**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **02-06-2020** | **Name:** | **Kiran N** |
| **Course:** | **Logic design** | **USN:** | **4al16ec031** |
| **Topic:** |  | **Semester & Section:** | **8th and A** |
| **Github Repository:** | **Kiran-course** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| REPORT:  HDL is an abbreviation of Hardware Description Language. Any digital system can be represented in a REGISTER TRANSFER LEVEL (RTL) and HDLs are used to describe this RTL. Verilog is one such HDL and it is a general-purpose language easy to learn and use. Its syntax is similar to C.  The idea is to specify how the data flows between registers and how the design processes the data. To define RTL, hierarchical design concepts play a very significant role.  Hierarchical design methodology facilitates the digital design flow with several levels of abstraction.  Verilog  HDL can utilize these levels of abstraction to produce a simplified and efficient representation of the RTL description of any digital design.  For example, an HDL might describe the layout of the wires, resistors and transistors on an Integrated Circuit (IC) Chip, i.e., the switch level or, it may describe the design at a more micro  Level in terms of logical gates and flip flops in a digital system, i.e., the gate level.  Verilog supports all of these levels.  Hierarchy of design methodologies:  Bottom  Up Design  The traditional method of electronic design Is bottom-up (designing from transistors and moving to a higher level of gates and, finally, the system). But with the increase in design complexity traditional bottom-up designs have to give way to new structural, hierarchical design methods.  Top Down Design For HDL representation it is convenient and efficient to adapt this design- style. A real top-down design allows early testing, fabrication technology independence, a structured system design and offers many other advantages. But it is very difficult to follow a pure top-  down design. Due to this fact most designs are mix of both the methods, implementing some key elements of both design styles.      •This level describes a system by concurrent algorithms (Behavioral).  •Each algorithm itself is sequential meaning that it consists of a set of instructions that are  executed one after the other.  •‘initial’, ‘always’ ,‘functions’ and ‘tasks’ blocks are some of the elements used to define the  system at this level.  •The intricacies of the system are not elaborated at this stage and only the functional description  of the individual blocks is prescribed. In this way the whole logic synthesis gets highly simplified and at the same time more efficient Register-  Transfer  Level Designs using the  Register  Transfer  Level specifies the characteristics of a circuit by operations and the transfer of data between  The registers.  An explicit clock is used.  RTL design contains exact timing possibility operations are scheduled to occur at certain  times. Modern definition of a RTL code is "Any code that is synthesizable is called RTL code".  Gate Level Within the logic level the characteristics of a system are described by logical links and their timing properties.  All signals are discrete signals. They can only have definite logical values(`0',`1', `X', `Z`). The usable operations are predefined logic primitives (AND, OR, NOT etc gates).  It must be indicated here that using the gate level modeling may not be a good idea in logic design.  Implement a simple T Flip-flop and test the module using a compiler:  module tff ( input clk, input rstn, input t, output reg q);  always @( posedge clk)  begin  if(!rstn)  q<=0;  else  if(t)  q <= ~q;  else  q<=q;  end  end  module |