Design of a Synchronous Mod-6 Counter using D flip-flops

Step 1 & 2 : remains the same.

D	Q	Q^+
0	0	0
0	1	0
1	0	1
1	1	1

Step : 3 Excitation Table.

Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

$\Rightarrow$ Whatever is the next state $Q_{n+1} = D$.

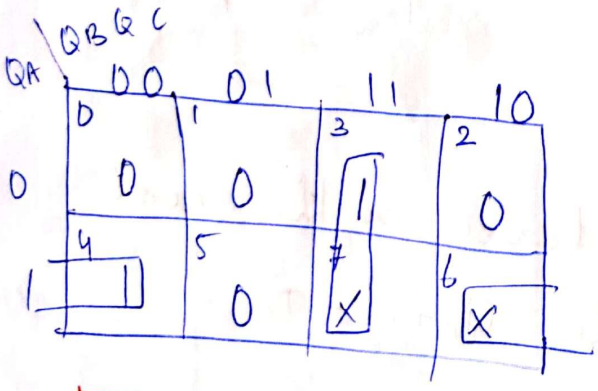
Q	Q^+	D
0	0	0
1	0	1
0	1	0
1	1	1

Step 4: Transition Table

P.S			N.S			I/P's of FF		
Q_A	Q_B	Q_C	Q_{A+1}	Q_{B+1}	Q_{C+1}	D_A	D_B	D_C
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	0
0	1	0	0	1	1	0	1	1
0	1	1	1	0	0	1	0	0
1	0	0	1	0	1	1	0	1
1	0	1	0	0	0	0	0	0
1	1	0	x	x	x	x	x	x
1	1	1	x	x	x	x	x	x

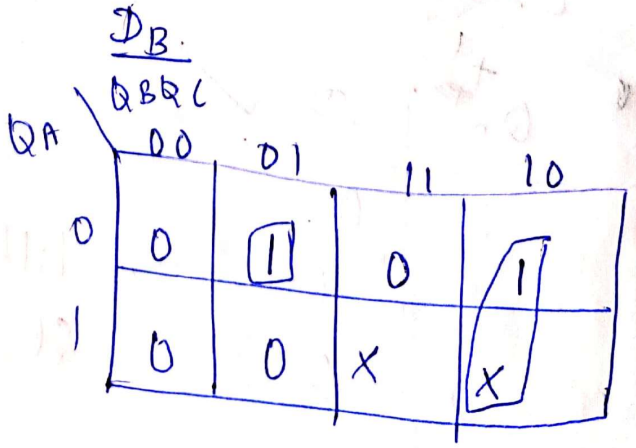
Step 5:- K-MAP

D_A



$$D_A = Q_B Q_C + Q_A \overline{Q_C}$$

D_B



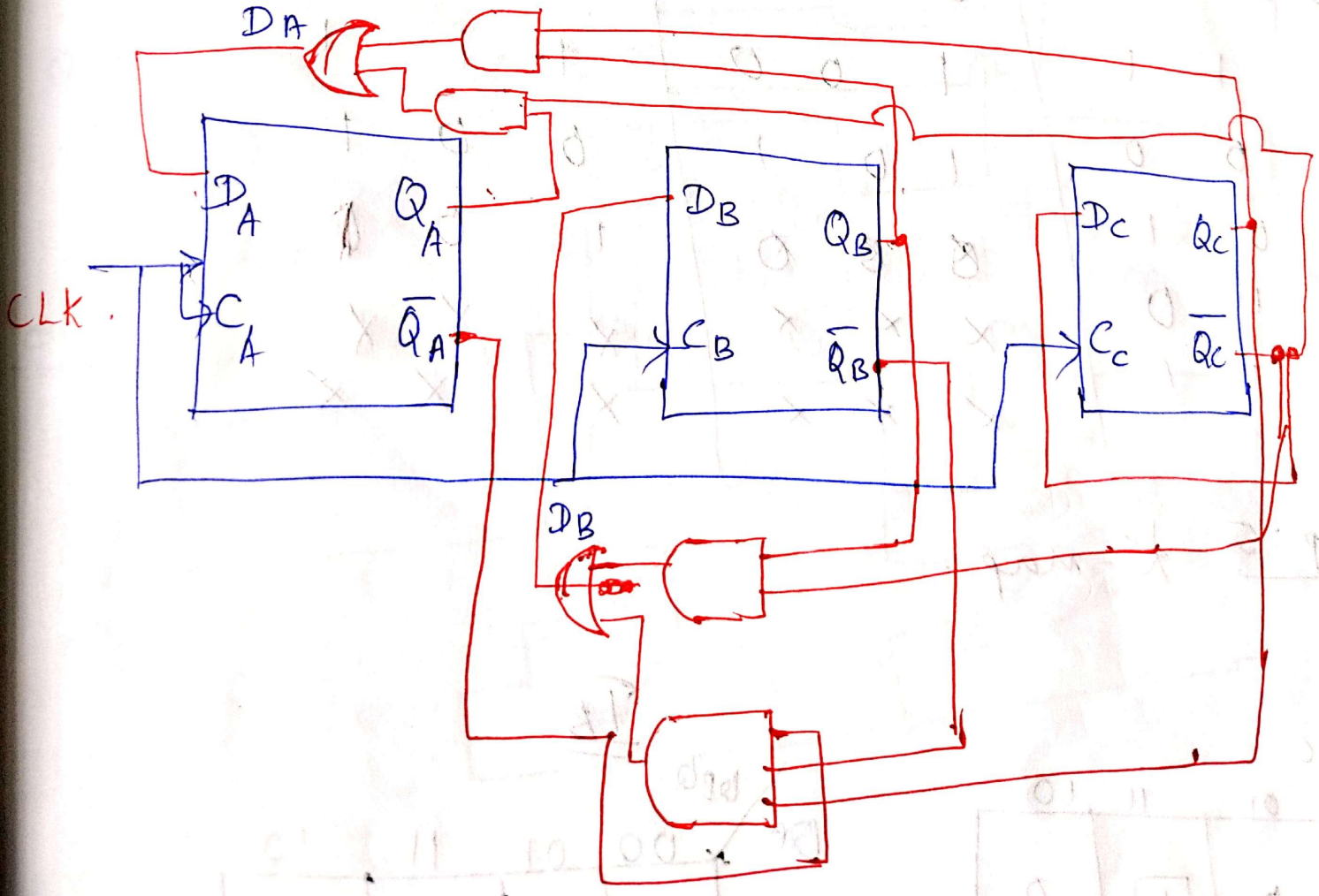
$$D_B = Q_B \overline{Q_C} + \overline{Q_A} Q_B Q_C$$

D_C

Q_A \ Q_BQ_C	00	01	11	10
0	1	0	0	1
1	1	0	x	x

$$D_C = \overline{Q_C}$$

Step 6 : Logic Diagram

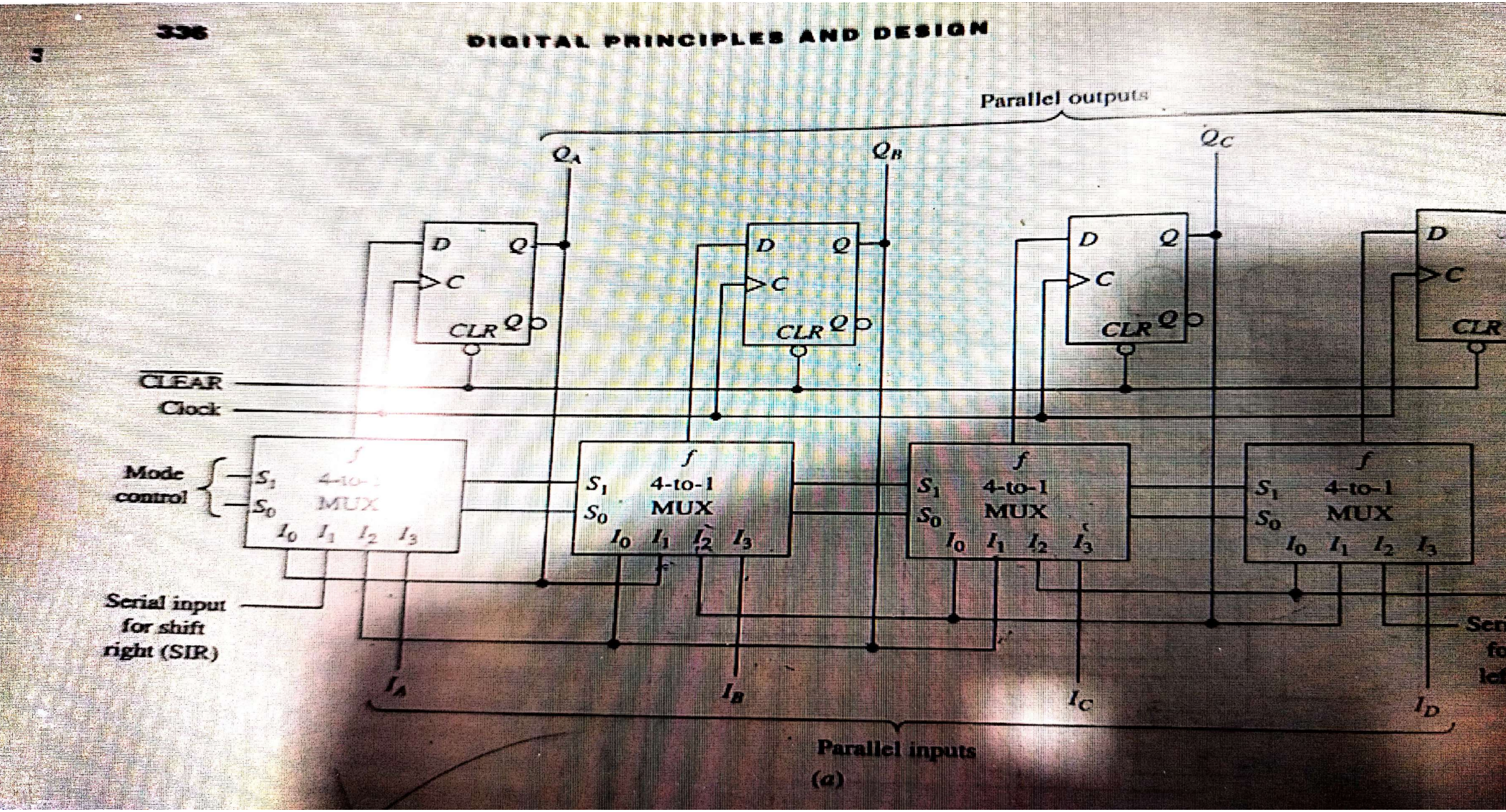


Bidirectional Shift Register :-

- The second general classification of shift registers consists of bidirectional shift register.
- These types of registers are capable of shifting their contents either left or right depending upon the signals present on appropriate control i/p lines.
- This register is also known as universal shift register.
- Depending upon the signal values on the select lines of the multiplexers, i.e the mode control lines, the register can retain its current state, shift right, shift left or be loaded in parallel.
- These operations are based on the occurrence of a positive edge on the clock line.
- The register is cleared asynchronously if a logic 0 is applied to the line labelled $\overline{\text{CLEAR}}$.
- When the select lines S_1S_0 of the multiplexers are 0 1 the register performs shift right operation.

- The i/p to the leftmost D flipflop is the signal
ter the serial-input for-shift-right line, the i/p to th
second leftmost D flipflop is the o/p of the leftmost
D flip-flop, the i/p to the third leftmost D flip
flop is the o/p of the second leftmost D flip-flop
the i/p to the fourth leftmost D flip flop is the o/
of the third leftmost D flip-flop.

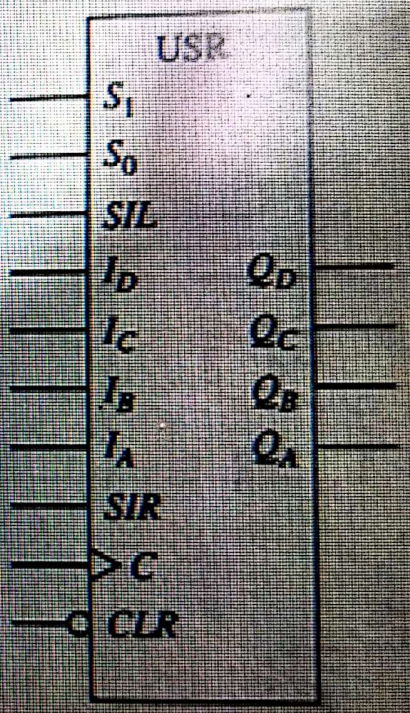
- Upon the occurence of the positive-edge signa
on the clock line, the register shift its conte
one position to the right (refer fig 6.28, 6.2
from


Parallel outputs
Q_A
Q_B
Q_C
D
C
CLR
Q
$\overline{CLEAR}$
Clock
Mode control
S_1
S_0
f
4-to-1 MUX
I_0 I_1 I_2 I_3
Serial input for shift right (SIR)
I_A
I_B
I_C
I_D
Parallel inputs

(a)

Select lines S_1	S_0	Register operation
0	0	Hold
0	1	Shift right
1	0	Shift left
1	1	Parallel load

(b)



(c)

Figure 6.29 Universal shift register. (*a*) Logic diagram. (b) Mode control. (c) Symbol.

Verilog Data Types

The several data types are nets, registers, vectors, integer, real, parameters and arrays.

Nets:

It is declared by the predefined word wire.

Nets have values that change continuously by the circuits that are driving them.

Verilog Data Types

- Nets:

  It is declared by the predefined word **wire**.

  It supports 4 values

Value	Definition
0	Logic 0 (false)
1	Logic 1 (true)
X	unknown
z	High impedance

Example:

```
wire sum;
wire S1 = 1'b0;
```

Verilog Data Types

Registers:

- It store values until they are updated.
- It is a data storage elements.
- It is declared by the predefined word reg.
- It supports 4 values of register.

Value	Definition
0	Logic 0 (false)
1	Logic 1 (true)
X	unknown
z	High impedance

Verilog Data Types

- Vectors:
- Vectors are multiple bits
- Register or net can be declared as a vector.
- Vectors are declared by brackets

Examples:

Wire [3:0]a=4'b1010;

Reg[7:0]total=8'd12;

Verilog Data Types

Integers:

- It is defined by a predefined word integer.

Real:

- Real (floating-point) numbers are declared with predefined word r
- Example:

1. 2.4,56.3 and 5e12(5*10^12)

Verilog Data Types

- Vectors:
- Vectors are multiple bits.
- A register or a net can be declared as a vector.
- Vectors are declared by brackets [ ].
- Example:
- Wire [3:0] a = 4'b1010
- Reg [7: 0] total = 8'd12;
- First statement declares the net a. It has 4 bits and the values are 1010

Verilog Data Types

Parameters:

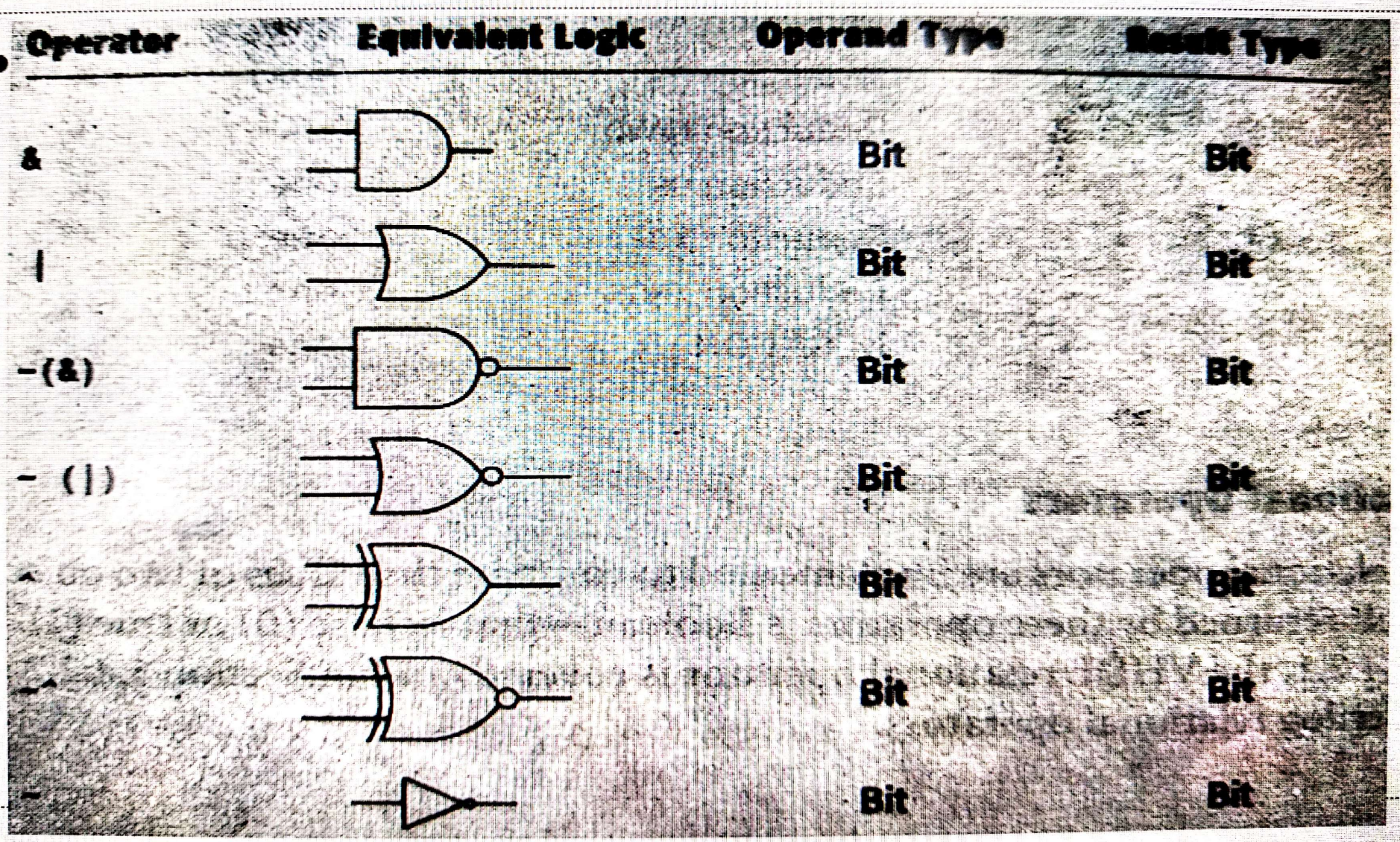
- It represents the global constants.
- It is declared by the predefined word parameter.
- Example:

```
Module compr(X,Y,Xgty,Xlty,Xegy)
Parameter N=3;
Input [N:0] X,Y;
Output Xgty,Xlty,Xegy;
Wire [N:0] sum, yb;
```

OPERATORS

- Logical: AND, OR and XOR
- Relational: Express relation between the objects. It includes equality, Inequality, less than, less than or equal, greater than or greater than or equal.
- Arithmetic
- Shift: To move the bits of an object in a certain direction, right or left

Verilog Bitwise Logical Operators



Verilog Boolean logical operators

- Opeartor: &&, ||
- Operation: AND, OR
- Number of operands: two
- These operators operate on two operands and the result is false (0) and true(1)
- !x… ! Negation operator

TABLE 1.4 Verilog Reduction Logical Operators

Operator	Operation	Number of Operands
&	Reduction AND	one
|	Reduction OR	one
~&	Reduction NAND	one
~|	Reduction OR	one
^	Reduction XOR	one
~^	Reduction XNOR	one
!	NEGATION	one

Relational Operators

- Compare the values of two objects
- These operators operate on two operands and the result is false (0) and true(1)

Click to add title

TABLE 1.6 Verilog Relational Operators

Operator	Description	Result Type
==	Equality	0,1,x
!=	Inequality	0,1,x
===	Equality inclusive	0,1
!==	Inequality inclusive	0,1
<	Less than	0,1,x
<=	Less than or equal	0,1,x
>	Greater than	0,1,x
>=	Greater than or equal	0,1,x

Verilog Arithmetic Operators

TABLE 1.8 Verilog Arithmetic Operators

Operator	Description	A or B Type	Y Type
+	Addition A + B	A numeric B numeric	numeric
-	Subtraction A – B	A numeric B numeric	numeric
*	Multiplication A ? B	A numeric B numeric	numeric
/	Division A + B	A numeric B numeric	numeric
%	Modulus A % B	A numeric, not real B numeric, not real	numeric, not real
**	Exponent A ** B	A numeric B numeric	numeric
{,}	Concatenation {A , B}	A numeric or array B numeric or array	Same as A

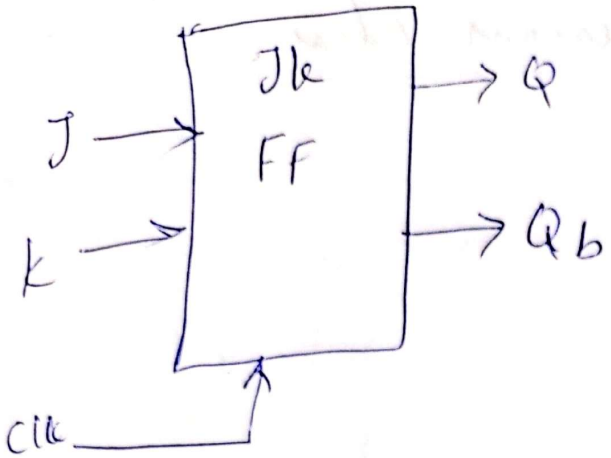
Verilog Shift Operators

1.4.4.2 Verilog Shift Operators

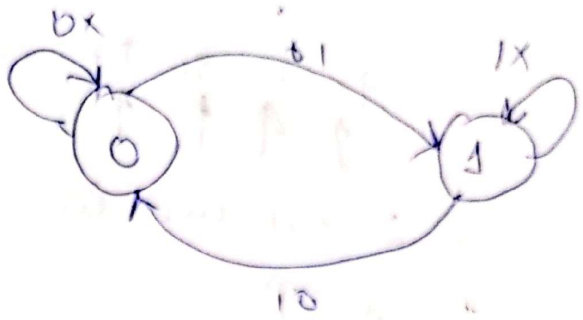
Verilog has the basic shift operators. Shift operators are unary operators; they o
erate on a single operand. To understand the function of these operators, assur
operand A is the 4-bit vector 1110. Table 1.10 shows the Verilog shift operators
they apply to operand A.

TABLE 1.10 Verilog Shift Operators

Operation	Description	Operand A Before Shift	Operand A After Shift
A << 1	Shift A one position left logical	1110	110x
A << 2	Shift A two positions left logical	1110	10xx
A >> 1	Shift A one position right logical	1110	x111
A >> 2	Shift A two positions right logical	1110	xx11



Logic Symbol



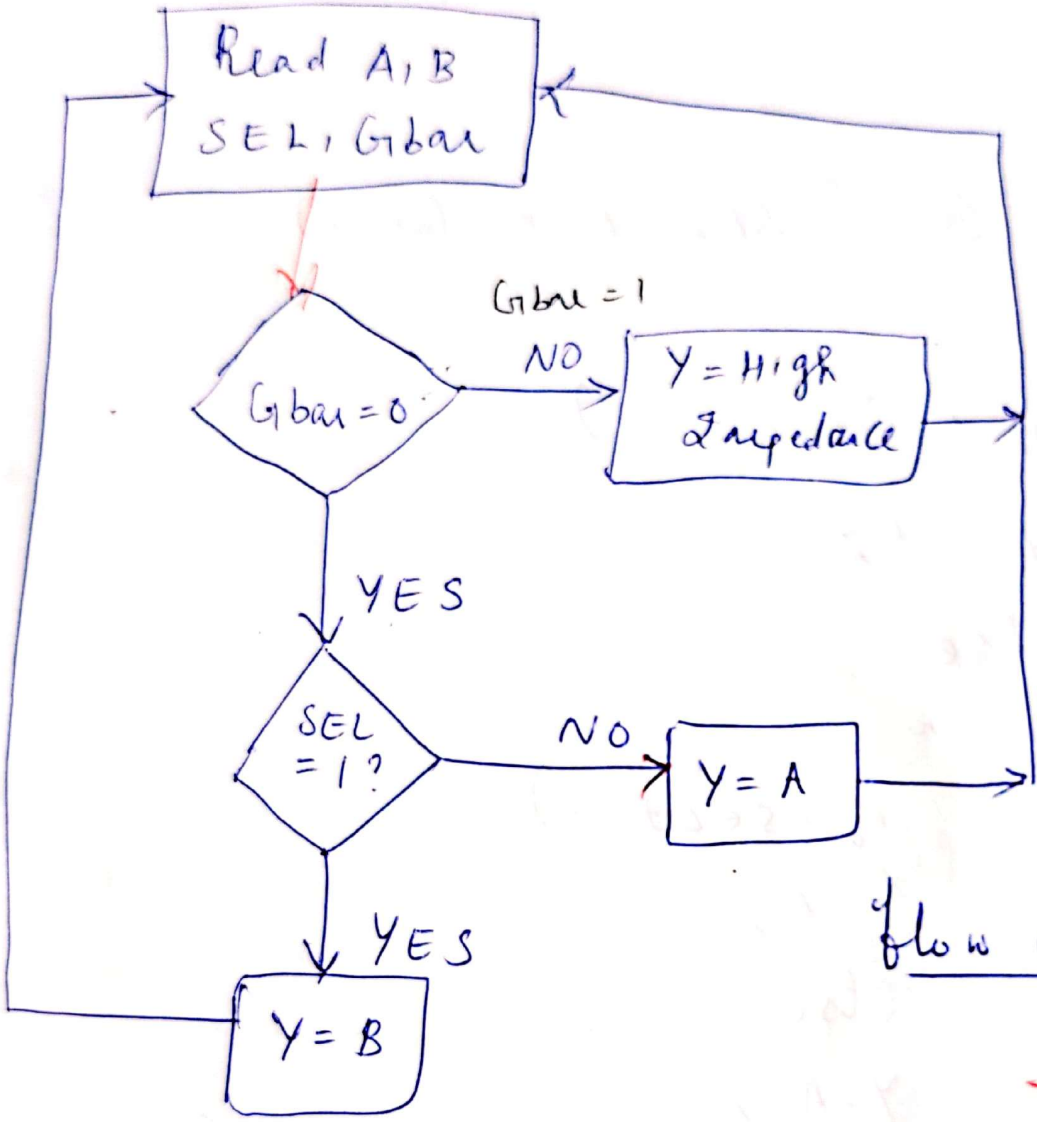
State diagram

Verilog:

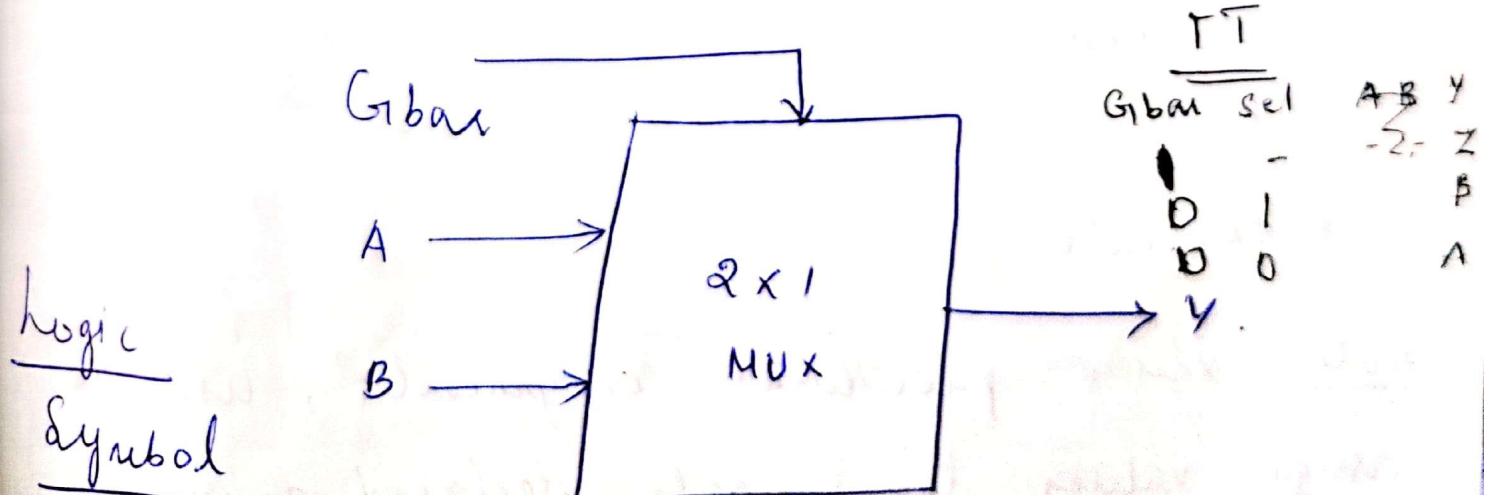
```
Module Jk-Ff ( Jk, clk, q, qb)
 input [1:0] Jk;
 input clk;
 Output q, qb;
 reg q, qb;
 always @ ( posedge clk)
 begin
 case (Jk)
      2'd0 : q = q;
      2'd1 : q = 0;
      2'd2 : q = 1;
      2'd3 : q = ~q;
```

Behavioral Description of a 2x1 MUX With Tristate O/P

The behaviour is described by the flow chart



flow chart of 2x1 MUX



Logic Symbol

TT

Gbar	Sel	Y
1	-	Z
0	1	B
0	0	A

Verilog 2x1 MUX using If-Else

```
Module MUX 2x1 ( A, B, SEL, Gbar, Y);
Input A, B, SEL, Gbar;
O/p Y;
reg Y;
always @ ( SEL, A, B, Gbar)
begin
      if ( Gbar == 1)
      Y = 1'bz
      else
      begin
            if (SEL != 1)
            Y = B;
            else
            Y = A;
      end
end
endmodule
```

Note:- This a procedural assignments, used to

Verilog 2x1 MUX using else-if

```
module MUXBH ( A, B, SEL, Gbar, Y);
input A, B, SEL, Gbar;
Output Y;
reg Y;
always @ ( SEL, A, B, Gbar)
begin
   if (Gbar == 0 & SEL == 1)
   begin
        Y = B;
   end
   else if ( Gbar == 0 & SEL == 0)
    Y = A;
    else
    Y = 1'bz ; // Y is assigned to high impedance
    end
endmodule.
```

(Simulation waveform from textbook).

Verilog Behavioral Description

```
module half_add (I1, I2, O1, O2);
input I1, I2;
output O1, O2;
reg O1, O2;
always @(I1, I2)
//The above abatement is always
//The module consists of always construct
begin
    #10 O1 = I1 ^ I2;
    #10 O2 = I1& I2;
end
endmodule
```

Example 2

Verilog Structural Description

```
module system(a, b, sum, cout);
input a, b;
output sum, cout;
    xor X1(sum, a, b);
//The above statement is EXCLUSIVE-OR gate
    and a1(cout, a, b);
//The above statement is AND gate
endmodule
```

Example 3

Verilog Data-Flow Description

```
module halfadder (a, b, s, c);
input a;
input b;
output s;
output c;
      assign s = a ^ b;
      assign c = a & b;
/* The module has no always, gat
    tranif0, tran, or tranif0*/
endmodule
```

Shift Registers:

- Registers that are capable of moving information positionwise upon the occurrence of a clock signal is called Shift Registers.

- These registers are normally classified by whether they can move the information in one or two directions, i.e unidirectional or bidirectional.

- The manner in which the informations are entered into and outputted from a register is another way in which they are categorized.

- There are two basic ways in which these transfers are done:
  Serially or in parallel.

- When information is transferred in parallel, all the 0-1 symbols that comprise the information are handled simultaneously in a single unit of time. Such information transfer requires as many lines as symbols being transferred.

- The Serial handling of information

Shift Registers

- Registers that are capable of moving information positionwise upon the occurrence of a clock signal is called Shift Registers.

- These registers are normally classified by whether they can move the information in one or two directions, i.e unidirectional or bidirectional.

- The manner in which the informations are entered into and outputted from a register is another way in which they are categorized.

- There are two basic ways in which these transfers are done:
  Serially or in parallel.

- When information is transferred in parallel, all the 0-1 symbols that comprise the information are handled simultaneously in a single unit of time. Such information transfer requires as many lines as symbols being transferred.

- The serial handling of information

of the information in a time sequence. It requires only a single line to perform the transfer.

- Thus, there are four possible ways registers can transfer information:

1). Serial-in / Serial out.

2). Serial-in/ Parallel out.

3). Parallel-in/ Parallel out.

4). Parallel-in / Serial out.

Serial-in, Serial-Out Unidirectional Shift register

- It constructed from positive edge-triggered D flip-flops.
- The Q o/p of each flip flop is connected to the D i/p of the next flip flop.
- The control i/p's of all the flip flops are connected together to a common synchronizing signal called clock.
- Thus, upon the occurence of a positive edge of the clock signal, the content of each flip flop is shifted one position to the right.
- The content of the left most flip flop

Signal value on the serial-data-in line and the content of the rightmost flip-flop prior to the clock signal is lost.

- The output from the shift register occurs at the rightmost flip-flop on the serial-data-out line.

-

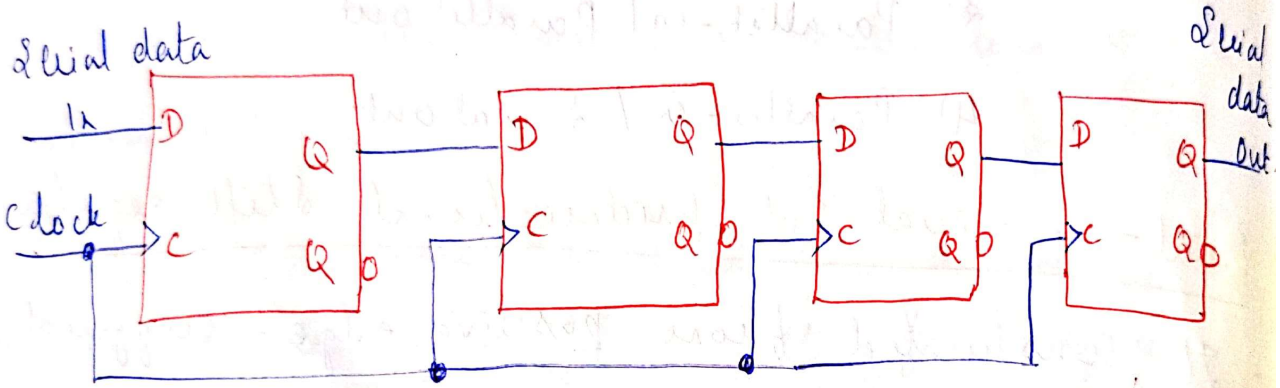


figure:- Serial-in, Serial-out Unidirectional Shift register

- If the initial content of the four flip-flops is 1011 and a logic-0 is applied to the serial-data-in line prior to the positive edge of the clock signal, then the content of the register becomes 0101 after the positive edge of the clock signal.

- The logic 0 becomes available on the serial-data-out line after four clock

- In some applications, the information within a register must be preserved, but only a reorientation is desired
- To achieve this, serial-data-out line is connected to the serial-data-in line
- The content of the register is again shifted one position to the right upon the occurrence of each clock signal, but the state of the leftmost flip flop is replaced by the state of the rightmost flip-flop.
- Example :-

After Shifting 1101 Circular shift register

- Shift registers having this type of connection is called Circular shift register.

Note :-

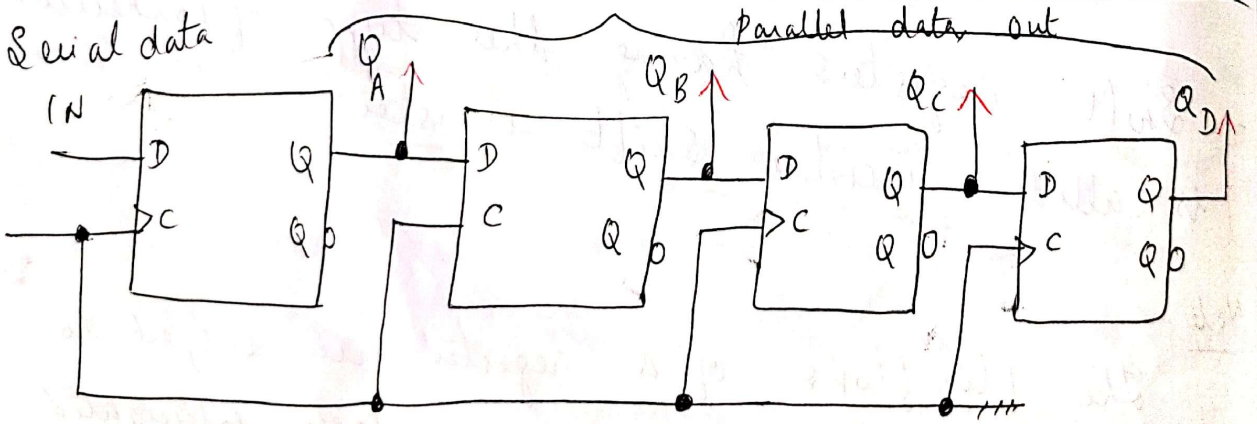
The flip flops of a register are subject to a change in state while they are being interrogated by a next flip flop in the cascade connection. A flip flop is simultaneously being read into while being read by another flip flop

The master-slave flip flop

- Latches are not appropriate, as their o/p's are subject to changes during the entire period in which they are enabled.

Serial-in, Parallel-out Unidirectional Shift Register

- o/p's are provided from each flip-flop.
- Once information is shifted into the register, i.e, Serial in, the information is available as a single entity i.e parallel out, at the flip-flop o/p terminals.
- Since information is transferred into this register Serially and after an appropriate number of shifts made available in parallel, this type of register is called Serial-to-parallel Conversion of information.



Parallel-IN, Serial out Unidirectional Shift Register

- The operation is controlled by the load/Shift line.
- When a logic-0 signal appears on this

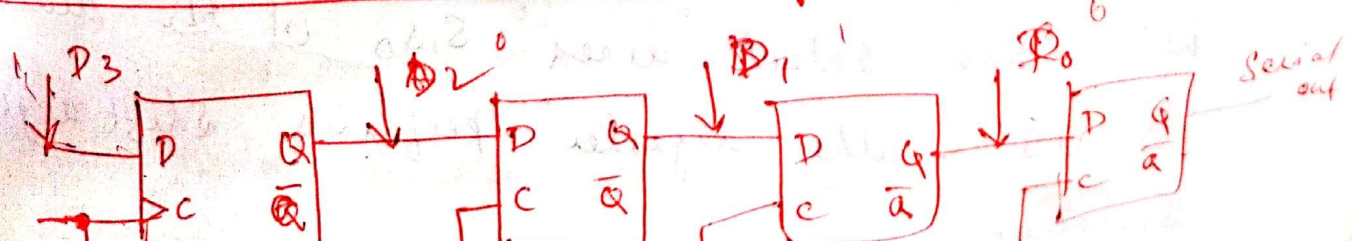
I_A I_B I_C I_D are transferred into the register upon the occurrence of positive edge clock signal.

- Then, when logic 1 signal appears on the load/shift line, the D-flip flop become a cascade connection that functions as a unidirectional shift register providing the serial o/p.
- In this way, it provides parallel-to serial conversion of information. (diagram from Book)

Parallel-in, Parallel-out Unidirectional Shift Register :-

- From the same diagram, if o/p's are taken from each individual flip flops, the register functions are parallel-in, parallel-out register.
- The same ckt will work as serial-in, parallel out & serial-in, serial-out register as well.

Parallel-IN, Serial Out : dgm



USN

Internal Assessment Test 2 – Dec.2024

Sub:	Digital System Design using Verilog					SubCode:	BEC302	Branch:	ECE	
Date:	13-12-2024	Duration:	90 Minutes	MaxMarks:	50	Sem/ Sec:	3/A,B,C,D		OBE	

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1	Design a MOD-6 synchronous counter using clocked D flip flop	[10]	CO3	L3
2	Explain Universal shift register with neat diagrams.	[10]	CO3	L2
3	Illustrate with examples the data types used to define nets, registers, vectors and arrays.	[10]	CO4	L2
4	Explain verilog logical operators with examples.	[10]	CO4	L2

5	Write the verilog code for a Positive Edge-Triggered JK flip-flop using the CASE statement with the timing waveform.	[10]	CO3	L2
6	Write the Verilog code for 2*1 multiplexer using If-Else statement and Else-If statement using the below logic (i) when enable is low and selection line is high, the output is equal to input 2 (A) (ii) when enable is low and selection line is low, the output is equal to input 1 (B) (iii) If enable signal is high, the output reaches high impedance state (z)	[10]	CO3	L2
7	Write the Verilog code for the Half Adder using the data-flow description, behavioral and gate level description.	[10]	CO4	L2
8	Explain the different types of shift registers with neat diagram.	[10]	CO3	L2

COURSE INSTRUCTORS

CCI

HOD