Dylan Thornburg & Alan Rieger
6/6/2023

Lab 9: Clock Divider and Mealy Machines

Introduction:
In this lab we will use a clock and mealy state machine to create a counter that counts up each second until it reaches its max value in which it resets to zero. We are using the FPGA to do this and our prelab focussed on coding the th value and states for our state machine.

Procedures:
For the first part, we just copied all of the code and inputted our th value (50000000). This resulted in a working demo of a 1 Hz blinking light after compiling and uploading. For the next part, we copied all of the code into the Lab9_fsm file and filled out the code that was needed. We then compiled and uploaded a few more times after fixing some small errors in the verilog code and it eventually worked.

Conclusion:
After some analysis of our code, we finally got the whole thing working. Our initial mistakes mostly just revolved around if-statement logic and incorrect 0s and 1s (they were switched). After solving these problems, it worked perfectly and we even finished early. This lab taught us how to implement a mealy state machine into something practical like a clock.

Report Questions:
Were the state diagrams/state tables you created in your pre-lab correct or not? If it was incorrect, in what way was it incorrect? What do you think led you to your incorrect diagram/table?
Yes they were correct. Although some of our code needed polishing, the logic found from the table and the diagram were still correct.

If your pre-lab was incorrect, provide a corrected answer.
Our pre-lab was correct. There were some minor translation errors when copying in the verilog but it was mostly correct. The prelab code can be seen in the normal code below.

Consider the transition table of S_RUN that you completed in your pre-lab. Suppose that the transition condition of transition #2 is changed from “PR’ · (CNT_out < max_CNT)” to “(CNT_out < max_CNT)”. What is the problem of this change? In what situations, does this cause problems?
Any time you try to pause the count while CNT is less than MAX, it won’t work since PR is not a part of the and statement which allows the counter to be paused.

What would happen if one keeps pressing the PR button for a long time (at least for more than 2 seconds)? If you believe that this is a problem, suggest a new state machine that can resolve this issue.
If the code works correctly, this shouldn't change anything. It should remain paused after letting go.

Figures:
Figure 1: Verilog Code:

```
module Lab9_fsm (CLK, RST, PR, CNT, Max_Val, CntEn, S);
input CLK, RST;                         // Clock and Reset
input PR;                                       // Pause/Reset
input [5:0] CNT;                        // current counter value
input [5:0] Max_Val;            // counter maximum value
output reg CntEn;                       // output, CntEn: counter enable
output reg [1:0] S;      // current state
reg [1:0] S_star;        // next state
parameter [1:0] S_RUN=3'b00, S_PAUSED=3'b01, S_MAX=3'b10;
// Flip-flops for state (state memory)
always @(posedge CLK or posedge RST)
        if (RST == 1) S <= S_RUN;   // initialization (to S_RUN)
        else S <= S_star;
// Next state logics
// input: S, PR, CNT, Max_Val
// output: S_star
always@(S,PR, CNT, Max_Val)
begin
  case(S)
    S_RUN:
        if (PR==1) S_star = S_PAUSED;
        else if (PR==0 & CNT < Max_Val) S_star = S_RUN;
        else S_star = S_MAX;
    S_PAUSED:
        if (PR==0) S_star = S_PAUSED;
        else S_star = S_RUN;
    S_MAX:
        if (PR==0 & CNT < Max_Val) S_star = S_RUN;
        else S_star = S_MAX;
    default:
        S_star = S_PAUSED;
```

```
  endcase
end
// Output logics
// input: S, PR, CNT, Max_Val
// output: CntEn
always@(S, PR,CNT, Max_Val)
begin
  case(S)
    S_RUN:
          if(PR==1) CntEn = 1'b0;
          else if(PR==0 & CNT < Max_Val) CntEn = 1'b1;
          else if(PR==0 & CNT >= Max_Val) CntEn = 1'b0;
    S_PAUSED:
          if(PR==0) CntEn = 1'b0;
          else CntEn = 1'b1;
    S_MAX:
          if(PR==0 & CNT < Max_Val) CntEn = 1'b1;
          else CntEn = 1'b0;
    default:
          CntEn = 1'b0;
  endcase
end
endmodule
```

Figure 2: Successful Upload:

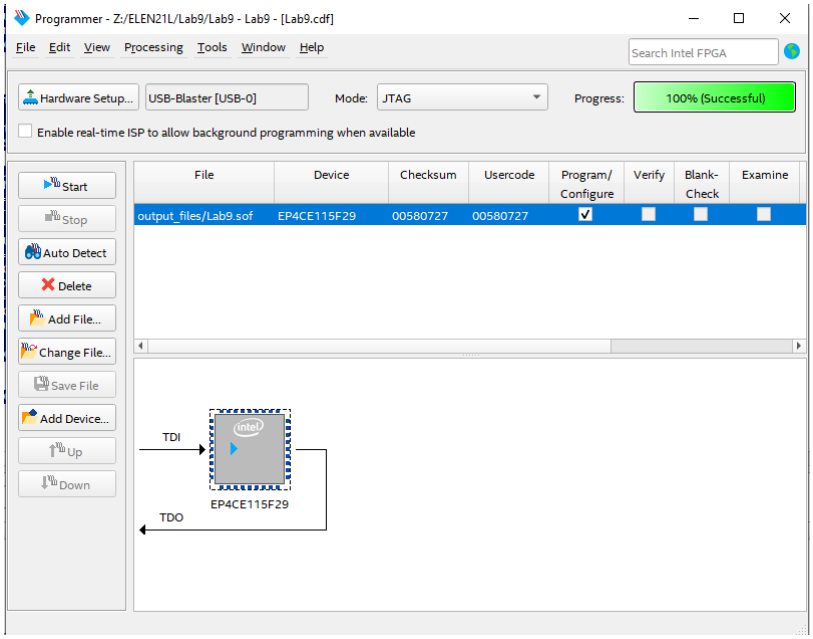

Figure 3: Board Working: