Victor Yuan

November 10, 2016

Lab 11

Introduction:

This lab will examine master-slave J-K flip-flops. The first of three parts will examine how J,K,Q and Q+ interact, next the interaction of CL with variables J,K,Q, and CK, finally what happens when J and K change while CK is high.

Team Member Responsibilities:

Lab partner: Noah Settler

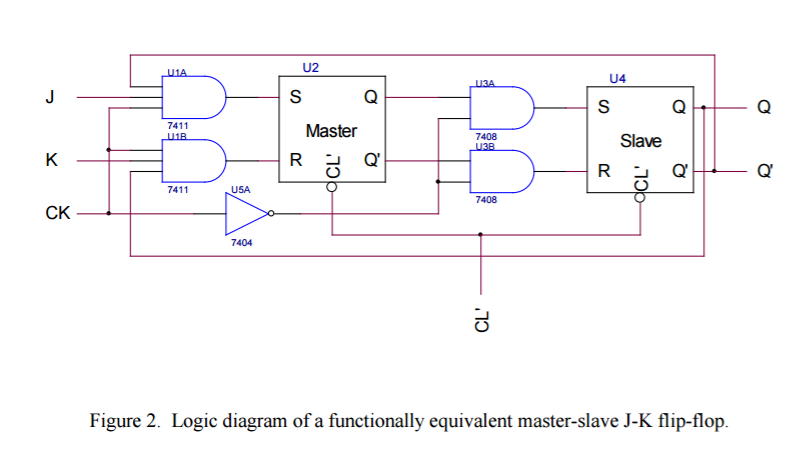
Work was divided evenly, I did some of the coding Noah did some of the Code.

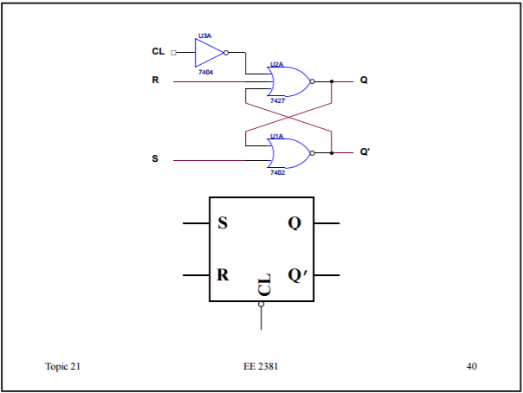
Materials:

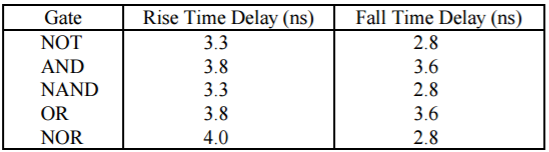
This lab was done completely in Verilog, and Putty on a Lab Computer

Procedure:

Part 1:



First convert Figure 2 into Verilog code. The logic gates for U2 and U4 are shown in Topic 21, slide 40 on the website and shown below, use it to write the Verilog simulation.



Remember to use the proper delay times listed in the table above. Then develop the stimulus pattern for the inputs J, K, CL’ and CK. Ck should start at 0 and cycle on its own every 100 ns. CL’ should start at 0 to reset the circuit then remain 1 until the simulation ends. J and K should cycle between 1 and 0 every 300 ns. Run the simulation and determine the partial Q+ truth table.

Part 2:

Now run the simulation again only with Cl turned on (set to 0), after this simulation the whole Q+ truth table should have been derived.

Part 3:

For part 3 create a separate stimulus pattern. First reset Q to 0 with the following partial pattern:

#300 CL= 1; J=0; K=0; CK=0;

#300 CL= 0; J=0; K=0; CK=0;

The rest of the simulation should follow the following sequence with CL set to high.

|  |  |
| --- | --- |
| Time | Change that should happen |
| Every  300ns | CK -> 1 |
| J -> 1 |
| J -> 0 |
| K -> 1 |
| K -> 0 |
| CK->0 |
| //the above is the first pulse |
| CK-> 1 |
| K-> 1 |
| K-> 0 |
| J-> 1 |
| J-> 0 |
| CK-> 0 |

Run the simulation and screenshot the results.

Questions:

I. Simulations:

1. Derive the truth-table for the master-slave J-K flip-flop shown in Fig. 2 under the assumptions that J, K and Q are the inputs of the truth-table, Q+ is the desired output of the truth-table, the clock CK has a single positive pulse, the J-K inputs are stable while the clock CK is high and the clear CL’ is “off”.

|  |  |  |  |
| --- | --- | --- | --- |
| J | K | Q | Q+ |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

2. Re-derive the truth-table in question I-1 above under the assumption that the clear is CL’ “on”.

|  |  |  |  |
| --- | --- | --- | --- |
| J | K | Q | Q+ |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 |

3. Based upon the above two truth-tables, does the circuit in Fig. 2 correctly realize the function table shown in Table 1?

Yes, the truth table supports table 1 outputs.

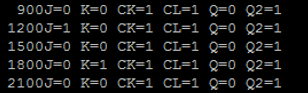
4. What is the relationship between the clear and clock? Which has precedence over the other?

They are independent of each other CK will turn on and off no matter what CL does.

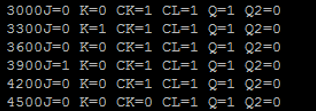
Clear has precedence over Clock since whenever close is low CK, J, K are all X, don’t cares.

5. Does the flip-flop shown in Fig. 2 show any evidence of “catching a pulse” on the J-K inputs when the clock is high? If so, please state explicitly the conditions under which it will “catch a pulse”.

Yes, it catches the pulse and freezes it whenever CK is high.



As seen above when CK is high, no matter what the J, K and Q inputs are the Q+ is always the same



Another example of pulse catching.

Conclusions:

1. How do the truth-tables for the next-state of a master-slave J-K flip-flop compare as derived from Verilog simulations and theory? Are there any differences?

The Given Truth table assumes that Q is an output while the derived truth table assumes that Q is an input.

1. Is the relationship between the clear and clock the same for the Verilog simulations and theory? Are there any differences?

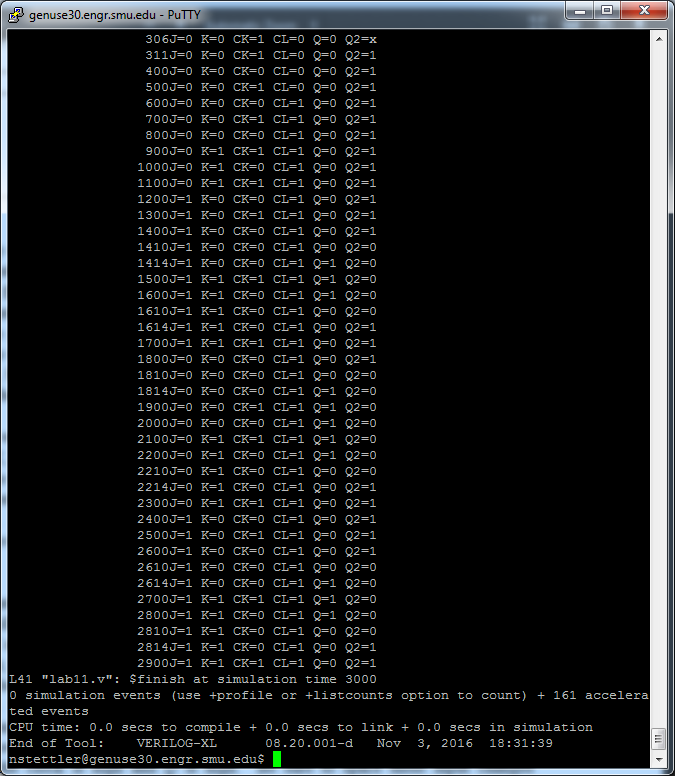
They are the same in both simulation and theory.

1. Is the “pulse catching” behavior the same for the Verilog simulations and theory? Are there any differences?

Pulse catching behavior is the same for simulation and theory.

All three areas of interest was studied in this lab.

Raw Data:



module main;

reg J,K,CK,CL;

wire Q, Q2;

and #(3.8,3.6) U1a(a,J,Q2,CK);

and #(3.8,3.6) U2a(b,CK,K,Q);

not #(3.3,2.8) U3a(c,CK);

//-----------------------

not #(3.3,2.8) U4a(d,CL);

nor #(4.0,2.8) U5a(e,d,b,f);

nor #(4.0,2.8) U6a(f,a,e);

//-----------------------

and #(3.8,3.6) U7a(g,c,e);

and #(3.8,3.6) U8a(h,c,f);

//----------------------

not #(3.3,2.8) U9a(i,CL);

nor #(4.0,2.8) U10a(Q,i,h,Q2);

nor #(4.0,2.8) U11a(Q2,g,Q);

initial CK=0;

always #100 CK=~CK;

initial

begin

$monitor($time, "J=%b K=%b CK=%b CL=%b Q=%b Q2=%b",J,K,CK,CL,Q,Q2);

#300 CL= 0; J=0; K=0;

#300 CL= 1; J=0; K=0;

#300 CL= 1; J=0; K=1;

#300 CL= 1; J=1; K=0;

#300 CL= 1; J=1; K=1;

#300 CL= 1; J=0; K=0;

#300 CL= 1; J=0; K=1;

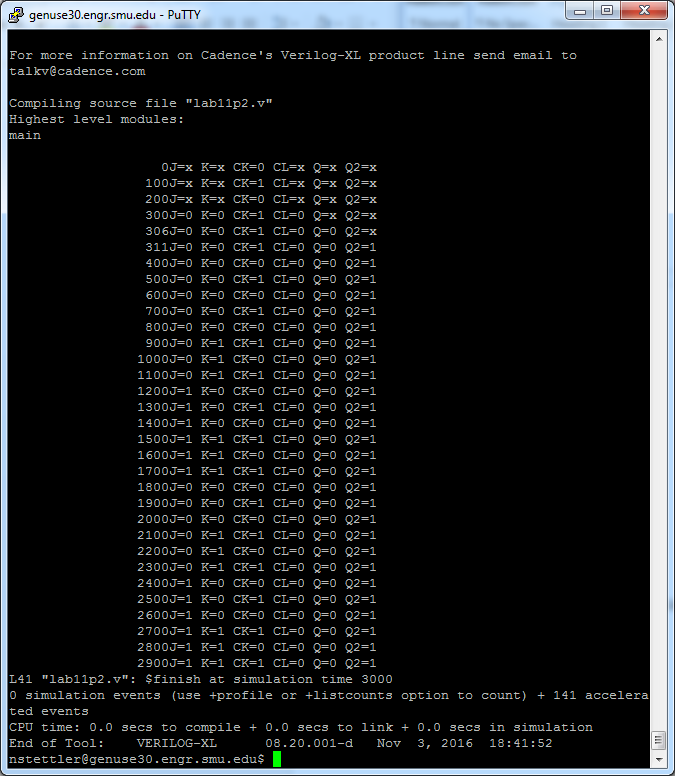
#300 CL= 1; J=1; K=0;

#300 CL= 1; J=1; K=1;

#300 $finish;

end

endmodule



//part2

module main;

reg J,K,CK,CL;

wire Q, Q2;

and #(3.8,3.6) U1a(a,J,Q2,CK);

and #(3.8,3.6) U2a(b,CK,K,Q);

not #(3.3,2.8) U3a(c,CK);

//-----------------------

not #(3.3,2.8) U4a(d,CL);

nor #(4.0,2.8) U5a(e,d,b,f);

nor #(4.0,2.8) U6a(f,a,e);

//-----------------------

and #(3.8,3.6) U7a(g,c,e);

and #(3.8,3.6) U8a(h,c,f);

//----------------------

not #(3.3,2.8) U9a(i,CL);

nor #(4.0,2.8) U10a(Q,i,h,Q2);

nor #(4.0,2.8) U11a(Q2,g,Q);

initial CK=0;

always #100 CK=~CK;

initial

begin

$monitor($time, "J=%b K=%b CK=%b CL=%b Q=%b Q2=%b",J,K,CK,CL,Q,Q2);

#300 CL= 0; J=0; K=0;

#300 CL= 0; J=0; K=0;

#300 CL= 0; J=0; K=1;

#300 CL= 0; J=1; K=0;

#300 CL= 0; J=1; K=1;

#300 CL= 0; J=0; K=0;

#300 CL= 0; J=0; K=1;

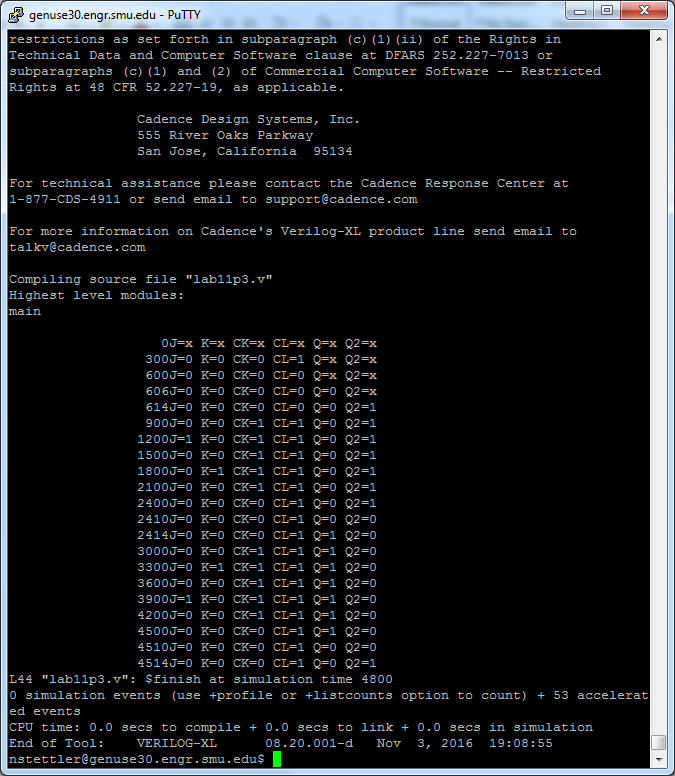
#300 CL= 0; J=1; K=0;

#300 CL= 0; J=1; K=1;

#300 $finish;

end

endmodule



//part3

module main;

reg J,K,CK,CL;

wire Q, Q2;

and #(3.8,3.6) U1a(a,J,Q2,CK);

and #(3.8,3.6) U2a(b,CK,K,Q);

not #(3.3,2.8) U3a(c,CK);

//-----------------------

not #(3.3,2.8) U4a(d,CL);

nor #(4.0,2.8) U5a(e,d,b,f);

nor #(4.0,2.8) U6a(f,a,e);

//-----------------------

and #(3.8,3.6) U7a(g,c,e);

and #(3.8,3.6) U8a(h,c,f);

//----------------------

not #(3.3,2.8) U9a(i,CL);

nor #(4.0,2.8) U10a(Q,i,h,Q2);

nor #(4.0,2.8) U11a(Q2,g,Q);

initial

begin

$monitor($time, "J=%b K=%b CK=%b CL=%b Q=%b Q2=%b",J,K,CK,CL,Q,Q2);

#300 CL= 1; J=0; K=0; CK=0;

#300 CL= 0; J=0; K=0; CK=0;

#300 CL= 1; J=0; K=0; CK=1;

#300 CL= 1; J=1; K=0; CK=1;

#300 CL= 1; J=0; K=0; CK=1;

#300 CL= 1; J=0; K=1; CK=1;

#300 CL= 1; J=0; K=0; CK=1;

#300 CL= 1; J=0; K=0; CK=0;

#300 CL= 1; J=0; K=0; CK=0;

#300 CL= 1; J=0; K=0; CK=1;

#300 CL= 1; J=0; K=1; CK=1;

#300 CL= 1; J=0; K=0; CK=1;

#300 CL= 1; J=1; K=0; CK=1;

#300 CL= 1; J=0; K=0; CK=1;

#300 CL= 1; J=0; K=0; CK=0;

#300 $finish;