CAD 4: Master-slave Flip Flop

EE 476 | University of Washington

Notes

1. As with the previous cad's, pin should be named as in the `specifications.json` file (and this document). likewise, all measurements are reported in the `specifications.json` file, and any plots or short answer questions go in the lab report.

2. **From this CAD onward, assume ideal input signals (including the clock) are buffered before reaching your circuit.** The buffer is composed of two inverters, both with size W_pmos = 0.4um and W_nmos = 0.3um. You will still turn in a circuit with exposed pins, but during tests and for measurements, you should send ideal signals to the ideal (schematic-level) buffers, which then feed into your circuit. The energy and power of the driving buffer should not be considered during measurements.

Setup

Use the same general directory structure as with CAD's 1-3, review *Tutorial 1* for details.

CAD 4 Overview

In this CAD, students will build and characterize a keystone of sequential designs, the flip flop. Students create and test a flip flop schematic in Part 1, and observe the effects of supply voltage on the flip flop's timing characteristics. Parts 2 and 3 involve the design layout and subsequent simulation. Compared to previous CAD assignments, the flip flop adds significant complexity in both DRC/LVS verification, and the measurement of performance metrics.

1 Flip Flop Schematic

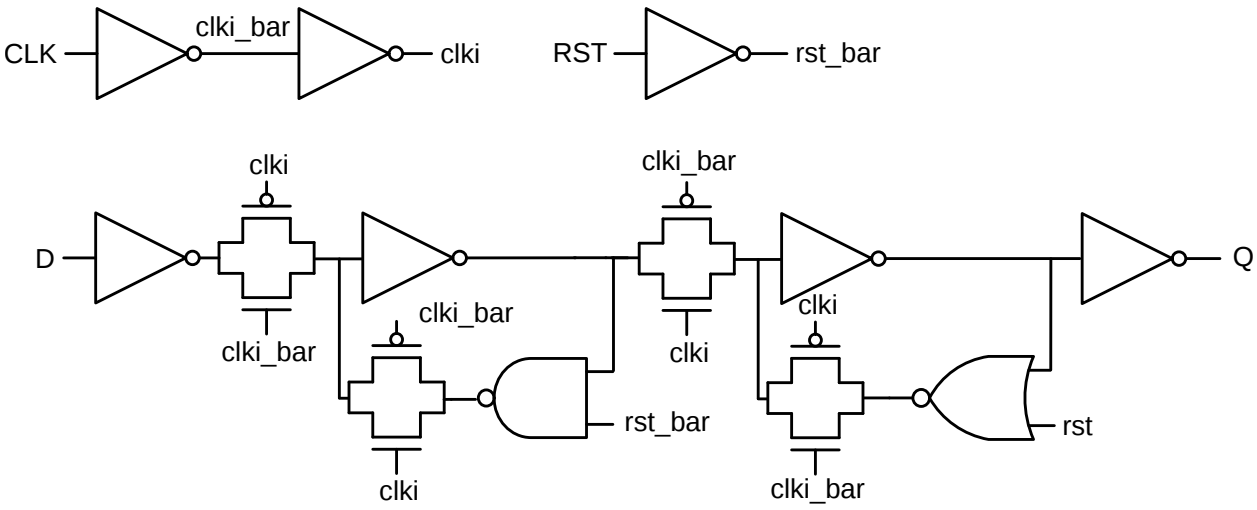

Figure 1

a. Create the schematic for the master-slave flip flop in Figure 1, and use it to build the circuit in Figure 2. Students are free to choose device sizes, create additional standard cells, and incorporate multiple levels of hierarchy (so long as the top-level pins and circuit name are unchanged). Call the flip flop `DFAR` ("D-flip flop with asynchronous reset"), and the circuit in Figure 2 `loaded_flip_flop`.

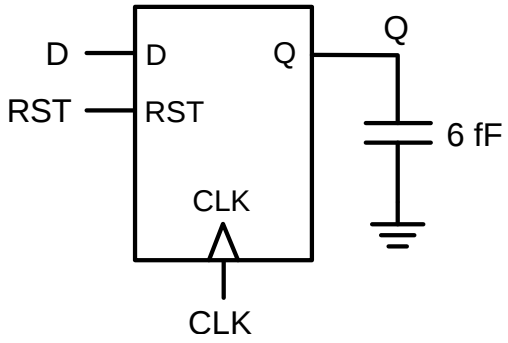

Figure 2

b. Collect the measurements listed in Table 1 for the `loaded_flip_flop` circuit, **with VDD at both 0.8V and 1.2V**, and a **clock period of 2n**. When making measurements, buffer your ideal HSPICE voltage inputs as shown in Figure 3 to mimic realistic inputs. Measurements involving `D`, `RST` or `CLK` should be made at the actual `D`, `RST` and `CLK` pins - for example, `t_cq` refers to the elapsed time from when `CLK` (as opposed to `CLK_ideal`) reaches 50% to when `Q` reaches 50%. Be sure to remove the input buffers and turn in only the `.ckt` file for Figure 2 after completing the measurements. Note: Conventionally, the size of an inverter is written as W_pmos/W_nmos.

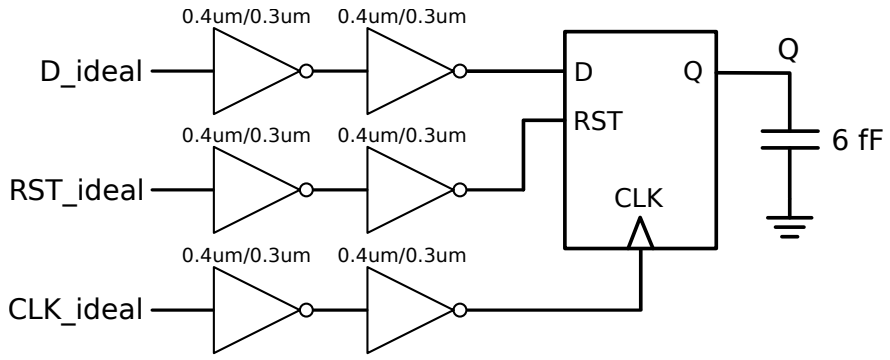

Figure 3

For this CAD, the setup-time is defined as the time-difference between the data arrival and the following clock edge that causes a delay of 10ps at the flip flop output ("10ps pushout"), compared to when the data arrives "safely early[1]." Setup-time is a measure of how early the input data must arrive *before* the rising clock edge. For setup-time, the "rise" and "fall" are with respect to transitions of the output.

Hold-time is defined as the more stringent of two criteria, the "10ps pushout" and a "10% glitch" criteria. The"10ps pushout time" refers to the time-difference between the rising clock edge and a following change in the input data that causes a delay of 10ps at the flip flop output, compared to when the change occurs "safely late." The "10% glitch time" is the time-difference between the rising edge of the clock and a change in the input data that causes the output data to change by $0.1 \times VDD$. In other words, hold-time is a measure of how long the input data must be "held steady" *after* the clock edge. As with setup-time, the "rise" and "fall" are with respect to the output.

Measurement	**Description**
`t_cq_rise`	T_{CQ} (clock-to-q delay) for output-rise transitions
`t_cq_fall`	T_{CQ} (clock-to-q delay) for output-fall transitions
`t_su_rise`	T_{SU} (setup-time) for output-rise transitions
`t_su_fall`	T_{SU} (setup-time) for output-fall transitions
`t_h_rise`	T_{HOLD} (hold-time) for output-rise transitions
`t_h_fall`	T_{HOLD} (hold-time) for output-fall transitions
`en_deliv_steady`	The energy delivered by VDD during one clock period when `Q=0`, `D=0`
`en_deliv_switch`	The energy delivered by VDD during one clock period when `Q=0`, `D=1`

Table 1

Delivery

a.-b. The schematic netlist for `loaded_flip_flop`, and the measurements in Table 1

[1]Review the lecture slides on timing elements for a more precise description

2 Flip Flop Layout

a. Create the layout for the `DFAR` circuit in Figure 1 from Part 1. Students are encouraged to create new cells to support different gate sizes and drive strengths, as well as incorporate multiple levels of hierarchy. As design complexity grows, it is increasingly important to use hierarchy not only for schematic and layout readability, but also for ease of verification. Using a divide-and-conquer approach for DRC and LVS can easily make the difference between a single-pass inspection over a few dozen layout shapes, and multiple inspections over hundreds of shapes. *In order to implement Part 5 of this CAD effectively, the flip flop you make now must be no wider than 5.0 um. Standard cells can be stacked vertically by flipping one and overlapping the* `vdd` *or* `vss` *regions.*

b. Collect the same measurements as in Part 1, but for the post-layout `loaded_flip_flop` (and only at VDD=1.2V). For the parasitics extraction, use `C+CC`.

Delivery

a. The netlist for the post-layout `loaded_flip_flop`, and DRC/LVS reports

b. Measurements from Table 1 for the post-layout `loaded_flip_flop`

3 Flip Flop Layout II

a. Collect the same measurements from Parts 1 and 2, but for the flip-flop extracted with `R+C+CC` parasitics.

Delivery

a. The netlist for the `loaded_flip_flop` extracted with `R+C+CC`, and the corresponding measurements from Table 1

4 LFSR Schematic

A Linear Feedback Shift Register (LFSR) is a clock-driven device to generate a sequence of psuedo-random numbers. In the context of Built-In Self Testing, a LFSR can be used to create input test data for a device under test and we can create another circuit to verify that the outputs match what's expected. Whereas the HSPICE testing we've done so far verifies design correctness, the LFSR approach can verify that a physical device was fabricated with no errors. This testing system will be used in future CADs, so we first need a working LFSR.

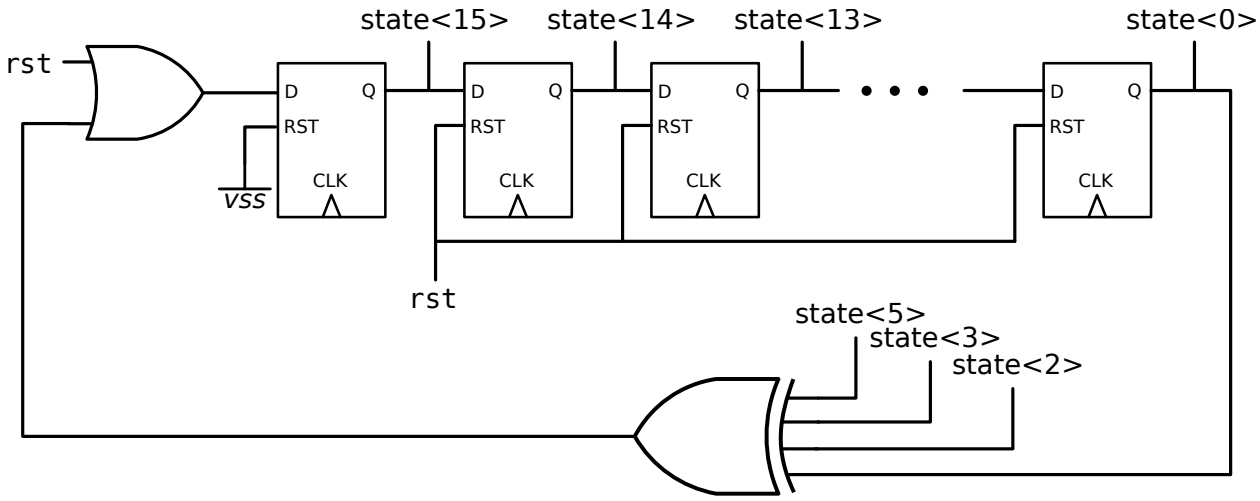

Figure 4

a. Create the schematic for the LFSR in Figure 4. `state<15>` through `state<0>` should be outputs (and the connections from `state<5>`, `state<3>` and `state<2>` to the XOR gate should be made internally). Although not shown, the `clk` input of all flip flops should be driven by a common `clk` input.

Note: you're free to add extra clock buffers (the clock is driving a large fan-out here) or to implement the four-input XOR gate by composing other, simpler gates, etc. Your schematic need only be logically equivalent to the one shown above.

Verify the correctness of your LFSR and take the following measurements at $V_{DD} = 1.2$ V using a clock period of 2ns. For verification, you may find it useful to use a HSPICE "vector file". An example vector file can be found under `/<476_FOLDER>/common/spice/examples/`.

Measurement	Description
`en_per_clock`	average energy per clock cycle during the first 15 clock periods following a reset

Table 2

Note: a reset operation consists of pulling `rst` high, sending a positive clock edge, and then pulling `rst` low again in a way that doesn't violate the flip flop's timing requirements. A correct reset operation will leave `state<15:0>` as `1000000000000000`.

5 LFSR Layout

a. Create the layout for the `LFSR` circuit in Part 4. Your layout should be arranged such that each `state<i>` pin is placed at a 5.0 um pitch. i.e. `state<15>` is 5.0 um away from `state<14>`, `state<14>` is 5.0 um away from `state<13>`, and so forth. This "bitslice" of 5.0 um will make connecting your LFSR to future portions of your processor much easier. The XOR gate, OR gate, and any clock buffers can be placed wherever they fit (e.g. to the left or right of the 16 flipflops) - this won't disrupt the bitslice so long as the above requirements are followed.

b. Collect the same measurements as in Part 4, but for the post-layout `LFSR`. For the parasitics extraction, use `C+CC`.

Delivery

a. The netlist for the post-layout `LFSR`, and DRC/LVS reports

b. Measurements from Table 2 for the post-layout `LFSR`

6 Additional Questions

a. Using the post-layout flip flop from Part 2, create a plot of clock-to-q delay as a function of data arrival before the clock edge; students are free to choose between a rising or falling transition, but the choice of transition should be stated. Assuming the clock edge occurs at $t = 0$, the plot should depict the clk-to-q delay for when the data arrives in the interval $t \in [-200ps, T_{FAIL}]$, where T_{FAIL} corresponds to the time where the flip flop is unable to capture the data correctly.

b. For the flip flop in Part 1, according to the measurements taken at 0.8V and 1.2V, which voltage would be better for energy performance? What impact does the voltage have on timing performance (for instance, T_{CQ}, T_{SU}, T_{HOLD})?

c. Regarding the `R+C+CC` extraction in Part 3, does the parasitic network include resistances connected to VDD or VSS? Why or why not?

d. Comment on some of the differences between `C+CC` and `R+C+CC`. When do you expect simulations with the two extraction types to differ significantly?

Delivery

a.-b. Nothing (plots and questions can be included in the report)

File Submission

Plots and answers to questions should be submitted in a report named `cad4_report.pdf`. Measurements should be submitted using the provided `specifications.json` file, and files should be in the specified locations (see the *CAD Submission* document for more information).