Lab 5 - latches, flip flops, counters

Digital Systems and Microcontrollers, IIIT-H Monsoon'24

26 September, 2024

1 Deliverable details

- Take inputs through serial monitor, slide switch or push button as per your TA's guidance.
- Clearly annotate/label the inputs/LEDs. Give the wires different colors for further clarity.
- Submit 3 separate circuits, one for each part of the experiment. Preferably each circuit should be in a separate Tinkercad project.

2 Objective

After completing this lab experiment, you will have built an RS latch, a JK Master-Slave Flip-Flop, and a 4-bit Up-Down Counter in Tinkercad.

3 Part A: SR Latch

3.1 Experiment

- Assemble a NOR latch using two NOR gates as shown in figure 1.
- R and S are to be taken as input from user.
- Display Q and Q' outputs on two LEDs.

3.2 Observation

• Observe and tabulate the sequence of Q and Q' in response to the following input sequence: S R = 01, 00, 10, 00, 01, 10, 01, 00, 11, 00, 10, 11, 00, 01, 11, 00. • When given the above inputs, explain till when the latch can be expected to operate correctly and why. Cross check your theoretical understanding with the observed behaviour of the latch.

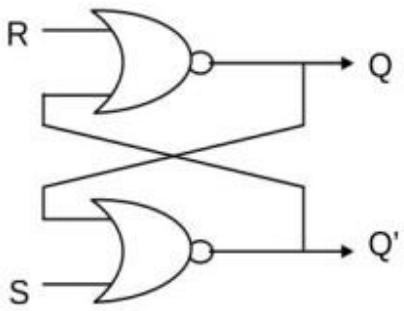

Figure 1: Schematic diagram for one RS latch

3.3 Tinkercad Simulation - Issue and Fix

Tinkercad simulation will work at a very slow speed when we criss-cross outputs of two gates to each other's inputs, like we do in a SR Latch. You can confirm your **simulation is stuck** when the **"Simulator time"** displays **00:00:00.000** for a long time.

We know that the **initial state** of the SR Latch circuit cannot be determined *before* running the circuit. In real life, one of the two NOR gates will always give an output earlier than the other gate. This quicker gate will end up determining the initial state of the state. This is very well highlighted in this Ben Eater video (timestamped at 11:32)

However, in Tinkercad, because of the slowness of the simulation, both NOR gates will remain stuck! Therefore, you need to give an **initial nudge** to **any one** of the NOR gates. Here, by "nudge", we mean temporarily setting one of its indeterminate inputs to 5V, so that they'll acquire a fixed initial state. Once the initial state of our circuit is fixed, we can then proceed with our usual experiment.

In this Tinkercad project, convince yourself how the push button at the bottom acts as a nudge. Start the simulation, and first confirm that the simulation is currently stuck. Now, push (and hold for 1 second) the pushbutton, and note that the simulation is not stuck anymore.

You can confirm your simulation is **not stuck** if the "Simulator time" is increasing continuously.

4 Part B: JK Master-Slave Flip-Flop

4.1 Experiment

- The JK master-slave flip flop consists of two latches: a master latch and a slave latch. The master latch changes its values on the leading edge of the clock, whereas the slave latch changes its values on the trailing edge of the clock.

- Add a ”Power supply” to your working area. Make sure Power supply is set to 5V and 5A

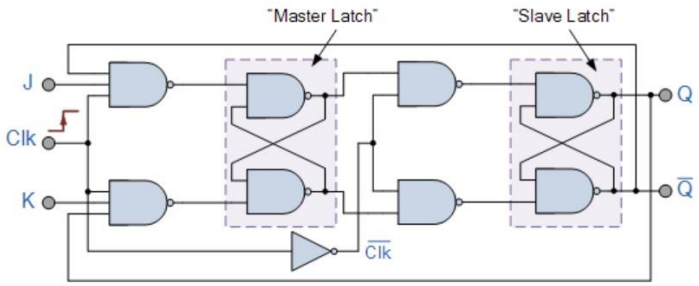

Figure 2: Schematic diagram for one master slave JK flip flop, figure extracted from [1]

J	**K**	**ACTION**	Q_{n+1}
0	0	HOLD	Q_n
0	1	CLEAR	0
1	0	SET	1
1	1	TOGGLE	Q'_n

Table 1: JK master slave flip flop function table

- Assemble NAND gates as shown in the figure 2 to form this flip flop. Ideally, you can complete this circuit using only one 74HC10 (triple input NAND) gate and two 74HC00 gates (two input NAND). However, feel free to use more gates for convenience.

4.2 Experimental advice!

Do NOT make the full circuit and then debug it. It will take you hours that way. Instead, preferably do the experiment in this order:

1. Make the master latch and test it
2. Make the slave latch and test it
3. Forward outputs from master latch to slave latch and test the combined result.
4. Send the outputs of slave latch back to master latch and implement the nudge switchas described previously.

In all the above cases, use a 5V signal as placeholder if some signal input is not ready yet.

4.3 Observations

Verify experimentally that the JK master slave flip flop satisfies the following function table given in table 1.

Observe and tabulate the sequence of Q and Q' in response to the following input sequence: J K = 10, 00, 01, 10, 01, 00, 11, 00, 10, 11, 00, 01, 11, 00.

5Part C: 4-bit Up-Down Counter (Optional)

5.1 Starter code(Optional)

You are provided a starter code for this section of the experiment. Clickable link. Look for the section //// YOUR CODE STARTS HERE and //// YOUR CODE ENDS HERE: this is where you have to write your code. The starter code demonstrates how to blink an LED in set intervals of time using t.oscillate.

The starter code demonstrates how to blink an LED in set intervals of time using t.oscillate.

In desktop versions of Arduino, you can externally add any library like Timer.h. However, Tinkercad only supports some pre-defined libraries which excludes Timer.h. **Therefore, we'll not be implementing Tinkercad version for this part**. Implementing this part is optional.

5.2 Experiment

- Using the Timer library, implement a 4-bit counter. The bit outputs of the 4-bit ripple counter will be represented by LEDs. (One LED for each bit)
- Initialize a Timer t and use t.oscillate function to toggle the pin values in a predefined time period (each pin will have a different time period)
- **Down counter**: once the ripple counter reaches 15 (1111), make it go down to 0. You will need to stop existing timers using t.stop. You can use t.every to fire a function after a set-interval of time (hint: 2^4 time units) that will do two tasks: stop all timers and restart them in opposite direction.

5.3 Observation

The ripple counter first goes UP from 0 (0000) to 15 (1111), then goes DOWN from 15 to 0, then goes UP, and this cycle repeats until the simulation is stopped.

5.4 Note

Please increment the ripple counter in time intervals of 200-250ms instead of 1000ms, so that observing the full ripple counter cycle is faster.

6 Appendix: Timer documentation

You only need the oscillate, every, and stop methods on the Timer class. See its documentation on the Arduino website (click here)

7 References

1. Electronics Tutorials for the flip flop diagram: page link