ELEC-4200

Digital System Design

FROM: Jacob Howard

TO: Prof. Ujjwal Guin

DUE DATE: 4/8/21

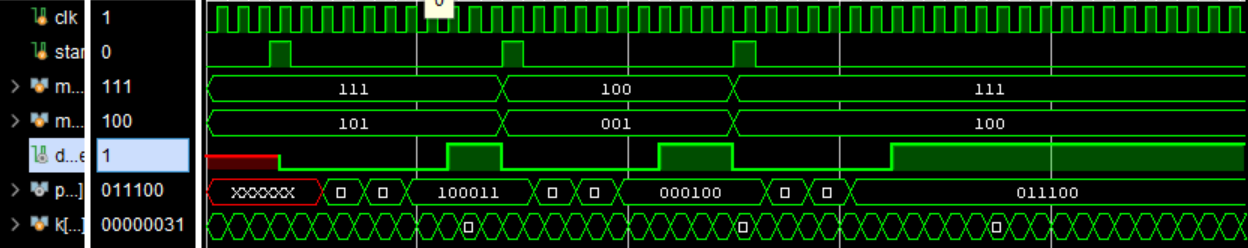
Lab 9

# **Introduction**

The goal of this lab was to familiarize ourselves with Algorithmic State Machines (ASM). We were required to learn and use ASM chart techniques for development in this lab.

# **Task 1**

In Task 1, we were asked to design a 3-Bit x 3-Bit binary multiplier. The design was a bit more complicated when we were told we had to use a 3-bit accumulator, a 3-bit multiplier register, a 3-bit adder, a counter, and a 3-bit shifter. Instead of having Verilog do all the hard work of multiplication while we would just code, for example, a \* b = output, we were designed a multiplier ourselves with logic design. We did this using ASM and coding a finite state machine. The code can be seen below in Code 1. We were also asked to create a test bench and simulate the code. The testbench can be seen in Testbench 1 and the simulation in Simulation 1 below.

****

*Simulation 1*

|  |
| --- |
| module Task1\_new (multiplicand, multiplier, clk, start, product, done);  input [2:0] multiplicand;  input [2:0] multiplier;  input clk, start;  output reg [5:0] product;  output reg done;  reg [6:0] acc;  integer n = 0;  reg [6:0] Q;  reg [1:0] state, nextstate;  parameter [1:0] S0=0, S1=1, S2=2;  //Main Code  ////////////////////////////////////////////////////  always @ (posedge clk)  state <= nextstate;  //begin  always @(state or posedge start)  begin  nextstate = 1'b0;  // done = 0;  case(state)  //Start State  S0: if (start)  begin  acc[6:3] = 4'b0000;  acc[2:0] = multiplier;  done = 0;  n = 0;  nextstate = S1;  end  //if start 0  else  nextstate = S0;  //accumulate state  S1: if (acc[0] == 1 & n <= 2)  begin  acc[6:3] = acc [6:3] + multiplicand;  nextstate = S2;  end  else  nextstate = S2;  //shift state  S2: if (n <= 2)  begin  acc = {1'b0, acc[6:1]};  product = acc[5:0];  n = n + 1;  nextstate = S1;  end  else  begin  done = 1;  nextstate = S0;  end  endcase  end  endmodule |

*Code 1*

|  |
| --- |
| module Task1\_tb();  reg [2:0] multiplicand;  reg [2:0] multiplier;  reg clk, start;  wire [5:0] product;  wire done;  integer k;  Task1\_new MUL (.multiplicand(multiplicand), .multiplier(multiplier),  .clk(clk), .start(start), .product(product), .done(done));  initial begin  #500 $finish; //runs simulation 100 times  end  //clock  initial begin  clk = 0;  for (k = 0; k <= 1000; k = k+1)  begin  #5 clk = 1;  #5 clk = 0;  end  end  //start  initial begin  start = 0;  #30 start = 1;  #10 start = 0;  #100 start = 1;  #10 start = 0;  #100 start = 1;  #10 start = 0;  end  initial begin  multiplicand = 3'b111;  multiplier = 3'b101;  #140  multiplicand = 3'b100;  multiplier = 3'b001;  #110  multiplicand = 3'b111;  multiplier = 3'b100;  end  endmodule |

*Testbench 1*

# **Task 2**

In Task 2, all we were asked to do was physically test the design on the board. We were asked to assign switches 5-7 as the multiplier input, switches 2-4 as the multiplicand input, switch 15 as the clock input, and button U as the start input. My design worked as intended when testing on the board and was verified by the TA.

# **Task 3**

In Task 3, we were asked to modify the design of task 1 to perform a 4-bit by 4-bit unsigned multiplication. We were told to store the multiplicand and multiplier input values in a 32x4 ROM file. We were also asked to use the onboard clock at 100 Mhz as the clock and have the output displayed on 7-segment displays. Sadly, due to errors in my Task 1 code through the lab, I was unable to get to Task 3 to design and implement the task. I later fixed Task 1 with help of the TA but was never able to design and test Task 3.

# **Conclusion**

In conclusion, this lab was more difficult than it first seemed. I had trouble implementing the multiplier with the desired specifications at first. I had multiple solutions that physically worked, but would not give me the correct simulation results. I fully designed a code that worked on the board and displayed the correct simulation with the help of the TA. Overall, this lab was more challenging than I first assumed but was able to figure out most of the lab by the end of lab time.