CMR INSTITUTE OF TECHNOLOGY

USN | | | | | | | | | | |

Internal Assessment Test 1 –Mar 2018

Sub:	Digital System Design using Verilog			**Code:**	10ECE663
Date: 14/03/2018	Duration: 90mins	Max Marks: 50	**Sem:** Ist	**Branch:**	BE ECE \NON CBCS

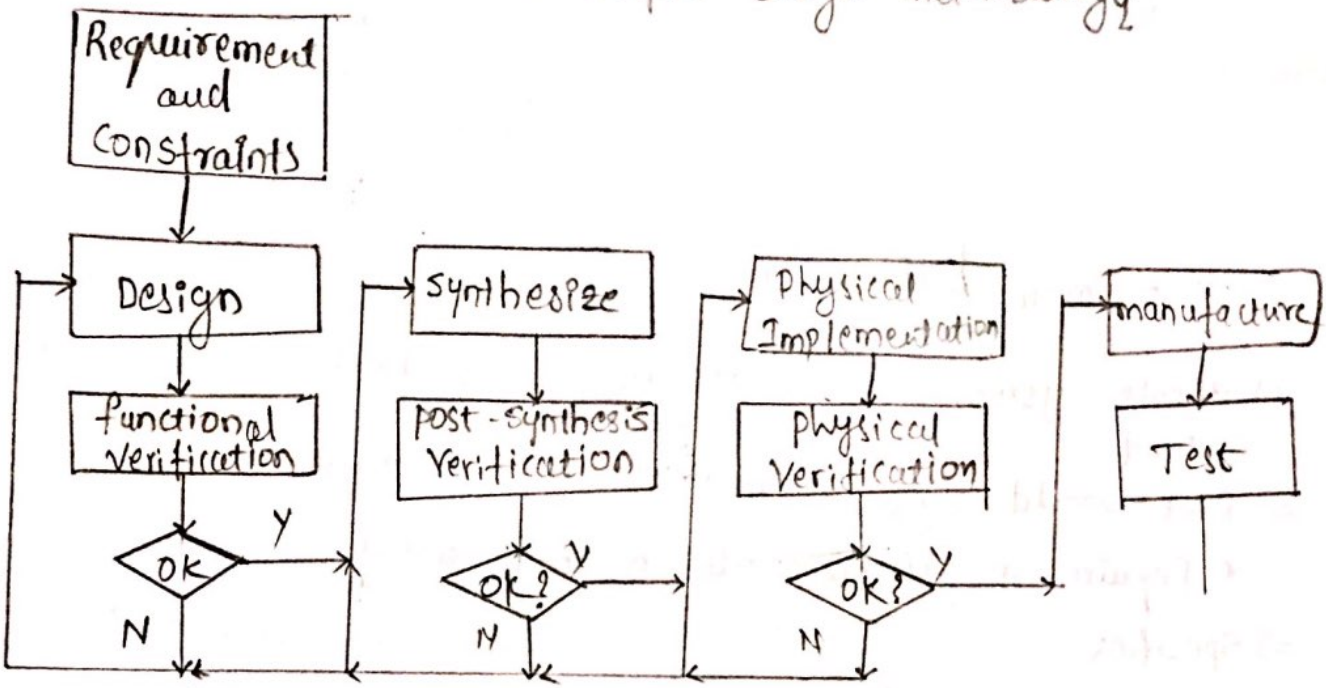
Note: **- Answer all questions**

1	Explain the systematic process of digital system from design to Embedded system design.	10M
2	Develop verilog code for priority encoder of 8:3	10M
3	Construct a multiplier for two 16-bit operands containing just one adder that adds successive partial products over successive clock cycles.	10M
4	Construct a datapath to perform a complex multiplication. All inputs are 4 pre binary and 12 post binary points.	10M
5	Explain flip-flops, counters and shift registers. Explain with syntax and proper example	10M

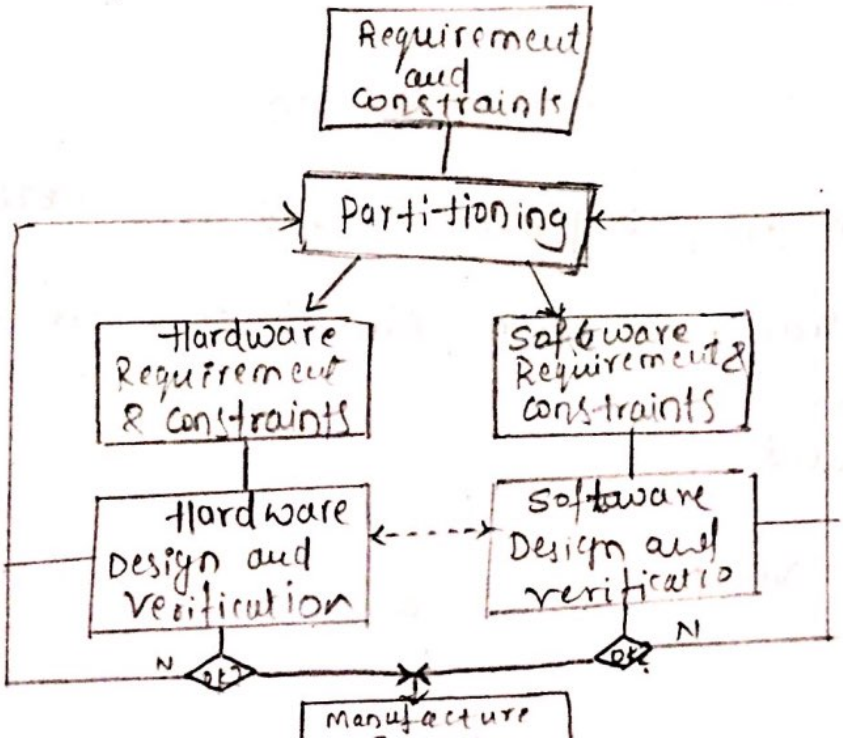
1) Explain the systematic process of digital system from design to embedded system design

Design Methodology :-

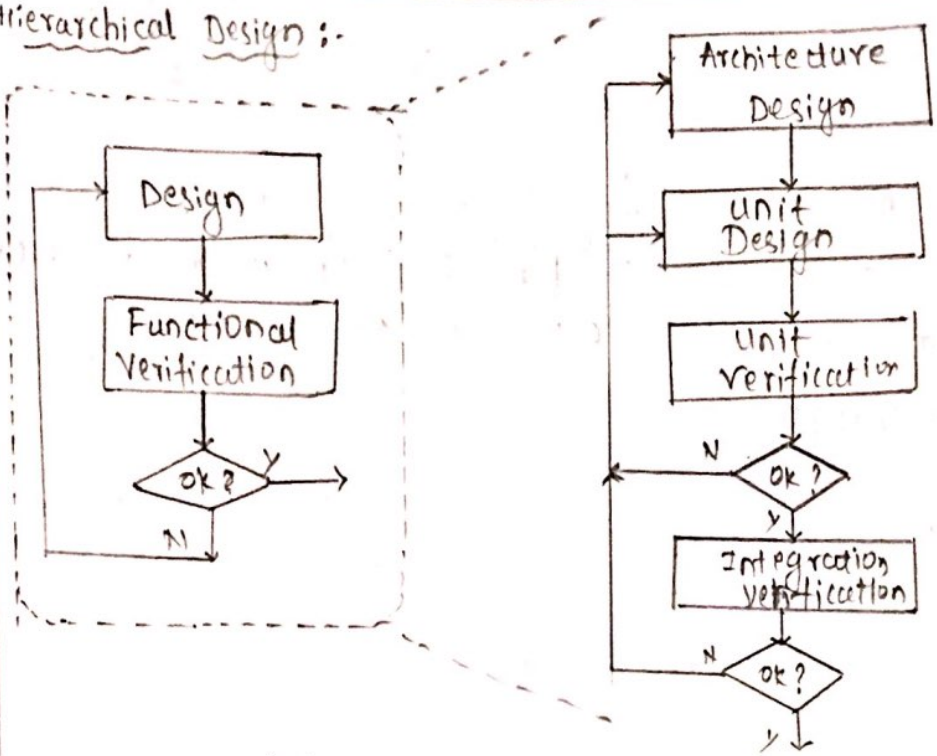
A simple Design methodology



Code Design methodology



Hierarchical Design :-



Design methodology

⇒ Simple systems can be design by one person using ad hoc method

⇒ Real-world systems are design by teams

* Require a systematic design methodology

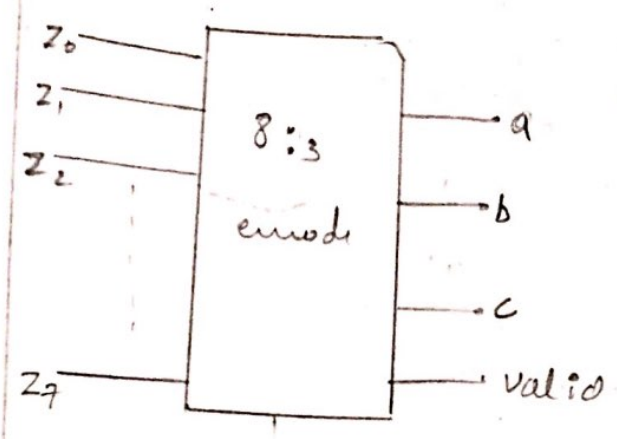
⇒ Specifies

* Tasks to be undertaken
* Information needed and produced
* Relationships b/w tasks
* EDA tools used

⇒ Synthesis

* we usually design using register-transfer level (RTL) verilog
* Synthesis tool translates to a circuit of gates that perform the same function
* Specify to the tool.
* post synthesis verificat

v) 8:3 Encoder :-



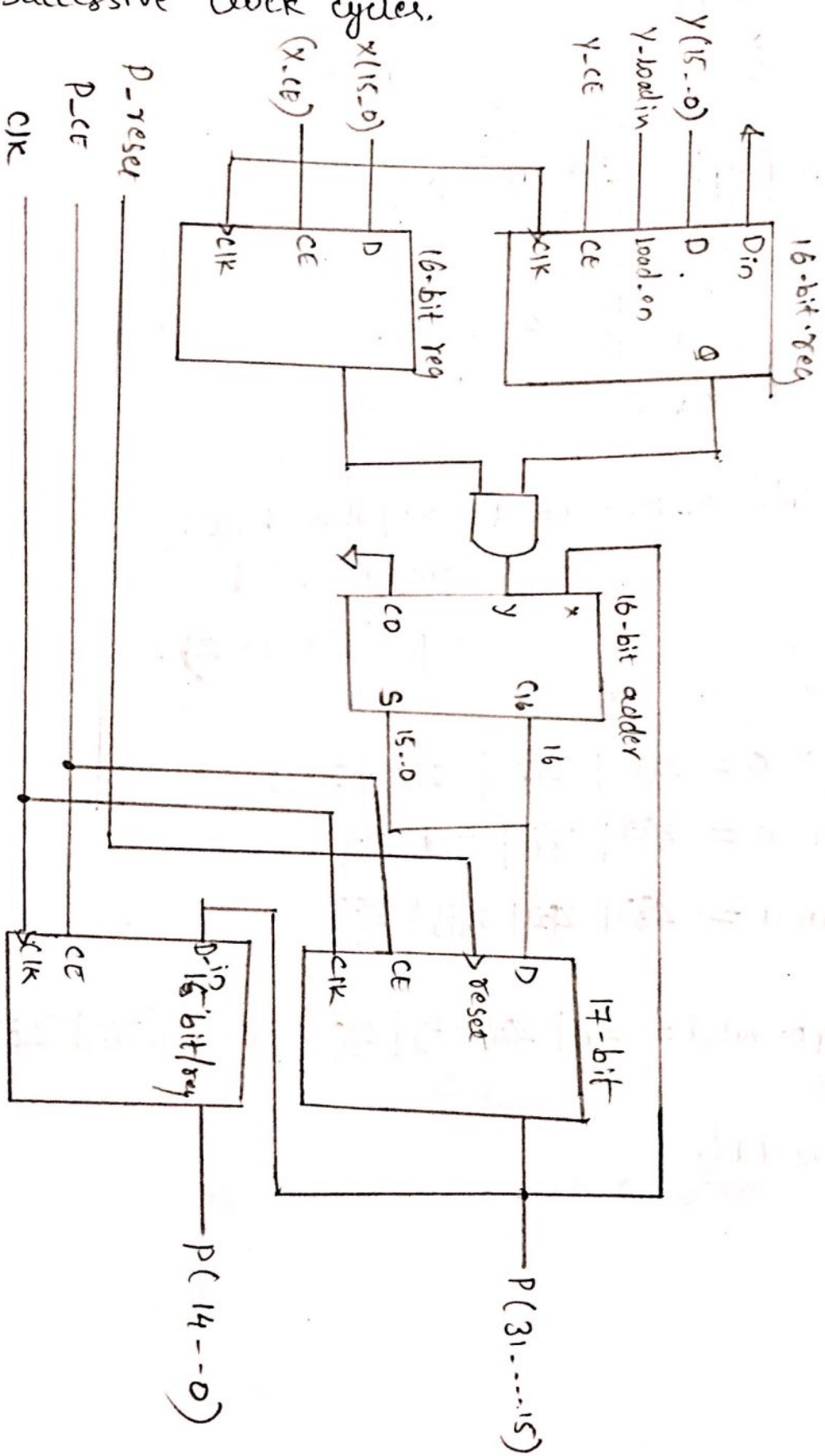
z_7	z_6	z_5	z_4	z_3	z_2	z_1	z_0	a	b	c	Valid
1	x	x	x	x	x	x	x	0	0	0	1
0	1	x	x	x	x	x	x	0	0	1	1
0	0	1	x	x	x	x	x	0	1	0	1
0	0	0	1	x	x	x	x	0	1	1	1
0	0	0	0	1	x	x	x	1	0	0	1
0	0	0	0	0	1	x	x	1	0	1	1
0	0	0	0	0	0	1	x	1	1	0	1
0	0	0	0	0	0	0	1	1	1	1	1

```
module 8_to_3_encod ( output a, b, c,
                      output valid,
                      input [0:7] Z);
begin
assign c = Z[6] | Z[4] | Z[2] | Z[0];
assign b = Z[5] | Z[4] | Z[1] | Z[0];
assign a = Z[3] | Z[2] | Z[1] | Z[0];

assign valid = Z[7] | Z[6] | Z[5] | Z[4] | Z[3] | Z[2] | Z[1] | Z[0];

endmodule
```

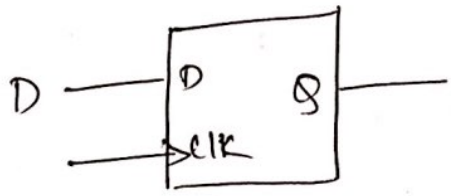
3) construct a multiplier for two 16-bit operands containi just one adder that adds successive partial products over successive clock cycles.



Explain flip-flop, counters and shift register, explain with syntax & proper example.

flipflop or latch is a circuit that has two stable states and can be used to store state information

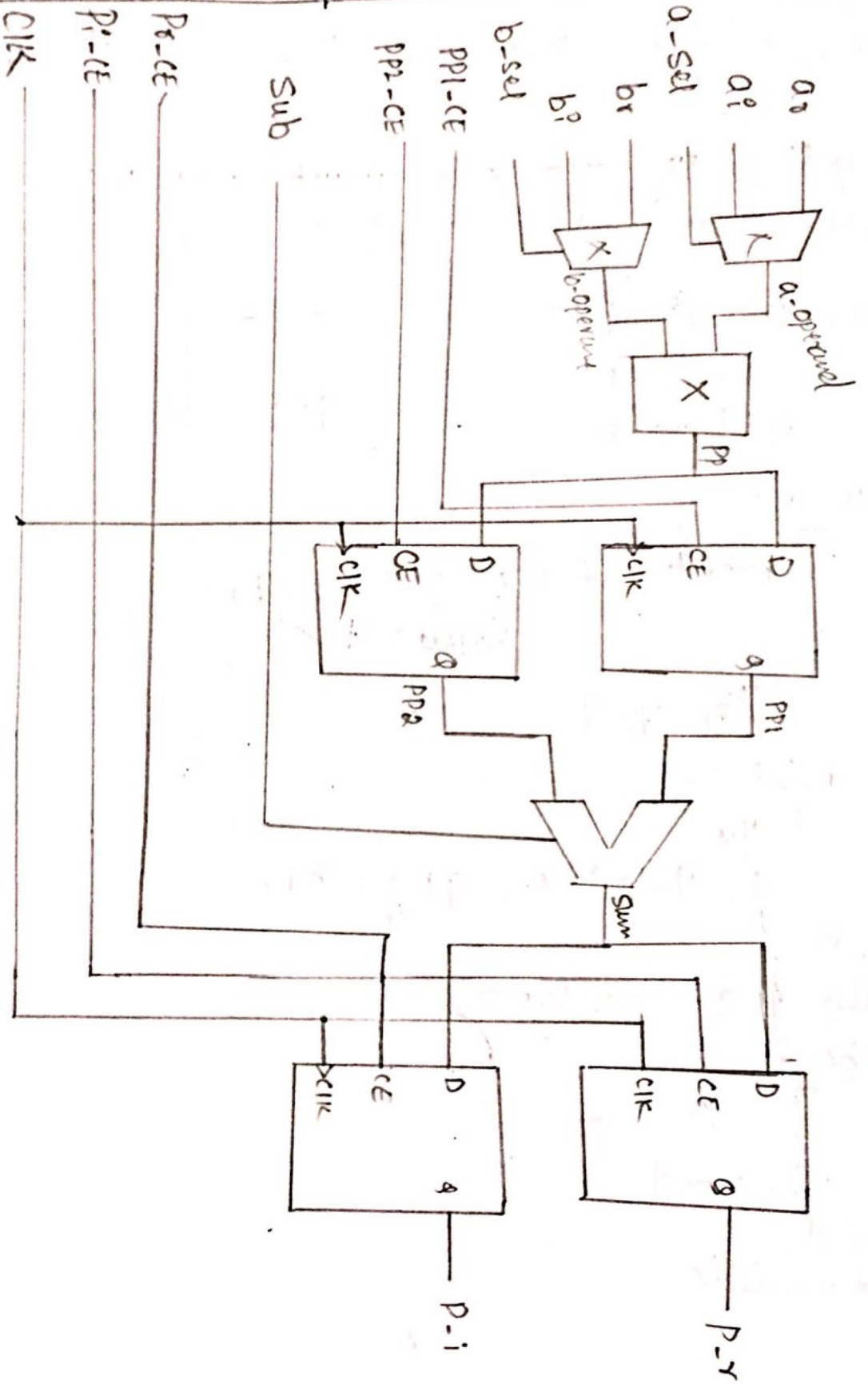
eg: D-flipflop
T-flip flop
JK-Hip flop



Syntax :-

```
module DFF (output q, q1,
            input d, c);
  reg q, q1;
 intial
  begin
     q = 1'b0; q1 = 1'b1;
  end
 always @ (posedge c)
  begin
    q = d;
    q1 = ~d;
  end
end module
```

(4) construct a datapath to perform a complex multiplication. inputs are 4 pre binary & 12 post binary point.



counters :- A simple form of counter is composed of an edge-triggered register & an incrementer. The value stored in the register is interpreted as an unsigned binary integer. The incrementer can be implemented using the circuit we described for an unsigned incrementer.

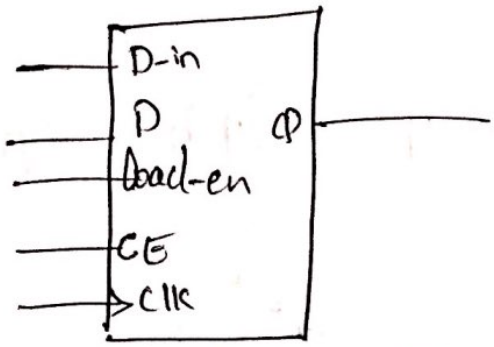
Ex: up-down counter, Asynch, synch.

syntax

```
module decoded_counter (output ctrl,
                        input clk);
reg [3:0] count_value;
always @(posedge clk)
  count_value <= count_value + 1;
assign ctrl = count_value == 4'b0111 ||
              count_value == 4'b1011;
endmodule
```

shift register

A register, as we have seen stores data and makes it available at the output unchanged. A shift register, on the other hand, can perform a shift operation on the stored data



syntax:

always @ (LE or D)

if (LE) Q <= D;