ECE 152A – Spring 2021
Lab 6

Objective: Microwave controller

In this lab, you will implement a state machine in Verilog. The purpose of this lab is to imitate the working of a simple microwave controller.

1. Understand the behavior of the microwave controller and map the behavior of the controller to a state machine. Define the inputs/outputs of the system and draw a state diagram.
2. Implement the state machine in https://www.edaplayground.com/ using Verilog.
3. Construct testbenches to test the working of the controller in all scenarios and with the help of waveforms, prove that the behavior of the controller is correct.

Lab Schedule

Lab Dates	**Deliverables**
05/17 and 05/28	**Part 2 and 3 due by the end of lab**

Grading

The lab will be graded as follows for steps that are completed on time.

Part	**Weight**
Part 1	30 %
Part 2	50 %
Part 3	20 %

Part	**Points**
Part 1	45
Part 2	75
Part 3	30
Total	150

Please note that we take the Honor Code very seriously. You must write the Verilog Code with only your own team.

Design Strategy Outline

Part 1

Understand the behavior of the microwave controller and map the behavior of the controller to a state machine. Define the inputs/outputs of the circuit and draw a state diagram.

Part 2

Implement the state machine in edaPlayground using Verilog.

Part 3

Construct a testbench and use a waveform to prove that the behavior of the controller is correct.

Note: please watch the demo in place of this part.
Download your design onto the FPGA and wire up a circuit that demonstrates your design's functionality.

Necessary Supplies

For this lab, you will need:

- UCSB DigiLab FPGA (Information about the FPGA)
- Breadboard
- 10-input DIP Switch
- Seven Segment Display (Common Cathode)
- Resistors for the seven segment display and the switches.

Detailed Design Strategy

Detailed working of the microwave controller

The microwave controller is governed by the following rules:

1. The microwave has a door (*door_status*), a power button (*power*), a start button (*start_button*), a cancel button (*cancel_button*), a timer (*timer*), a state display (*state_display*) and a time display(*time_display*).
2. When the microwave is not in use the microwave display reads 'IdLE' i.e. idle state
3. When the food is placed in the microwave and the door is closed ($door\_status = 1$), the power button is used to set the power – FULL/ HALF.
4. Then the timer is set (in seconds) and the start button is pressed. Note that none of buttons or the timer has any effect when the door is open. The timer cannot be set for more than 120s.
5. If the power or timer are not set before pressing the start button, assume power – HALF and timer – 60 seconds by default.
6. After the start button is pressed, the microwave display reads 'PrOC' i.e. process state and the countdown timer begins (Hint: For every clock cycle decrement the time by 1 till it reaches zero).
7. Once the timer reaches 0, the microwave display reads 'dOnE'. The display should read 'dOnE', i.e. completed state, till the door is opened and then read 'IdLE'. Note that this display change occurs only when the microwave is in completed state. If the microwave is in idle/process state, the display remains the same when the door is opened.
8. If the door is opened before the timer reaches 0, the timer is paused and resumes when the door is closed and the start button is pressed.
9. If the door is opened before the timer reaches 0 and the timer can be reset to new values, the microwave operates with the new values (the microwave is still in process state, it need not start from idle state).
10. Whenever cancel button is pressed, irrespective of the current state the microwave transitions to idle state.

Parameter description

- Power button: power can be set to HALF or FULL. If power is not set, then default setting is HALF. The power button has no effect when the controller is in the process state.
- Start button: When start button is pressed the timer begins. The start button can be pressed anytime and the timer begins at the clock edge and the start button is reset to zero. Start button is 1 $\rightarrow$ timer begins at the clock edge $\rightarrow$ Start button is reset to 0 in the same clock cycle.

- Cancel button: When cancel button is pressed the controller resets to initial settings. Cancel button is 1 $\rightarrow$ controller transitions to initial state at clock edge $\rightarrow$ Cancel button is reset to 0 in the same clock cycle.
- Timer: The timer can be set to any positive integer less than 120 seconds.
- Door status: the door status is 0 when the door is opened and the door status is 1 when the door is closed
- State display: the current state of the controller has to be displayed on the 7-segment display as 'IdLE', 'PrOC' or 'dOnE'. Whenever the controller transitions into a new state, the state display has to change at the first clockedge in the new state. ('I' can be displayed as '|').
- Time display: Output the countdown timer - decrements at every clockedge. (not 7-segment display)

Part 1 - State Machine Design

First, we need to map the behavior of the controller to a state machine. To do this, follow these steps:

1. Before you start your state machine design, answer the following questions (which are meant to help you keep the design simple).
   - How many states should your state machine have (i.e. what should the minimum number of states be)?
   - What is the default state and the corresponding default output?
   - How is each state attained from the default state?
2. Clearly define the state variables that you will use (and how you will encode/represent them in your code). In addition, clearly define the inputs and outputs of the controller.
3. Draw a Mealy state diagram for the controller. Show all the valid states in the diagram with the values of the inputs clearly shown beside each edge connecting the nodes.

Part 2 - Verilog Implementation and Simulation

The controller can be broken down into two parts.

- State Machine – The state machine is the sequential circuit that takes care of determining what the output pattern should be as a result of the current state and the current inputs.
- Combinational Logic – This is needed to implement the timer as a countdown counter. This logic block takes the state of the microwave from the state machine and displays the state (IdLE, PrOC and dOnE) on the display. Depending on the state of the microwave the timer is implemented as countdown timer or paused (when start_button is pressed) or reset (when cancel_button is pressed).

Part 3 - Construction of testbench

Possible scenario 1 is provided in the testbench template given below. Using that as reference, complete the testbench for the remaining three scenarios described below.

1. open door → place food → close door → set power (FULL) → set timer (100 seconds) → wait for cooking to finish → open door → remove food → close door
2. open door → place food → close door → set power (HALF) → set timer (100 seconds) → wait for 40 seconds → open door → remove food → place food back → close door → press start button → wait for cooking to finish → open door → remove food → close door
3. open door → place food → close door → set power (FULL) → set timer (100 seconds) → push button → wait for 30 seconds → open door → remove food → place food in oven → close door → reset timer (50 seconds) → wait for cooking to finish → open door → remove food → close door
4. open door → place food → close door → set power (FULL) → set timer (100 seconds) → push button → wait for 30 seconds → press cancel button → open door

Keeping the structure of the controller in mind, implement the controller using Verilog by using the following template.

```
// this module takes the inputs of the microwave controller
// power = 0 for HALF and 1 for FULL
// timer is the heat time
// door_status = 0 for OPEN and 1 for closed
// start_button = 0 initially and 1 when pressed
// cancel_button = 0 initially and 1 when pressed
module controller(input clk,
                  input power,
                  input [6:0] timer,
                  input door_status,
                  input start_button,
                  input cancel_button,
                  output reg [6:0] state_display1,
                  output reg [6:0] state_display2,
                  output reg [6:0] state_display3,
                  output reg [6:0] state_display4,
                  output reg [6:0] time_display);
  // fill out code
  // TODO:
  //1. Implement a countdown counter for the timer
  //2. Monitor door_status and cancel_button(analogous to reset in Lab4
   )
  //3. Monitor start_button (analogous to enable in Lab4)
  //4. state_display consists of 4-alphabets based on the state of the
   controller. Use 7-segment display representation to display the
   state.

endmodule
```

```
module controller_tb();

// declare variables
parameter CLK = 10;
reg clk = 0;
reg power, door_status, start_button, cancel_button;
reg [6:0] timer;
wire [6:0] state_display1; // I or P or d
wire [6:0] state_display2; // d or r or O
wire [6:0] state_display3; // L or O or n
wire [6:0] state_display4; // E or C or E
wire [6:0] time_display;

  initial begin
    forever begin
      clk <= ~clk;
      #5;
    end
  end

  controller ctl(.clk(clk),
                 .power(power),
                 .timer(timer),
                 .door_status(door_status),
                 .start_button(start_button),
                 .cancel_button(cancel_button),
                 .state_display1(state_display1),
                 .state_display2(state_display2),
                 .state_display3(state_display3),
                 .state_display4(state_display4),
                 .time_display(time_display)
                );


  initial begin
    $dumpfile("dump.vcd");
    $dumpvars;

    // Possible scenario 1
    //default settings
    power <= 0; // power is HALF
    timer <= 7'b111100; // timer is 60s
    door_status <= 0; // Door is open
    start_button <= 0; // start button not pressed
    cancel_button <= 0; // cancel button not pressed

    #CLK //1 clock cycle to place food
    door_status <=1; // Door is closed
    power <= 1; // Power is set to FULL
    timer <= 7'b1100100; // timer is set to 100s
    start_button <= 1; // start button pressed
    #5 start_button <= 0; // start button reset at negedge

    #(100*CLK+5) door_status <= 0; //Door opened after 101s - 1s extra
   to display dOnE

    #CLK timer <= 7'b111100; // timer is reset to 60s
    door_status = 1; // 1 clock cycle to remove food and door is closed

    // write test bench for possible scenarios 2, 3 and 4


    // end simulation
    $finish;

  end

endmodule
```

Lab Report

Please submit a report that summarizes your work for Lab 6. The report should include:

- The state machine of the microwave controller with description about each state.
- All of the Verilog code you wrote for the lab.
- Your waveform screenshots and the paragraphs in which you explain how your waveforms prove the functionality of the microwave controller.
- The challenges/ difficulties you faced during the lab

The lab report is due on GauchoSpace by Sunday, May 30, 2021 before 11:59 PM. One report is sufficient for each group.