ECEN 454/714 – Lab4: Design & Simulation of 1-bit adder

1. Introduction

In this lab you will be designing the 1-bit adder circuit, which will be then used to form the 4-bit adder in the next lab. In the following sections, you will be given instructions on how to perform post-layout simulations which you will need to do on each of the standard cells that have been designed and then use these standard cells to build the layout of the 1-bit adder. After this, you will verify its overall performance by performing post-layout simulations on the entire circuit.

2. General idea of Post-Layout Simulations

The electrical performance of a full-custom design can be best analyzed by performing a post-layout simulation on the extracted circuit net-list. At this point, the designer should have a complete mask layout of the intended circuit/system, and should have passed DRC&LVS steps with no violations. The detailed simulation performed using the extracted net-list will provide a clear assessment of the circuit speed, the influence of circuit parasitic and any glitches that may occur due to signal delay mismatches. Note that a satisfactory result in post-layout simulation is still no guarantee for a complete successful product.

The layouts that have been designed by you so far have included power lines (VDD, GND), input and output signals. But you have not checked whether the circuit will perform as you would want it to. Hence you will apply appropriate voltages and loads at the corresponding pins of the layout and then measure the circuit's performance. This would be a more realistic measure of its performance since it would include the circuit parasitics as well. This process is something similar to what you did in Lab1 where you designed a schematic and then simulated its performance (NOTE: In lab1 you only verified the logic performance of the circuit in terms of 1's and 0's and you did not measure its timing performance either) by applying a verilog test bench, but in this case you will be applying Pulse, DC and other supply voltages and measuring timing performance of the circuit using spectre.

To perform such simulations, you will need to create a test-schematic which will contain the “ symbol” of the CUT (Circuit-Under-Test) and you will then need to apply the appropriate inputs to measure its performance.

3. Simulation

Before the simulation, modify the mos size in your schematic and layout. Use the width and length of the mos you got in lab 3. This step is for decreasing the error percentage of delays.

• Open a new schematic in the same library where your Inverter circuit is by choosing “File >New > Cellview” in the Library Manager and selecting “Schematic” from the Tool selection widget. Let the Cellview name be something like “inverter_simulate”.

• Next you select and place the components (You will wire them up in the next step). Choose your library and place the symbol of Inverter you created for the inverter design. (If you forget how to choose and place a component, refer back to Lab3).

Change the library in the Component Browser to “NCSU_Analog_Parts” and place “vdd” and “gnd” by choosing “Supply Nets” from the component browser. Also place a DC voltage source “vdc” and a pulse waveform generator “vpulse” from the “Voltage Sources”. Also place a capacitor in front of the output of Inverter by selecting “cap” from R_L_C section of the Component Browser. For now, please just place the components in your schematic, and let us set the parameters of those later in this section.

• Next you wire up these components. You use the dc voltage source to supply the VDD voltage to the circuit. For this you simply connect the “vdd” supply net with the positive terminal of the DC voltage source “vdc” and the “gnd” with the negative terminal of the voltage source. Also connect the VDD and GND of the Inverter to the positive and negative terminals of the DC supply voltage “vdc”, or appropriate supply nets mentioned here.

• Now connect the “+” terminal of the pulse generator “vpulse” to the input pin of the Inverter symbol. Also connect the “-“ terminal of “vpulse” to “gnd” net. The pulse generator generates a pulse of desired duration, pulse width, and voltage to provide the input stimuli to the CUT.

• Connect the capacitor to the output pin of the Inverter symbol. This completes the wiring of the circuit. Your schematic will be very similar to the design below:

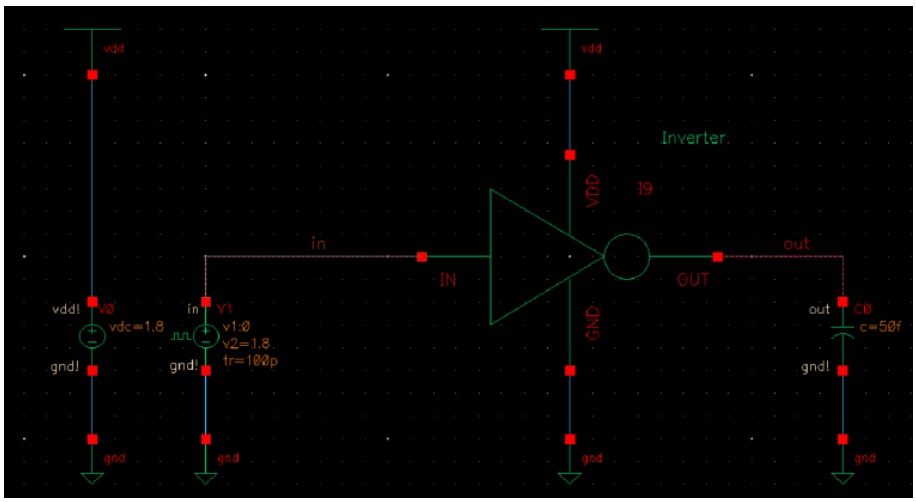

Figure 1. A sample schematic to simulate Inverter.

• You may wonder that you are making a symbol from the schematic and trying to simulate it, then how can it be called a Post-Layout Simulation. But remember you had performed a LVS check on this circuit and if the layout and the schematic netlists match then the circuit would be the same. All you will have to make sure is that the simulation is performed with the transistor and parasitic parameters extracted from the layout instead of the schematic. This can be easily taken care by inserting an option as you will see later.

• You now edit the values of the parameters of the voltage sources and the load. For the DC voltage, you edit its voltage by first clicking it and then hitting "q" on the keyboard (or choosing "Edit > Properties >Object"). The Edit properties window pops up. Edit the "DC Voltage field" and fill in "1.8" as shown in the figure 2.

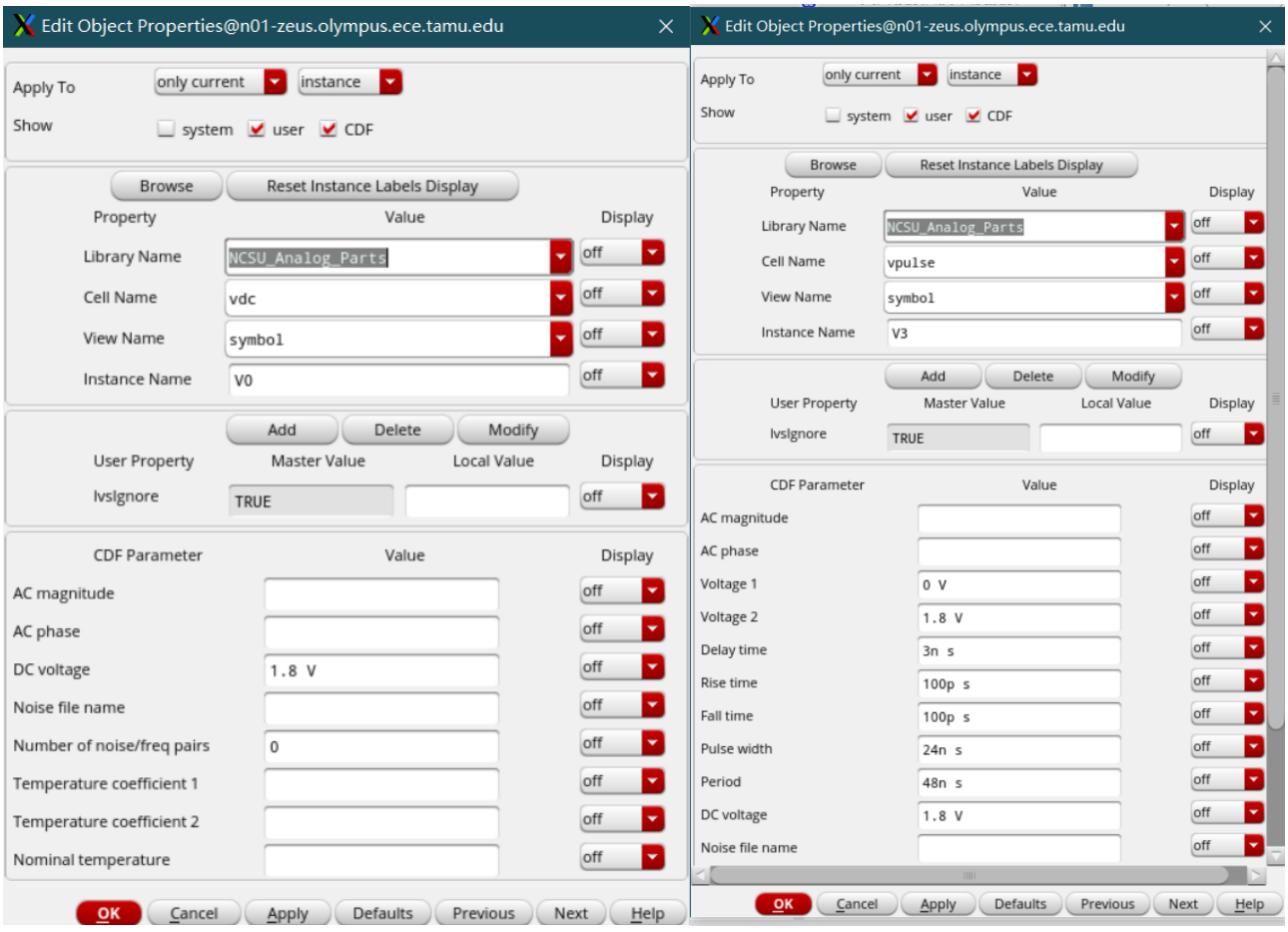

Figure 2. Setting the parameters of voltage sources (dc voltage and pulse voltage)

• Now you edit the properties of the pulse generator. You may try experimenting with the pulse width and the period of the clock as you like, but as an example you may fill up the properties as shown in

figure 2. The delay time indicates the time after 0 after which the waveform is generated. Voltage 1 and Voltage 2 may correspond to the respective voltages.

• Edit the properties of the output load (capacitor). A default value of 1pF will be filled in. But this can be lowered down to a lower value of around 50fF.

• Next you add label names to the nets you want to observe after simulation. Choose “create >Wire Name” and then type in a desired wire name in the pop-up window and then click on the corresponding net you want to name. For the case of the inverter, you will observe the input and the output signals and name them as “in” and “out” as shown in the schematic design above in Figure 1.

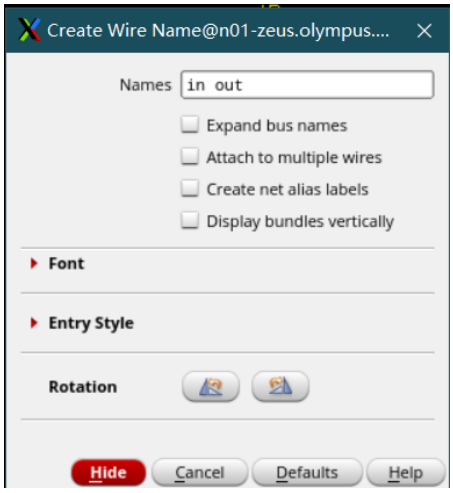

Figure 3. Naming a wire

• Now you will start the simulation. Cadence will internally generate a netlist of the circuit being simulated and use Spectre for all the timing simulations and then generate a waveform for the signals that need to be probed. Since Spectre will be looking for model parameters for nmos and pmos transistors you have to include the model cards in a certain location and ask Spectre to look for the model cards by indicating the path to the simulation tool.

• Create a directory named “models” under your “cadence” directory. And create a directory named “spectre” in the “models” directory.

cd ~/cadence

mkdir models

cd models

mkdir spectre

• Download the model cards (tsmc20N.m, tsmc20P.m) that have been put up on the website and place them in "~/cadence/models/spectre/" after creating the directory. Make sure these are the only 2 files in that particular directory so as to avoid confusion and also retain their names.

• To be able to run simulations, please source the following first (make sure cadence is closed):

source /opt/coe/cadence/SPECTRE211/setup.SPECTRE211.linux.bash

Now run Cadence as usual. Open the schematic you want to simulate, select launch> ADE L, (if there's license warning, click always), then go to:

Setup > Simulator/Directory/Host". Change the Simulator from Hspice to Spectre. Click OK.

Now you have to make sure that the simulator has the right path to pick up the model card definitions. Choose "Setup > Model Library" and the following window will pop up. Make sure you have to add tsmc20N.m and tsmc20P.m into model library and delete the old ones. Then click on "OK".

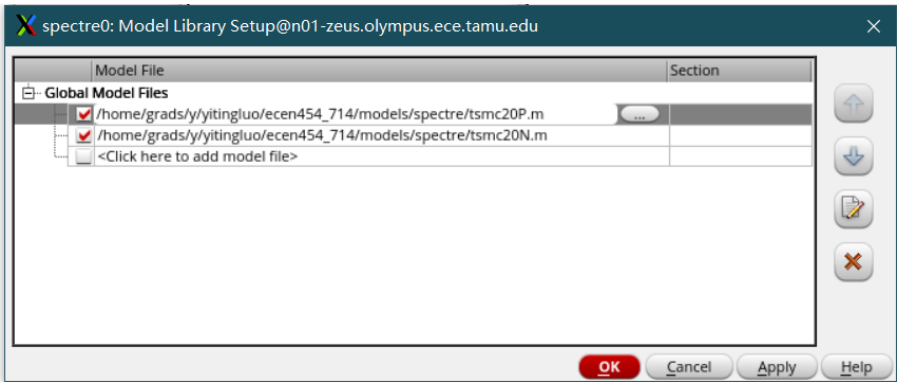

Figure 4. Setting model Path Window

• Now you will choose "Analyses > Choose". By default, the "tran" (tran stands for transient) option has been selected. Since you choose to do transient analysis on our circuit you will continue with the option. But you still have to fill in the Stop Time which indicates the time until which the simulation will be performed. Let this around 50ns. Click OK.

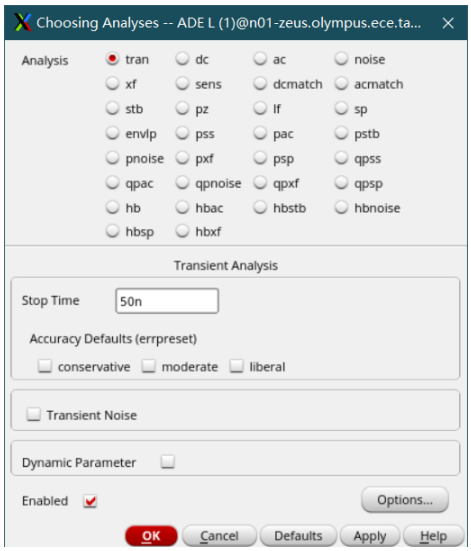

Figure 5. Choosing Analyses window

• Next you select the output signals that need to be plotted. Choose “Outputs > To be Plotted > Select on Schematic”. Then click on the nets in the schematic that you had named “in” and “out” to be plotted and then click on ESC. Verify if the “Outputs” section of the Analog environment main window is populated with the correct signal names. If the waveform didn't change at all, after changing the width of pmos or nmos, in the simulation window select simulation>recreate> netlist, then re-run the simulation. The main window will look similar to following at this stage.

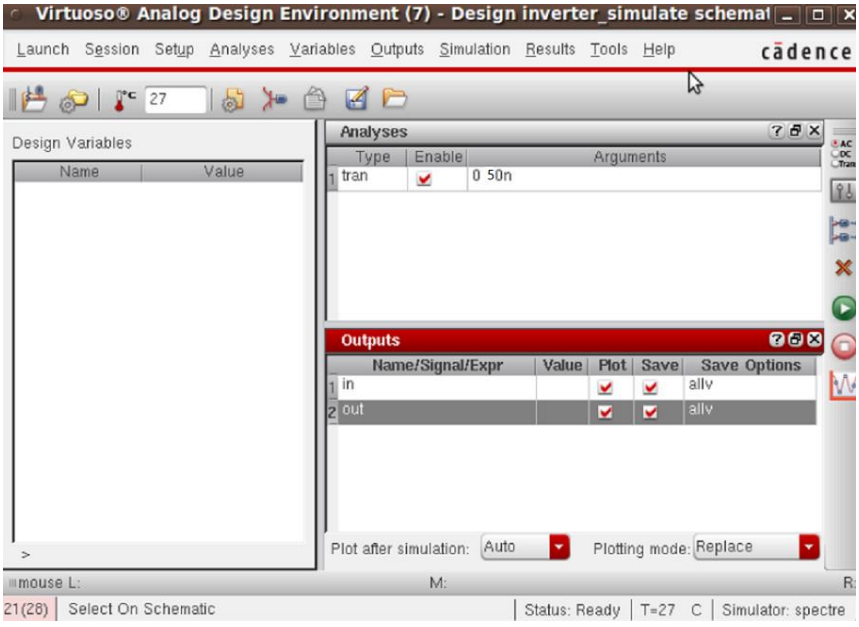

Figure 6. Analog Environment Window with proper simulation setup

• Now start the simulation by selecting “Simulation > Run”. The waveforms for the selected nodes should appear in a new window as follows. You may change the color, type, or style by right click related signal you are observing.

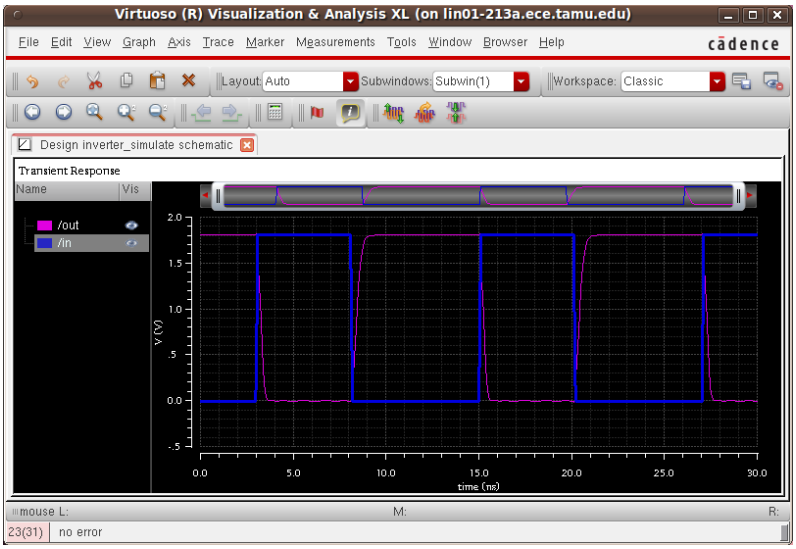

Figure 7. Waveform Window with results from transient simulation

• You will see the waveforms plotted together on the same axis. To separate them you can choose the option “Graph > Split Current strip”. This will display the signals on different axes.

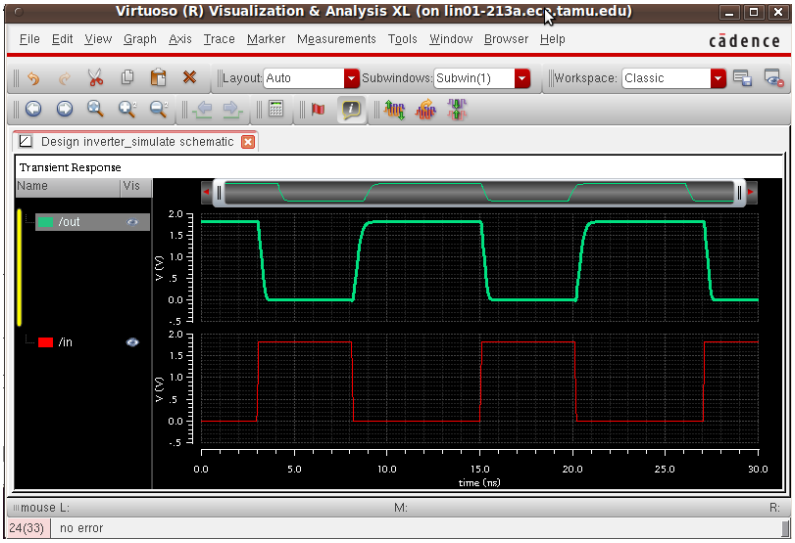

Figure 8. Waveform Window with results from transient simulation (split form)

• So far is the procedure to generate the waveforms on the schematic design that you created for the inverter, not the layout with extracted parasitic. Although the schematic and the layout designs are equivalent, the transistors in these designs have different sizes. The LVS check that was performed only checks if the devices, nets and terminals in both the schematic and the layout (By layout, you mean the "extracted" view of the layout) are equivalent. It however has not been programmed to check if the device sizes are equivalent. The device sizes will actually determine the timing of the circuit. Also the extracted view of the layout may include a lot of parasitics which can also affect the timing.

• Now, let us make the simulation to takes into account the extracted parameters of the layout. You need to make a small modification in the "Analog Environment" window. Choose "Setup > Environment". You will see window like Figure 9 (a).

• Now you will make a change in the line "Switch View List" by including "extracted" exactly in front of the word "schematic". This makes sure that the simulator picks up the extracted view parameters (if one exists for the same cell).

The Environment Options will look something like Figure 9 (b)after the change.

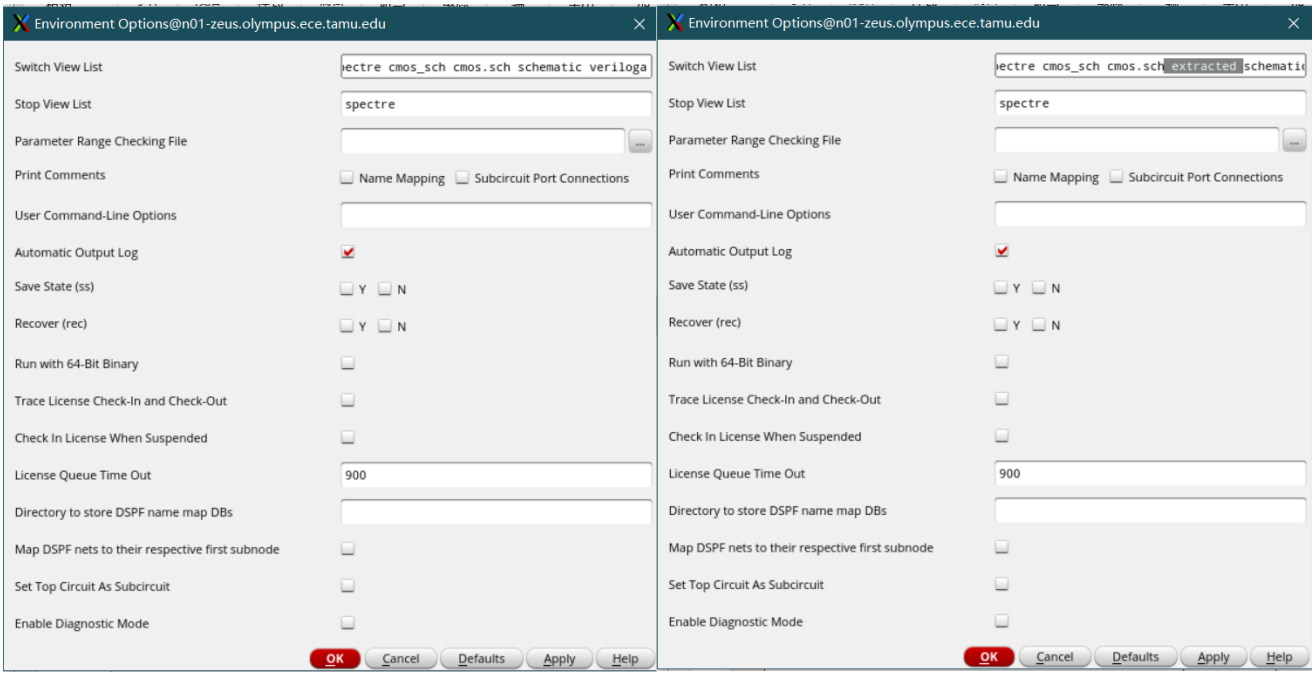

Figure 9 (a) & (b). Environment Option Window and Modification.

• Click "OK" and then rerun the simulations. The waveforms will now be according to the parameters that have been generated from the extracted layout.

• From the waveforms generated, you can also measure the rising and falling delays of the output signal. There are two ways to get delay (introduced in Lab 3), you could choose either way to implement it. Ideally, we would want to match the rising and falling delays of the standard cells to achieve the best trade-off between area and frequency and to make sure that the worst case and best case delays of the circuit do not differ a lot. But this might require modifying your earlier layouts

• You may save the simulation setups to reuse it later so as not to resetting the environment again, by choosing "Session > Save State" and putting a setup name on "Save As" field. Later you can use your saved session by choosing session > load state.

Figure 10. Save simulation setups

4. Power & Cell Height.

• Power Measurement: To measure the power delivered by the voltage source during the simulation (post-layout), you need to add the current which goes through the voltage node. You can edit the

outputs signal in ADL window. To calculate the power supply provided by vpluse and vdd in the inverter simulation, we need to add V0 and V1 node to the outputs. The schematic and ADL window would be like Figure 11 and Figure 12.

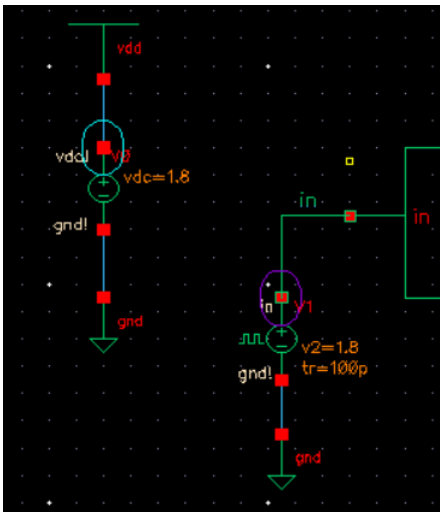

Figure 11. Choose the output current node on schematic

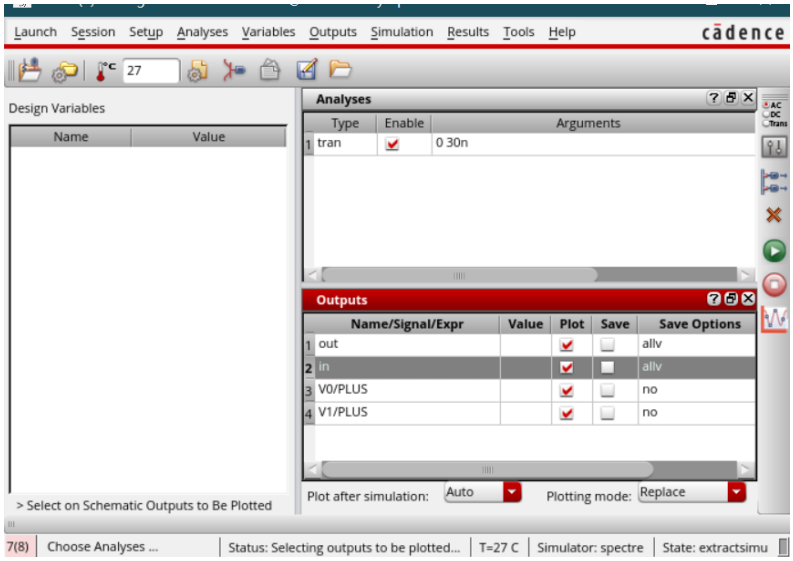

Figure 12. ADL window Outputs

Run simulation, at the waveform window, right click a wave and send it to calculator.

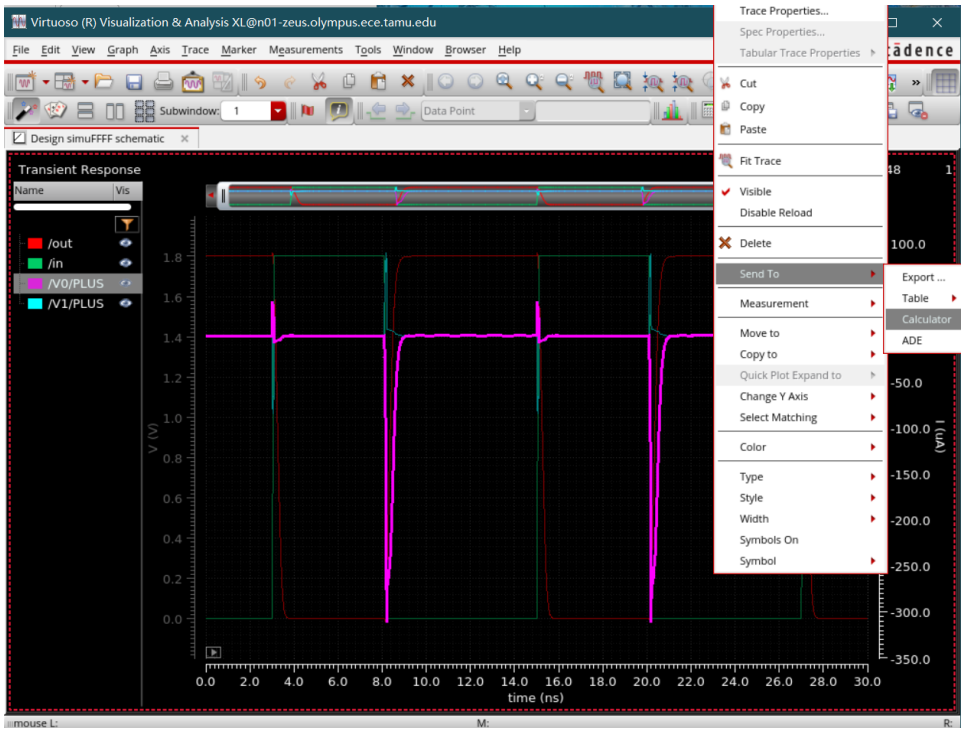

Figure 13. Simulation Result Browser.

The calculator window is shown below in Figure 14. Now choose average from the function panel in the calculator.

Next select Tools->Table in calculator window. Then the average current delivered from the power/voltage source is calculated. In the sample Figure 15, the average current delivered through vdd is calculated.

For vdd source, power supply is calculated by: 1.8 * average current

(vdd power supply = 1.8 * average(i("/V0/PLUS" ?result "tran")) in this case)

For vpulse source, power supply is calculated by: average voltage * average current

(vin power supply = average(v("/in" ?result "tran")) * average(i("/V1/PLUS" ?result "tran")) in this case)

The power would be around ~$(10)^{-7}\ watt$.

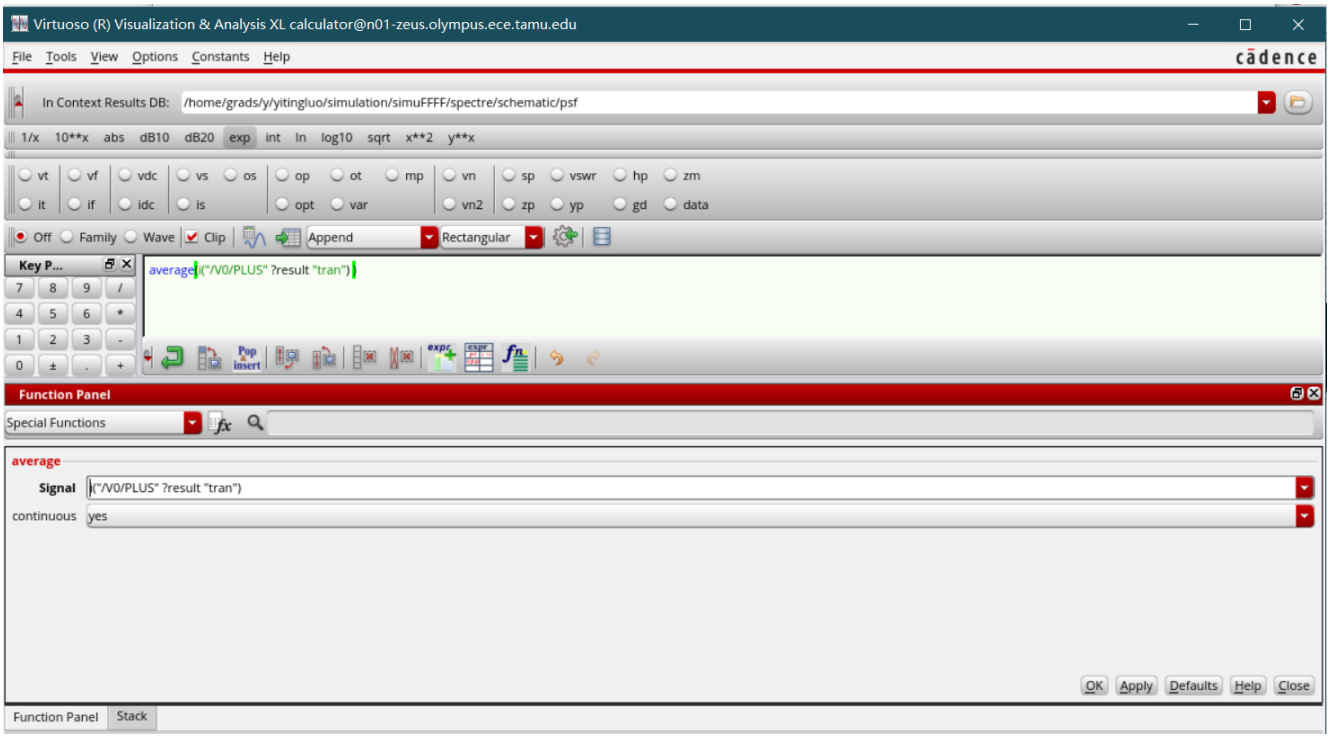

Figure 14. Calculator function panel

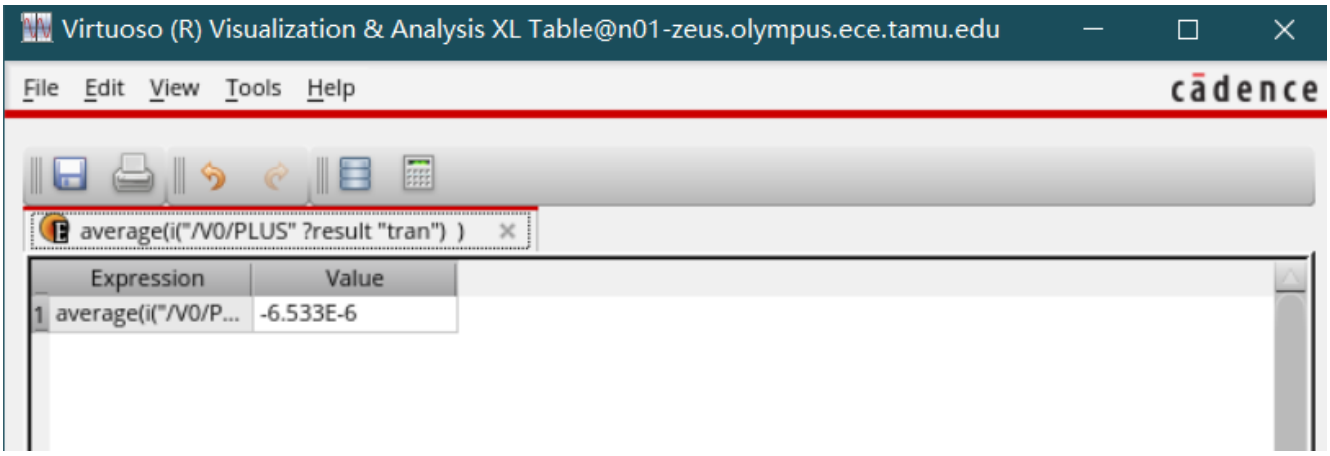

Figure 15. Calculator result display

Repeat the above steps for other standard cells you've designed (XOR2 and NAND2 gates).

• You need to make sure all the standard cell layouts have the same height. This is done because you will eventually be using all these gates to build the adder circuit and you would need to place these gates adjacently. When they are of the same height the power lines VDD and GND can be connected to each other and also the Nwell regions of all the cells can be merged.

• Hence it must be made sure that the Nwell regions of the cells have the same size and also the power lines have to be at the same height so that it is easy to connect to cells with common power lines. To make it easier to understand, a sample 1-bit adder layout design is attached below. It's a 2-row design.

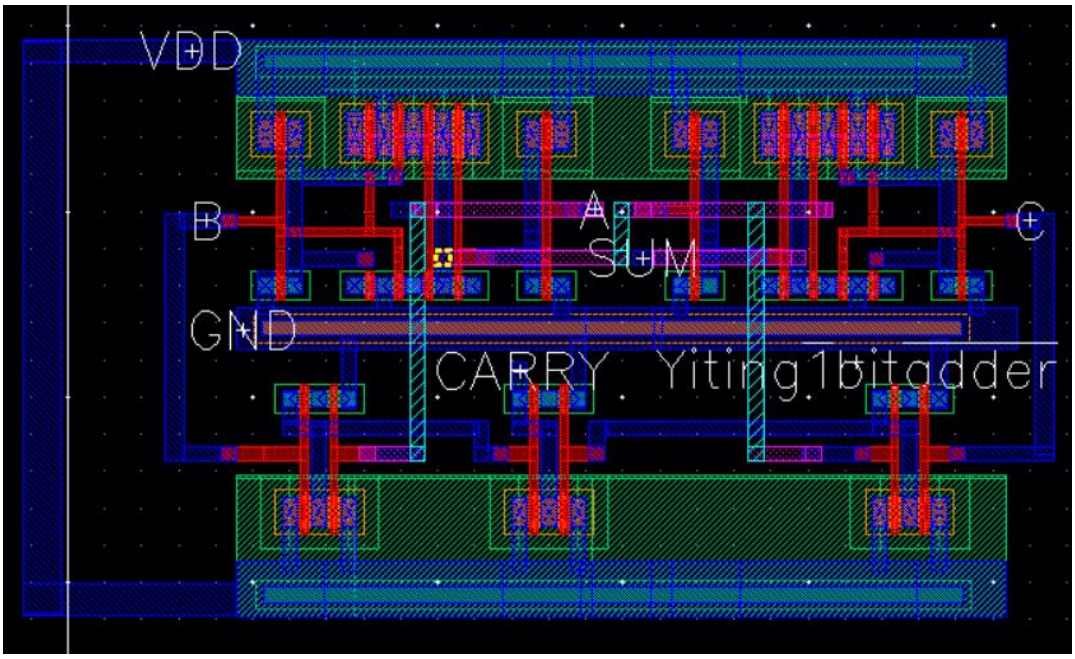

Figure 16. Sample 1-bit adder layout design.

5. Single bit Adder.

• Next you will use the standard cells (XOR, Nand and Inverter) to build the layout of a single bit full adder circuit.

• You have a free-hand to place the cells and route the signals accordingly. Only make sure that you use a row-based (use multiple cells in each row and the design may contain 2-3 rows) design to place the standard cells adjacently. You may connect the power lines of adjacent cells. (Since all are of the same height).

• The single bit adder should not take more than a couple of rows. You have to be careful during routing. Make sure you connect the appropriate pins without violating design rules or creating shorts between crisscrossing layers of the same type. You are free to use metal layers 1-4. (Do not use higher metal layers).

• Also you will need to be careful with pin names. No 2 nets can have the same pin name. DRC will flag an error. So you will have to give different names to pins that need to be created for the design.

• After you complete the routing, run a DRC check on the circuit. You will have to debug any errors in the layout. After the layout is DRC clean, perform extraction on the circuit with the "extract_parasitic_capacitances" switch selected.

• Next you will need to perform the LVS check, for which you will first need to create a schematic of the same design. This is similar to the schematic of the full-adder that you created in Lab1, the difference being that this time you will use gate symbols from your library instead of the inbuilt Cadence library parameters.

• Wire the schematic and make sure it has no warnings. Now open the extracted view of the layout that was generated and perform a LVS check on the circuit. Debug the errors if the LVS netlists don't match.

• Now perform post-layout simulations on this circuit (This will be helpful to you since this block will be repeated 4 times to create the complete 4-bit adder in the next lab). Refer to the previous sections to perform the post-layout simulations.

• How to get maxim frequency: There are three inputs for 1bit adder. Make sure that two of them have long enough pulse and cycle. We just focus on the other input. Then decrease the pulse as well as cycle for this input little by little. Then the output including carry and sum would become more and more abnormal little by little, until the rising can not completely arrive to VDD or the falling can not completely fall to zero, we measure this threshold as minimum period, you could get maximum frequency through $1/T_{min}$. For example, in the waveform below, we set A 40ns to be its period,

while we set B 20ns as its cycle. Then you set C 10ns first. We may find its output is normal, then decrease it step by step, maybe when T=3ns it is still could arrive VDD or Zero, but when you go on to decrease it to T=2.9ns it is abnormal, then your threshold is T=3ns, your maximum frequency is 1/3ns.

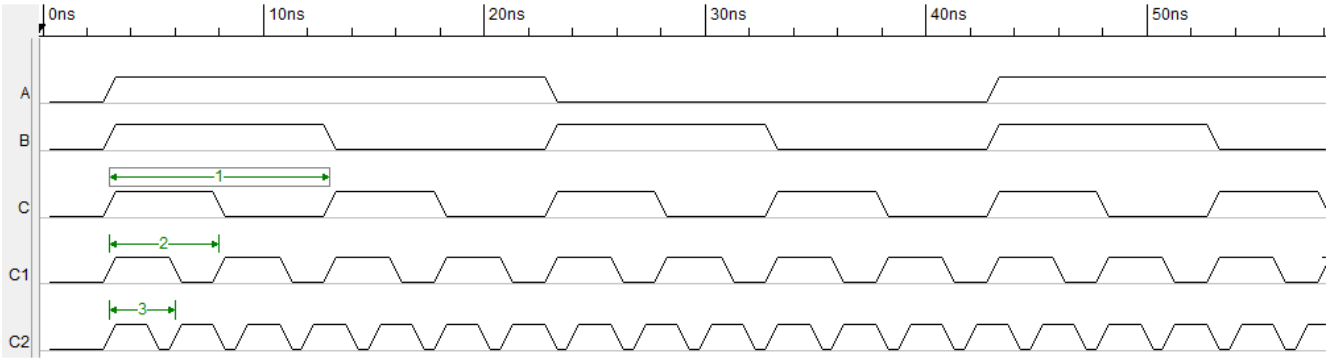

Figure 17. Maximum Frequency measure

Report Requirements:

1. The report should contain the schematic, layout, DRC and LVS reports of the 1-bit adder.

2. Report rising delay as well as falling delay for all the gates you designed including inverter, nand2, xor2 and 1bit adder. Make sure the error with respect to each other for standard cells (inverter, nand2, xor2 gates) should be less than 10%.

3. Please also report the schematic, layout, and LVS result of each gate only if you update anyone of those or you have failed LVS of any gates so far in previous labs (if you resize the nmos or pmos, you should also include new version).

4. Turn in the waveforms of the outputs of all gates and 1-bit adder with all possible combinations of all inputs. Please make all your input signals (the pulses) represent all the possible logic values. For example, pulses into two inputs to NAND2 will show 4 logical values: 00, 01, 10, and 11. This can be done easily with putting multiples of pulse voltage sources into your simulation schematic.

5. Report power dissipations from all voltage sources of all the gates and adder. When acquiring power dissipations from pulses, you need to multiply the average current with the actual average value of the pulses, not just 0.9V.

6. Report the maximum frequency which your 1-bit adder can achieve. Provide waveforms to support. (note that your outputs should completely rise to VDD and fall to GND)

For this, you may want to resize your transistors to reduce delay. This means you also need to modify the layout of the standard cells. In case you choose to modify your layout, make sure you perform i)DRC check ii)LVS check before going ahead with the post-layout simulations to measure the new delays. This is important since while changing the layout you may have accidentally changed some wiring which you did not intend to do.

7. Please try to reduce delay. Additional scores will be given according to your effort.

Lab 4 Rubric:

1. Schematic (0.5), layout (1), DRC (0.5), LVS (1) reports of adder

2. Rising and falling delay for inverter, nand2, xor2 and 1 bit adder (0.5 each, 0.5*4=2)

3. Waveform of outputs - 0.5 each (inverter, nand2, xor2 and 1 bit adder)

4. Power dissipations - 0.5 each (inverter, nand2, xor2 and 1 bit adder)

5. Max frequency (1)

6. If the report shows the student is trying to reduce delay, add 1 point as bonus. (Max is still 10)