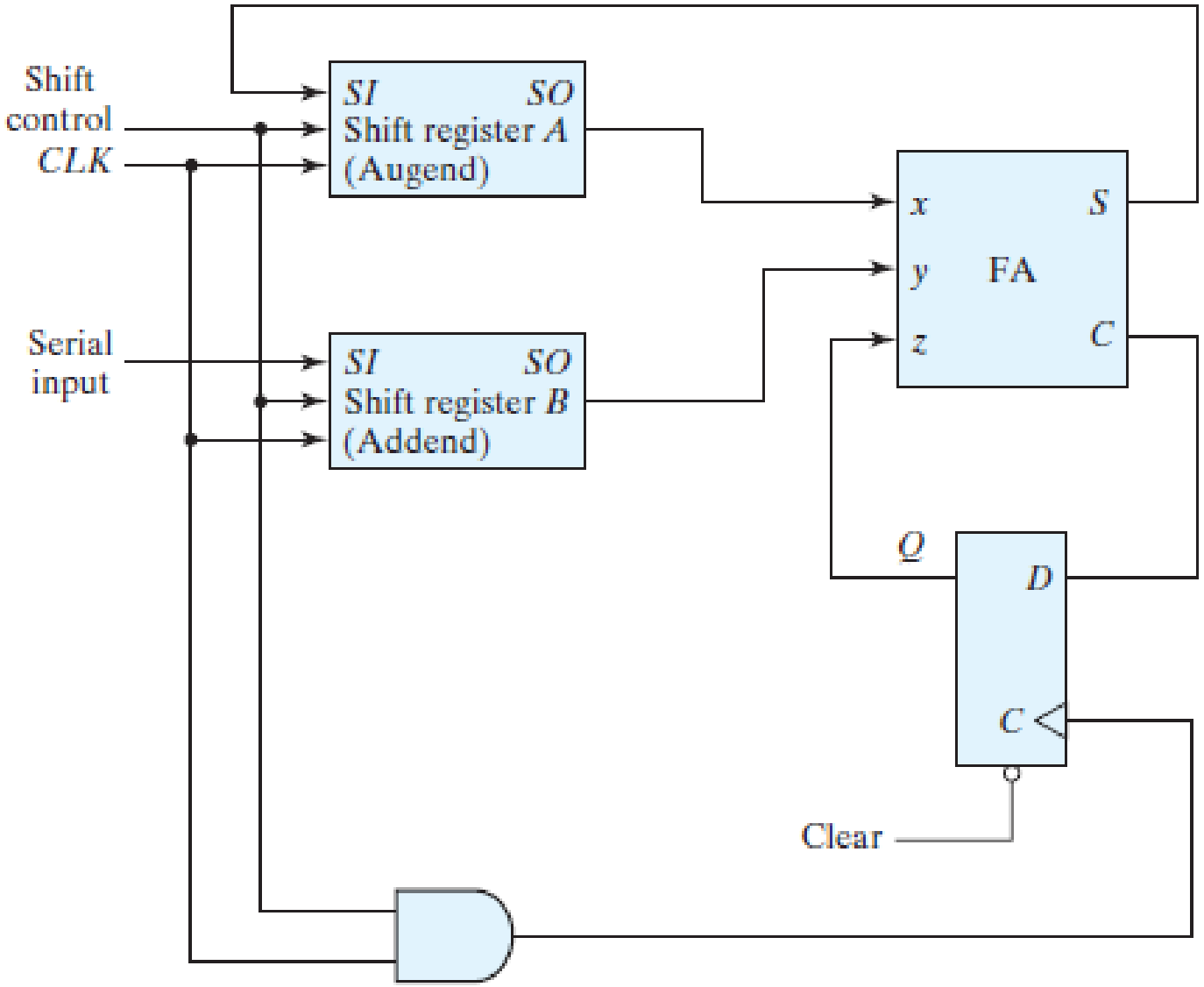


- In the parallel mode, information is available from all bits can be transferred simultaneously during one clock pulse.
- In the serial mode, the registers have single serial input and a single serial output.
- The information is transferred one bit at a time while the registers are shifted in the same direction.

Serial Addition

- Operations in digital computers are usually done in parallel because that is a 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.
- The bits of two binary numbers are added one pair at a time through a single full adder (FA) circuit as shown in below Fig.

Serial Addition



Serial Addition

- operation of the serial adder is as follows: Initially, register A holds the augend, register B holds the addend, and the carry flip-flop is cleared to 0.
- Shift control enables the triggering of clock and 1-bit addition of two operands from LSB to MSB.
- A new sum (S) bit is transferred to shift register A
- A carry-out (C) of the Full Adder is transferred to Q as the z input of the next addition.
- Finally, when the shift control is disabled, summation result is stored in shift register A.

Serial Addition

- Initially, register A and the carry flip-flop are cleared to 0, and then the first number is added from B.
- While B is shifted through the full adder, a second number is transferred to it through its serial input.
- The second number is then added to the contents of register A , while a third number is transferred serially into register B.
- This can be repeated to perform the addition of two, three, or more four-bit numbers and accumulate their sum in register A.

Serial v.s. Parallel

- Serial adders:
  - Use shift registers
  - A sequential circuit
  - Require only one FA and a carry flip-flop
  - Slower but require less equipment
- Parallel adders:
  - Use registers with parallel load for sum
  - Basically a pure combinational circuit
  - ***n*** FAs are required
  - Faster

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

- 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.
- The block diagram symbol and the circuit diagram of a four-bit universal shift register that has all the capabilities just listed are shown in below Fig.

Universal Shift Register

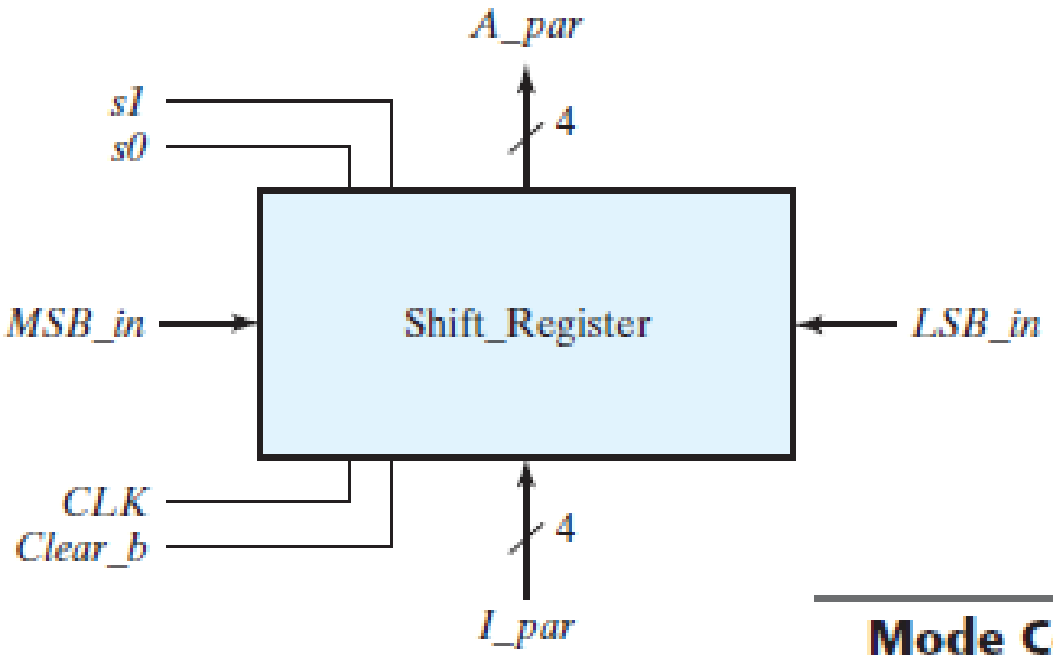
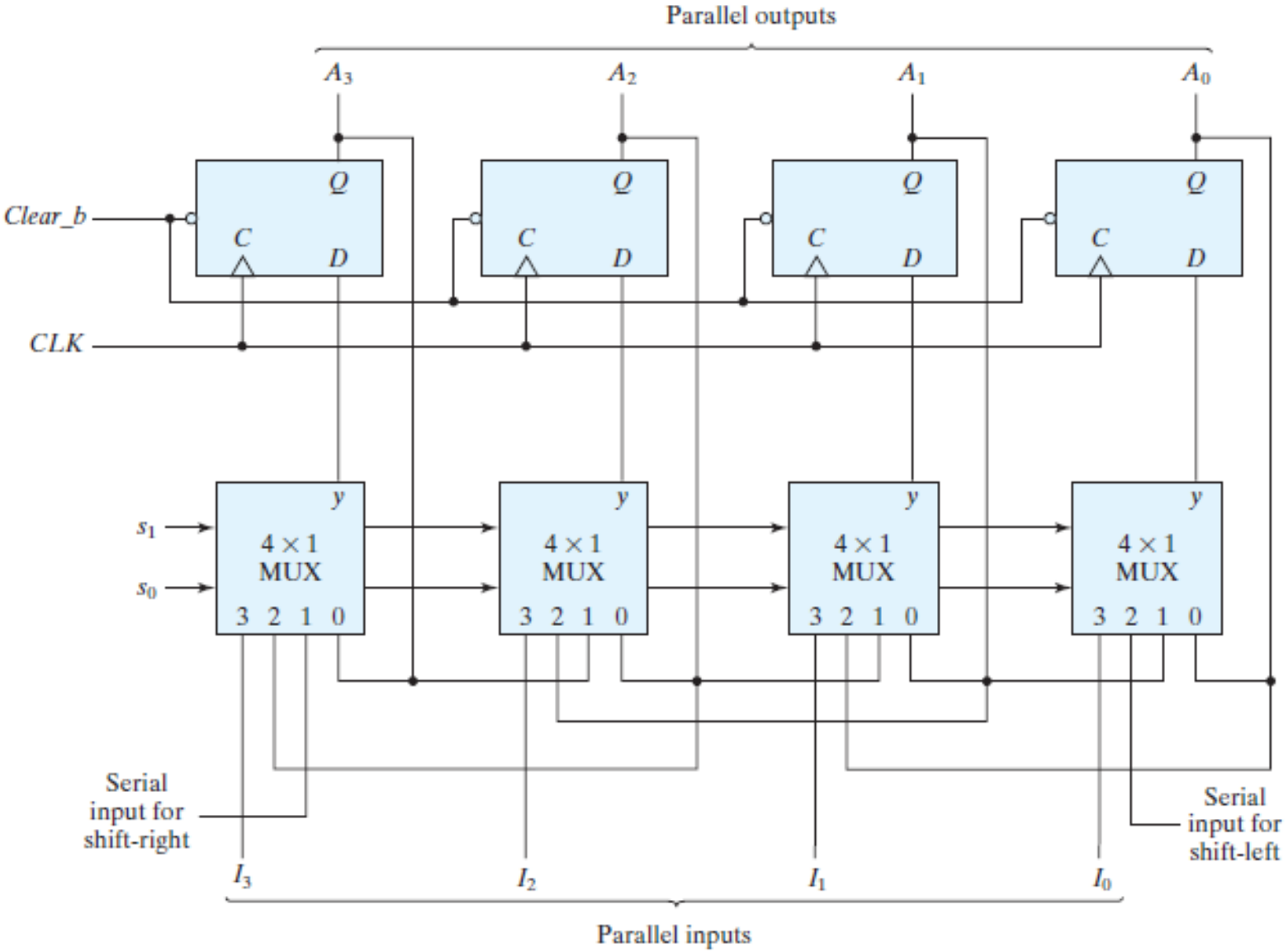


Fig: Graphic symbol

Functional Table

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

Universal Shift Register



Universal Shift Register

- The circuit consists of four D flip-flops and four multiplexers.
- The four multiplexers have two common selection inputs s_1 and s_0.
- Input 0 in each multiplexer is selected when s_1s_0=00, input 1 is selected when s_1s_0=01, and similarly for the other two inputs.

Universal Shift Register

- Shift registers are often used to interface digital systems situated remotely from each other.
- If the distance is far, it will be expensive to use n lines to transmit the n bits in parallel.
- It is more 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.
- The transmitter performs a parallel-to-serial conversion of data and the receiver does a serial-to-parallel conversion.

Ripple Counters

- A register that goes through a prescribed sequence of states (i.e counts upward or downward) upon the application of input pulses is called a counter.
- The input pulses may be clock pulses, or they may originate from some external source and may occur at a fixed interval of time or at random.
- The sequence of states may follow the binary number sequence or any other sequence of 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 through 2^n-1.

Ripple Counters

- 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. i.e., the C input of some or all flip-flops are 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 Counter

- 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. i.e., the output of each flip-flop is connected to the C input of the next flip-flop in sequence.
- The flip-flop holding the least significant bit receives the incoming count pulses.
- A complementing flip-flop can be obtained from
  - JK flip-flop with the J and K inputs tied together
  - T flip-flop
  - D flip-flop with the complement output connected to the D input.

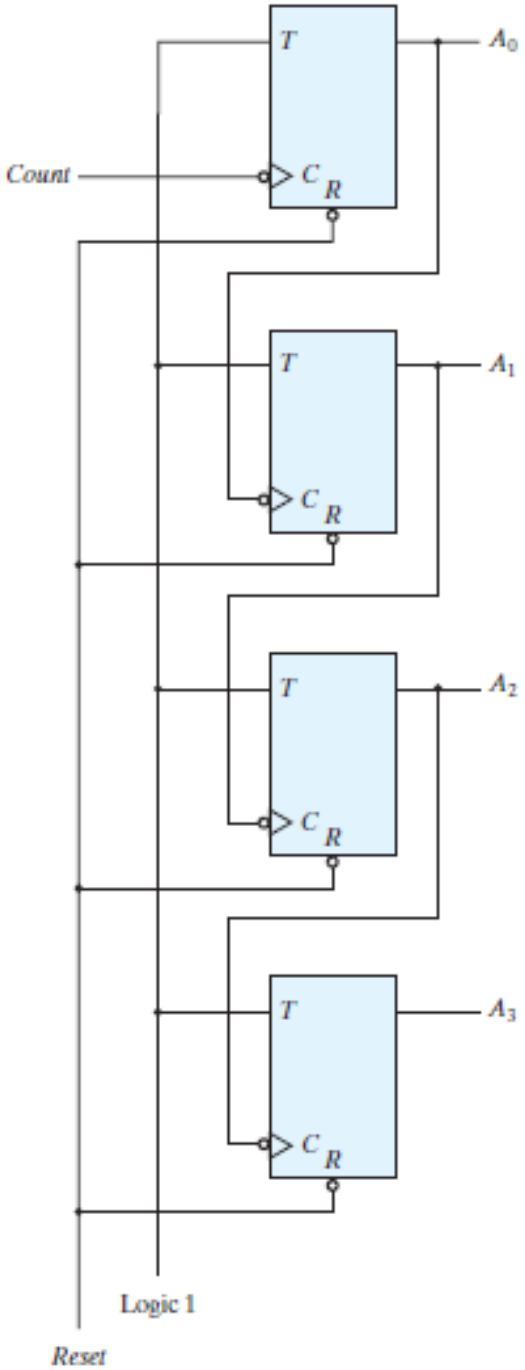


Fig: With T flip-flops

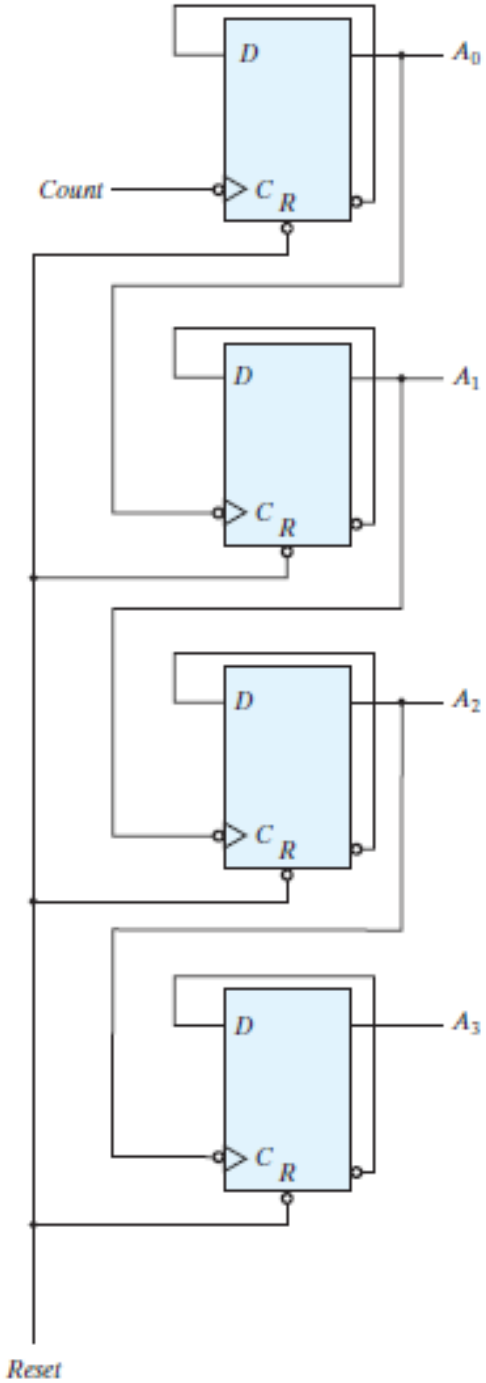


Fig: With *D flip-flops*

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 A_i goes from 1 to 0, it complements A_{i+1} (Negative trigger)

Binary countdown counter

- A binary counter with a reverse count is called a binary countdown counter.
- In a countdown counter, the binary count is decremented by 1 with every input count pulse.
- The count of a four-bit countdown counter starts from binary 15 and continues to binary counts 14, 13, 12, . . . , 0 and then back to 15.
- A list of the count sequence of a binary countdown counter shows that the least significant bit is complemented with every count pulse.
- Any other bit in the sequence is complemented if its previous least significant bit goes from 0 to 1.

Binary countdown counter

- Diagram: The bubble in the C inputs must be absent in binary ripple counter.
- If negative-edge-triggered flip-flops are used, then the C input of each flip-flop must be connected to the complemented output of the previous flip-flop.
- Then, when the true output goes from 0 to 1, the complement will go from 1 to 0 and complement the next flip-flop as required.

BCD Ripple Counter

- A decimal counter follows a sequence of 10 states and returns to 0 after the count of 9.
- Such 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.
- If BCD is used, the sequence of states is as shown in the state diagram of below Fig.
- 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).

BCD Ripple Counter

- If BCD is used, the sequence of states is as shown in the state diagram of below Fig.

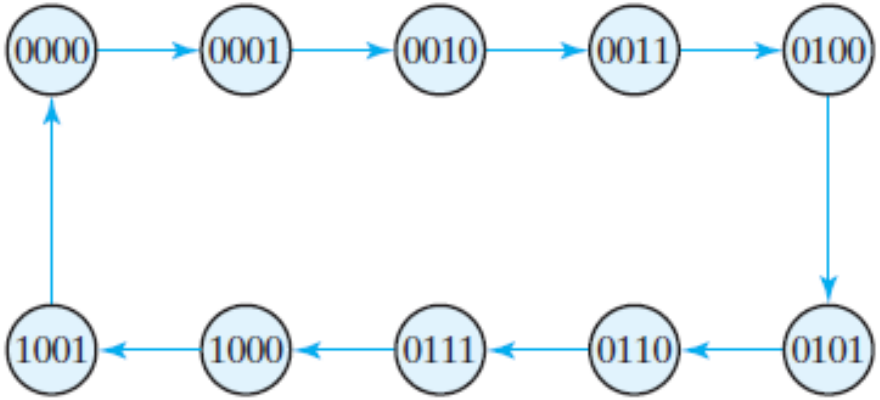
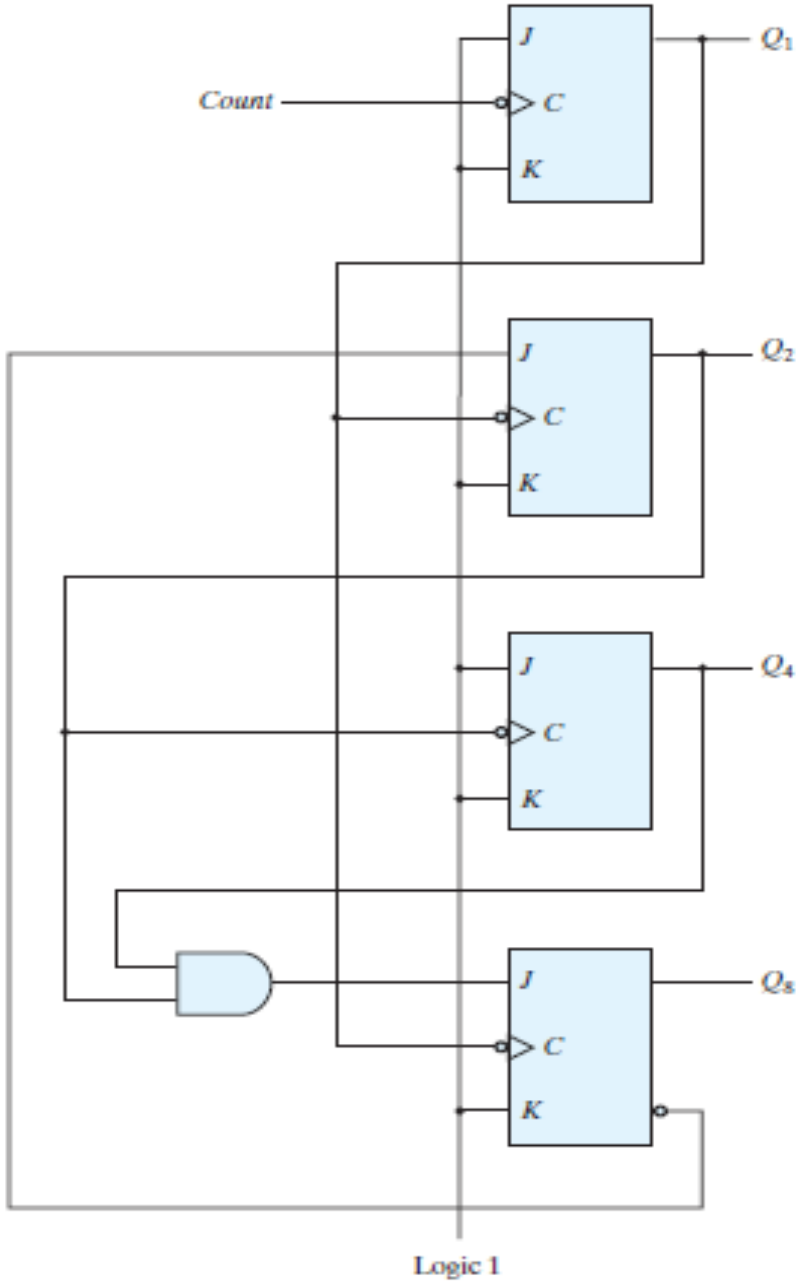


Fig: State diagram of a decimal BCD counter

- The logic diagram of a BCD ripple counter using JK flip-flops is shown in below Fig.

BCD Ripple Counter



- The four outputs are designated by the letter symbol Q, with a numeric subscript equal to the binary weight of the corresponding bit in the BCD code.

Present State				Next State			
Q8	Q4	Q2	Q1	Q8	Q4	Q2	Q1
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	0	0	0	0

BCD Ripple Counter

- The four outputs are designated by the letter symbol Q, with a numeric subscript equal to the binary weight of the corresponding bit in the BCD code.
- Note: 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.

BCD Ripple Counter

- A ripple counter is an asynchronous sequential circuit.
- Signals that affect the flip-flop transition depend on the way they change from 1 to 0.
- The operation of the counter can be explained by a list of conditions for flip-flop transitions.
- These conditions are derived from the logic diagram and from knowledge of how a JK flip-flop operates.
- Remember that 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.

BCD Ripple Counter

- To verify that these conditions result in the sequence required by a BCD ripple counter, it is necessary to verify that the flip-flop transitions indeed follow a sequence of states as specified by the state diagram.
  - Q1: always complemented
  - Q2: inverted when Q8 = 0 and Q1 = 1 → 0
  - Q4: inverted when Q2 = 1 → 0
  - Q8: when Q1 = 1 → 0
    if (Q2 = Q4 = 1) Q8 is inverted
    else Q8 = 0

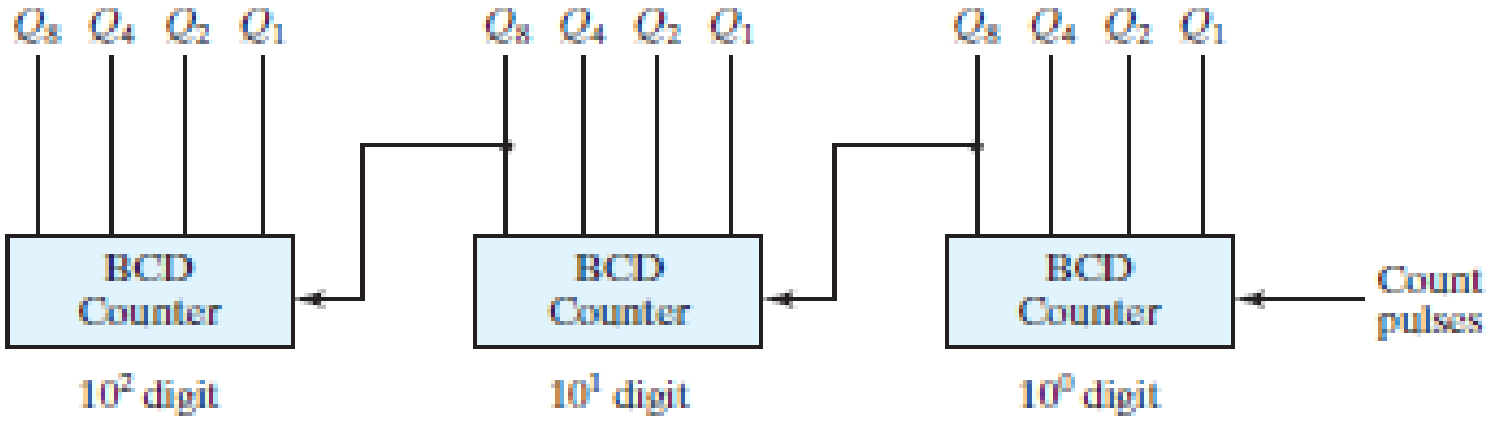
JK Flip-Flop

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

Three-Decade BCD Counter

- The BCD counter of above Fig. is a decade counter, since it counts from 0 to 9.
- 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 cascade, one for each decade.
- A three-decade counter is shown in above Fig.

Three-Decade BCD Counter



- 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 own decade goes from 9 to 0.

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.

Binary Counter

- The design of a synchronous binary counter is so simple that there is no need to go through a sequential logic design process.
- 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 .

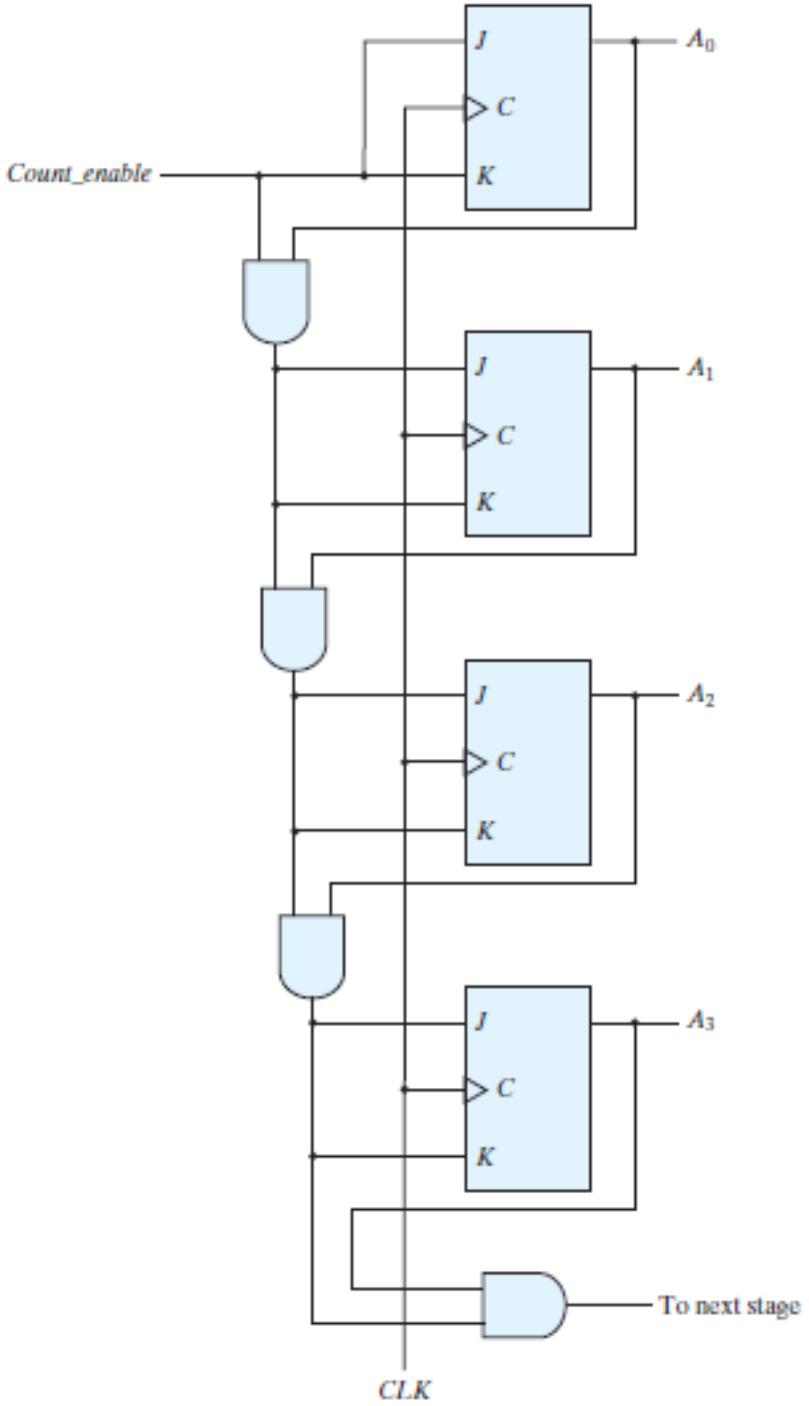
Binary Counter

- For example, if the present state of a four-bit counter is $A_3A_2A_1A_0$ = 0011, the next count is 0100.
- A_0 is always complemented.
- A_1 is complemented because the present state of A_0 = 1.
- A_2 is complemented because the present state of A_1A_0 = 11.
- However, A_3 is not complemented, because the present state of $A_2A_1A_0$ = 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
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

Binary Counter

- Synchronous binary counters have a regular pattern and can be constructed with complementing flip-flops and gates.
- The regular pattern can be seen from the four-bit counter depicted in Fig.



Binary Counter

- The C inputs of all flip-flops are connected to a common clock.
- 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 first stage, A0, has its J and K equal to 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.

Binary Counter

- 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.

Note:-

- The flip-flops trigger on the positive edge of the clock.
- The polarity of the clock is not essential here, but it is with the ripple counter.
- 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.

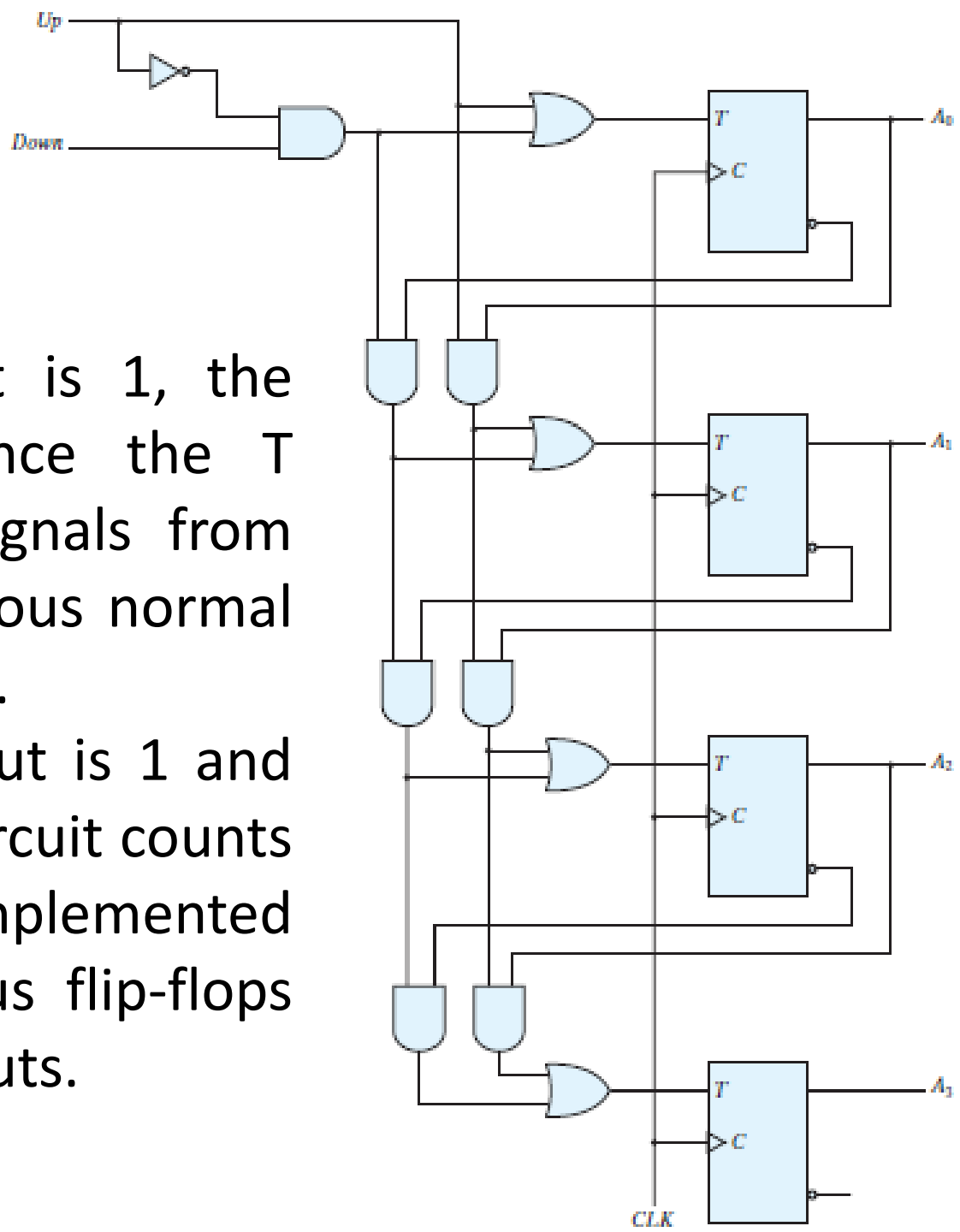
Up–Down Binary Counter

- The two operations 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 below Fig.
- It has an up control input and a down control input.

Up	Down	Operation
0	0	No change
0	1	Count Down
1	x	Count Up

Up–Down Binary Counter

- When the up input 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 input is 1 and the up input is 0, the circuit counts down, since the complemented outputs of the previous flip-flops are applied to the T inputs.



Up–Down Binary Counter

- 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.
- This set of conditions ensures that only one operation is performed at any given time.

Note:

- The up input has priority over the down input.

BCD Counter

- A BCD counter counts in binary-coded decimal from 0000 to 1001 and back to 0000.
- Because of the return to 0 after a count of 9, a BCD counter does not have a regular pattern, unlike a straight binary count.
- To derive the circuit of a BCD synchronous counter, it is necessary to go through a sequential circuit design procedure.

BCD Counter

- The state table of a BCD counter is listed in below Table.

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

BCD Counter

- The input conditions for the T flip-flops are obtained from the present- and next-state conditions.
- Also shown in the table is an output y, which is equal to 1 when the present state is 1001.
- In this way, y can enable the count of the next-higher significant decade while the same pulse switches the present decade from 1001 to 0000.

BCD Counter

- The flip-flop input equations can be simplified by means of maps.
- The unused states for minterms 10 to 15 are taken as don't-care terms.
- The simplified functions are -

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

BCD Counter

- The circuit can easily be drawn with four T flip-flops, five AND gates, and one OR gate.
- Synchronous BCD counters can be cascaded to form a counter for decimal numbers of any length.
- The cascading is done as in Fig. (Block diagram of a three-decade decimal BCD counter), except that output y must be connected to the count input of the next-higher significant decade.

Binary Counter with Parallel Load

- Counters employed in digital systems quite often require a parallel-load capability for transferring an initial binary number into the counter prior to the count operation.
- The below Fig. shows the top-level block diagram symbol and the logic diagram of a four-bit register that has a parallel load capability and can operate as a counter.
- When input load control equal's to 1 it disables the count operation and causes a transfer of data from the four data inputs into the four flip-flops.
- If both control inputs are 0, clock pulses do not change the state of the register.

Binary Counter with Parallel Load

- The carry output becomes 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.
- In going from state 1111 to 0000, only one gate delay occurs.
- Similarly, each flip-flop is associated with an AND gate that receives all previous flip-flop outputs directly instead of connecting the AND gates in a chain.

Binary Counter with Parallel Load

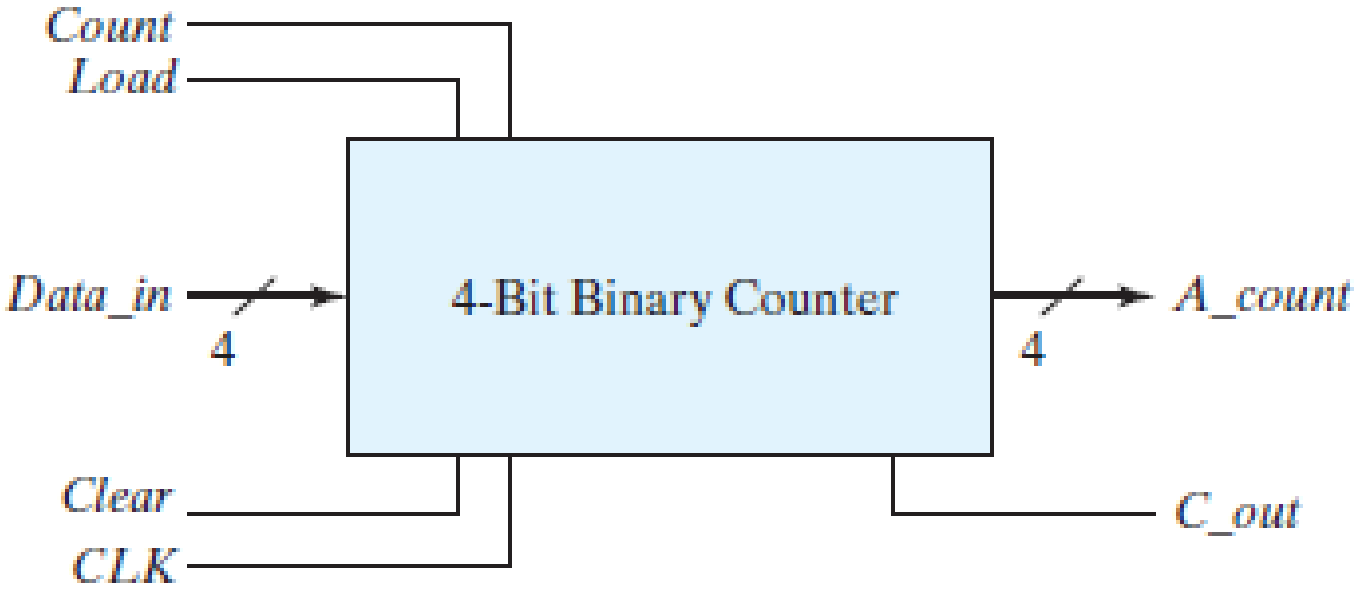
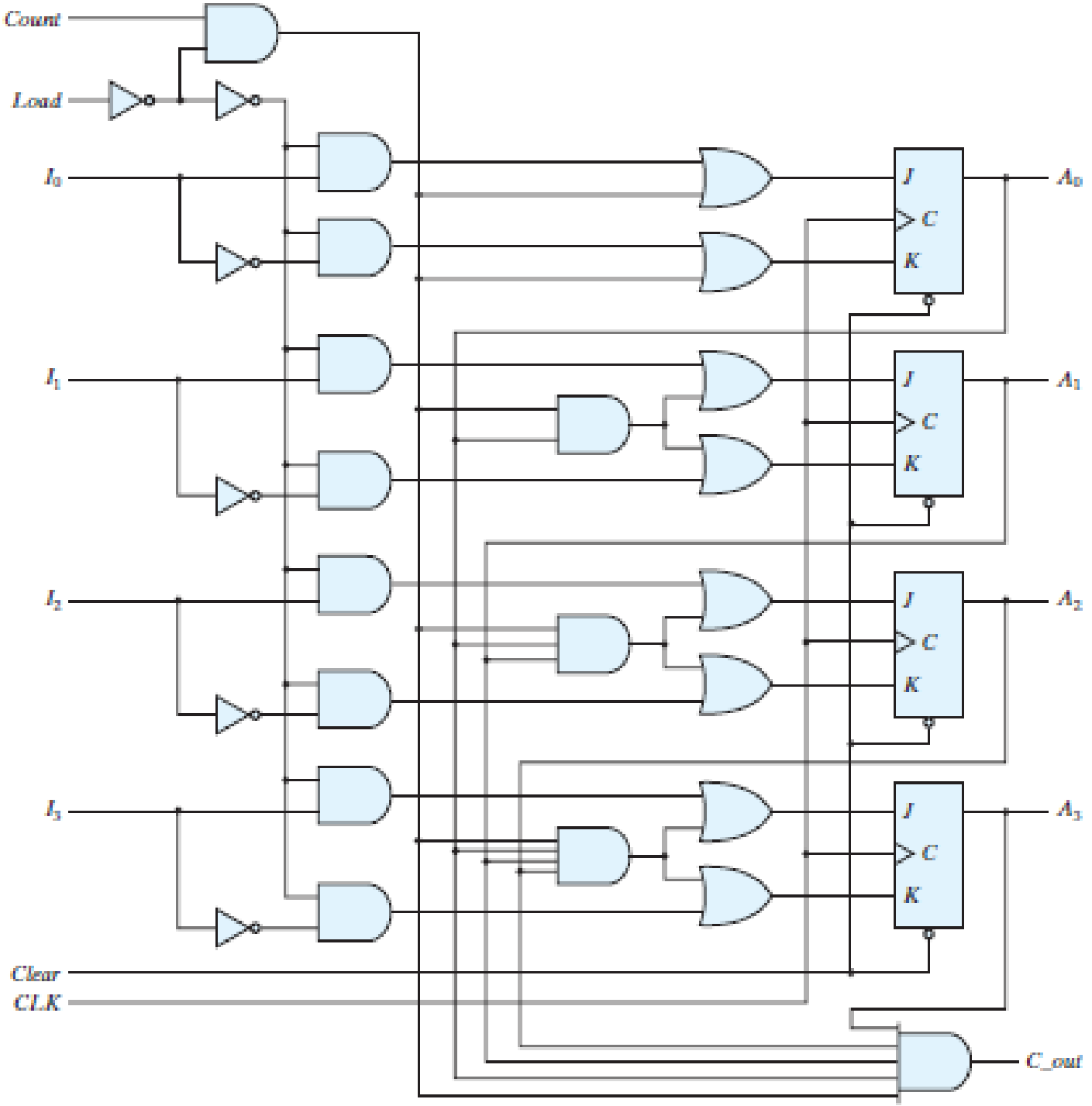


Fig: **Block Diagram of a four-bit register that has a parallel load capability and can operate as a counter**

Table: **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



Binary Counter with Parallel Load

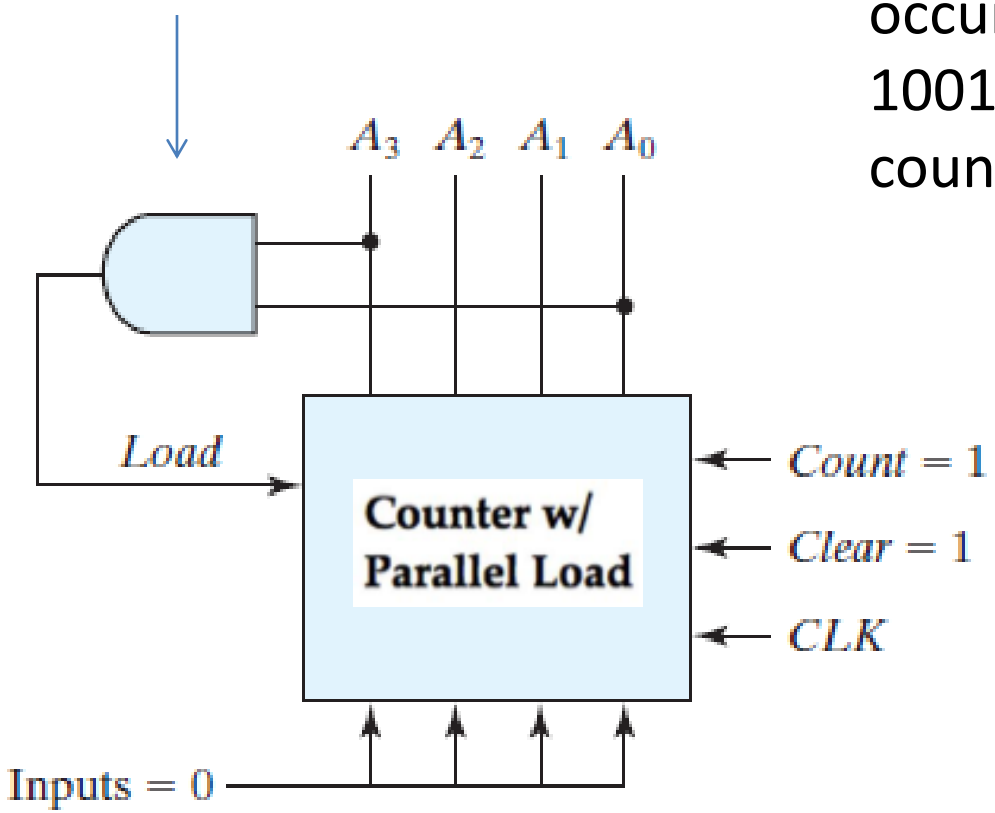
- A counter with a parallel load can be used to generate any desired count sequence.
- Below Fig. shows two ways in which a counter with a parallel load is used to generate the BCD count.
- In each case, the Count control is set to 1 to enable the count through the CLK input.
- The Load control inhibits the count and that the clear operation is independent of other control inputs.

Binary Counter with Parallel Load

- The AND gate in below Fig. 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.
- As long as the output of the AND gate is 0, each positive-edge clock increments the counter by 1.
- When the output reaches the count of 1001, both A_0 and A_3 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.
- Since all four inputs are connected to logic 0, an all-0's value is loaded into the register following the count of 1001.
- Thus, the circuit goes through the count from 0000 through 1001 and back to 0000, as is required in a BCD counter.

Binary Counter with Parallel Load

Load “0000” after “1001”



(a) Using the load input

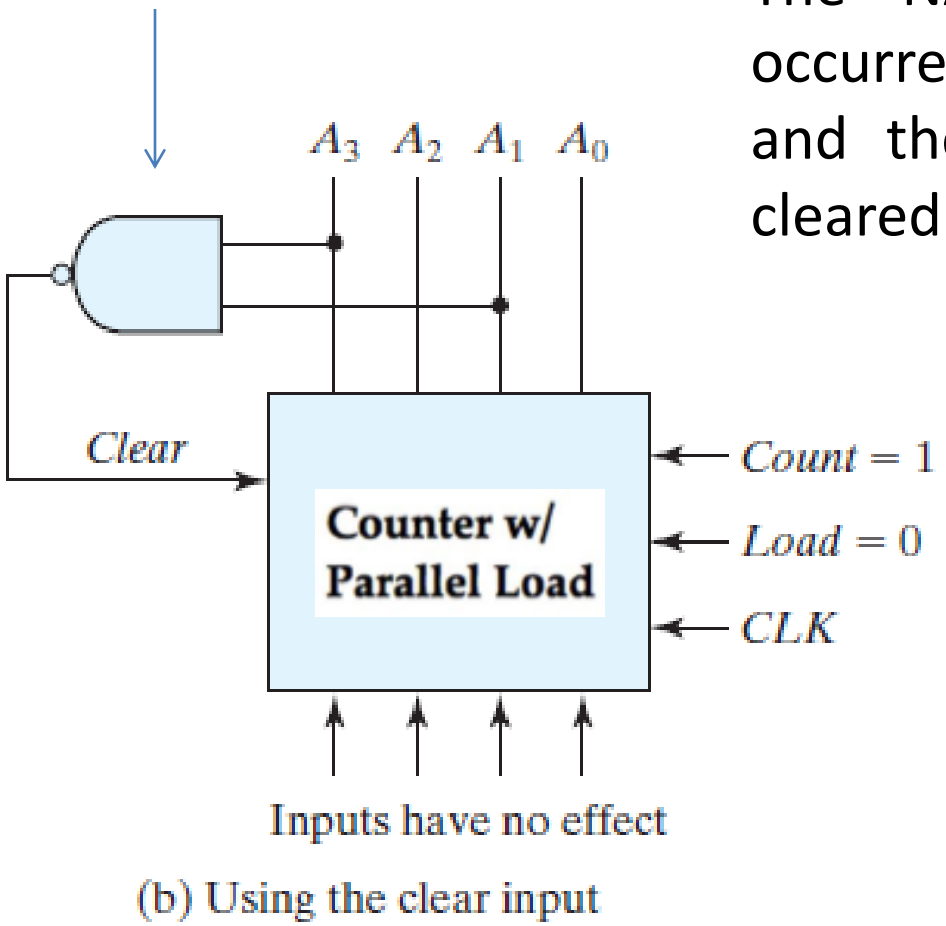
The AND detects the occurrence of state 1001 and then the counter reloads 0000

Binary Counter with Parallel Load

- In below Fig. 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 A_0 as the count goes from 1010 to 1011 and 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.

Binary Counter with Parallel Load

Clear to “0000” immediately at “1010”



(b) Using the clear input

The NAND detects the occurrence of state 1010 and then the counter is cleared to 0