JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

TEST-3 EXAMINATIONS-MAY 2025

M.Tech-II Semester (ECE)

COURSE CODE (CREDITS): 21M11EC211 (3) MAX. MARKS: 35

COURSE NAME: Digital System Design using Verilog HDL

COURSE INSTRUCTOR: Dr. Pardeep Garg MAX. TIME: 2 Hours

Note: (a) All questions are compulsory. (b) The candidate is allowed to make suitable numeric assumptions wherever required for solving problems.

Q. No	Question	CO	Marks
Q1	'*In Verilog, if there are no timing control statements, the simulation time does not advance*' justify the use of timing controls in Verilog in context of this statement.	CO-3	1
Q2	Discuss the use of regular delay control, and intra-assignment delay control using suitable example for each.	CO-3	1.5+1.5=3
Q3	Declare a register called *oscillate*. Initialize it to 0 and make it toggle every 30 time units. **Do not** use an *always* statement.	CO-3	2
Q4	Design a clock with time period=40 and a duty cycle of 25% by using the always and initial statements. The value of clock at time = 0 should be initialized to 0.	CO-3	3
Q5	Blocking and Nonblocking assignments can be used in Verilog modules, differentiate between these two by discussing their execution style, operator used etc by taking suitable example. Out of these two, which one is preferred in digital design and why? Which assignment out of these suffers from race condition and which one does not suffer, discuss using a suitable example.	CO-3	2+1+2=5
Q6	Is Verilog HDL a concurrent language? If yes, Justify the concurrent nature of Verilog HDL language with the help of a programming example which uses multiple initial statements and show the output also.	CO-3	4
Q7	It is said that in some situations *while* loop is appropriate and in some situations *for* loop is more appropriate to be used. Justify this statement with proper logic and test cases for both loops.	CO-4	4
Q8	Employing suitable example of each, discuss the usefulness of *forever* and *repeat* loop in Verilog HDL.	CO-4	4
Q9	Write a Verilog design module using a case statement to implement a 2-to-4 decoder. Include the module definition, input/output declarations, and always block.	CO-4	4
Q10	The design code of a Verilog HDL language is as follows:		2.5+2.5=5

```
reg [7:0] data;
integer i;

initial begin
   data = 8'b1010_0100;
   i = 0;

   search_block: begin
      while (i < 8) begin
         if (data[i]) begin
            $display("Found a '1' at position %d", i);
            disable search_block;
         end
         i = i + 1;
      end
   end
end
```

a) What is the special feature used in the above Verilog code within the search_block? Explain how this feature controls the execution of the code and describe the output when the first '1' is encountered.

b) If a similar logic were to be implemented in C language, which construct would replace the Verilog disable statement used here? Compare how this feature behaves in Verilog and C.