



APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

LOGIC SYSTEM DESIGN (CST-203)

MODULE 5
NOTES

PREPARED BY : JIBIN EP, ASSISTANT PROFESSOR, EKC TC MANJERI

INDEX

SL No	Topic	Slide Link	Video Link
0	Syllabus	Slide 4	NA
1	Introduction to Registers	INTRODUCTION TO REGISTERS	https://youtu.be/yAAZpC968CY
2	Classification of Shift Register	TYPES OF SHIFT REGISTER	https://youtu.be/9bV9otTD6ls
3	Bidirectional Shift Registers with Parallel load	BIDIRECTIONAL SHIFT REGISTER WITH PARALLEL LOAD	https://youtu.be/1paBzP5WoiU
4	The Ring Counter	THE RING COUNTER	https://youtu.be/EuWodsvzYOk
5	The Johnson Counter	TWISTED RING COUNTER/ JOHSON COUNTER	https://youtu.be/rpZAH7XVA70

INDEX

SL No	Topic	Slide Link	Video Link
6	Arithmetic Algorithm for Signed Magnitude Number	<u>ARITHMETIC ALGORITHM FOR SIGNED MAGNITUDE NUMBER</u>	<u>https://youtu.be/zcz35pSAI4Q</u>
7	Arithmetic Algorithm addition of Floating Point Number	<u>ARITHMETIC ALGORITHM FOR FLOATING POINT ADDITION AND SUBT...</u>	<u>https://youtu.be/cgKf9I7p6EA</u>
8	Arithmetic Algorithm for addition of BCD numbers	<u>ARITHMETIC ALGORITHM FOR ADDITION AND SUBTRACTION OF BCD...</u>	<u>https://youtu.be/hfbvmGZ79SA</u>
9	Arithmetic Algorithm for 2'S Complement arithmetic's	<u>ARITHMETIC ALGORITHM FOR 2'S COMPLEMENT ARITHMETIC</u>	<u>https://youtu.be/-2Ext37PzPU</u>
10	Read Only Memory	<u>PROGRAMMABLE LOGIC DEVICES : READ ONLY MEMORY</u>	<u>https://youtu.be/SXhbEO1ZS4E</u>
11	Programmable Logic Array	<u>PROGRAMMABLE LOGIC ARRAY</u>	<u>https://youtu.be/PdpYb16mEaw</u> <u>https://youtu.be/1aHDDJEUg7g</u>

Module V

Shift registers

Shift registers – Serial In Serial Out, Serial In Parallel Out, Bidirectional Shift Register with Parallel load. Ring counter. Johnson counter- timing sequences and state diagrams.

Arithmetic algorithms

Algorithms for addition and subtraction of binary numbers in signed magnitude and 2's complement representations. Algorithm for addition and subtraction of BCD numbers. Representation of floating point numbers, Algorithm for addition and subtraction of floating point numbers.

Programmable Logic devices

ROM. Programmable Logic Array(PLA)- Implementation of simple circuits using PLA.

MODULE 5 (TOPIC-1)

INTRODUCTION TO REGISTERS

[INDEX](#)

<https://youtu.be/yAAZpC968C>



YOUTUBE : SHASTRA TECHNICAL INSTITUTE

Introduction to Registers

- Flipflop is 1 bit memory cell
- To increase the storage capacity, we have to use group of flipflops. This group of Flipflops is known as **Registers**
- The 'n' bit register consist of 'n' number of flipflops and is capable of storing 'n-bit' word

Classification of Registers

- Data can be entered in Serial or Parallel form



Classification of Registers

Depending upon input and output

SISO

- SERIAL INPUT SERIAL OUTPUT

SIPO

- SERIAL INPUT PARALLEL OUTPUT

PISO

- PARALLEL INPUT SERIAL OUTPUT

PIPO

- PARALLEL INPUT PARALLEL OUTPUT

Classification of Registers

Depending upon Application

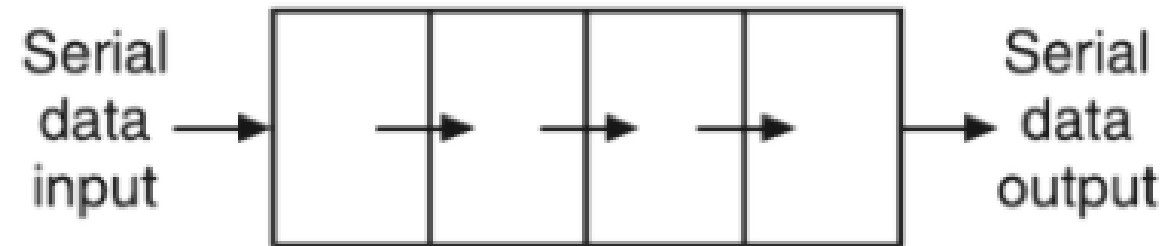
SHIFT REGISTERS

- SISO, SIPO, PISO

STORAGE REGISTERS

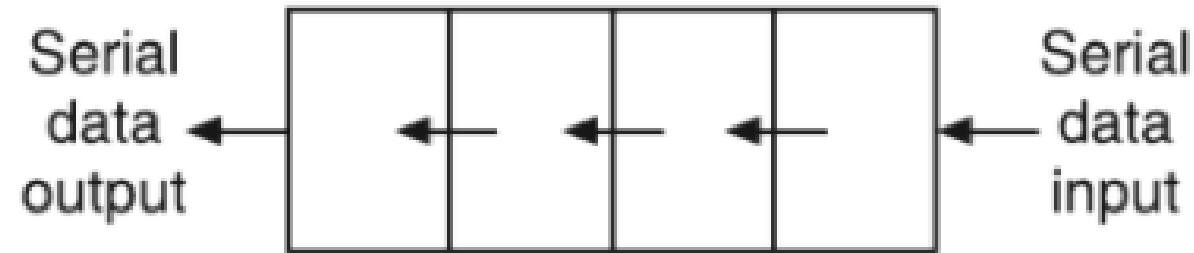
- PIPO

Data Transfer in Registers



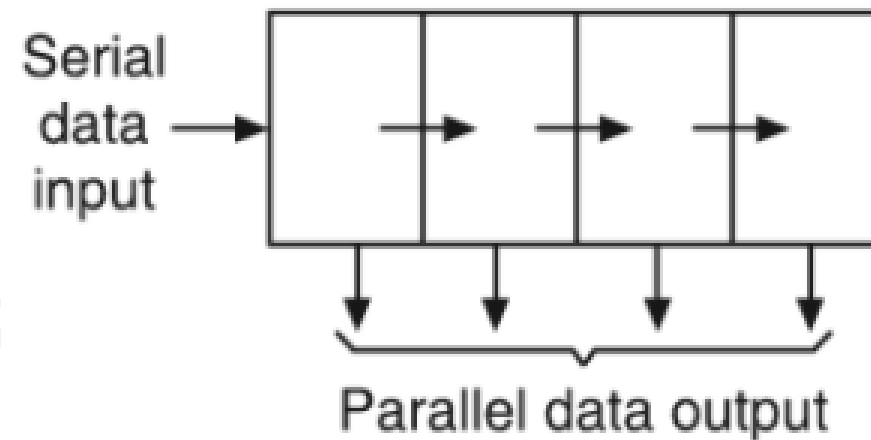
(a) Serial-in, serial-out,
shift-right, shift register

Data Transfer in Registers



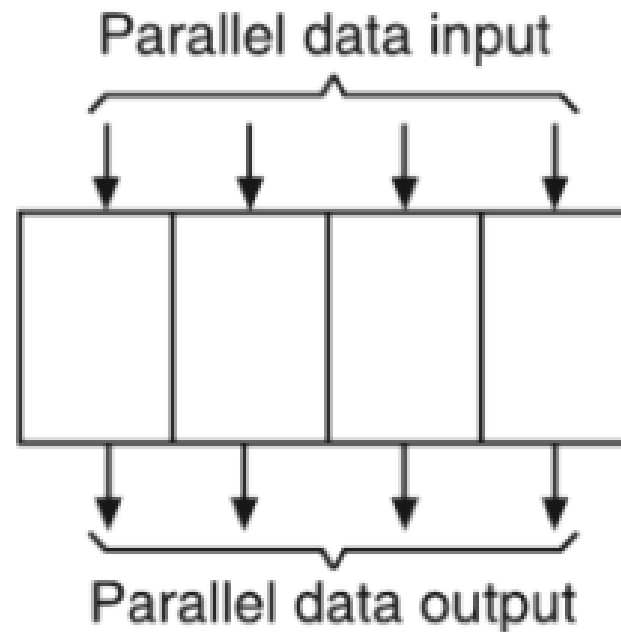
(b) Serial-in, serial-out,
shift-left, shift register

Data Transfer in Registers



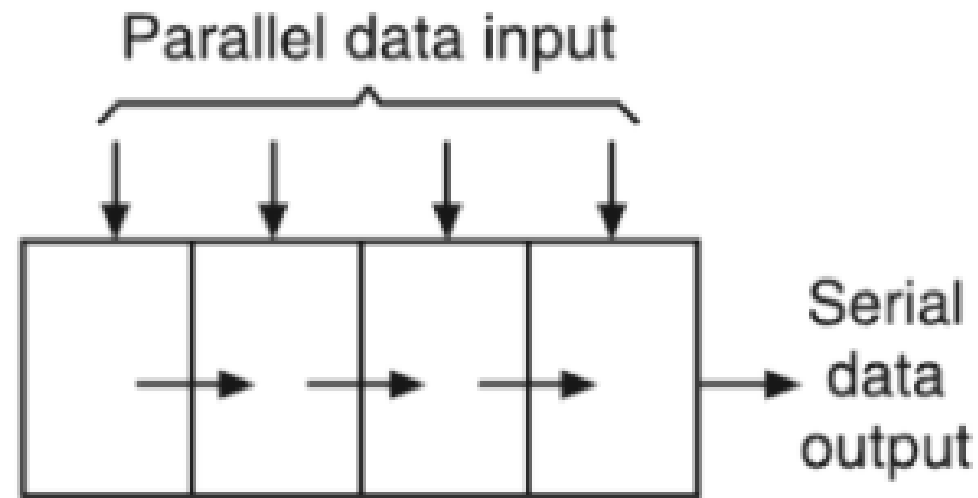
(c) Serial-in, parallel out, shift register

Data Transfer in Registers



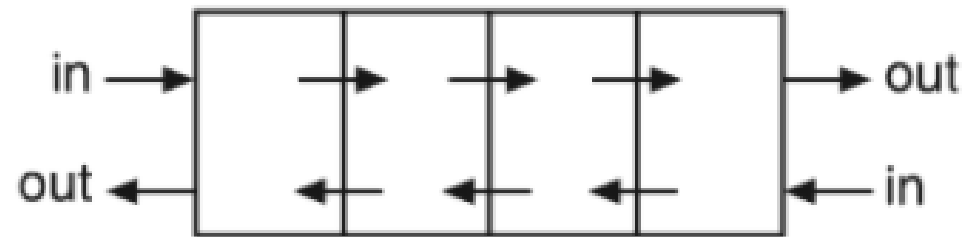
(d) Parallel-in, parallel-out, shift register

Data Transfer in Registers



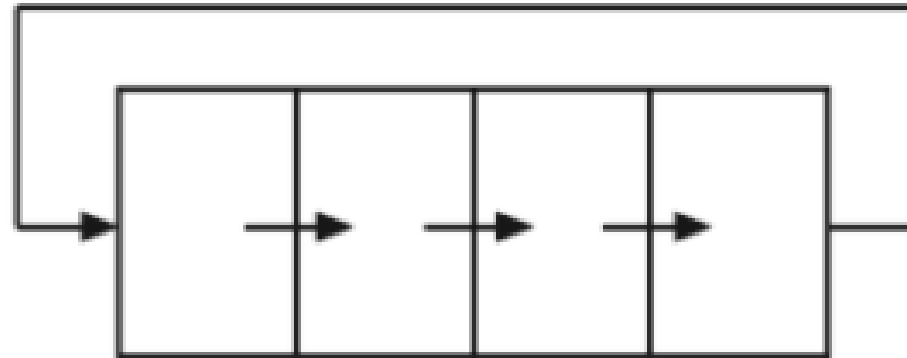
(e) Parallel-in, serial-out,
shift register

Data Transfer in Registers



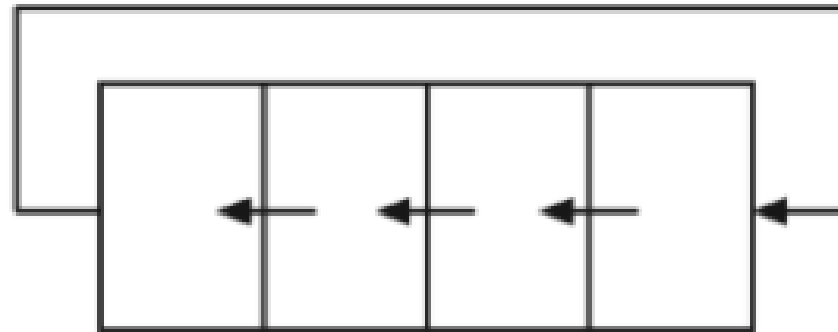
(f) Serial-in, serial-out, shift-left, shift-right, (bidirectional) shift register

Data Transfer in Registers



(g) Rotate-right shift register

Data Transfer in Registers



(h) Rotate-left shift register

MODULE 5 (TOPIC-2)

TYPES OF SHIFT REGISTER

[INDEX](#)

<https://youtu.be/9bV9otTD6ls>



YOUTUBE : SHASTRA TECHNICAL INSTITUTE

SISO

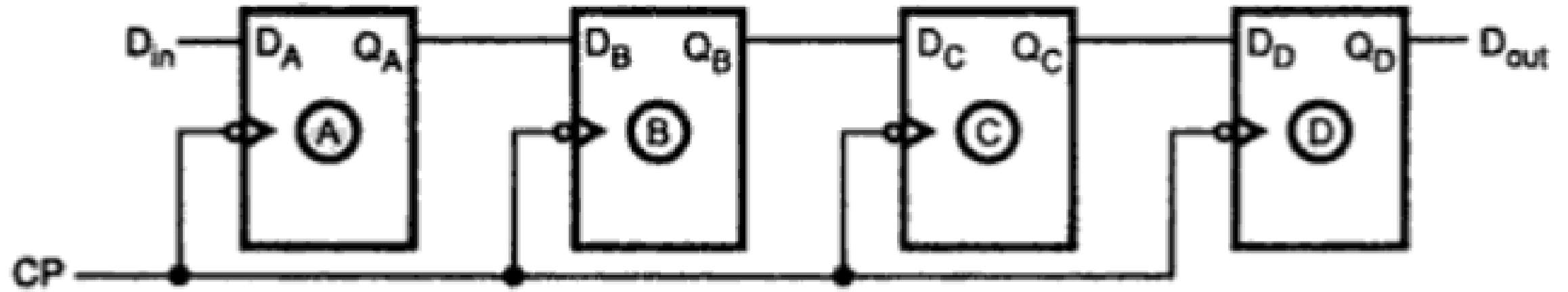
SHIFT
REGISTER

SISO SHIFT REGISTER

- The shift register, which allows serial input and produces serial output is known as Serial In - Serial Out **SISO** shift register
- This block diagram consists of D flip-flops, which are **cascaded**. That means, output of one D flip-flop is connected as the input of next D flip-flop.
- All these flip-flops are synchronous with each other since, the same clock signal is applied to each one.
- In this shift register, we can send the bits serially from the input of left most D flip-flop. Hence, this input is also called as **serial input**.
- For every positive edge triggering of clock signal, the data shifts from one stage to the next.
- So, we can receive the bits serially from the output of right most D flip-flop. Hence, this output is also called as **serial output**.

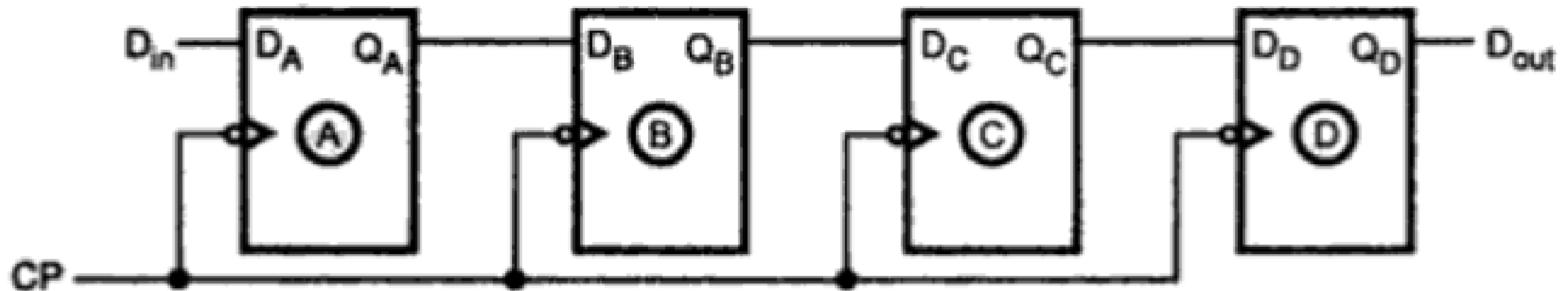
SISO SHIFT RIGHT REGISTER

- Figure shows Serial-in-Serial-Out Shift Register

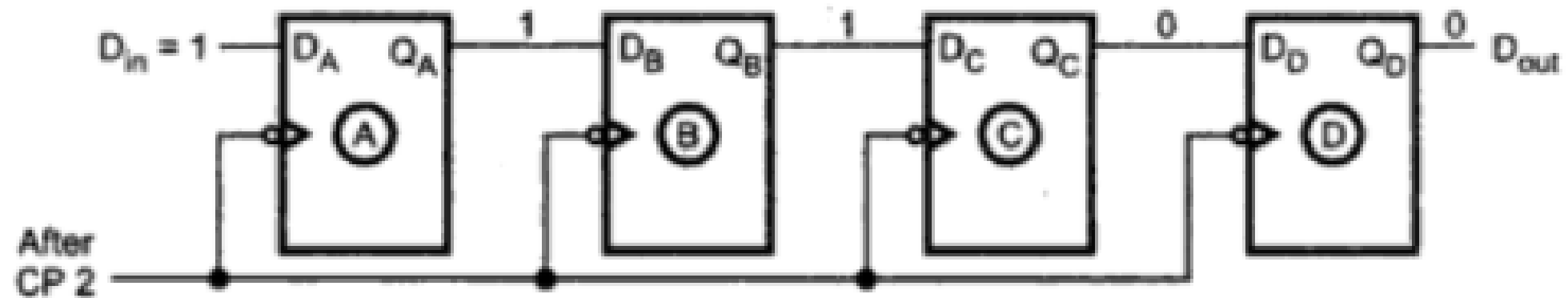
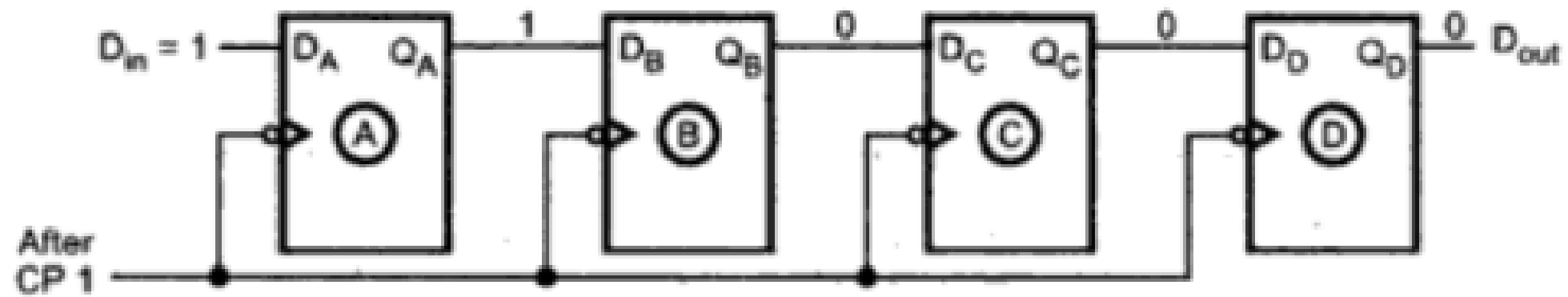
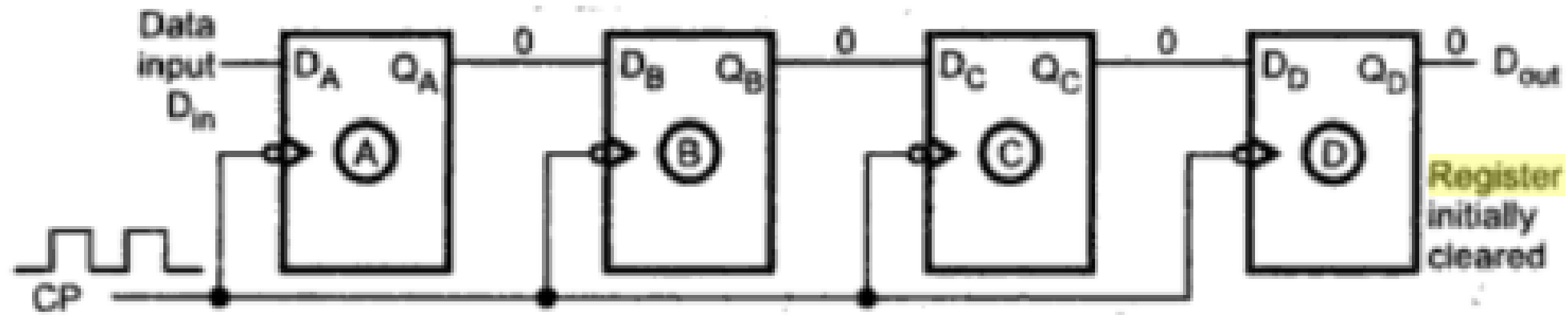


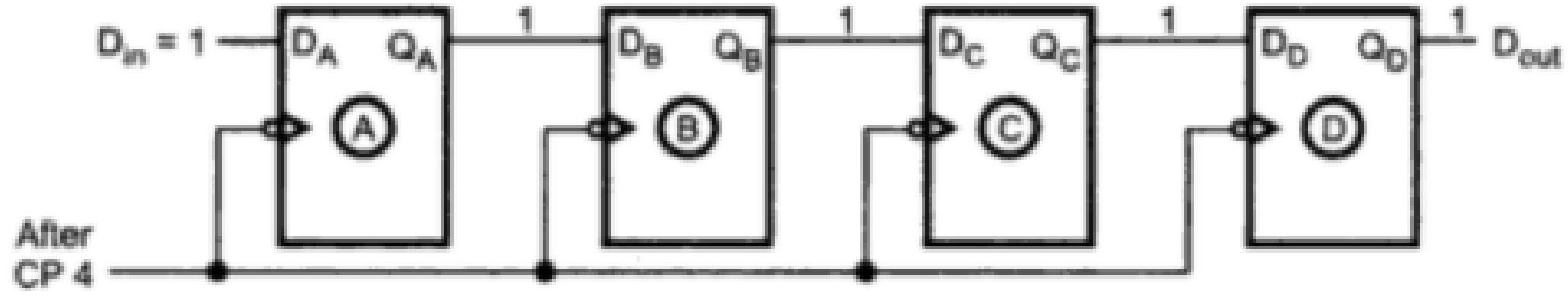
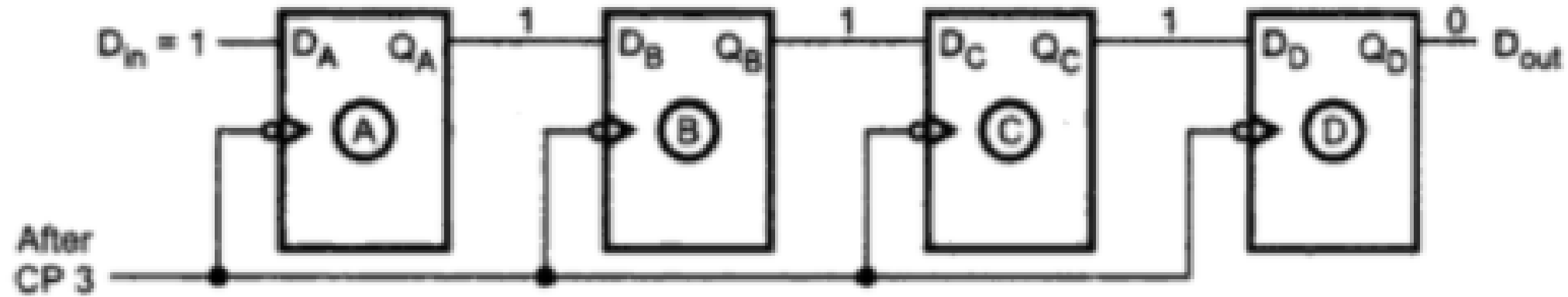
SISO SHIFT RIGHT REGISTER

- We will illustrate the Entry of four bit binary number **1111** into the register, beginning with the leftmost bit



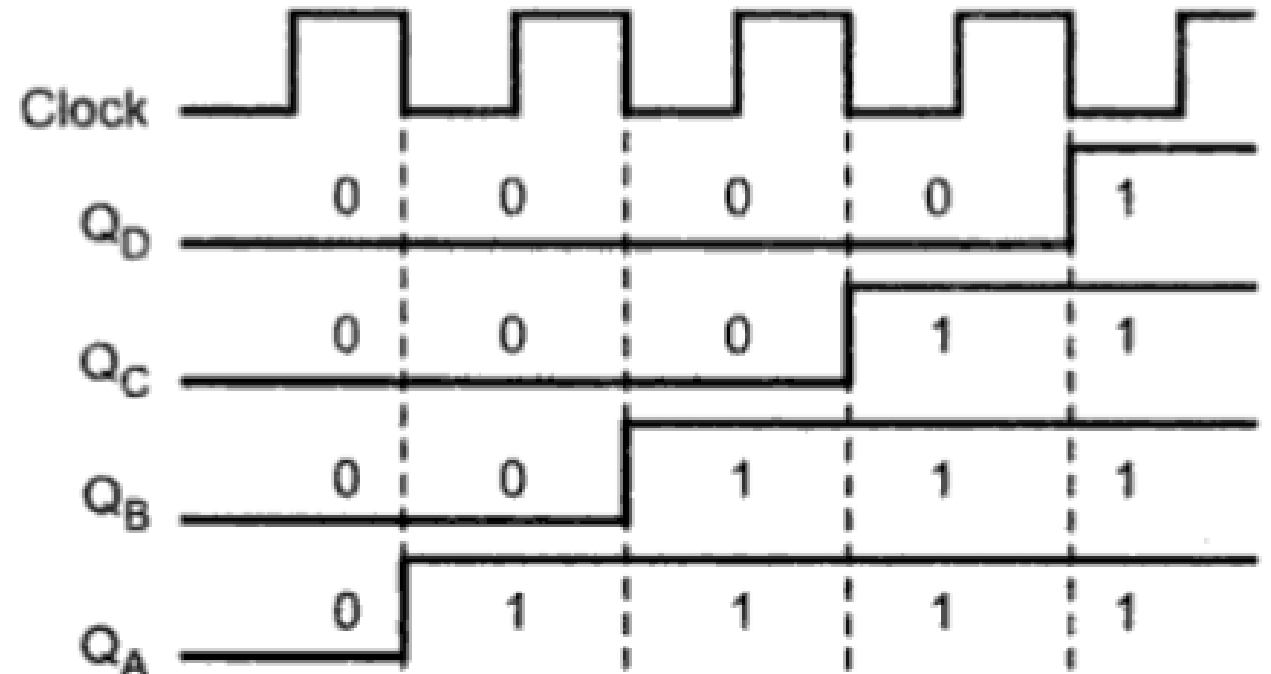
- Initially the register is cleared so $Q_A Q_B Q_C Q_D = 0 0 0 0$





Waveform of SISO

Clock	Qa	Qb	Qc	Qd
Initial	0	0	0	0
1st CP	1	0	0	0
2nd CP	1	1	0	0
3rd CP	1	1	1	0
4th CP	1	1	1	1

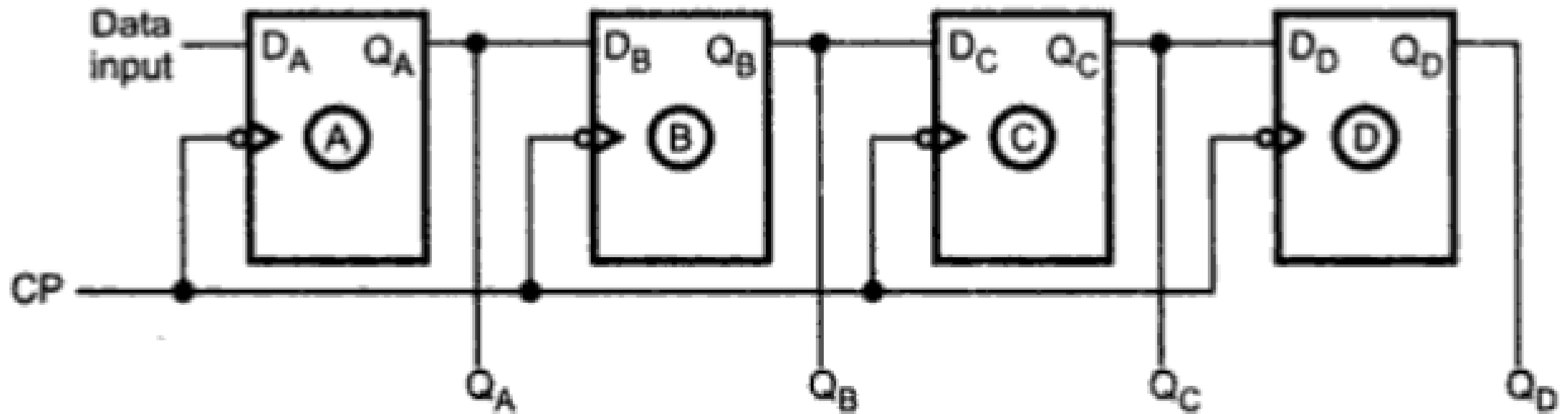


SIPO

SHIFT
REGISTER

SIPO SHIFT REGISTER

- In this case the data bits are entered into the register serially but the output is taken parallel
- Once the data are stored, Each bit appears on respective output line and all bits are available simultaneously

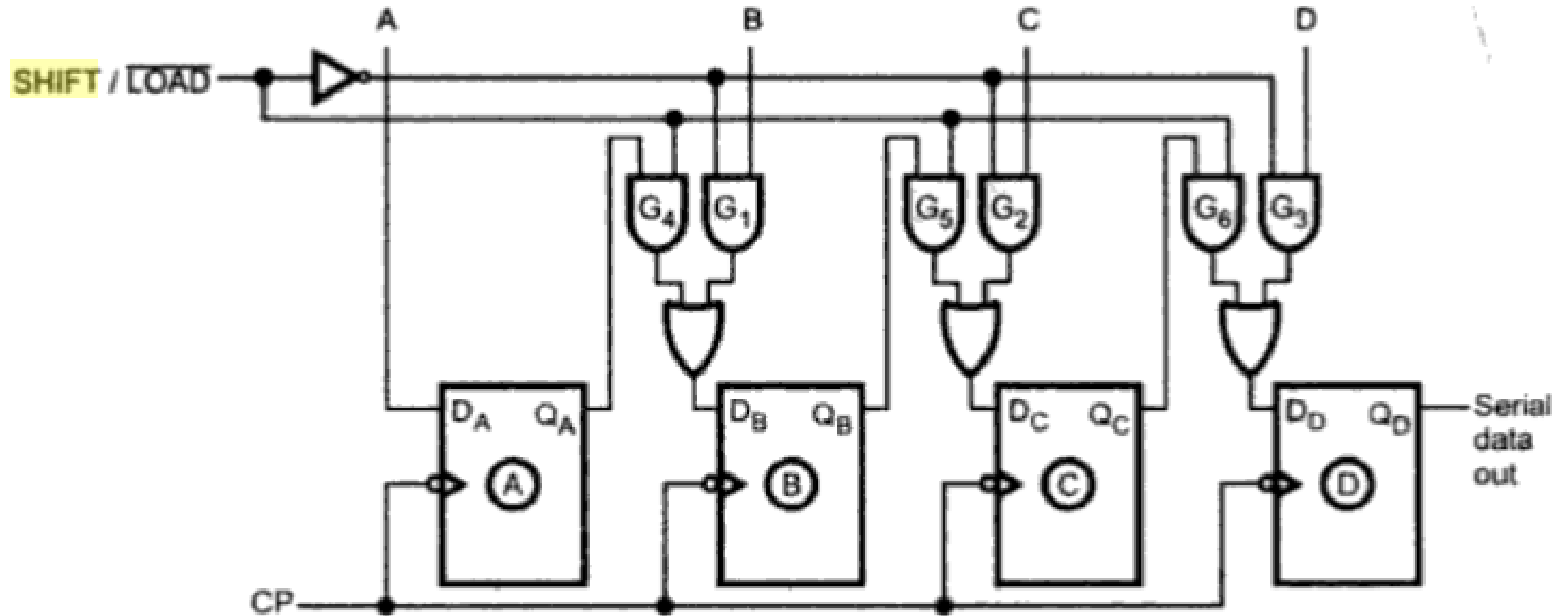


PISO

SHIFT
REGISTER

PISO SHIFT REGISTER

- In this type, the bits are entered in parallel



PISO SHIFT REGISTER

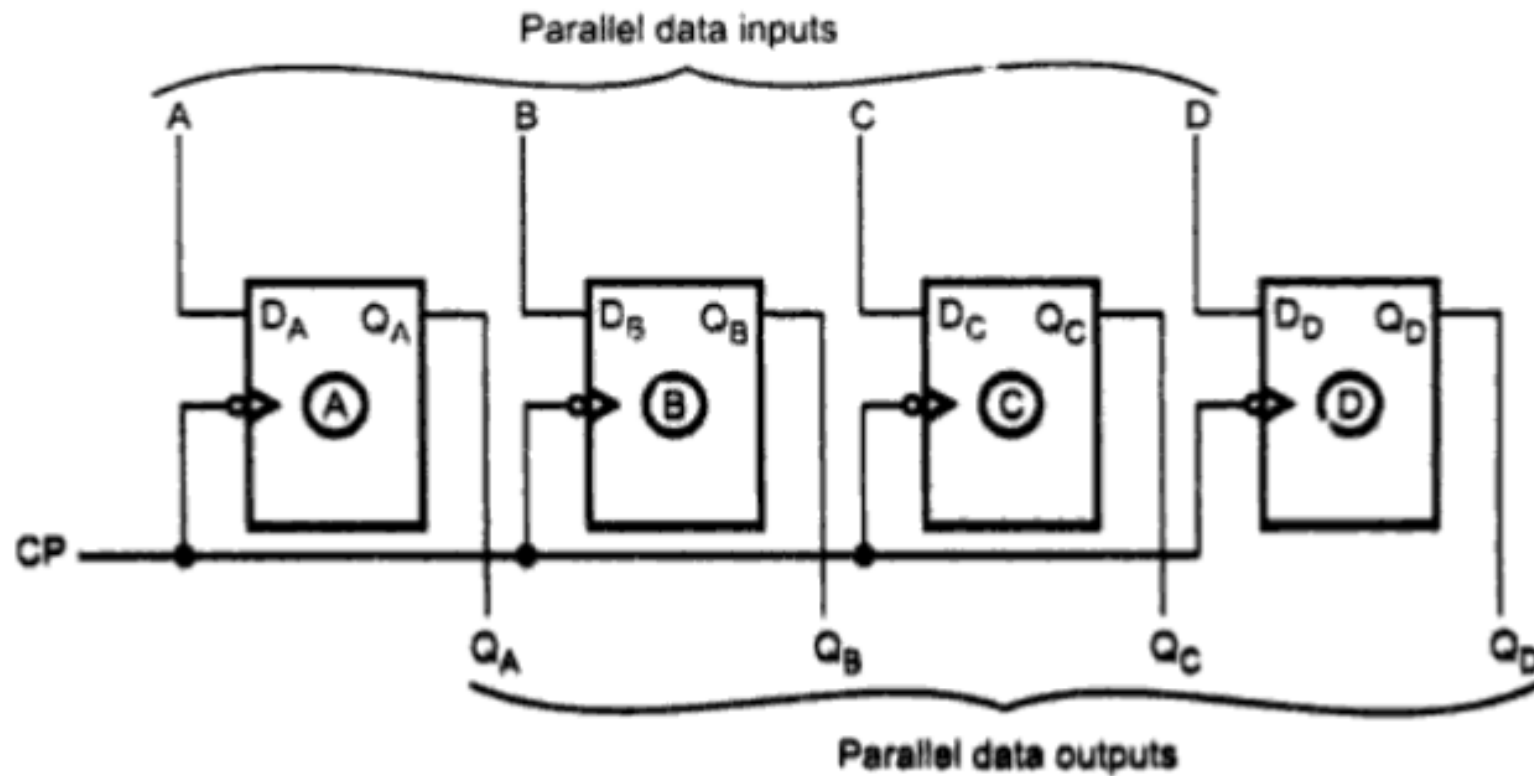
- There are 4 input lines for entering data in parallel
- Shift / Load is the control input which allows shift or loading data operation of the register
- When Shift / Load is low, the gates G1, G2, G3 are enabled, allowing each input data bit to be applied to D input of respective Flipflops
- When Shift / Load is High, G1, G2 and G3 disabled and G4, G5 and G6 are enabled. This allow data bits shift left from one stage to next
- The OR Gates at D Input of Flipflops allow either the parallel data entry operation or shift operation , Depending upon which AND Gate are enabled by the level on Shift / Load Input

PIPO

REGISTER

PIPO

- In PIPO, There is a simultaneous entry of all data bits and the bit appears on parallel output simultaneously



MODULE 4 (TOPIC-3)

BIDIRECTIONAL SHIFT REGISTER WITH PARALLEL LOAD

[INDEX](#)

<https://youtu.be/1paBzP5WoiU>  **YouTube**

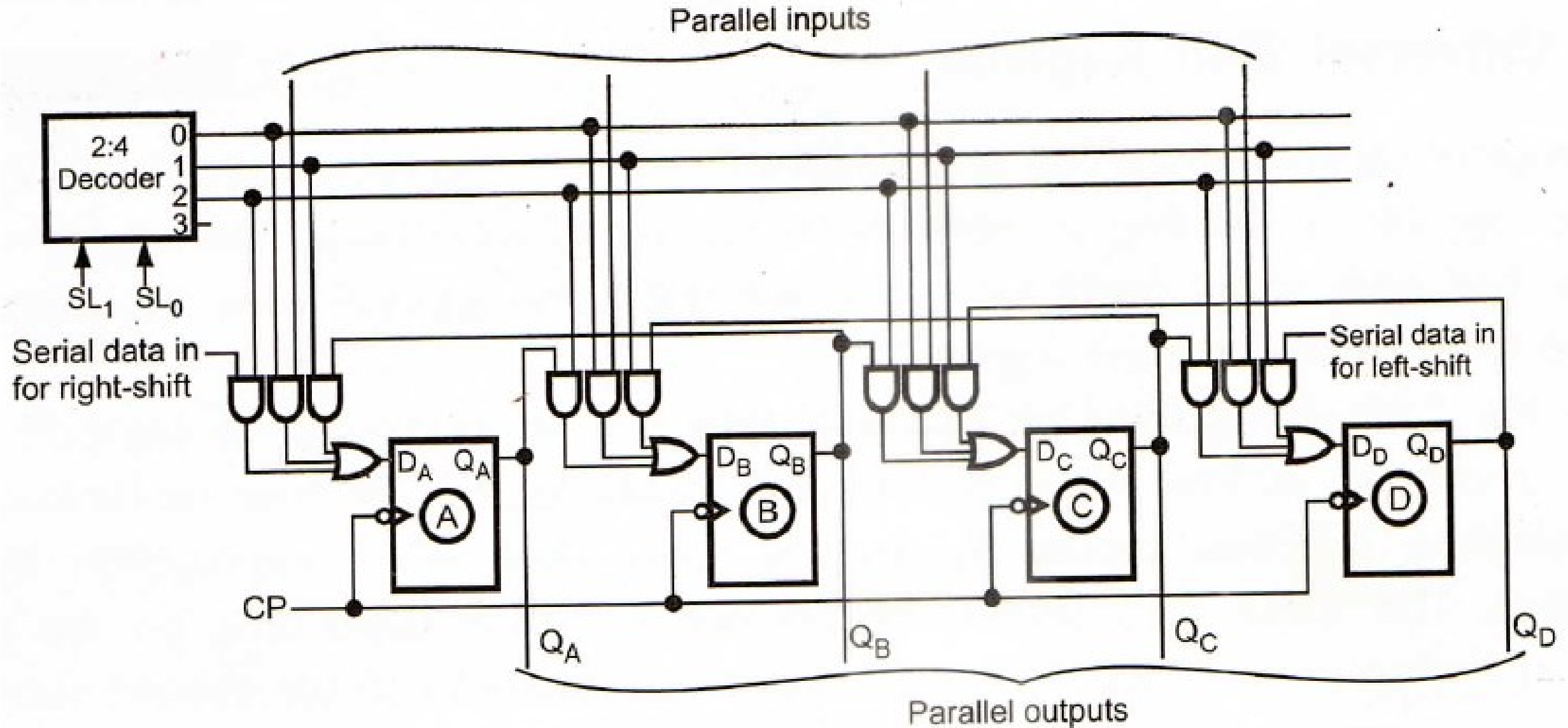
YOUTUBE : SHASTRA TECHNICAL INSTITUTE

BIDIRECTIONAL SHIFT REGISTER WITH PARALLEL LOAD

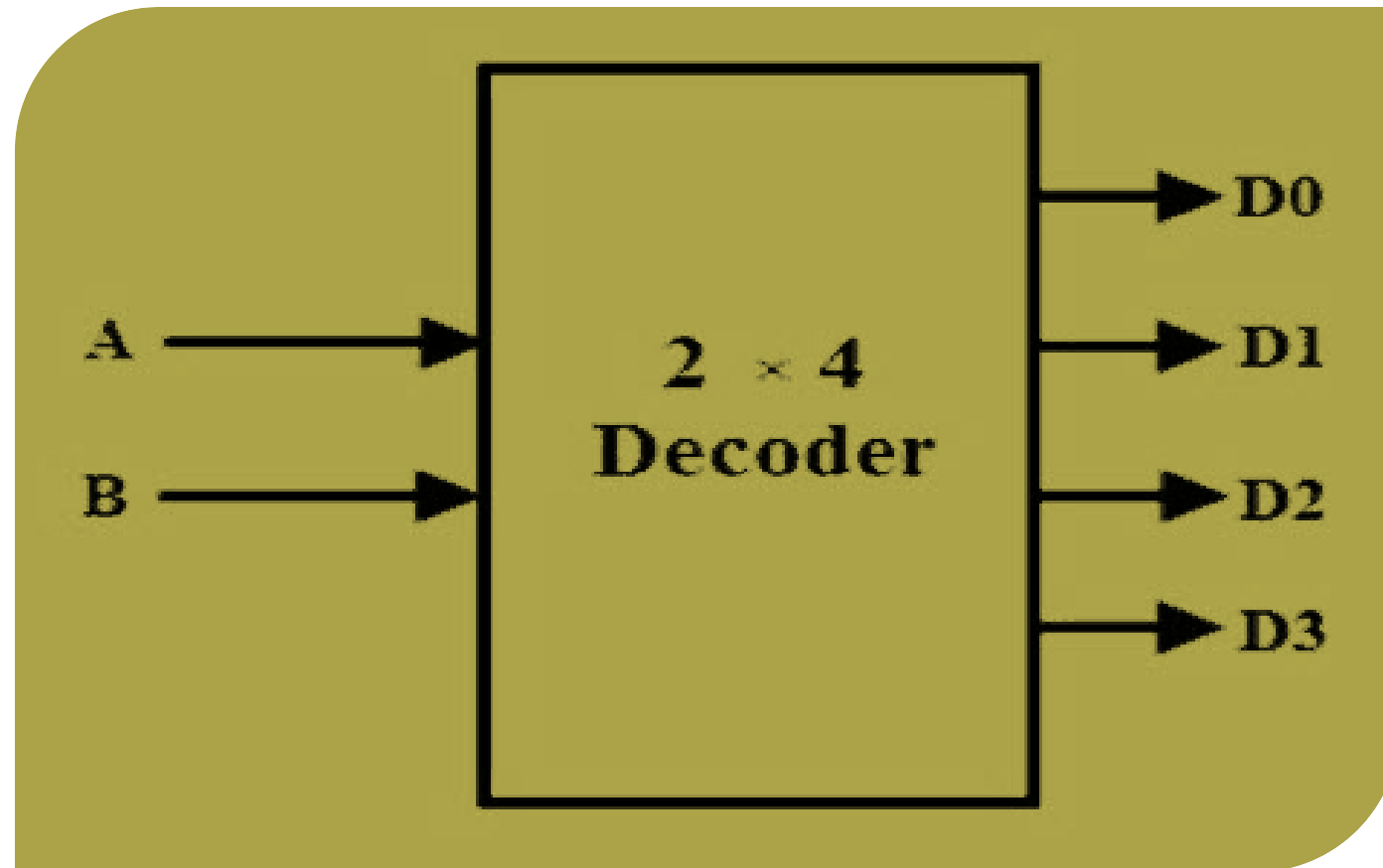
- Bidirectional shift registers are the storage devices capable of shifting the data either right or left, depending on the mode selected.
- Here the parallel load capability is added to the shift register, the data entered in parallel can be taken out serial fashion by shifting data stored in the register



BIDIRECTIONAL SHIFT REGISTER WITH PARALLEL LOAD

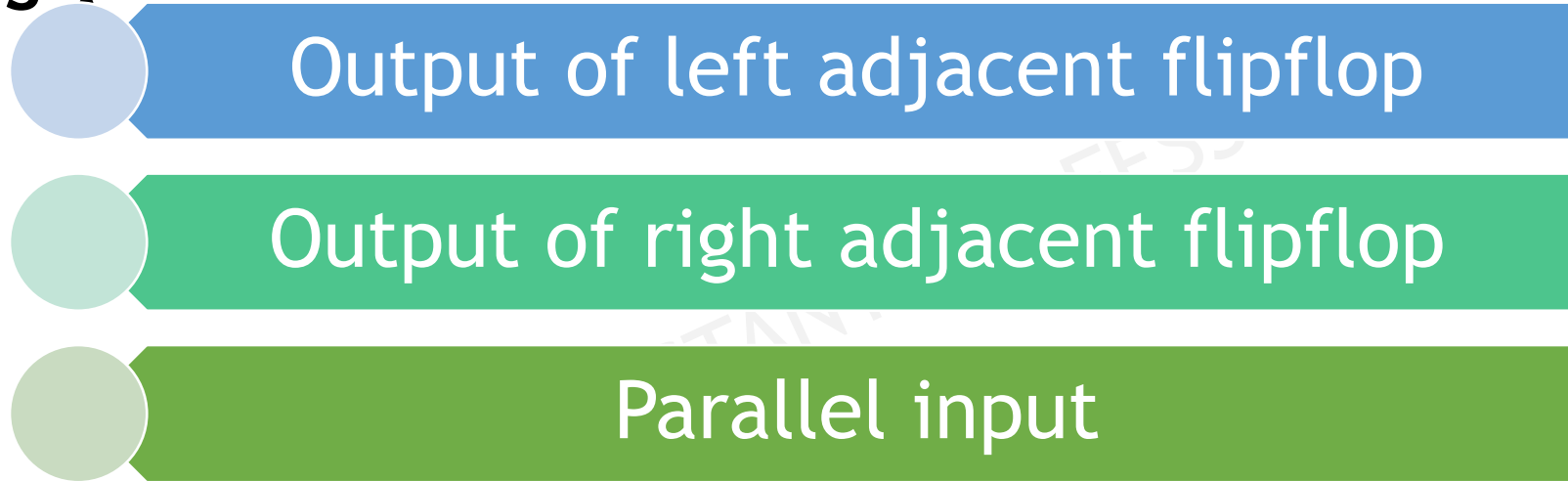


HOW DECODER WORKS



BIDIRECTIONAL SHIFT REGISTER WITH PARALLEL LOAD

- As Shown in figure, D Input of each flipflop has 3 sources :



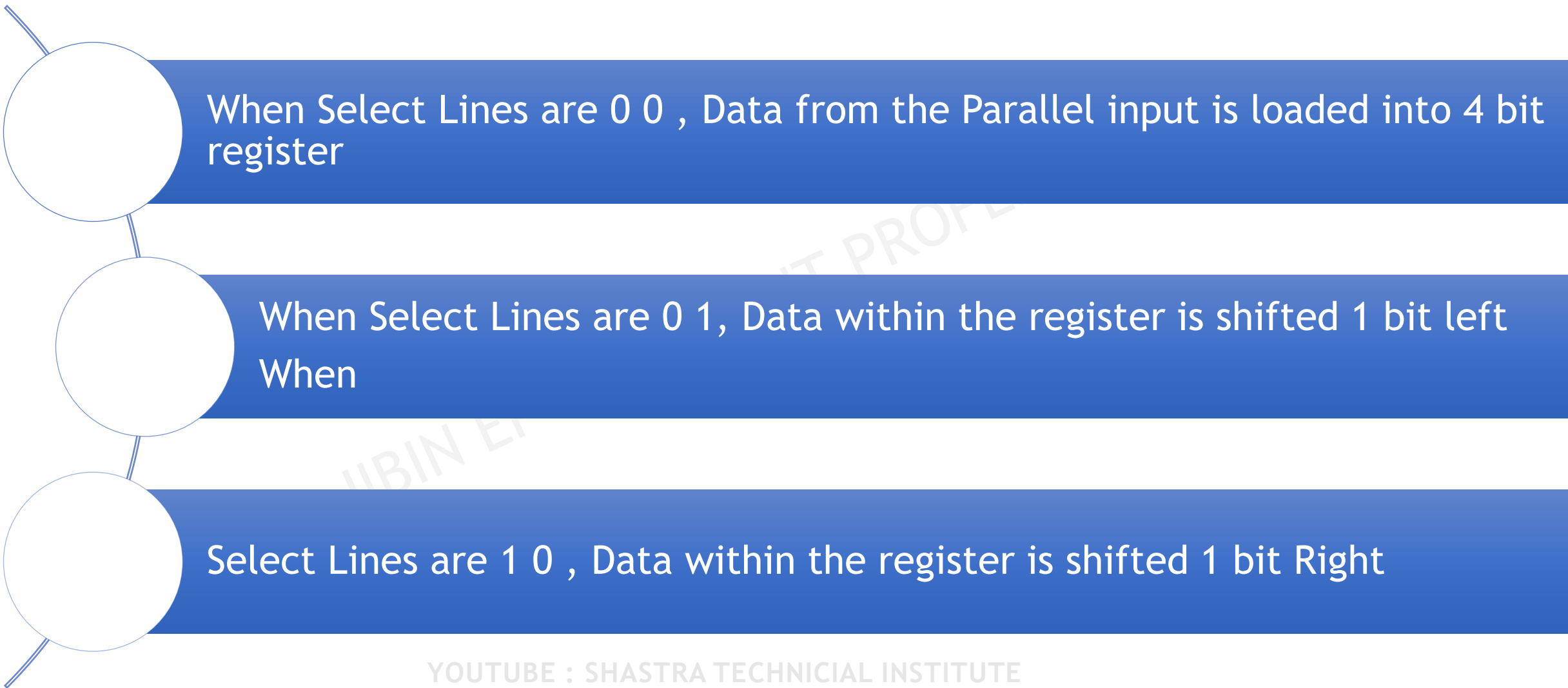
- Out of these three sources, one source is selected at a time and it is done with the help of decoder

BIDIRECTIONAL SHIFT REGISTER WITH PARALLEL LOAD

- The decoder select lines (SL_1 and SL_0) Select the one source out of three

SL_1	SL_0	Selected Source
0	0	Parallel Input
0	1	Output of Right Adjacent Flipflop
1	0	Output of Left Adjacent Flipflop
1	1	Not Used

BIDIRECTIONAL SHIFT REGISTER WITH PARALLEL LOAD



When Select Lines are 0 0 , Data from the Parallel input is loaded into 4 bit register

When Select Lines are 0 1, Data within the register is shifted 1 bit left
When

Select Lines are 1 0 , Data within the register is shifted 1 bit Right

MODULE 5 (TOPIC-4)

THE RING COUNTER

[INDEX](#)

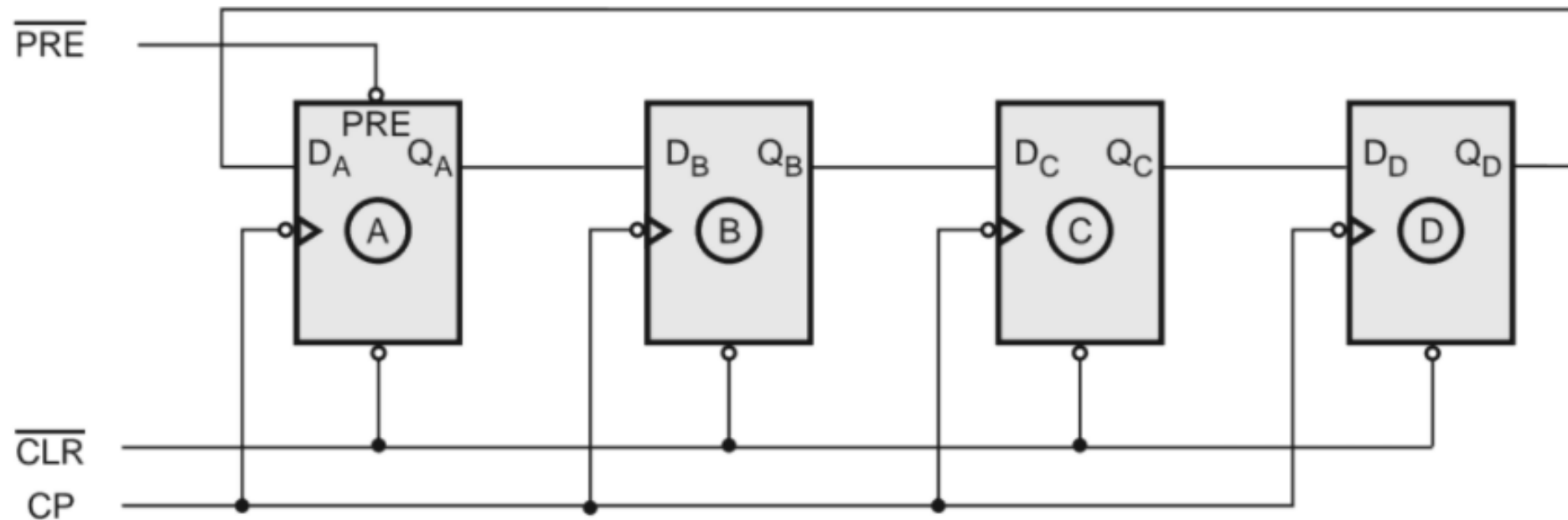
<https://youtu.be/EuWodsvzYQk>



YOUTUBE : SHASTRA TECHNICAL INSTITUTE

THE RING COUNTER

- In ring counter the Q output of each stage is connected to D input of Next stage and the output of last stage is fed back to input of first stage
- The CLR followed by PRE makes the first stage to 1 and remaining outputs are zero. Q_A is one Q_B, Q_C, Q_D are zero
- The first clock pulse produces $Q_B = 1$ and remaining outputs are zero

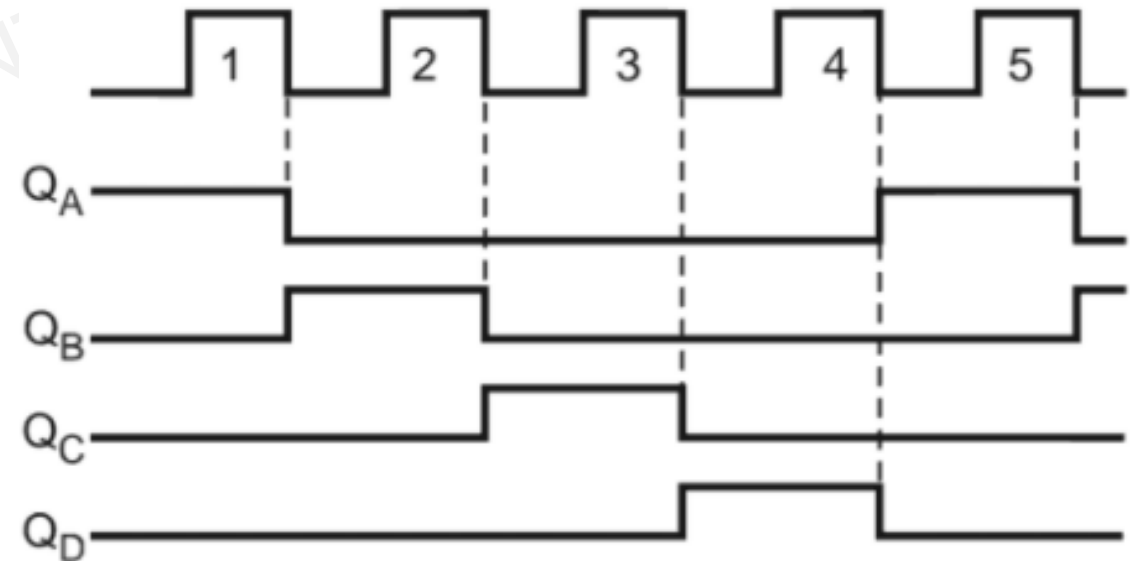


YOUTUBE : SHASTRA TECHNICAL INSTITUTE

THE RING COUNTER

- According to the clock pulses applied at the input CP, a sequence of four states is produced
- 1 is always retained in counter and simply shifted around the ring advancing one stage for each clock pulse. In this case four stages of flipflops are use
- So a sequence of four states is produced and repeated

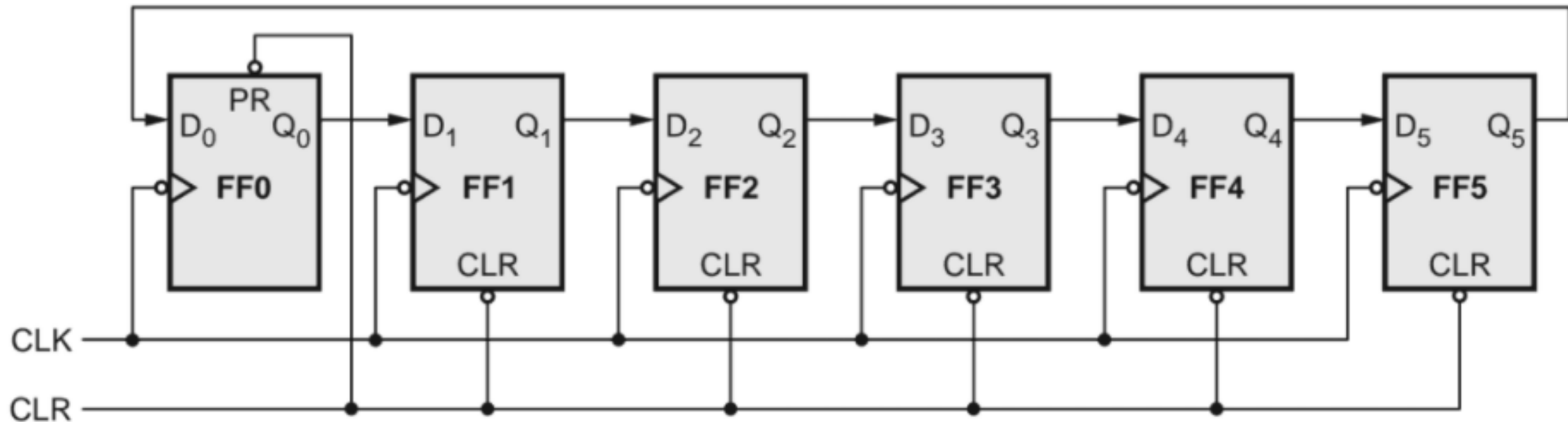
Clock pulse	Q_A	Q_B	Q_C	Q_D
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1
4	1	0	0	0



YOUTUBE : SHASTRA TECHNICAL INSTITUTE

ANOTHER EXAMPLE OF RING COUNTER

Draw a six stage ring counter and explain its operation.




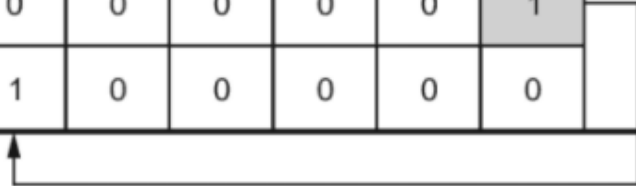
YOUTUBE : SHASTRA TECHNICAL INSTITUTE

ANOTHER EXAMPLE OF RING COUNTER

Draw a six stage ring counter and explain its operation.

Operation : The Fig. 9.5.4 shows the operation of six-stage ring counter. On preset, FF0 (flip-flop 0) is set and FF1 to FF5 are reset. After each falling edge of the clock contents of ring counter are shifted 1 bit from LSB to MSB.

CLR	CLK	Q ₀	Q ₁	Q ₂	Q ₃	Q ₄	Q ₅	Operation
	X	1	0	0	0	0	0	Preset
1	↓	0	1	0	0	0	0	'1' Bit follows a circular path to form ring counter
1	↓	0	0	1	0	0	0	
1	↓	0	0	0	1	0	0	
1	↓	0	0	0	0	1	0	
1	↓	0	0	0	0	0	1	
1	↓	1	0	0	0	0	0	



YOUTUBE : SHASTRA TECHNICAL INSTITUTE

MODULE 5 (TOPIC-5)

TWISTED RING COUNTER/ JOHNSON COUNTER

[INDEX](#)

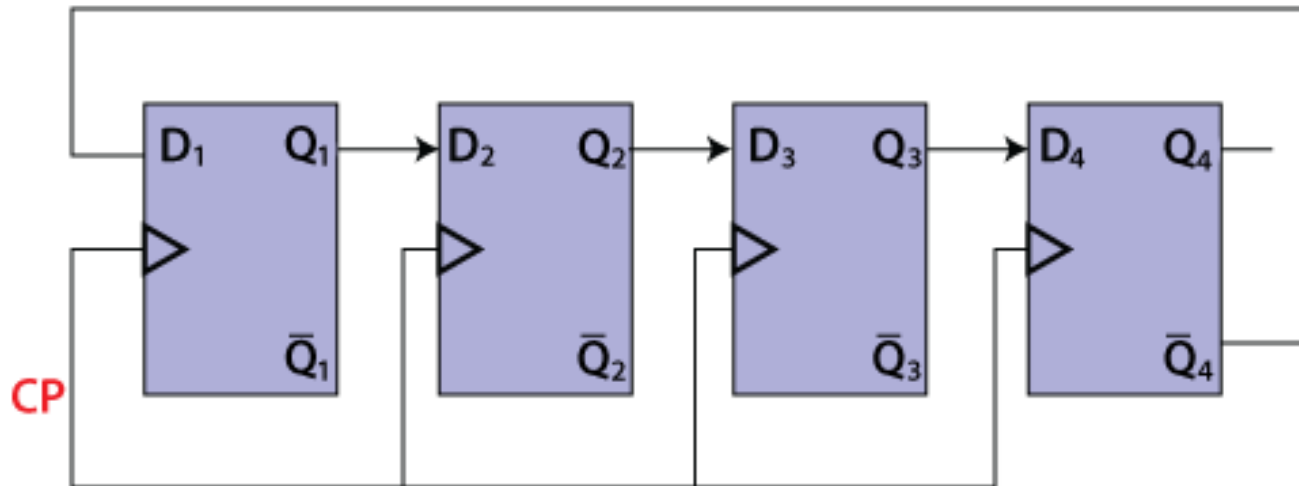
<https://youtu.be/rpZAH7XVA70>



YOUTUBE : SHASTRA TECHNICAL INSTITUTE

THE JOHNSON COUNTER

- the Johnson counter is similar to the Ring counter.
- The only difference between the Johnson counter and the ring counter is that the outcome of the last flip flop is passed to the first flip flop as an input.
- But in Johnson counter, the inverted outcome Q' of the last flip flop is passed as an input. The remaining work of the Johnson counter is the same as a ring counter. The Johnson counter is also referred to as the Creeping counter.
- Below is the diagram of the 4-bit Johnson counter. Like Ring counter, four D flip flops are used in the 4-bit Johnson counter, and the same clock pulse is passed to all the input of the flip flops.



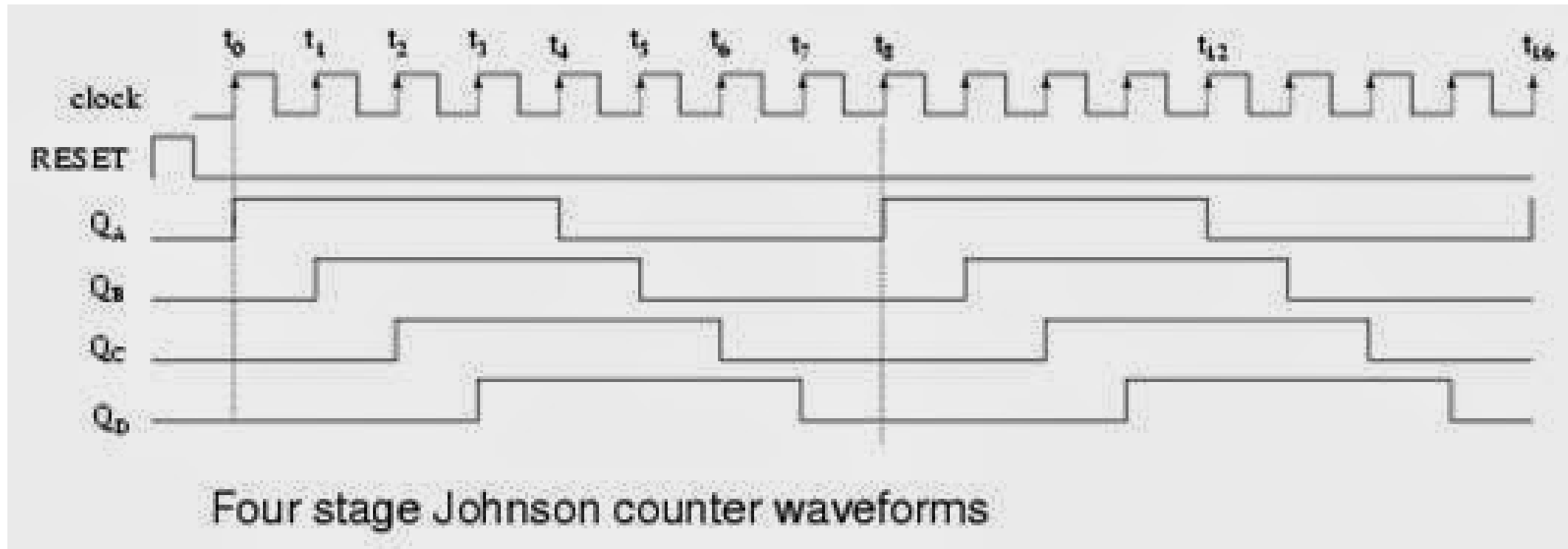
YOUTUBE : SHASTRA TECHNICAL INSTITUTE

THE JOHNSON COUNTER

- The above table state that
- The counter produces the output 0000 when there is no clock input passed(0).
- The counter produces the output 1000 when the 1st clock pulse is passed to the flip flops.
- The counter produces the output 1100 when the 2nd clock pulse is passed to the flip flops.
- The counter produces the output 1110 when the 3rd clock pulse is passed to the flip flops.
- The counter produces the output 1111 when the 4th clock pulse is passed to the flip flops.
- The counter produces the output 0111 when the 5th clock pulse is passed to the flip flops.
- The counter produces the output 0011 when the 6th clock pulse is passed to the flip flops.
- The counter produces the output 0001 when the 7th clock pulse is passed to the flip flops.

CP	Q1	Q2	Q3	Q4
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	1	1	1

TIMING DIAGRAM JOHNSON COUNTER



YOUTUBE : SHASTRA TECHNICAL INSTITUTE

MODULE 5 (TOPIC-6)

ARITHMETIC ALGORITHM FOR SIGNED MAGNITUDE NUMBER

[INDEX](#)

<https://youtu.be/zcz35pSAI4Q>



YOUTUBE : SHASTRA TECHNICAL INSTITUTE

ARITHMETIC ALGORITHM FOR SIGNED MAGNITUDE NUMBER

- Addition and Subtraction with Signed-Magnitude Data The representation of numbers in signed-magnitude is familiar because it is used in everyday arithmetic calculations. The procedure for adding or subtracting two signed binary numbers with paper and pencil is simple and straightforward.
- A review of this procedure will be helpful for deriving the hardware algorithm. We designate the magnitude of the two numbers by A and B. When the signed numbers are added or
- subtracted, we find that there are eight different conditions to consider, depending on the sign
- of the numbers and the operation performed. These conditions are listed in the first column of Table . The other columns in the table show the actual operation to be performed with the magnitude of the numbers. The last column is needed to prevent a negative zero.

ARITHMETIC ALGORITHM FOR SIGNED MAGNITUDE NUMBER

- In other words, when two equal numbers are subtracted, the result should be +0 not -0. The algorithms for addition and subtraction are derived from the table and can be stated as follows (the words inside parentheses should be used for the subtraction algorithm): Addition (subtraction) algorithm: when the signs of A and B are identical (different), add the two magnitudes and attach the sign of A to the result. When the signs of A and B are different (identical), compare the magnitudes and subtract the smaller number from the larger. Choose the sign of the result to be the same as A if $A > B$ or the complement of the sign of A if $A < B$. If the two magnitudes are equal, subtract B from A and make the sign of the result positive. The two algorithms are similar except for the sign comparison. The procedure to be followed for identical signs in the addition algorithm is the same as for different signs in the subtraction algorithm, and vice versa

ARITHMETIC ALGORITHM FOR SIGNED MAGNITUDE NUMBER

Addition and Subtraction of Signed-Magnitude Numbers

Operation	Add Magnitudes	Subtract Magnitudes		
		When $A > B$	When $A < B$	When $A = B$
$(+A) + (+B)$	$+(A + B)$			
$(+A) + (-B)$		$+(A - B)$	$-(B - A)$	$+(A - B)$
$(-A) + (+B)$		$-(A - B)$	$+(B - A)$	$+(A - B)$
$(-A) + (-B)$	$-(A + B)$			
$(+A) - (+B)$		$+(A - B)$	$-(B - A)$	$+(A - B)$
$(+A) - (-B)$	$+(A + B)$			
$(-A) - (+B)$	$-(A + B)$			
$(-A) - (-B)$		$-(A - B)$	$+(B - A)$	$+(A - B)$

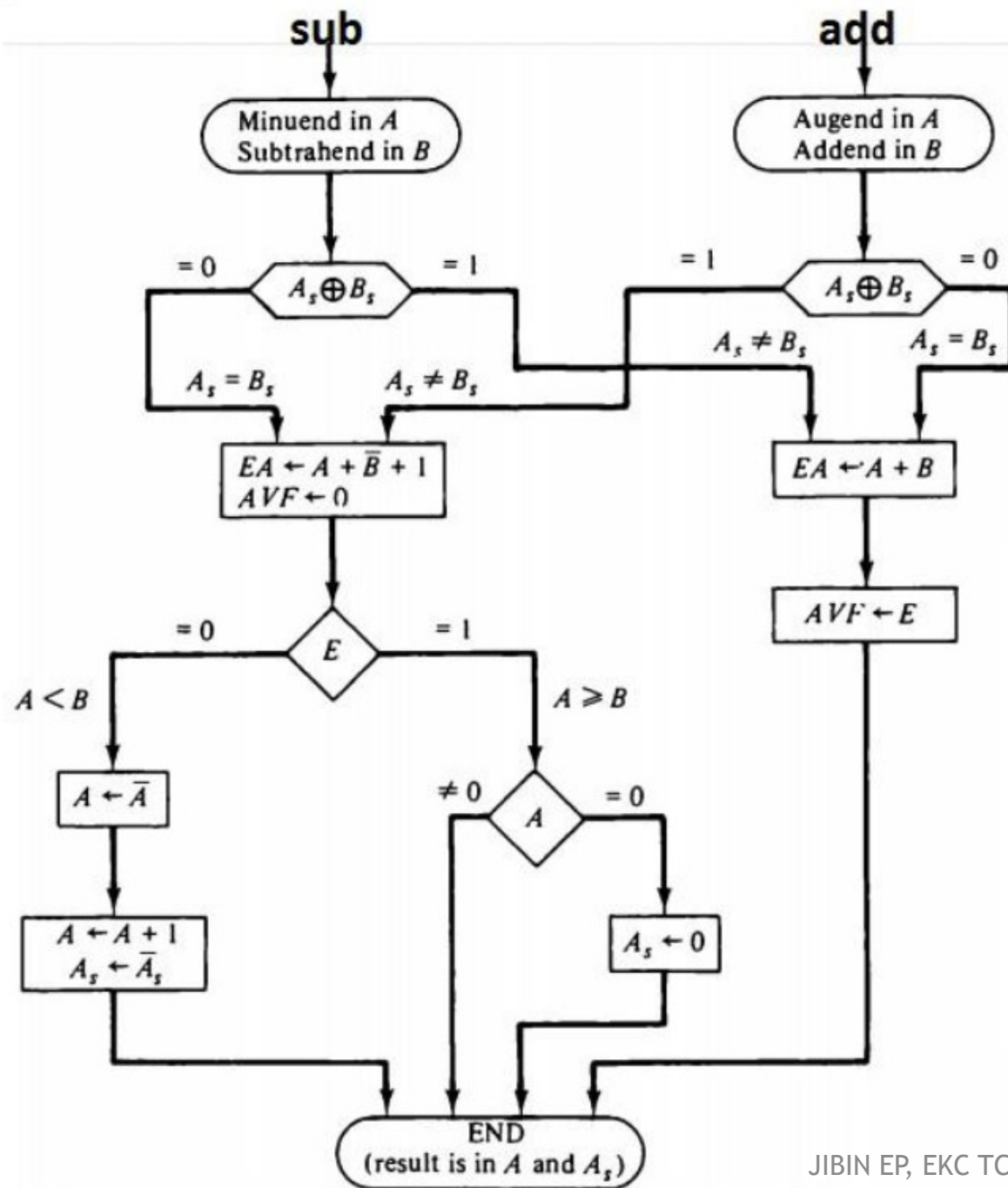
YOUTUBE : SHASTRA TECHNICAL INSTITUTE

Algorithm Explanation

The flowchart for the hardware algorithm is presented in Fig. . The two signs A, and B, are compared by an exclusive-OR gate. If the output of the gate is 0, the signs are identical; if it is 1, the signs are different. For an add operation, identical signs dictate that the magnitudes be added. For a subtract operation, different signs dictate that the magnitudes be added. The magnitudes are added with a microoperation $EA \leftarrow A + B$, where EA is a register that

combines E and A. The carry in E after the addition constitutes an overflow if it is equal to 1. The value of E is transferred into the add-overflow flip-flop AVF. The two magnitudes are subtracted if the signs are different for a subtract operation or identical for a subtract operation. The magnitudes are subtracted by adding A to the 2's complement of B. No overflow can occur if the numbers are subtracted so AVF is cleared to 0. A 1 in E indicates that $A < B$.

and the number in A is the correct result. If this number is zero, the sign A, must be made positive to avoid a negative zero. A 0 in E: indicates that $A < B$. For this case it is necessary to take the 2's complement of the value in A. This operation can be done with one microoperation $A \leftarrow A'' + 1$. However, we assume that the A register has circuits for microoperations complement and increment, so the 2's complement is obtained from these two microoperations. In other paths of the flowchart, the sign of the result is the same as the sign of A, so no change in A, is required. However, when $A < B$, the sign of the result is the complement of the original sign of A. It is then necessary to complement A, to obtain the correct sign. The final result is found in register A and its sign in A,. The value in AVF provides an overflow indication. The final value of E is immaterial.



Description

- A_s Sign of A
- B_s Sign of B
- A_s & A Accumulator
- AVF Overflow bit for $A + B$
- E Output carry for parallel adder

Addition and Subtraction of Signed-Magnitude Numbers

Operation	Add Magnitudes	Subtract Magnitudes		
		When $A > B$	When $A < B$	When $A = B$
$(+A) + (+B)$	$+(A + B)$			
$(+A) + (-B)$		$+(A - B)$	$-(B - A)$	$+(A - B)$
$(-A) + (+B)$		$-(A - B)$	$+(B - A)$	$+(A - B)$
$(-A) + (-B)$	$-(A + B)$			
$(+A) - (+B)$		$+(A - B)$	$-(B - A)$	$+(A - B)$
$(+A) - (-B)$	$+(A + B)$			
$(-A) - (+B)$	$-(A + B)$			
$(-A) - (-B)$		$-(A - B)$	$+(B - A)$	$+(A - B)$

CIAL INSTITUTE

MODULE 5 (TOPIC-7)

ARITHMETIC ALGORITHM FOR FLOATING POINT ADDITION AND SUBTRACTION

[INDEX](#)

<https://youtu.be/cgKf9l7p6EA>

YOUTUBE : SHASTRA TECHNICAL INSTITUTE



ARITHMETIC ALGORITHM FOR FLOATING POINT ADDITION AND SUBTRACTION

- During addition or subtraction, the two floating-point operands are in AC and BR. The sum or difference is formed in the AC. The algorithm can be divided into four consecutive parts:
 - 1. Check for zeros.
 - 2. Align the mantissas.
 - 3. Add or subtract the mantissas.
 - 4. Normalize the result.
- A floating-point number that is zero cannot be normalized. If this number is used during the computation, the result may also be zero. Instead of checking for zeros during the normalization process we check for zeros at the beginning and terminate the process if necessary. The alignment of the mantissas must be carried out prior to their operation. After the mantissas are added or subtracted, the result may be unnormalized. The normalization procedure ensures that the result is normalized prior to its transfer to memory. The flowchart for adding or subtracting two floating-point binary numbers is shown in Fig. 10-15. If BR is equal to zero, the operation is terminated, with the value in the AC being the result. If AC is equal to zero, we transfer the content of BR into AC and also complement its sign if the numbers are to be subtracted. If neither number is equal to zero, we proceed to align the mantissas

ARITHMETIC ALGORITHM FOR FLOATING POINT ADDITION AND SUBTRACTION

- The magnitude comparator attached to exponents a and b provides three outputs that indicate their relative magnitude. If the two exponents are equal, we go to perform the
- arithmetic operation. If the exponents are not equal, the mantissa having the smaller exponent is shifted to the right and its exponent incremented. This process is repeated until the two exponents are equal. The addition and subtraction of the two mantissas is identical to the fixed-point addition and subtraction algorithm. The magnitude part is added or subtracted depending on the operation and the signs of the two mantissas. If an overflow occurs when the magnitudes are added, it is transferred into flip-flop E . If E is equal to 1, the bit is transferred into A_1 and all other bits of A are shifted right. The exponent must be incremented to maintain the correct number. No underflow may occur in this case because the original mantissa that was not shifted during the alignment was already in a normalized position. If the magnitudes were subtracted, the result may be zero or may have an underflow. If the mantissa is zero, the entire floating-point number in the AC is made zero. Otherwise, the mantissa must have at least one bit that is equal to 1. The mantissa has an underflow if the most significant bit in position A_1 is 0. In that case, the mantissa is shifted left and the exponent decremented. The bit in A_1 is checked again and the process is repeated until it is equal to 1. When $A_1 = 1$, the mantissa is normalized and the operation is completed.

MODULE 5 (TOPIC-8)

ARITHMETIC ALGORITHM FOR ADDITION AND SUBTRACTION OF BCD NUMBERS

[INDEX](#)

<https://youtu.be/hfbvmGZ79SA>

YOUTUBE : SHASTRA TECHNICAL INSTITUTE



BCD ADDITION

1

$Sum \leq 9;$
 $Carry\ 0$

- Answer is correct
- Valid BCD

2

$Sum \leq 9;$
 $Carry\ 1$

- Invalid BCD
- Correct it by adding 6, 0110

3

$Sum > 9$

- Invalid BCD
- Correct it by adding 6, 0110

BCD SUBTRACTION USING 9'S COMPLEMENT

Take 9s complement of the number to be subtracted

1

Add it to the result using BCD Addition

2

If the result is invalid, correct it by adding 6 (0110)

3

Shift the carry to next bit

4

If End around carry is generated , add it to the result

5

BCD SUBTRACTION USING 10'S COMPLEMENT

Take 10's complement of the number to be subtracted

1

Add it to the result using BCD Addition

2

If the result is invalid, correct it by adding 6 (0110)

3

Shift the carry to next bit

4

If End around carry is generated , discard the carry.
(If number is negative, take 10's complement)

5

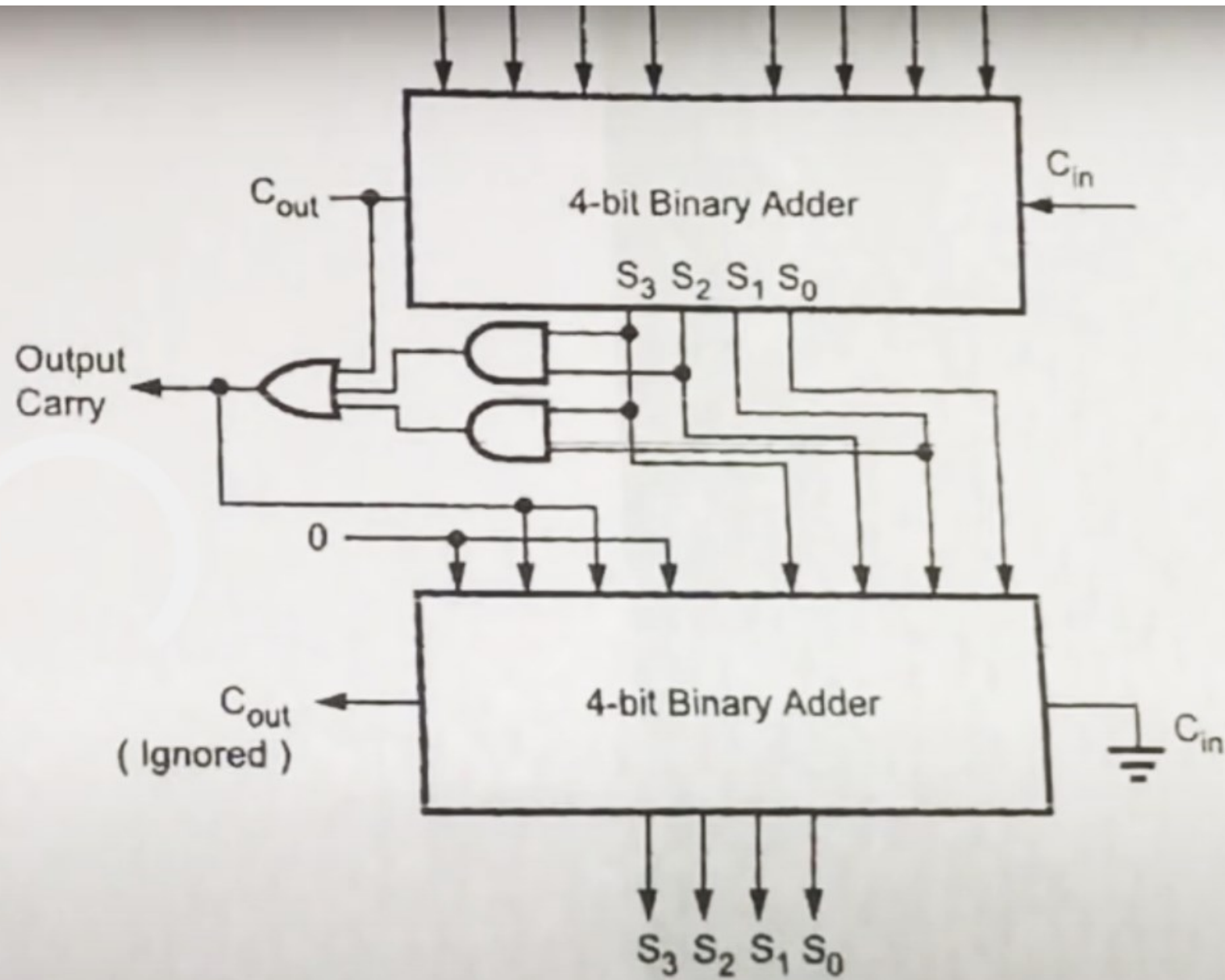


Fig. 3.32 Block diagram of BCD adder

MODULE 5 (TOPIC-9)

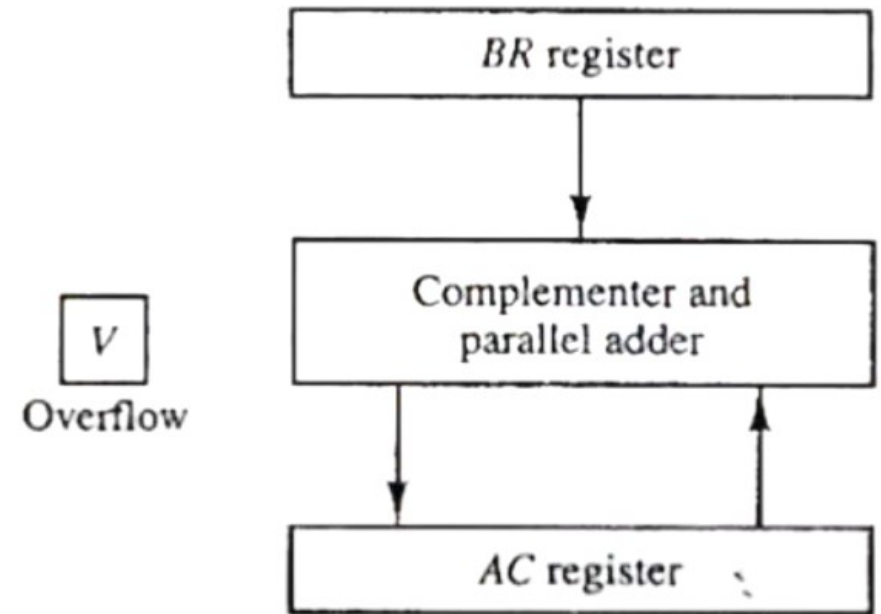
ARITHMETIC ALGORITHM FOR 2'S COMPLEMENT ARITHMETIC

[INDEX](#)

YOUTUBE : SHASTRA TECHNICAL INSTITUTE
<https://youtu.be/-2Ext37PzPU> 

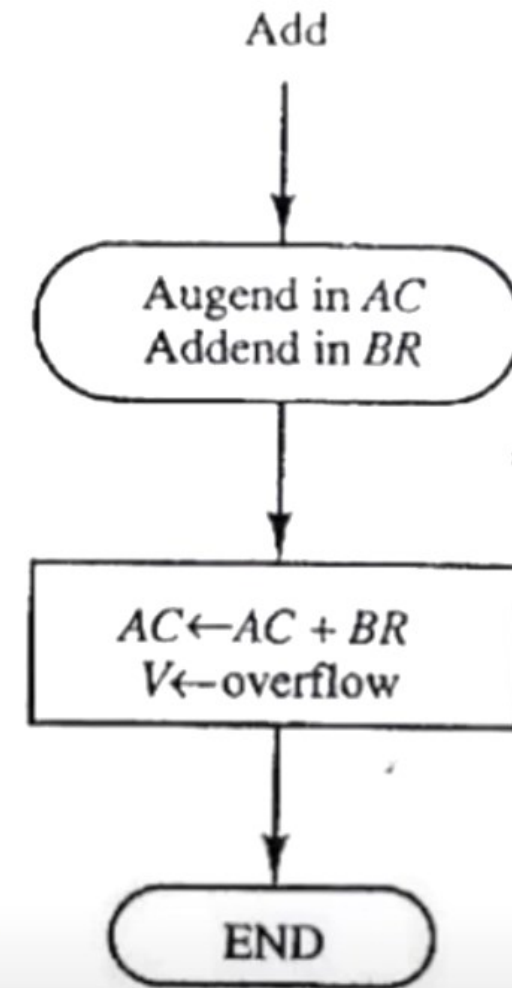
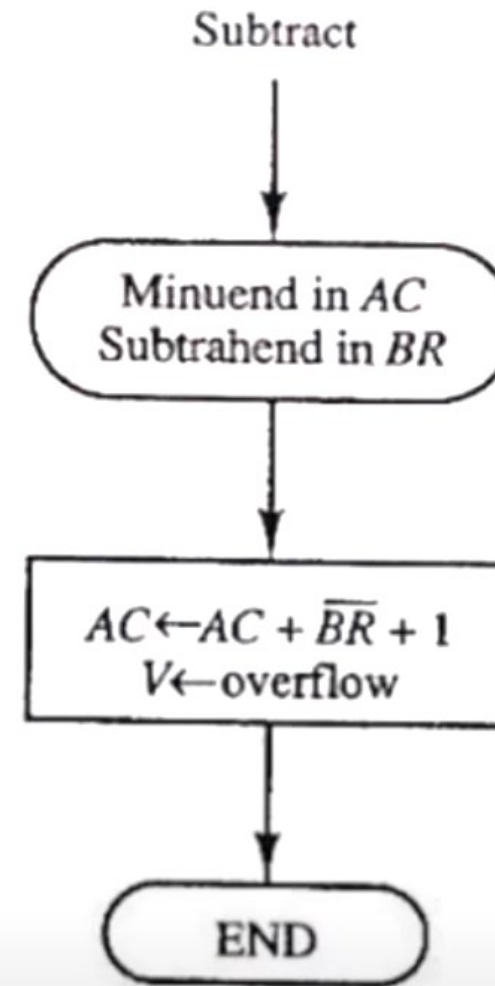
The Register Configuration

- The register configuration of hardware implementation is shown in figure
- We name A Register AC (Accumulator) and B Register BR
- The Leftmost bit AC and BR represent the sign bit of the numbers
- The sign bit added or subtracted with together with other bits in complementor and parallel adder
- The overflow flipflop V is set to 1 if there is overflow. The output carry in this case is discarded



The Addition and Subtraction Algorithm

- The Sum is obtained by adding the contents of AC and BR
- The Overflow bit V is set to 1 if the exclusive OR of the last two carries is 1, and it is cleared to 0 otherwise
- The Subtraction Operation is accomplished by adding the content of AC with 2's complement of BR
- An Overflow must be checked during this operation because two numbers added could have the same sign



MODULE 5 (TOPIC-10)

PROGRAMMABLE LOGIC DEVICES : READ ONLY MEMORY

[INDEX](#)

<https://youtu.be/SXhbEO1ZS4E>

YOUTUBE : SHASTRA TECHNICAL INSTITUTE



Programmable Logic Devices (PLD)

- Programmable Logic Devices are used to **Implement logic functions**
- The Main Advantage of Programmable Logic Devices approach is that PLD Can be **easily configurable** by individual user for specific application

Classification of PLD

- According to the architecture, complexity and flexibility in programming PLDs are classified as

PROM

- Programmable Read only Memory

PLA

- Programmable Logic Array

PAL

- Programmable Array Logic

FPGA

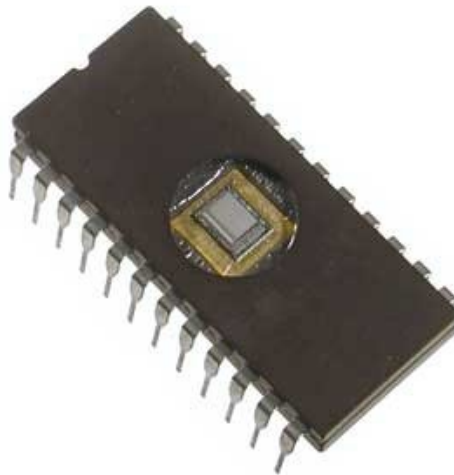
- Field Programmable Gate Array

CPLD

- Complex Programmable Logic Devices

ROM (READ ONLY MEMORY)

- ROM stands for Read Only Memory.
- The memory from which we can only read but cannot write on it.
- This type of memory is **Non-volatile**. **non-volatile storage** is a type of computer **memory** that can retain stored information even after power is removed.
- The information is stored permanently in such memories during manufacture.



CLASSIFICATION OF ROM

PROM

- Programmable Read Only Memory
- PROM is read-only memory that can be modified only once by a user
- It can be programmed only once and is not erasable.

EPROM

- Erasable and Programmable Read Only Memory
- EPROM can be erased by exposing it to ultra-violet light for a duration of up to 40 minutes.

EEPROM

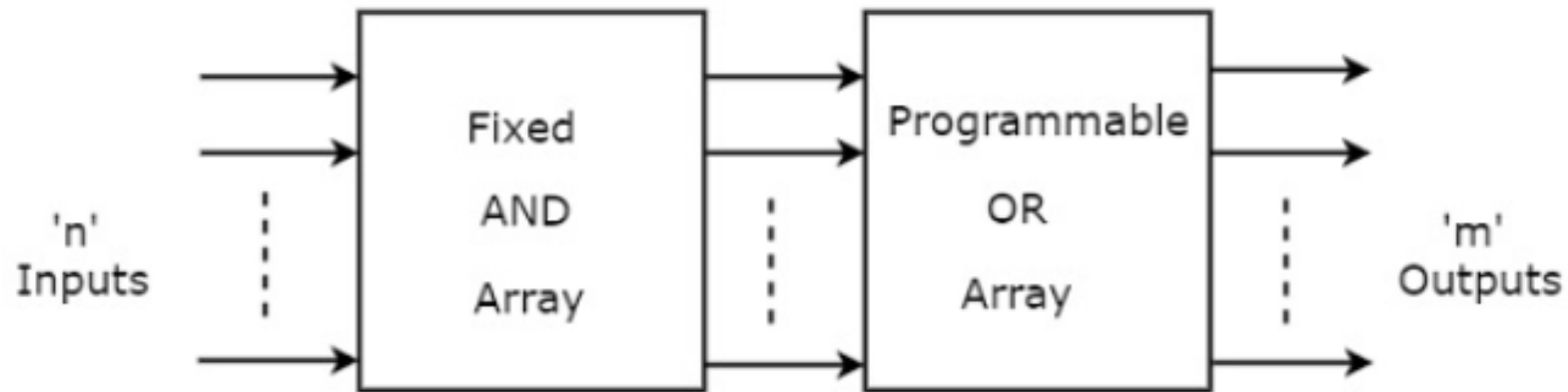
- Electrically Erasable and Programmable Read Only Memory
- EEPROM is programmed and erased electrically.
- It can be erased and reprogrammed about ten thousand times.
- Both erasing and programming take about 4 to 10 ms

PROM (PROGRAMMABLE READ ONLY MEMORY)

- Read Only Memory ROM is a memory device, which stores the binary information permanently.
- That means, we can't **change that stored information by any means later.**
- If the ROM has programmable feature, then it is called as **Programmable ROM PROM.**
- The user has the flexibility to program the binary information electrically once by using PROM programmer.

PROM (PROGRAMMABLE READ ONLY MEMORY)

- PROM is a programmable logic device that has fixed AND array & Programmable OR array.
- The **block diagram** of PROM is shown in the following figure.



EXAMPLE OF PROM

Implement the Boolean function

- $A(X,Y,Z) = \sum m(5,6,7)$

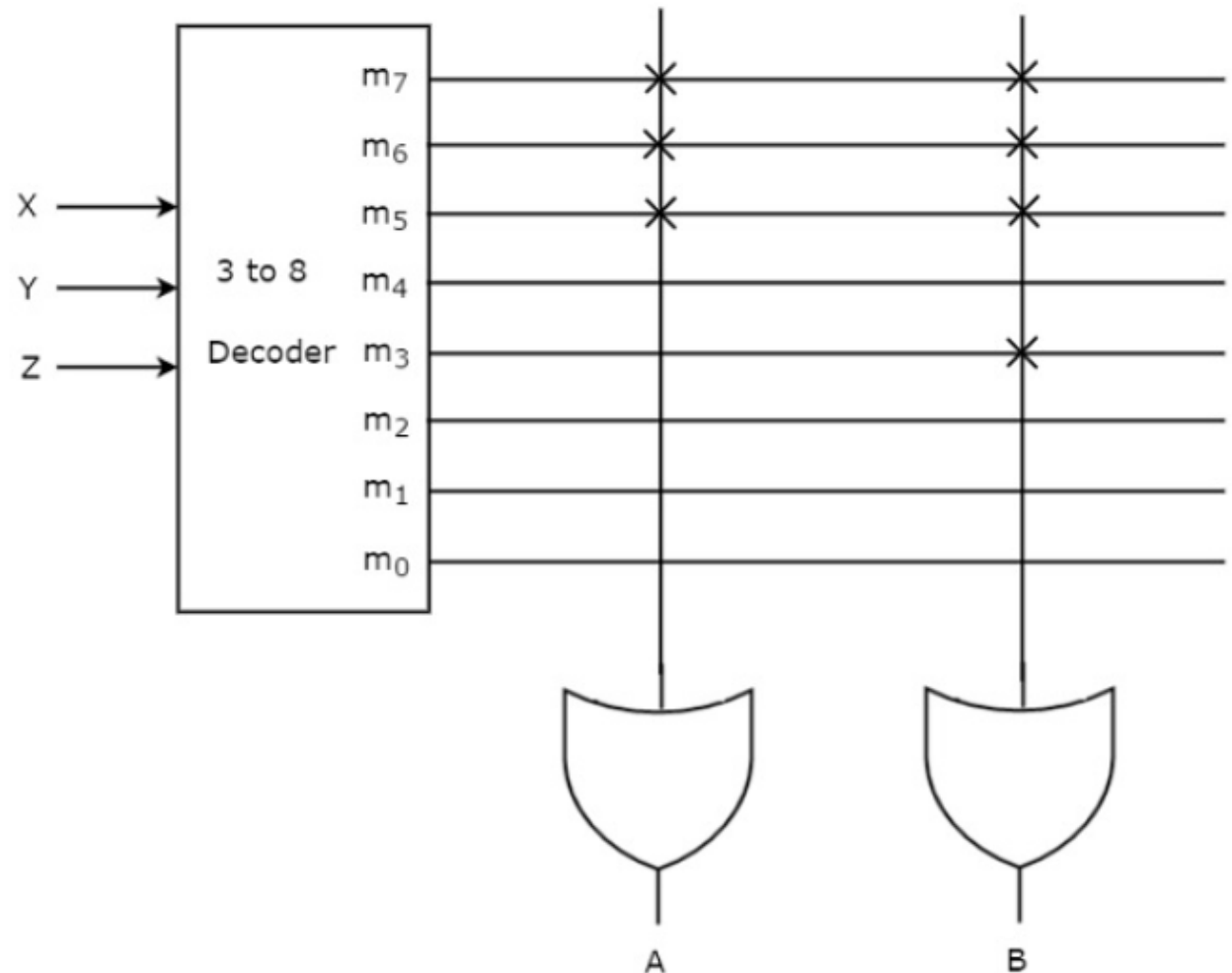
$$B(X,Y,Z) = \sum m(3,5,6,7)$$

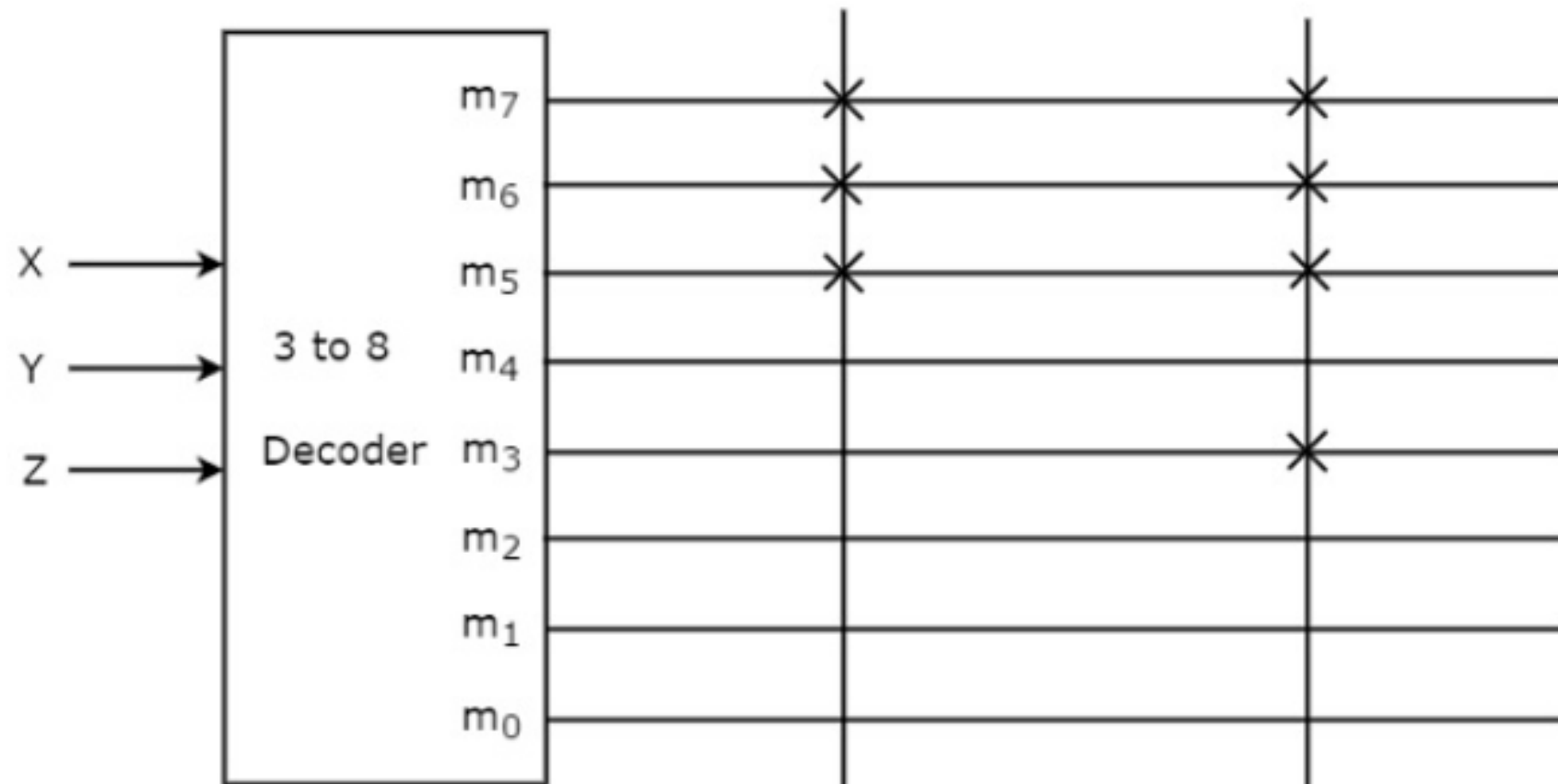
The given two functions are in sum of min terms form and each function is having three variables X, Y & Z.

So, we require a 3 to 8 decoder and two programmable OR gates for producing these two functions.

The corresponding **PROM** is shown in the following figure.

Here, 3 to 8 decoder generates eight min terms. The two programmable OR gates have the access of all these min terms. But, only the required min terms are programmed in order to produce the respective Boolean functions by each OR gate. The symbol 'X' is used for programmable connections.





- $A(X,Y,Z)=\sum m(5,6,7)$
- $B(X,Y,Z)=\sum m(3,5,6,7)$



MODULE 5 (TOPIC-11)

PROGRAMMABLE LOGIC ARRAY

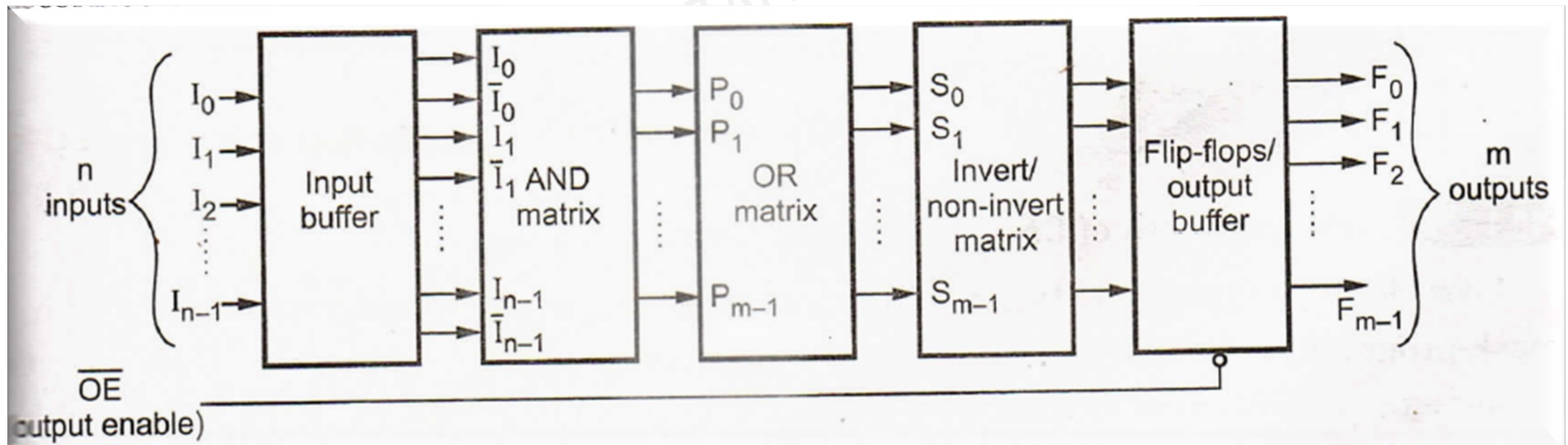
[INDEX](#)

<https://youtu.be/PdpYb16mEaw>
<https://youtu.be/1aHDDJEUg7g>



PROGRAMMABLE LOGIC ARRAY

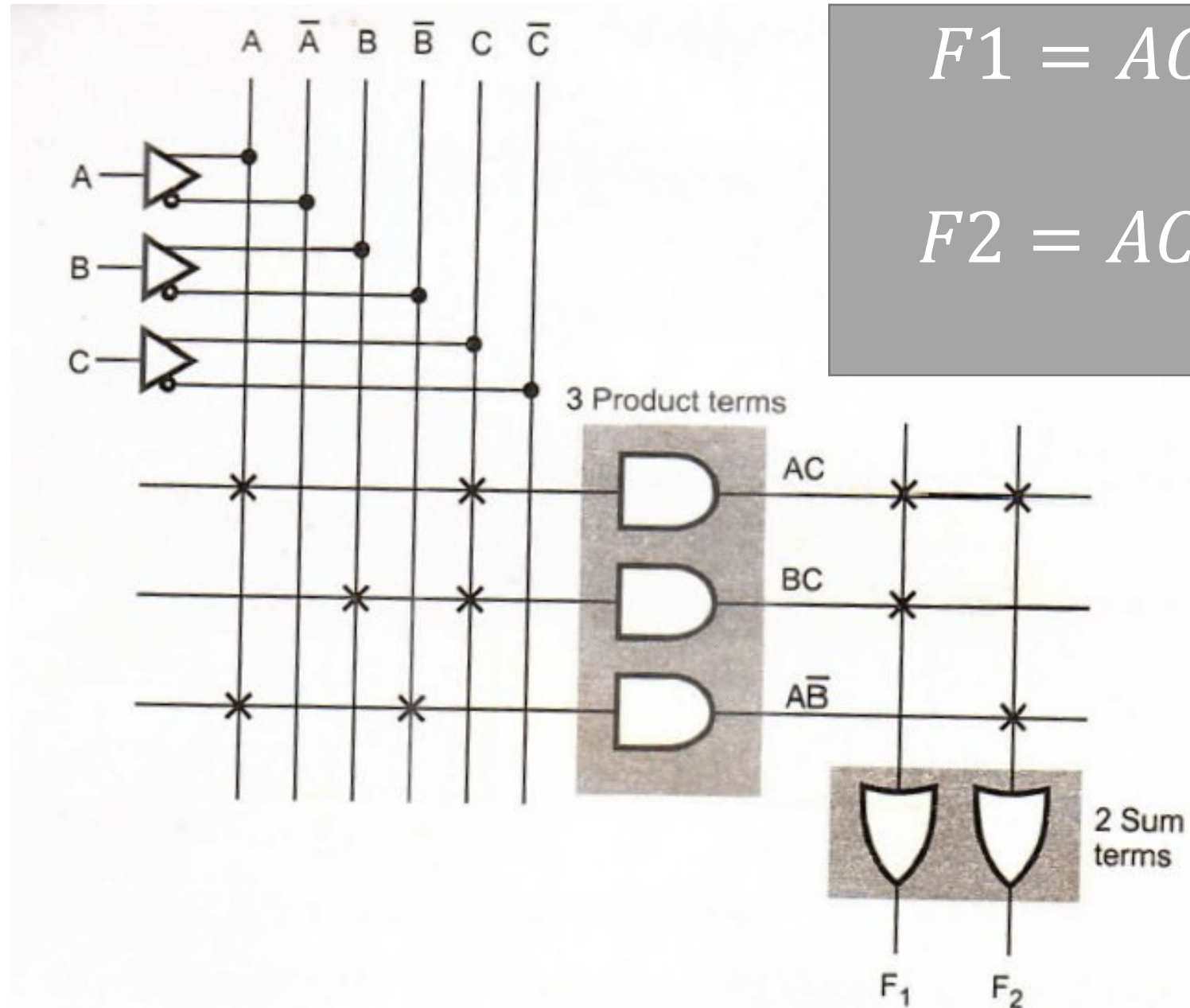
- PLA is a programmable logic device that has both Programmable AND array & Programmable OR array. Hence, it is the most flexible PLD.
- It Consist of n inputs, output buffer, wit m outputs, m product terms, m sum terms, input and output buffers



PROGRAMMABLE LOGIC ARRAY

- The product term constitutes group of m AND Gates, and sum term constitute a group of m OR Gates
- The Fuses are inserted between all n inputs and their complement values to each of the AND Gates
- Fuses are also provided between the output of the AND Gate and Inputs or OR Gate
- The Third set of fuses in the output inverters allows the output functions to be generated either in the AND-OR form or in the AND-OR-Invert form

IMPLEMENTATION EXAMPLES OF PLA



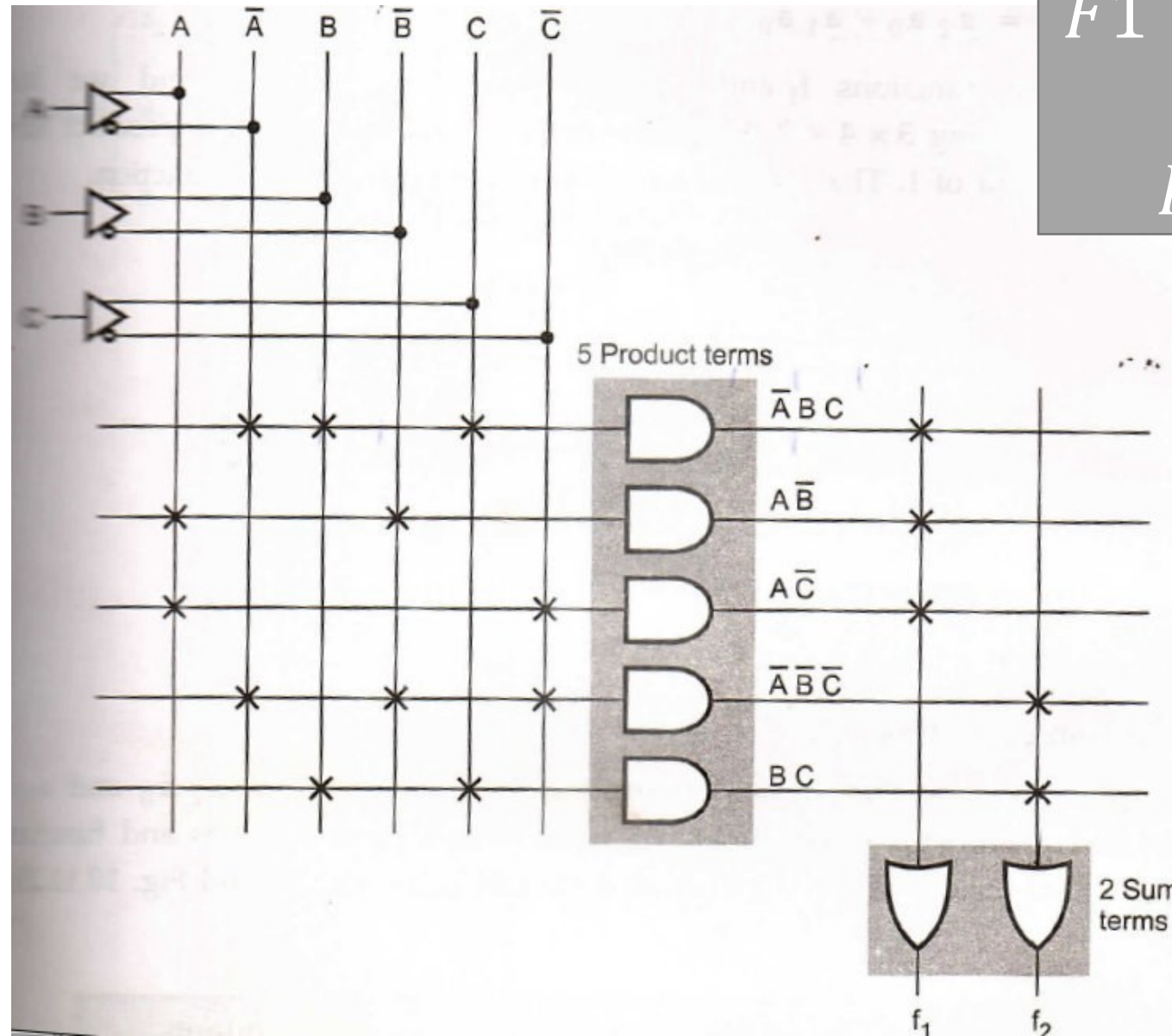
$$F_1 = AC + BC$$

$$F_2 = AC + AB'$$

IMPLEMENTATION EXAMPLES OF PLA

$$F1 = A'BC + AB' + AC'$$

$$F2 = A'B'C' + BC$$



PREPARED BY



Jibin EP

**SHAstra
TECHNICAL
INSTITUTE**

KTU | PSC | UPSC



Jibin EP

Assistant Professor

Department of Electronics and Communication
Eranad Knowledge City Technical Campus Manjeri

Contact : 9633908979, 70121716607

Email : epjibin@gmail.com, jibin@ekc.edu.in



TECHNICAL INSTITUTE

<https://www.youtube.com/ShastraTechnicalInstitute>