Lab 1

Justin Roessler

January 16, 2019

1 Executive Summary

The purpose of this lab is to learn how to use verilog and how the clock edge and reset effects the results of q when changing d in a D flip flop. Different cases were added of different time lengths to show that q is only updated on the positive edge of the clock cycle. The Register module is set up to set q to 0 when reset is set, otherwise q is equal to d on a positive clock edge. The lab was successful and demonstrated the significance of being aware of clock cylces. If we wanted to set q to 6 when d is 6, we must let d be set long enough for a positive clock edge to occur so q will be set to d.

2 Test Report

To verify operation of this/these module(s), this lab requires 1 test bench.

1. Register Test Bench

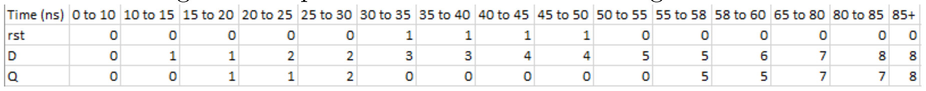
2.1 Register Test Bench

The register test bench contains:

1. Inputs
   (a) clk - the oscillator that is the clock signal on a wire
   (b) reset - a register that always sets Q to 0 when set
   (c) D - a register that is a 64 bit number
2. Outputs
   (a) Q - a wire that is a 64 bit number and will equal D unless a reset is set

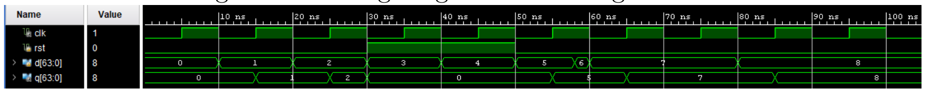
The test bench tested the behavior of the program counter when it was fed different numbers for a differant amount of time. Much of the code and test bench was already prepared before the lab. However, several test cases were

Figure 1: Expected Results Table of the register test.



Time (ns)	0 to 10	10 to 15	15 to 20	20 to 25	25 to 30	30 to 35	35 to 40	40 to 45	45 to 50	50 to 55	55 to 58	58 to 60	65 to 80	80 to 85	85+
rst	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
D	0	1	1	2	2	3	3	4	4	5	5	6	7	8	8
Q	0	0	1	1	2	0	0	0	0	0	5	5	7	7	8

Figure 2: Timing diagram for the register test.



added to the test bench. One of the cases set d to 6 for one fifth of a clock cycle. Since d was set to 6 for less than a clock cylce, Q was never set to 6 because of the absense of a positive clock edge. Also D was set to 7 for 2 clock cycles. In this case, Q was set to 7 for two clock cycles as well.

Operation of the testbench is verified by comparing the Simulation Results with the Expected Results Table. This showed me the register module was operating as expected and the Program Counter was behaving as needed.

3 Code Appendix

Listing 1: Verilog code for implementing a register.

```
`include "definitions.vh"

module register(
    input wire clk,
    input wire reset,
    input  wire [`WORD-1:0] D,
    output reg [`WORD-1:0] Q=`WORD'b0
    );

    always @(posedge(clk),posedge(reset))begin
        if (reset==1'b1)
            Q<=`WORD'b0;
        else
            Q <= D;
    end

endmodule
```

Listing 2: Verilog code for testing a register.

```
`include "definitions.vh"


module register_test;

wire clk;
reg rst;
reg[`WORD - 1:0] d;
wire[`WORD - 1:0] q;

oscillator clk_gen(clk);

register UUT(
    .clk(clk),
    .reset(rst),
    .D(d),
    .Q(q)
    );

initial
begin
    rst = 0;
    d<=`WORD'd0; #`CYCLE;
    d<=`WORD'd1; #`CYCLE;
    d<=`WORD'd2; #`CYCLE;
    rst = 1;
    d<=`WORD'd3; #`CYCLE;
    d<=`WORD'd4; #(`CYCLE);
    rst = 0;
    d<=`WORD'd5; #(`CYCLE*4/5);
    d<=`WORD'd6; #(`CYCLE/4);
    d<=`WORD'd7; #(`CYCLE*2);
    d<=`WORD'd8; #(`CYCLE);
end

endmodule
```