ECEN 454/714 – Lab3: Cell Characterization using Spectre

1. Introduction

In this lab session you will be required to characterize the standard cells that you have used in the design of the pipelined adder so far. Although cell characterization includes a wide number of parameters such as propagation delay, power, area, timing constraints in the case of sequential elements, input capacitance and global parameters like PVT(Process, Voltage, Temperature), corner selection, etc. In this lab you will be dealing only with the **propagation delay** and **input capacitance of a standard cell** by performing **circuit simulations at the transistor level.**

The simulation tool is Cadence Spectre. You need to write simulation files according to Spectre grammars, run simulation, and then observe results using another tool "WV". You might need to download files of CMOS models, standard cell description, and one simple example through the lab website. **Please create a new directory called "cellcharacs" under your "cadence" directory, and then copy all necessary files into your "cadence/cellcharacs" directory.**

2. Circuit Simulation

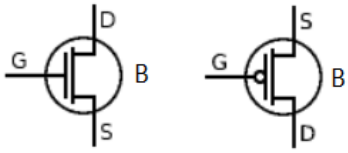
● In the first lab, you did logic simulation using NC-Verilog simulator of Virtuoso. However, as mentioned before, that logic simulation does not cover any timing information extracted from the circuit layout. In real circuits, signals inside the circuit are not ideal logic high or logic low. You can capture analog information like voltages, currents, timing and frequency in circuit simulation. SPICE simulation is one well-known simulation approach. There are multiple simulation models and levels in SPICE simulation. The basic idea of SPICE simulation is to construct a time-domain system based on a set of state variables, and then to solve it by numerical computation. SPICE simulation has the highest accuracy among all known circuit simulation approaches, while it also has the slowest speed.

● In this section you will use Cadence Spectre, one SPICE simulation tool, to do circuit simulation. The first step is to write a simulation file using Spectre simulation language. In this file, you need to define the netlist of your circuit, including necessary libraries, and write simulation controls. For a better understanding of Cadence Spectre, please read Spectre Reference and Spectre User Guide posted on eCampus.

● Download the CMOS transistor model file "model18.spi" from google drive. The CMOS model is using BSIM level 3.1. You can find some basic transistor parameters inside the file. For example, oxide thickness "TOX"; threshold voltage "VTH0"; etc.

● Download the standard cell description file "cell18.spi". Open the file; you will see there is a definition for one of the cells, inverter "IV". The file is written in Spectre format. The standard cell is defined as a sub-circuit with inputs and outputs. Note VDD and VSS need to be defined too. Inside the sub-circuit, you use several NMOS and PMOS transistors to construct the gate structure. The names of transistors are defined as "M1, M2 …" For each transistor, its terminals are defined in the order of "drain **gate source body(substrate)** ". "tsmc18P" and "tsmc18N" are PMOS and NMOS models defined in "model18.spi". Note at the end of definition of a transistor, there are some parameters of gate width "w" and gate length "l". You can initialize these parameters inside the sub-circuit, but they

also can be changed when you refer the standard cell in your simulation file. Please understand this, later you will be required to write the sub circuits for the other standard cells that you have used in the design of the 16-bit pipelined adder. (NAND and XOR)



● Download the sample simulation file "sample.spi", make a copy, rename it as "inverter.spi". You will find the following statements:

; Spice netlist for an inverter and a capacitor
simulator lang=spectre
include "~/cadence/cellcharacs/model18.spi"
include "~/cadence/cellcharacs/cell18.spi"
vgnd (gnd 0) vsource dc=0
vVDD (VDD 0) vsource dc=1.8
vpwl (IV_in 0) vsource type=pwl wave=[0n 0 1n 0 1.2n 1.8 4n 1.8 4.2n 0]
X1 (IV_in IV_out VDD gnd) IV wp=0.9u lp=0.2u wn=0.4u ln=0.2u
R1 (IV_out 1) resistor r=1
C1 (1 0) capacitor c=100f
Transient Analysis tran start=0 stop=10ns step=1ps
save IV_in IV_out

● The first line is a line of comment, you can also use "*" or "//" in the beginning of a line to indicate comments. The second line indicates that the simulation language is using Spectre format. The third and fourth lines include necessary libraries to run simulation, the model card and the cell sub circuit definition. Also, always make sure the path is correct. Following two lines define voltage sources of power and ground. Note the general format for a component defined in Spectre is:

Name (node1 … nodeN) master [[param1 = value1] … [paramN = valueN]]

where *Name* is the name assigned to this instance of the component, *node1 … nodeN* are terminals that the component connects, *master* indicates what kind of component it is, and *params* are component parameters and defined by users. For example, in the definition of VDD, the component VDD is a kind of voltage source (vsource) and it provides a dc voltage with 1.8 volts.

For the next component vpwl, it is a voltage source generating a piecewise linear voltage waveform. There are several parameters for this voltage source. Type pwl indicates it is a piecewise linear voltage signal. Parameter wave describes the shape of the waveform. In this case, the voltage increases from 0 to 1.8 during time period 1n - 1.2n and decreases from 1.8 to 0 during time period 4n - 4.2n. (Note "n" stands for nanosecond.

Component X1 is an inverter. In library cell18.spi you can find its definition. Note the order of its four terminals. You might want to make sure that **the order is corresponding to the order defined in the library** . Note the parameters of the inverter. Simulation will always use the values you write in the simulation file.

Component R1 is a resistor with 1Ω. It connects the output of gate X1(IV_out) and a node 1. Note you can name a node with a number or a word, and you do not need to claim the node before you use it. Note 0 is default for the ground.

Component C1 is a capacitor with 100fF. Note 'f', femto, indicate 1.0e-15.

The last two sentences claim the running time and the step accuracy of the simulation. You can use a smaller step value to increase the accuracy of your simulation results, but you will run a longer time. The save sentence indicates which node you want to observe. The default value is voltage. You will see it in the waveform display tool.

● After your simulation file (inverter.spi) is ready, type the following command to source Cadence Spectre:

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

Remember every time you want to launch Spectre, you need to use this command to source it first. Then type " **spectre inverter.spi** " and simulation begins. The working directory must be your "cadence/cellcharacs" directory. You will see simulation information displayed and the final lines like "spectre completes with 0 errors, xx warnings …" if the simulation is done.

```
Simulation started at: 7:56:20 PM, Thur Sep 28, 2023, ended at: 7:56:20 PM,
        Thur Sep 28, 2023, with elapsed time (wall clock): 612 ms.
spectre completes with 0 errors, 2 warnings, and 8 notices.
```

● The waveform display tool we are going to use is Cadence Viva. Launch Cadence, at the CIW window, find Tools > VIVA XL > Results Browser. A waveform viewer window would pop up as shown below.

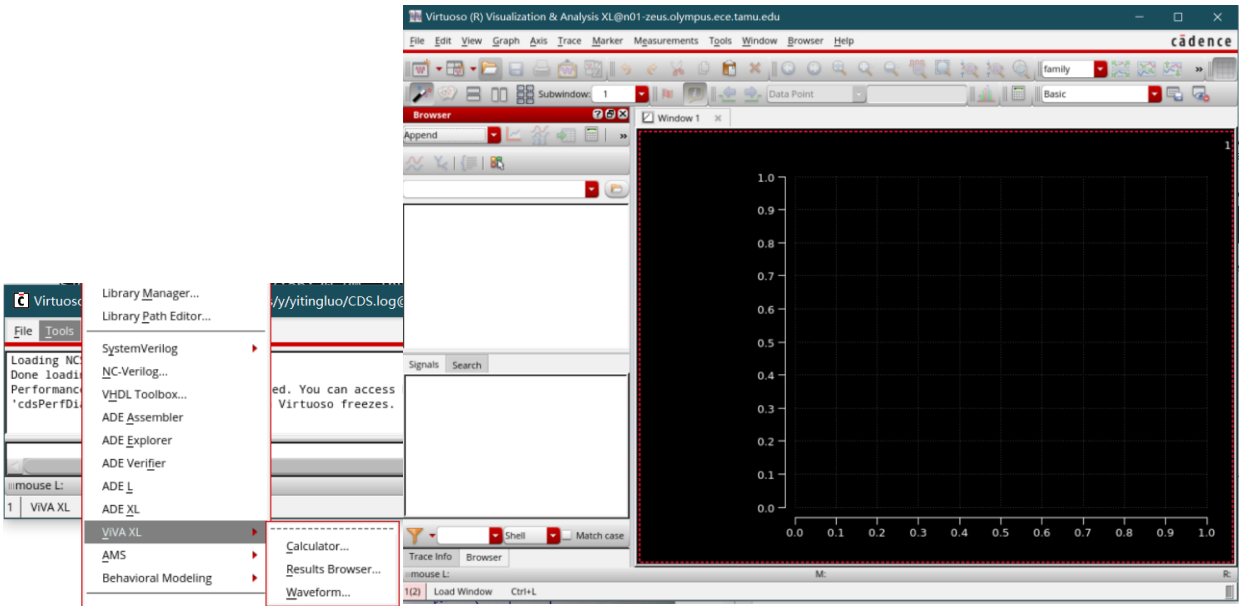


Figure 1. CIW window and waveform viewer window

In the waveform viewer window, choose File > Open Results, then find the inverter.raw. Click Open.

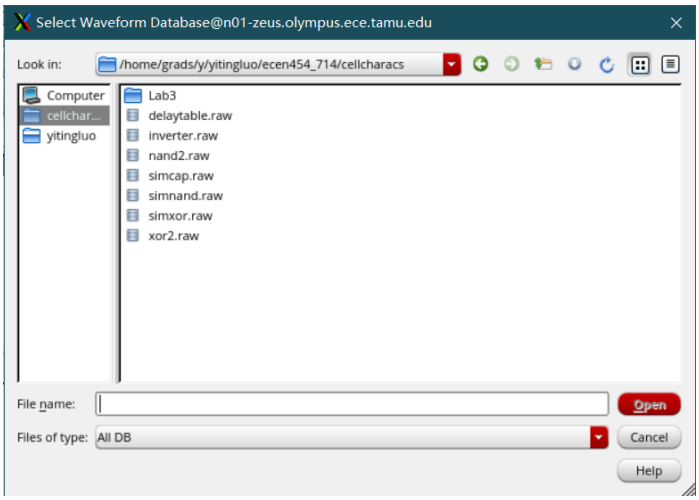


Figure 2. Load inverter.raw.

Open TransientAnalysis.tran, double click on IV_in and IV_out, the waveforms would show up.

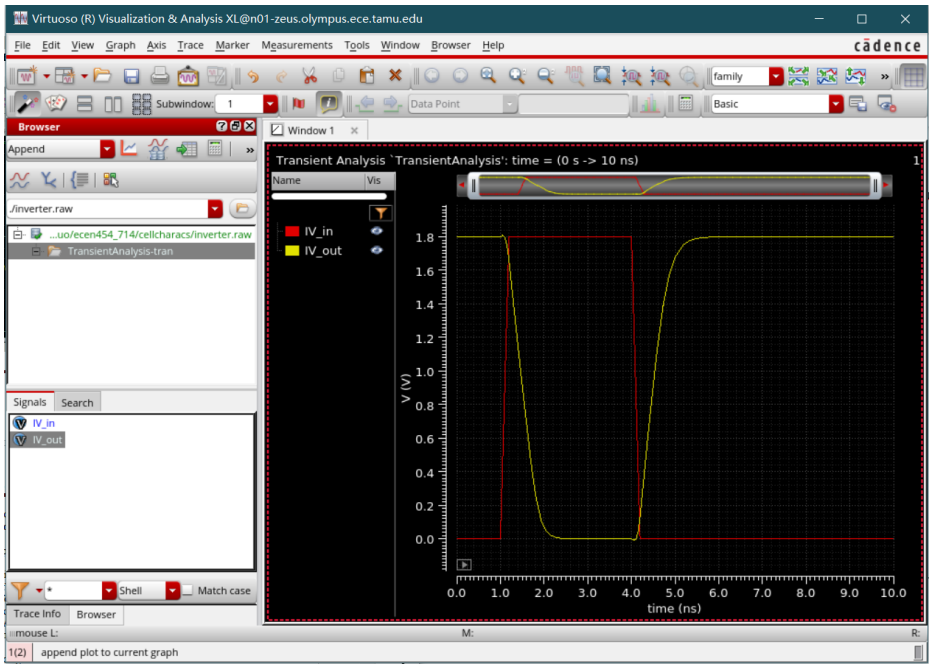
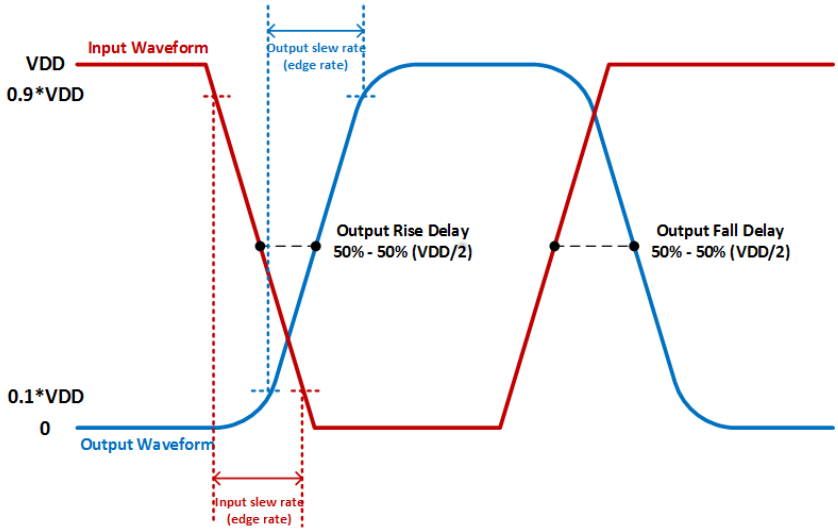


Figure 3. Inverter's input and output waveforms

● Next, we use the waveform to compute delay. In general, **delay is defined as the time from 50% point of the input waveform to 50% point of the output waveform**. In this lab, since the VDD is set to 1.8v, you can use 0.9v as 50% point for both input and output waveforms. Note the input node is IV_in and the output node is IV_out.



●Next, we measure the rising and falling delay. You can choose using marker or calculator to measure the delay.

●Method 1: Create marker

Select Marker > Create Marker. Set Y value of the marker to 0.9 and choose "on" to visualize it.

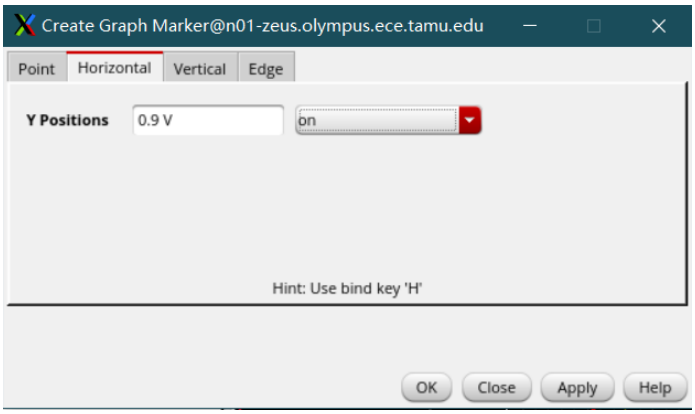


Figure 4. Create Marker

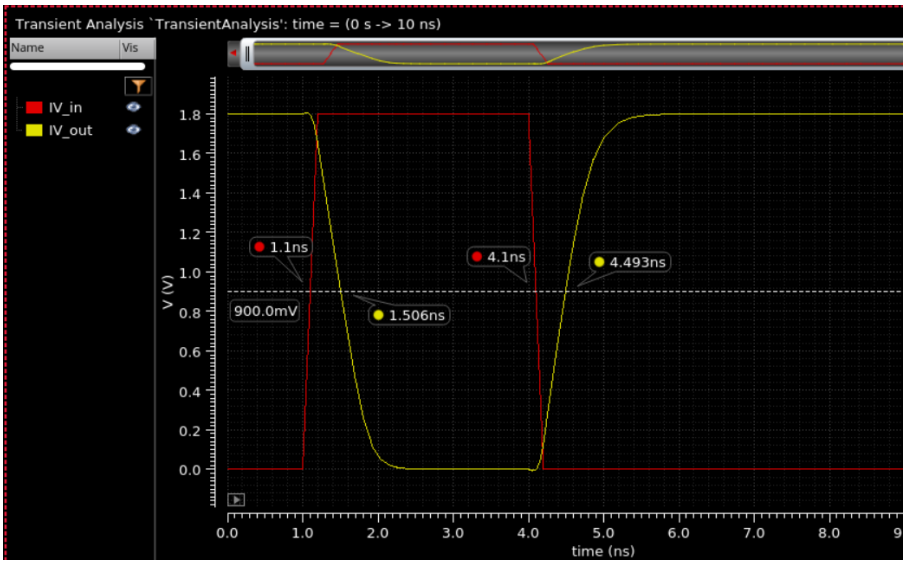


Figure 5. Created Marker

In such a case, the falling delay would be (1.506 – 1.1 = 0.406) ns, and the rising delay would be (4.493 – 4.1 = 0.393) ns.

● **Method2:** Use delay function of calculator to measure the delay
Choose either “in” or “out” signal, right click on it and send it to calculator. Then choose “delay” on “function panel” on it, then make changes on “delay” option table as below. Make sure signal1 and signal2 refers to “in” and “out”, their threshold values should be 0.9.
In this case we test falling delay, output signal should be “falling” edge type while input signal should be “rising” edge type. Number of occurrences should be “multiple”.
Then the results display window would show falling delay 406.2 ps as well as the occurrence time 1.1ns.

Repeat this for the rising delay. Output signal should be “rising” edge type while input signal should be “falling” edge type.

● Record the rising and falling delay, calculate the error percentage between the two delays.
Error percentage= (Abs(Rising delay – Falling delay)/ Min(Rising delay, falling delay)) x 100

● See if the error between the two delays falls within 10% with respect to each other. If not, modify the gate length and width in the simulation file (inverter.spi for the inverter simulation). The parameters defined in “cell18.spi” will be overridden by the values in the “inverter.spi”. After changing the size of the mos, rerun the simulation using “spectre inverter.spi”. Note that after each simulation, you need to “fresh” your data through double click on IV_in and IV_out.

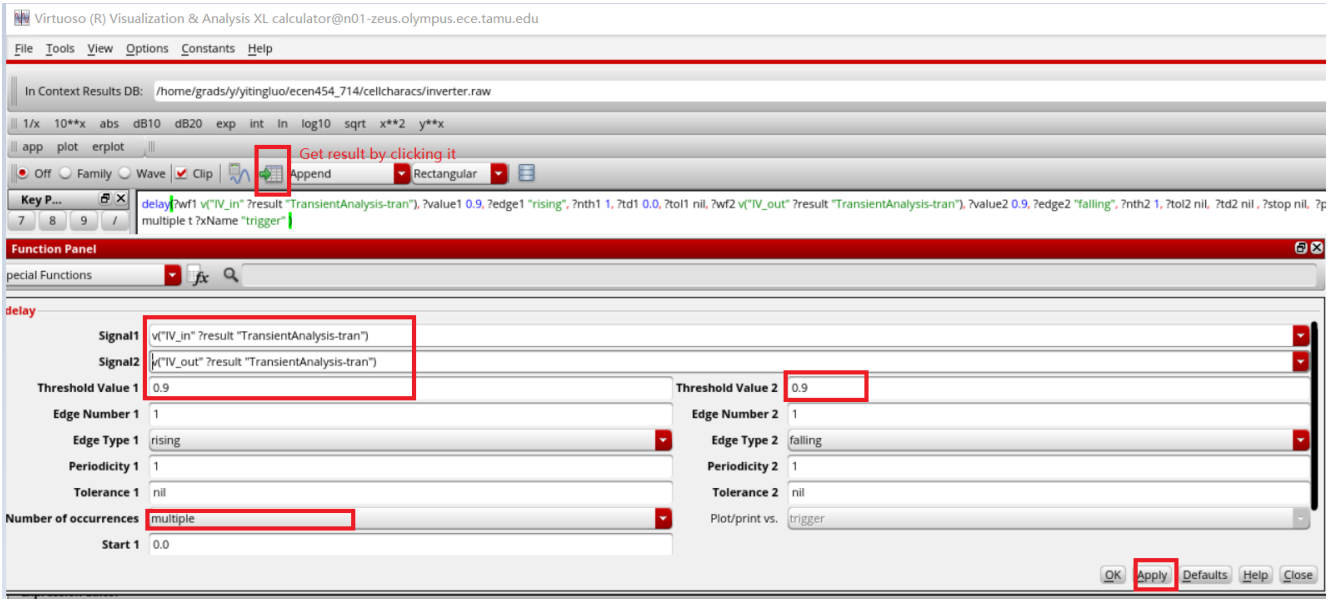


Figure 6. Calculator delay function panel settings.

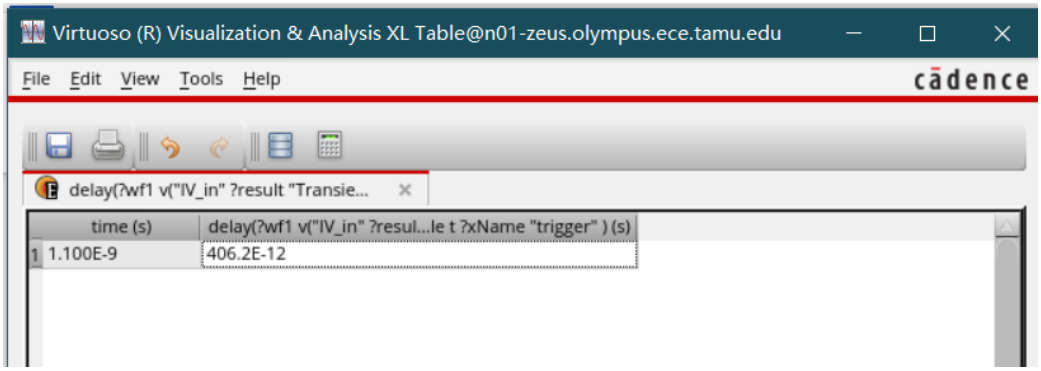


Figure 7. Calculated falling delay result.

3. Cell Delay Table

● If you use the resistance formula in the book and simply multiply it by the load capacitance and 0.7 (which is the timing constant for 50% delay), you will find the value does not match with the accurate delay that was obtained in the above section. One reason is because the resistance formula is too simple. Another reason is because you have a ramp input. Since the gate delay is dependent on many parameters, such as input signal transition time, load capacitance and working region, in general there is no simple closed form equation can give very accurate estimation in large circuits within 1% error. Using a delay table for each standard cell is a method used widely in industry. Two-dimensional table is very popular, where the input transition time and the load capacitance are two index variables. In this lab, you only build a one-dimension table, where the load capacitance is the variable.

● Create a new simulation file from your inverter simulation file, i.e., copy "inverter.spi" to a file with any name you would like to give, for example "inverter_delaytable.spi". Linux command is: cp sample.spi inverter_delaytable.spi . Remove the resistor connected to the output of the inverter by deleting or commenting out the line with R1 instance. Then, connect C1 directly to IV_out by changing net connections. Note your inverter should have similar rising and falling delays when C=100fF (within 10% error w.r.t. each other). The lines with R1 and C1 in your simulation file will look as below.

;R1 (IV_out 1) resistor r=1 **<- R1 is removed from circuit**
C1 (IV_out 0) capacitor c=100f **<- C1 is connecting IV_out and 0**

● Change your load capacitance value (C1) from 1fF to 100fF. Choose 15 capacitance values by changing the line with C1 and compute the rising delay and falling delay for each capacitance value. Keep in mind that every time you change the value of capacitance, you need to rerun Spectre again. Record every delay value and corresponding load capacitance, and you will have a one-dimension delay table as below. Now suppose there is a load capacitance whose value is in the range of (1fF, 100fF), you can use linear interpolation to compute the corresponding rising cell delay under a ramp input with the fixed slope value you set for your table (200ps in this case).

Delay table format

Capacitance/fF	Falling Delay/ns	Rising Delay/ns	Error Percentage
1			
7, 13, 19, 25…			
100			(Make sure it's < 10%)

4. Gate Input Capacitance

● Next you will compute the sink capacitance of your inverter using Spectre simulation. Normally the capacitance considered in delay evaluation in the circuit consists of two parts. One is from the

interconnect parasitic capacitance, for example, the capacitance between one metal layer and the substrate. The other part is from input pins of each cell in the circuit. The capacitance on the input pin is located between the gate and the substrate of the cell. It is one major effect for the delay computation of the upstream cell.

- To compute a capacitance by simulation, you first consider voltage-current relation. For a capacitor, you have $C\frac{dv(t)}{dt} = i(t)$, where C is the capacitance, $v(t)$ is the voltage on the capacitor and $i(t)$ is the current flowing through this capacitor. In frequency domain, considering amplitude only, you have $2\pi fC = \frac{I_f}{V_f}$, where f is the frequency, I_f and V_f are the amplitude of the current and voltage signal in frequency domain. If you feed an AC voltage signal to the capacitor with unit amplitude, then you have $= \frac{I_f}{2\pi f}$. However, the gate is not an ideal capacitor and the value varies at different working frequencies. A simple way to approximate the accurate value is to sample some points over a wide range of frequencies and then choose the mean value.

● Save a file called “simcap.spi” from lab webpage to “~/cadence/cellcharacs” directory.

● Please see the line claiming AC voltage signal with amplitude 1 as below. The grammar is similar to the case in which you create a pulse signal. The only difference is that it is an AC signal. Here “mag” defines an AC signal with amplitude 1. Now you have a voltage signal at IV_in. Note that “acinput” is the name of the voltage source. It connects IV_in to 0(ground).

acinput (IV_in 0) vsource dc=0 mag=1

● Since you want to monitor the current flowing to the inverter, you need to find this value first. Previously you use “save IV_out” to see the voltage at IV_out node. To save a current, you cannot save a node current. Instead, you can save a component current, which is the current flowing through that component. You will serially connect a 0Ω resistance from input to the inverter. Then the current flowing through this resistance is the current flowing through the inverter. The resistance claim is as follows:

R1 (IV_in IV_in1) resistor r=0

● Now you need to modify the inverter instance X1 by yourself to convert it to your own inverter circuit. Note that the input pin is IV_in1 in this case.

● AC analysis: Until now, you use transition analysis to analyze the transient behavior in timing domain. This time you would use AC analysis to see the circuit behavior in frequency domain. The analysis keyword is “ac”. Please see the following line in the file. It means you will do AC analysis from frequency 10 Hz to 1 GHz. The step is not specified here since Spectre will automatically choose logarithmic step when the ratio of stop start value is bigger than 10.

Freq ac start=1e+1 stop=1e+9

● Monitor the output. Following line is to save the current.

save R1: currents

●Now save your file. Type “spectre simcap.spi” to simulate your file. Remember to source Spectre and Cdence again if you are opening a new terminal.

Launch Cadence, at the CIW window, find Tools > VIVA XL > Results Browser. In the waveform viewer window, choose File > Open Results, then find the simcap.raw. Click Open. Click Freq.ac, double click R1:1, you will see the wave form. Make sure both X and Y axis in not in the “Log Scale”. You can find the current is almost linear to the frequency as shown below.

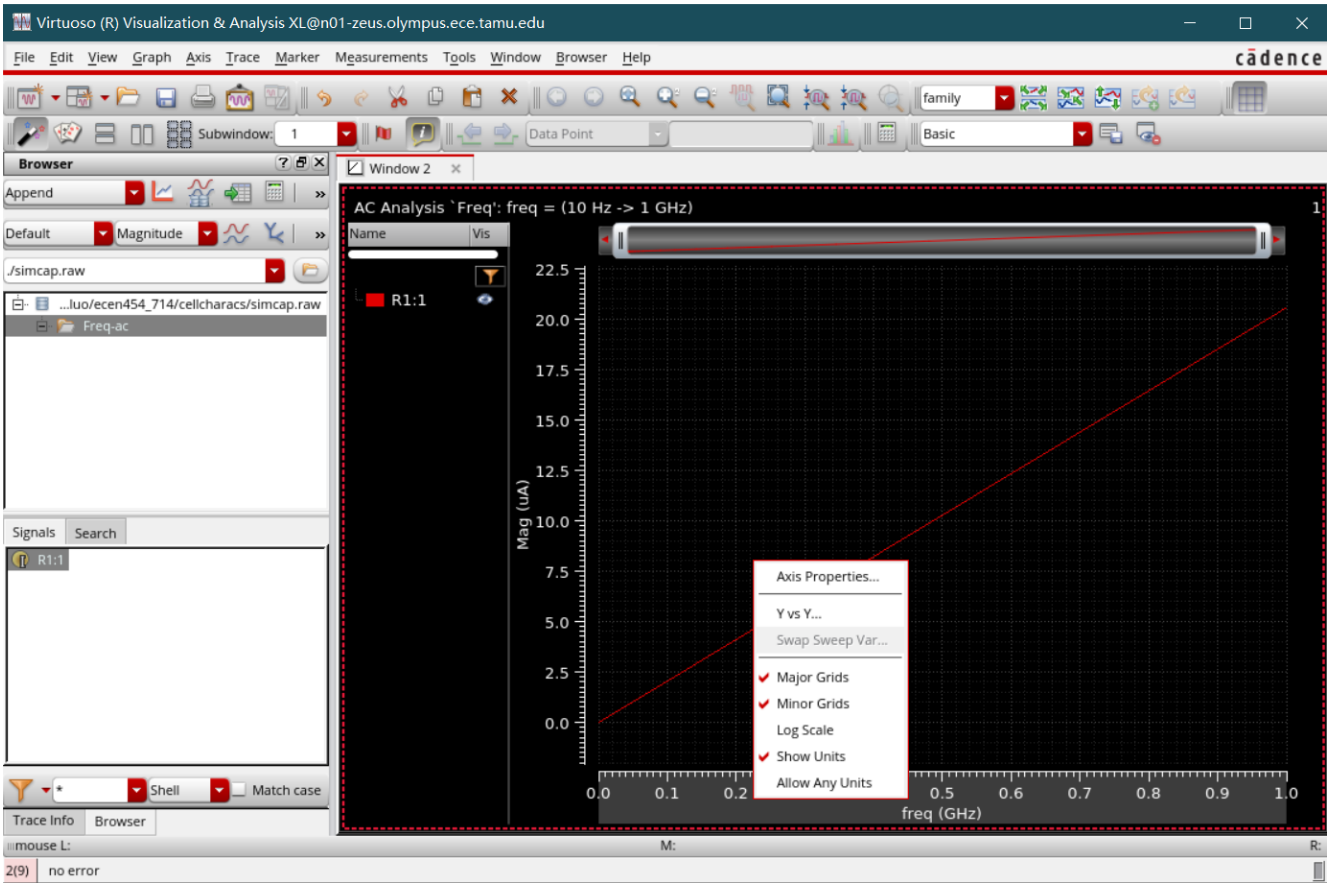


Figure 8. Frequency and current waveform

● Take 10 sample points on the wave form and compute the corresponding 10 sink capacitance. The result of sink capacitance would be the mean value of the calculated 10 sink capacitances.

Sink capacitance calculation table.

Frequency (f)	Current (I)	Sink Capacitance = I / ($2\pi f$)
(Choose 10 data points)		Calculate and record corresponding capacitance C1
		Calculate and record corresponding capacitance C2
		…
		…
		…
		Calculate and record corresponding capacitance C10
Final sink capacitance		Calculate mean value of C1, C2, …, C10

Report Requirement:

1. Plot waveforms of IV_in and IV_out clearly indicating rising delay and falling delay of inverter X1.

2. Include your simulation file (inverter.spi, nand2.spi, xor2.spi) with the assigned parameters.

3. Report the rising and falling delay as well as error with respect to each other of your inverter in table format. Please follow step 3 in this manual to create Delay Table correctly. Make sure both rising and falling delays are similar to each other (within 10% error w.r.t. each other) when the load capacitance is 100fF.

4. Plot the waveform from AC simulation. Report the sink capacitance value of the inverter by an average of 10 sample points taken along with the plot of the current vs. frequency obtained. Refer to step 4 in this manual to acquire sink capacitance.

5. Add to the existing library in **cell18.spi** a subckt for the NAND2 and XOR2 gates with equal topologies in the schematics of gates in lab 2 report. For two-input case, give one input **VDD**, change other input signal. Complete the above report requirements from 1 to 5 for both these gates.

6. Include your cell18.spi file.

7. Include the waveforms when the load capacitance is 100fF as screenshots (make sure the error percentage of rising and falling delays is with 10% range).

Rubric for Lab 3:

1. Waveforms /calculator result which clearly indicating rising delay and falling delay – 0.5 each (inverter, nand, and xor)

2. Delay table – 1 each (inverter, nand, and xor)

3. Waveform from AC simulation and sink capacitance – 1 each (inverter, nand, and xor)

4. Simulation spi file – 0.5 each (inverter, nand, and xor)

5. cell18.spi file capacitance – 1