**Digital Design (CS F215)** **14-10-2024**

# Lab 8: Finite State Machines

*Lecturer: Harikrishnan N B* *Student:*

**READ THE FOLLOWING CAREFULLY:**

**Honour Code for Students:** I shall be honest in my efforts and will make my parents proud. Write the oath and sign it on the assignment.
For the lab, you need a dedicated fresh notebook. Keep this notebook dedicated for solving the questions pertaining to lab.

There will be design questions in the lab, the students are supposed to theoretically show the correctness of their design function. Following that, the students are encouraged to use iverilog software to implement the same.

The lab evaluation consists of two parts.

1) Step 1 (Theory): The students are supposed to solve the design question in their dedicated lab notebook. They are supposed to get it checked and validated by TA's/ instructors. (**Deadline for theory evaluation: 10:15 AM**)

2) Step 2 (Lab): Once Step 1 is completed, they can use iverilog to implement their design. Once you have completed the design, inform the TA's/ instructors and get it verified. (**Deadline for lab file submission in local quanta: 10:40 AM**)

3) Step 3: Upload your solution to quanta.bits-goa.ac.in (local quanta). Create a zip file CampusID_Lab8.zip which contains all the files.

`Note: The clock period is 10 seconds (with 50 % duty cycle). Initialize the clock to 0 for all the testbenches.`

**All questions must be solved using the modeling mentioned in the question. Only solutions implemented using the mentioned modeling will be accepted for evaluation.**

---

1. **Theory(1 Mark):** Draw the block diagram for 1-bit JK flip-flop. Clearly label all inputs and outputs. Your diagram should include a posedge clock input (`clk`), an active-high asynchronous reset input (`reset`), two data inputs (`j and k`), and an output (`q`). Also, make the flip-flop characteristic table.

2. **Lab(3 Marks):** Complete the module `jkff_async_reset.v` in Verilog using a behavioral model to implement a 1-bit JK flip-flop.The module should take a posedge clock input (`clk`), an active-high asynchronous reset input (`reset`), and two data inputs (`j and k`), producing an output q. Test the output by running the `testbench_jkff_async_reset.v`, which has already been implemented for you.

3. **Theory(4 + 5 + 4 marks):** In the royal court of Emperor Akbar, he wishes to discuss important state matters with his clever advisor, Birbal, without anyone else overhearing their conversation. To ensure their secret communication, they decide to use a special coded language based on a series of binary inputs (0s and 1s) that represent various messages. Design a Mealy finite state machine (FSM) for the same based on a 3-bit binary input. The code operates as follows: if the input code is '101' or '110' it outputs '1' and for all other inputs, it outputs '0'. Implement the FSM using JK flip-flops to handle state transitions. Derive the state equations for the flip-flops and the output. You must use **binary encoding** for the states. You are required to draw the minimized state diagram, the reduced state table, and write the equations for each JK flip-flop input, along with the output equations for the FSM.

   **Note : The input consists of 3-bit codes in a continuous overlapping stream. The first 2 output bits are invalid, denoted by 00**

   **Sample Input : 0001011001**
   **Output : 0 0 0 0 0 1 0 1 0 0**
   The spaces are provided for clarity and are not part of the actual output.

4. **Lab(13 marks):** Complete the module `fsm_module.v` in Verilog using a structural model to implement the above circuit. The module takes posedge `clk`, active-high `reset` and `input_code` as the input and gives `out` as the output. Complete the corresponding testbench (`testbench_fsm.v`). The inputs to the testbench are provided in the table below. Verify the correctness of the module for the testcases provided in below Table 8.1. Testcases should be in the order given below.

Table 8.1: Testcases

| **Time** | **Reset** | **Input** |
|---|---|---|
| 0 | 1 | 0 |
| 10 | 0 | 0 |
| 20 | 0 | 1 |
| 45 | 0 | 0 |
| 55 | 0 | 1 |
| 65 | 0 | 0 |
| 75 | 0 | 1 |
| 85 | 0 | 1 |
| 95 | 0 | 0 |
| 105 | 0 | 0 |