**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **01-06-2020** | **Name:** | **Kiran N** |
| **Course:** | **Logic design** | **USN:** | **4al16ec031** |
| **Topic:** | **FPGA** | **Semester & Section:** | **8th and A** |
| **Github Repository:** | **Kiran-course** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | |
|  |  |  |  | |
| Verilog code to implement NAND gate in all different styles:  1.Gate level modelling in Verilog  Module nand\_2(output y, input A,B)  Wire yd;  And(yd,A,B);  Not (y,yd);  Endmodule;  2.Data flow modelling Verilog  module nand\_2\_data\_flow (output Y, input A,B);  assign Y= ~(A & B);  endmodule  3.Behaviorial modelling Verilog  module nand\_2\_behavioral (output reg Y, input A,B);  always @ (A or B)begin  if (A == 1’b1 & B == 1’b1)begin  Y = 1’b0;  End  Else  Y = 1’b1;  End  Endmodule  4. Test bench of the Nand gate using Verilog  include “nand\_2\_behavioral.v”  module nand\_2\_behavioral\_tb;  reg A, B;  wire Y;  nand\_2\_behavioral indtance0 (Y, A, B);  initial begin  A = 0; B = 0;  #1 A = 0; B = 1;  #1 A = 1; B = 0;  #1 A = 1; B = 1;  End  Initial begin  $monitor (“%t | A = %d| B = %d| Y = %d”, $time, A, B, Y);  $dumpfile(“dump.vcd”);  $dumpyars();  End  Endmodule | | | |