Spring 2021

California State University, Northridge

Department of Electrical & Computer Engineering

Experiment 3

D-Flip-Flop Design

February 11, 2021

ECE 526L

Written By: Jose Luis Martinez

Introduction

In this lab we will be designing a D-Flip-Flop provided in **Fig. 1** using an SR-Latch provided in **Fig. 2.** We are provided with the delay data in **Fig. 3** and it is up to us to figure out how to apply it properly to the design. The purpose of this lab is to set the foundation of creating multiple modules and learn how to properly apply time delays.

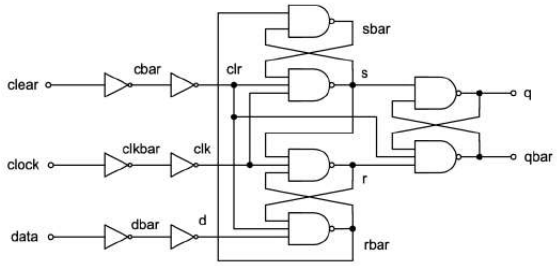


Fig. 1 D-Flip-Flop

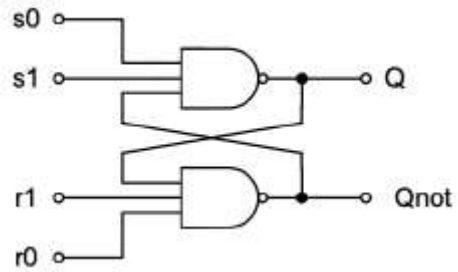


Fig. 2 SR-Latch

Single-input gates: intrinsic delay of 3 ns.
Two-input gates: intrinsic delay of 4 ns.
Three-input gates: intrinsic delay of 5 ns.
Capacitive loading of 0.5 ns for a fanout of one.
Capacitive loading of 0.8 ns for a fanout of two.
Capacitive loading of 1.0 ns for a fanout of three.
2.0 ns loading delay for a primary output.

Fig. 3 Delays

We are also to test this logic circuit using $monitor, $display, $write, and $strobe.

Methodology

For this lab we have 3 main tasks to do which is creating the SR-Latch with proper timing, D-FF with proper timing, and then creating a testbench to test out the D-FF using the compiler directives required.

Before any modules were made, I made a definitions.v file that only contains the ‘timescale and definitions that will be used throughout the files. This definition file has all of the delays from **Fig. 3** and a string that will output the results.

After I created the SR_Latch2.v file which has our SR-Latch module in it. The SR_Latch module has four inputs and two outputs. Inside the module there are two delay parameters initialized to zero and they will be placeholders for our delays. For the logic there are 2x three input NAND gates with their outputs connected to Q and Q_bar connected respectively. The s0 and s1 inputs are connected to the inputs of the NAND gate with the Q output and r0 and r1 inputs are connected to the inputs of the NAND gate with the Q_bar output. For the final inputs, their outputs simply go to the other NAND gate’s input.

After the SR-Latch is set up, I made another file, D_FF.v which contains the D_FF module. This module has q and q_bar as its outputs and clock, data, and clear as its inputs. Inside the module we have wire signals cbar, clkbar, dbar, clr, clk, d, sbar,s, r, and rbar to help connect all of our modules together. So looking at **Fig. 1** we can see that we need to make six NOT gates and three SR-Latches. So for the NOT gates we need to create a pair for clear, clock, and data. For the pair connected to the clear, the pair is connected like this: clear -> NOT1 -> cbar -> NOT2 -> clr. For the pair connected to the clock, the pair is connected like this: clock -> NOT3 -> clkbar -> NOT4 -> clk. For the pair connected to the data, the pair is connected like this: data -> NOT5 -> dbar -> NOT6 -> d. The next part is connecting the SR-Latches with each other and with the NOT gate wires. The first SR-Latch is the top one that will control the s for the flip-flop. This module is initialized with the proper wire/signals as shown in **Fig. 1** and immediately after, we use the defparam command to set the proper values of the delays to that SR-Latch. The other SR_Latches are then initialized the same way.

The testbench file is then made to test if our logic circuit works. By initializing the D_FF module we can then provide it with signals to test however many test cases we want. I made a Lab3_tb module with an instance of D_FF and connected it to registers and wires. I used the $monitor, $display, $write, and $strobe compiler directives to display the data to the log file.

Results

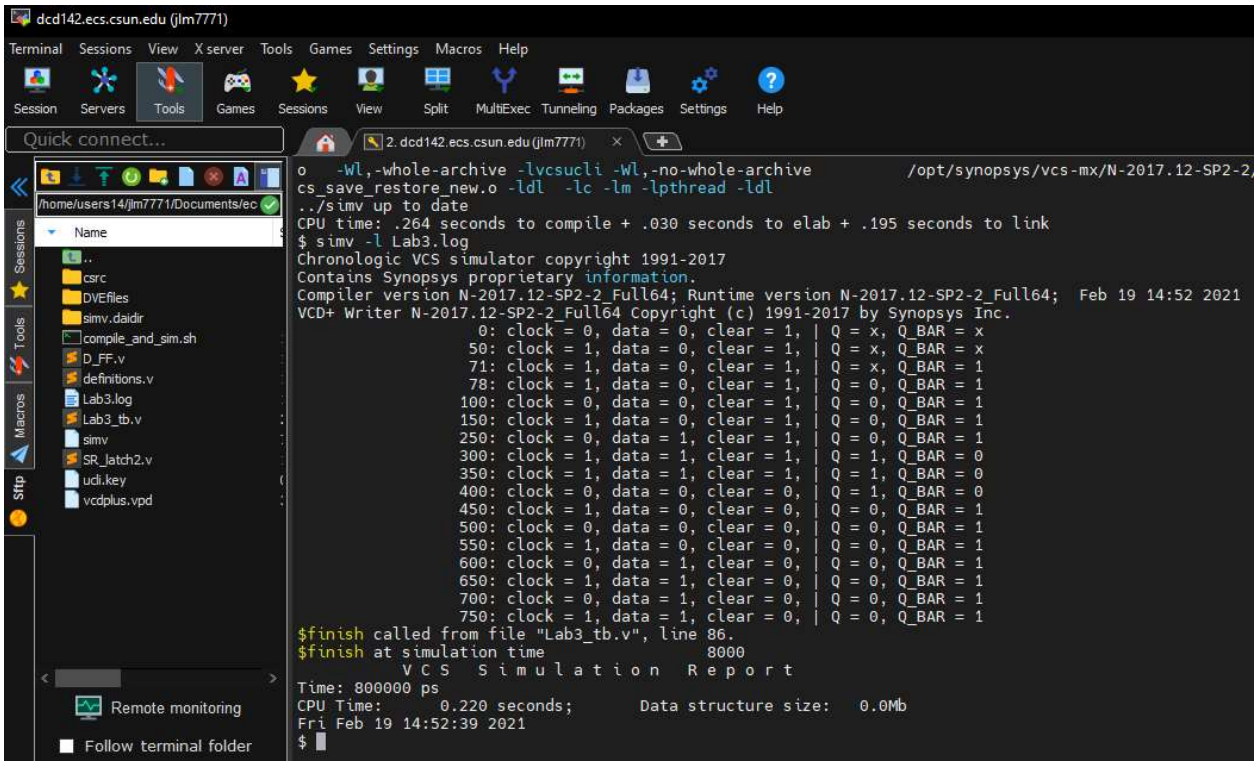


Fig. 4 simv output

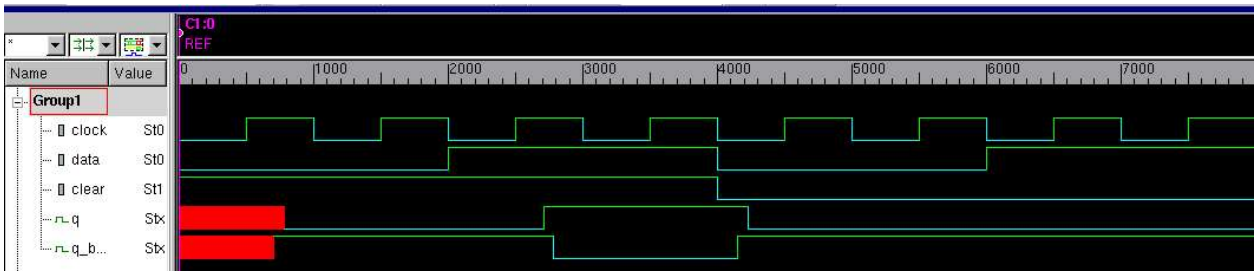


Fig 5. Waveforms

```
/*****************************************************************************
***                                                                        ***
*** ECE 526 L Experiment #3            Jose Luis Martinez, Spring, 2021    ***
***                                                                        ***
*** Experiment 3 - DFF                                                    ***
***                                                                        ***
*****************************************************************************
*** Filename: definitions.v Created by Jose Luis Martinez, Febuary 11, 2021 ***
***                                                                        ***
*****************************************************************************/

`timescale 1ns/100ps

`define FAN_OUT_1     0.5
`define FAN_OUT_2     0.8
`define FAN_OUT_3     1
`define TIME_DELAY_1  3
`define TIME_DELAY_2  4
`define TIME_DELAY_3  5
`define PRIMARY_OUT   2
`define MONITOR_STR_1 "%d: clock = %b, data = %b, clear = %d, | Q = %d, Q_BAR = %d"
`define MONITOR_STR_1_W "%d: clock = %b, data = %b, clear = %d, | Q = %d, Q_BAR = %d\n"
```

Fig 6. definitions.v

```
/*********************************************************************************
***                                                                            ***
*** ECE 526 L Experiment #3            Jose Luis Martinez, Spring, 2021    ***
***                                                                            ***
*** Experiment 3 - DFF                                                       ***
***                                                                            ***
*********************************************************************************
*** Filename: SR_latch2.v   Created by Jose Luis Martinez, Febuary 11, 2021 ***
***                                                                            ***
*********************************************************************************/

`include "definitions.v"


module SR_latch2(Q, Q_bar, s0, s1, r0, r1);
    output Q, Q_bar;
    input s0, s1, r0, r1;

    parameter DELAY_1 = 0;
    parameter DELAY_2 = 0;

    nand #(DELAY_1) NAND1(Q, s0, s1, Q_bar);
    nand #(DELAY_2) NAND2(Q_bar, r0, r1, Q);

endmodule
```

Fig. 7 SR_latch2.v

```
/*****************************************************************************
***                                                                        ***
*** ECE 526 L Experiment #3            Jose Luis Martinez, Spring, 2021    ***
***                                                                        ***
*** Experiment 3 - DFF                                                    ***
***                                                                        ***
*****************************************************************************
*** Filename: D_FF.v        Created by Jose Luis Martinez, Febuary 11, 2021 ***
***                                                                        ***
*****************************************************************************/

`include "definitions.v"

module D_FF(q, q_bar, clock, data, clear);
    output q, q_bar;
    input clock, data, clear;
    wire cbar, clkbar, dbar, clr, clk, d, sbar,s, r, rbar;

    not #(`TIME_DELAY_1 + `FAN_OUT_1) NOT1(cbar, clear);
    not #(`TIME_DELAY_1 + `FAN_OUT_3) NOT2(clr, cbar);
    not #(`TIME_DELAY_1 + `FAN_OUT_1) NOT3(clkbar, clock);
    not #(`TIME_DELAY_1 + `FAN_OUT_2) NOT4(clk, clkbar);
    not #(`TIME_DELAY_1 + `FAN_OUT_1) NOT5(dbar, data);
    not #(`TIME_DELAY_1 + `FAN_OUT_1) NOT6(d, dbar);

    SR_latch2 srl1( .Q(sbar), .Q_bar(s), .s0(rbar), .s1(1'b1), .r0(clr), .r1(clk));
    defparam srl1.DELAY_1 = `TIME_DELAY_3 + `FAN_OUT_1;
    defparam srl1.DELAY_2 = `TIME_DELAY_3 + `FAN_OUT_3;

    SR_latch2 srl2(.Q(r), .Q_bar(rbar), .s0(clk), .s1(s), .r0(clr), .r1(d));
    defparam srl2.DELAY_1 = `TIME_DELAY_3 + `FAN_OUT_2;
    defparam srl2.DELAY_2 = `TIME_DELAY_3 + `FAN_OUT_2;

    SR_latch2 srl3(.Q(q), .Q_bar(q_bar), .s0(s), .s1(1'b1), .r0(clr), .r1(r));
    defparam srl3.DELAY_1 = `TIME_DELAY_3 + `FAN_OUT_1 + `PRIMARY_OUT;
    defparam srl3.DELAY_2 = `TIME_DELAY_3 + `FAN_OUT_1 + `PRIMARY_OUT;

endmodule
```

Fig. 8 D_FF.v

```
/*********************************************************************************
***                                                                            ***
*** ECE 526 L Experiment #3            Jose Luis Martinez, Spring, 2021    ***
***                                                                            ***
*** Experiment 3 - DFF                                                       ***
***                                                                            ***
*********************************************************************************
*** Filename: LAB3_tb.v     Created by Jose Luis Martinez, Febuary 11, 2021 ***
***                                                                            ***
*********************************************************************************/

`include "definitions.v"

module Lab3_tb();
    reg clock, data, clear;
    wire q, q_bar;
    D_FF dff(.q(q), .q_bar(q_bar), .clock(clock), .data(data), .clear(clear));

    initial begin
        $monitor(`MONITOR_STR_1, $time, clock, data, clear, q, q_bar);
    end

    initial begin
        $vcdpluson;
            clock = 1'b0;
            data = 1'b0;
            clear = 1'b1;
        #50 clock = 1'b1;
            data = 1'b0;
            clear = 1'b1;
        #50 clock = 1'b0;
            data = 1'b0;
            clear = 1'b1;
        #50 clock = 1'b1;
            data = 1'b0;
            clear = 1'b1;
        #50 clock = 1'b0;
            data = 1'b1;
            clear = 1'b1;
        $monitoroff;
        #50 $display(`MONITOR_STR_1, $time, clock, data, clear, q, q_bar);
            clock = 1'b1;
```

```
            data = 1'b1;
            clear = 1'b1;
        #50 $write(`MONITOR_STR_1_W, $time, clock, data, clear, q, q_bar);
            clock = 1'b0;
            data = 1'b1;
            clear = 1'b1;

        #50 clock = 1'b1;
            data = 1'b1;
```

```
52              clear = 1'b1;
53              $strobe(`MONITOR_STR_1, $time, clock, data, clear, q, q_bar);
54          #50 clock = 1'b0;
55              data = 1'b0;
56              clear = 1'b0;
57              $strobe(`MONITOR_STR_1, $time, clock, data, clear, q, q_bar);
58          #50 clock = 1'b1;
59              data = 1'b0;
60              clear = 1'b0;
61              $strobe(`MONITOR_STR_1, $time, clock, data, clear, q, q_bar);
62          #50 clock = 1'b0;
63              data = 1'b0;
64              clear = 1'b0;
65              $strobe(`MONITOR_STR_1, $time, clock, data, clear, q, q_bar);
66          #50 clock = 1'b1;
67              data = 1'b0;
68              clear = 1'b0;
69              $strobe(`MONITOR_STR_1, $time, clock, data, clear, q, q_bar);
70          #50 clock = 1'b0;
71              data = 1'b1;
72              clear = 1'b0;
73              $strobe(`MONITOR_STR_1, $time, clock, data, clear, q, q_bar);
74          #50 clock = 1'b1;
75              data = 1'b1;
76              clear = 1'b0;
77              $strobe(`MONITOR_STR_1, $time, clock, data, clear, q, q_bar);
78          #50 clock = 1'b0;
79              data = 1'b1;
80              clear = 1'b0;
81              $strobe(`MONITOR_STR_1, $time, clock, data, clear, q, q_bar);
82          #50 clock = 1'b1;
83              data = 1'b1;
84              clear = 1'b0;
85              $strobe(`MONITOR_STR_1, $time, clock, data, clear, q, q_bar);
86          #50 $finish;
87      end
88
89
90  endmodule
```

Fig. 9 Lab3_tb.v

Analysis

From our waveform in **Fig. 5** we can see that the outputs of the d-flip-flop are correct. The clear is active low meaning that when the input to the clear is LOW, the d-flip-flop will reset to 0. The time delays each of the NOT gates connected to the inputs have a time delay of 3.5ns. The second NOT gate for the clear has a delay of 4ns. The second NOT gate for the clock has a delay of 3.8ns. The second NOT gate for the data has a delay of 3.5ns. For the first SR_Latch, the delay for the top NAND gate is 5.5ns and the bottom NAND gate has a delay of 6ns. For the second SR_Latch, the delay for the top NAND gate is 5.8ns and the bottom NAND gate has a delay of 5.8ns. For the last SR_Latch, the delay for the top NAND gate is 7.5ns and the bottom NAND gate has a delay of 7.5ns.

For the testbench file I made sure to include the $monitor, $strobe, $write, and $display compiler directives to show what they do.

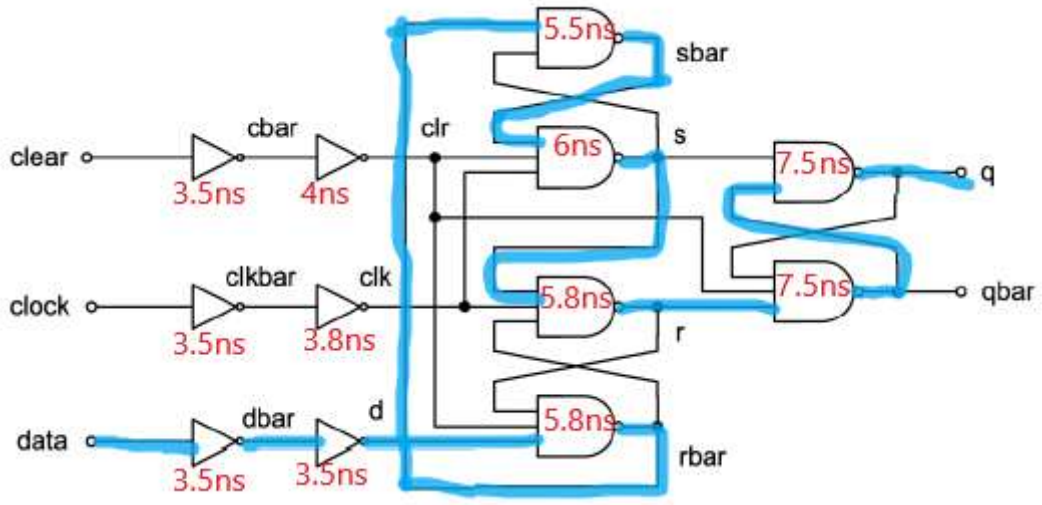
1. $monitor will print the specified string every time one of the specified variables changes.
2. $display will print the specified string at the point that it is called.
3. $write will print the specified string at the point that it is called but will not append a newline.
4. $strobe will print the specified string but will execute last among the statements of the same time.

Conclusion

In conclusion we designed a d-flip-flop using three SR-Latches and six NOT gates. We learned how to apply the proper timing using the parameter and defparam types in Verilog. We also learned the differences between $monitor, $strobe, $write, and $display.

Questions

1- What's the critical path (longest delay) of this design?



Taking the path above through NOT5 -> NOT6 -> NAND4 -> NAND1 -> NAND2 -> NAND3 -> NAND6 -> NAND5 gives us a delay of **45.1ns** which gives us the critical path delay.

2- What is the maximum operating frequency for your circuit?

If our critical path delay is **45.1ns** then we can use $Freq = \frac{1}{T}$ giving us a maximum frequency of around 22.2MHz.

Academic Dishonesty

Submitting any report that is not entirely your own work is a form of academic dishonesty and will not be tolerated. Each and every lab report must include the following statement, signed and dated by the student. Lab reports without the statement will be summarily rejected.

I hereby attest that this lab report is entirely my own work. I have not copied either code or text from anyone, nor have I allowed or will I allow anyone to copy my work.

Name (printed) _____

Name (signed) _____ Date _____