**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **05/06/2020** | **Name:** | **Lepakshi T V** |
| **Course:** | **Digital design using HDL** | **USN:** | **4AL17EC044** |
| **Topic:** | **Verilog Tutorials and practice programs, Building/ Demo projects using FPGA**  **Implement a verilog module to count number of 0’s in a 16-bit number in the compiler.** | **Semester & Section:** | **6th sem, A sec** |
| **Github Repository:** | **Lepakshi-044** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report – Report can be typed or hand written for up to two pages.**  Verilog is a HARDWARE DESCRIPTION LANGUAGE (HDL). A hardware description Language is a language used to describe a digital system, for example, a network switch, a microprocessor or a memory or a simple flip−flop.  **Design Styles**  Verilog like any other hardware description language, permits the designers to design a design in  either Bottom−up or Top−down methodology**.**  **Abstraction Levels of Verilog**  Verilog supports a design at many different levels of abstraction. Three of them are very important:   * Behavioral level * Register−Transfer Level * Gate Level   **Various stages of ASIC/FPGA:**   * Specification * Micro Design/Low level design: * RTL Coding * Simulation * Synthesis * Place & Route * Post Si Validation   **Simulation:** Simulation is the process of verifying the functional characteristics of models at any level of  abstraction.  **Synthesis:** Synthesis is process in which synthesis tool like design compiler or Simplify takes the RTL in  Verilog or VHDL, target technology, and constrains as input and maps the RTL to target  technology primitives. Synthesis tool after mapping the RTL to gates, also do the minimal amount  of timing analysis to see if the mapped design meeting the timing requirements**.**  **A Verilog code to count number of zeros:**  module num (input [15:0] A, output reg [4:0] ones);  integer i;  always@(A) begin  ones = 0; for(i=0;i<16;i=i+1)  if(A[i] == 0'b1)  ones = ones + 1;  end  endmodule  output  Input = "1010\_0010\_1011\_0010" => Output = "01001"  Input = "0011\_0110\_1000\_1011" => Output = "01000" |

**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **05/06/2020** | **Name:** | **Lepakshi T V** |
| **Course:** | **Python** | **USN:** | **4AL17EC044** |
| **Topic:** | **Build a Data Collector Web App** | **Semester & Section:** | **6th sem, A sec** |
| **Github Repository:** | **Lepakshi-044** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report – Report can be typed or hand written for up to two pages.**  **The output will be like**    **PostGreSQL database web app with Flash**  **Frontend: HTML, CSS**  **Backend: Getting User Input**  **Backend: The PostGreSQL Database model**  **Backend: Storing user data to the database**  **Backend: Emailing database values back to the user**  **Backend: Sending statics to users**  **Deploying the Web Application to a Live Server**  **Implementing download and upload in our Web app** |