University of Pennsylvania
Department of Electrical and System Engineering
Circuit-Level Modeling, Design, and Optimization for Digital Systems

ESE3700, Spring 2023

HW2: SPICE setup and Restoration

Friday, January 27

Due: Friday, February 3, 11:59PM

Read before starting: In this homework you will start to familiarize yourself with SPICE, which is what we will use to simulate all our circuits. It is recommended that you read through the entire handout and note the different sections before starting. The goals of this homework are to 1) use electric to build test setup for DC transfer function analysis, and 2) use electric to build test setup for transient analysis; load gate with FO4 load (fanout to 4 equivalent gate inputs). The big priority and outcome of this homework is to make sure that you have a work flow in place to facilitate all future work.

This handout is written assuming you are working on your Eniac account on one of the Linux computers in Detkin/Ketterer lab. Ngspice requires X11 tunneling for graphics processing (i.e plotting). To set up the tools for use from your personal machine (recommended) see the Ngspice setup guides linked from our course webpage
`https://www.seas.upenn.edu/~ese3700/#tools`.

Additionally you can install Ngspice on your personal computer, but note that it might require some trouble shooting and help by your peers, TA or instructor in office hours. **Start this homework early!**

- **SPICE Setup and Instructions:**
  1. Must do first: Setup Electric for Spice. (Page 3)
  2. Write DC analysis SPICE deck. (Page 5)
  3. Review SPICE deck. Convince yourself it represents the intended circuit. Identify key nodes in circuit.
     Hint: a common problem is wires aren't really connected in Electric. You'll eventually know what to watch for, but looking at the SPICE deck is a good sanity check.
  4. Generate DC transfer function $V_{out} = f(V_{in})$ for a controlling input of the nand2 gate.
  5. Using the DC transfer function $V_{out} = f(V_{in})$, identify the workable noise margins for your SPICE simulated nand2 gate.
  6. Run transient simulation in SPICE. (Page 8)
  7. View waveforms from SPICE run in ngspice.
  8. Measure propagation delays (both for 0→1 and 1→0 transitions) and output rise and fall times. Use definitions from textbook for propagation delays and rise/fall times.

9. Assuming the gate capacitance of the $W = L = 2$ PMOS or NMOS FET is 0.05fF, estimate the transistor's R_{on} based on observed signal rise time.
10. Work Problem R (Page 10).

- **Turn in:**

  1. nand2 test SPICE Decks (DC and transient simulations)
  2. plot of DC transfer function
  3. plot(s) of transient simulation; include versions at sufficient resolution to show delay and rise-time identification
  4. propagation delays and rise times
  5. noise margins for SPICE simulated nand2 gate
  6. R_{on} estimation from SPICE simulation results
  7. solution to Problem R

  Show all your work and steps for partial credit!

Setup electric for SPICE

Bring up File→Preferences.

Then select Categories→Tools→Spice.

- Set "Spice engine" to "Spice 3".
- Set "Spice Level" to "3".
- Set "Spice primitive set" to "SpicepartsS3"
- Select "User header cards from file:" and set the file to:

```
/home1/e/ese3700/ptm/22nm_HP.pm
```

Then click "Ok" to complete these changes.

The "header cards" in this case are the model for the transistors we're simulating. In this case, we are using the 22nm high performance (HP) transistors from the Predictive Technology Models (PTM) that came from ASU: `http://ptm.asu.edu/`
Note: We made a slight modification to the name of the models in the version stashed in `/home1/e/ese3700/ptm/22nm_HP.pm` so that it would work with Electric. If you grab the PTM version from the web, you will need to change the model names as well. We suggest you start with our version to avoid this technicallity.

The model file is also available on the course syllabus for download if you install Ngspice locally on your own machine.

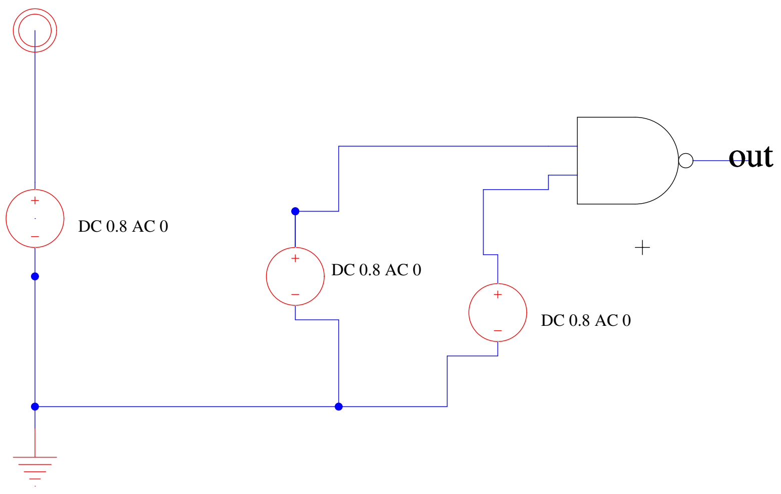
In future labs we may use different models (*e.g.* different device feature sizes) or compare the behavior of circuits between different technology models, so, at times, you will need to change the header files in use.

To finish setting this up for a 22nm technology, bring up File→Preferences again and go to: Categories→Technology→Scale. Select mocmos and set its "Technology scale':" (at bottom) to 22 (for 22 nm).

Test Setup: DC Analysis

In this section we will analyze a nand2 gate. In electric build a nand2 gate like you built in the previous lab with discrete transistors and create an icon. For DC analysis, create a schematic that includes an instance of your nand2. In addition to the nand2 instance, you will need to add:

- A voltage source (V_Generic under the "Spice" pallette [primitive set]) to drive the supply voltage (Vdd)
- Two voltage sources, one to drive each of the two inputs to the nand gate



Set all the voltage sources to DC 0.8V. In general, parameterized schematic elements (like transistors and voltage sources) have properites associated with them that you may need to set. Here, the voltage supplies have both an AC and DC voltage parameter. Leave the AC parameter at 0. You can change the value by double-clicking on the value (not the cell). You can actually change the entire text here, but you should leave it with "DC 0.8 AC 0".

Remember that the electric tutorial section you read last week on creating icons also discusses instantiating instances in schematics.

`http://www.staticfreesoft.com/jmanual/mchap01-12-04.html`

Note on Vdd and Gnd: Vdd and Gnd are global symbols in electric (and SPICE). So, you do not need to make them export terminals on your schematics/icons. Everything connected to the Vdd symbol at any level of your hierarchy is connected to the same net; similarly everything connected to the Gnd symbol is connected to the same ground net. You do, however, have to drive Vdd at some point in your circuit; you usually do this in the top level diagram as we've suggested above. If you don't do this, there will be no power to the circuit (just like not connecting, or forgetting to turn on, the power supply for a physical circuit).

Writing a SPICE Deck

To write out the SPICE netlist for simulation (also known as a "SPICE Deck"), use:

Tools→Simulation (Spice)→Write Spice Deck

Use this to write a file like: `nand2dc.spi`

The section of the electric manual on using SPICE is:

`http://www.staticfreesoft.com/jmanual/mchap09-04-03.html`

While it should be possible, we have not been able to read SPICE output back into electric, so we will be viewing the SPICE simulation results in `ngspice`.[1]

Understanding the SPICE Deck

Open your `.spi` file in your favorite text editor. Lines beginning with an astrisk (*) are comments.

- The `.include` line inserts text from elsewhere. In this case, it is the transistor models and should point to the PTM file you configured above when you set the SPICE preferences.
- The `.SUBCKT` ... .END section describes the subcircuit for your nand2 cell. This is how SPICE represents hierarchical schematics.
- The section at the end following the "TOP LEVEL CELL" comment is the netlist for your composed cell.

In both the top-level cell portion and the subcircuit portion, you have a description of the elements and their connectivity, with one element per line.

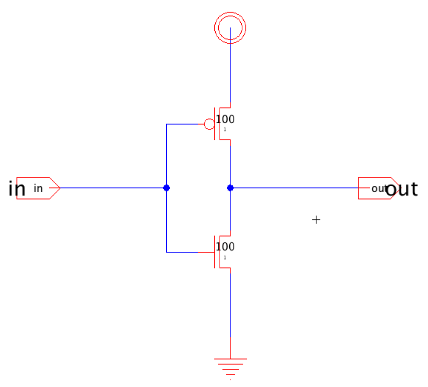
- In the subcircuit, you see the four transistors (two N and two P) in your design.
  - Following the name (first string on the line), you have the 4 terminals: source, gate, drain, body. You should convince yourself the connectivity this describes matches your schematic.
  - Following the four terminals, you have the model name (N or P)
  - Finally you have the model parameters, in this case the Length (L) and Width (W) of the transitor. If you made the change above to 22nm and left the transistors at their default size of 2 2, then both will be 0.044U, or 2×22nm=44nm.

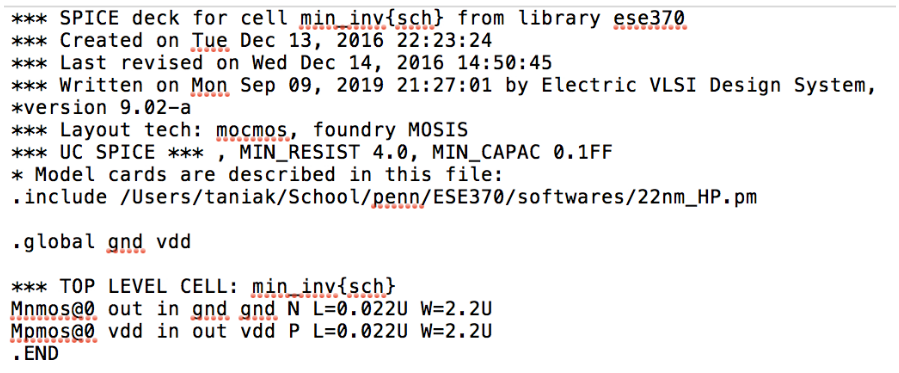
[1] If you figure out how to make it work, let us know!

- In the top-level circuit, you will see:
  - An instantiation of your nand2 (name, terminals, name-of-cell)
  - The voltage supplies you added to the cell.

What are the terminal names for the inputs and outputs to the nand2 gate? What is the name of the voltage supply instance driving one of the inputs to the nand2 gate? (You will need to know this for your DC analysis below.)

To help you learn how to parse your files for connectivity, below is an inverter and the corresponding SPICE deck. Note the device terminal node order and connectivity and device sizes.





```
*** SPICE deck for cell min_inv{sch} from library ese370
*** Created on Tue Dec 13, 2016 22:23:24
*** Last revised on Wed Dec 14, 2016 14:50:45
*** Written on Mon Sep 09, 2019 21:27:01 by Electric VLSI Design System,
*version 9.02-a
*** Layout tech: mocmos, foundry MOSIS
*** UC SPICE *** , MIN_RESIST 4.0, MIN_CAPAC 0.1FF
* Model cards are described in this file:
.include /Users/taniak/School/penn/ESE370/softwares/22nm_HP.pm

.global gnd vdd

*** TOP LEVEL CELL: min_inv{sch}
Mnmos@0 out in gnd gnd N L=0.022U W=2.2U
Mpmos@0 vdd in out vdd P L=0.022U W=2.2U
.END
```

ngspice

We will be using `ngspice` as a SPICE simulator this term. You can find the binary on CETS machines in: `/usr/local/bin/ngspice` (which should be in your path)

Start `ngspice` on your circuit nand2dc.spi using:

```
ngspice nand2dc.spi
```

Online, `ngspice` is available from source forge:

`http://ngspice.sourceforge.net/`

You can download the latest release (release 26) and build it for your laptop.

You can find a manual for `ngspice`:

`http://ngspice.sourceforge.net/docs.html`

You will probably need to consult the manual from time to time to review the full capabilities of various commands and features.

DC Analysis

To perform a DC sweep analysis, you use: `dc` *voltage-source-name min max step*

For this case, you want to sweep the voltage of one of the supplies driving the nand2 gate. If the name of that supply was VV_Generi@4, then your dc command might be:

```
dc vv_generi@4 0 0.8 0.01
```

This sweeps from 0 to 0.8V by increments of 10mV. Note: `ngspice` apparently converts the names to lower case, so even though there is mixed case in the SPICE Deck, `ngspice` won't recognize the node if you give it the name in anything other than all lower case.

This may give some warnings about features being too small (we will ignore those for now). Then it says there are some number of "Data Rows" and give you a prompt back.

At this point, you can plot the impact on the nand2 output by using the plot command. Assuming the node at the output of the nand2 is named `nand2@0_out`, then the plot command is:

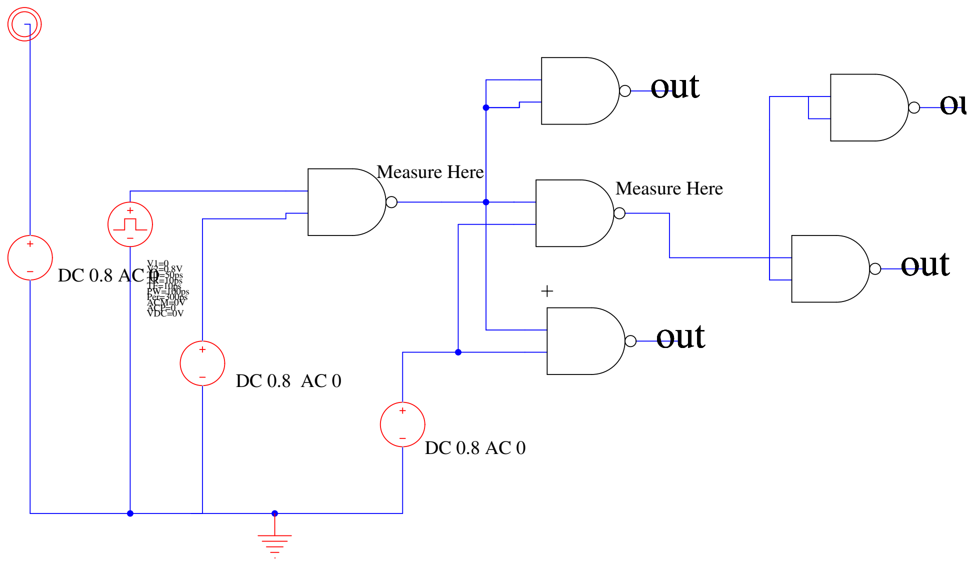
```
plot nand2@0_out
```

You can resize the plot window, zoom in on portions of it (right mouse click and drag right), and produce a postscript printout of the plot. You will want to produce and save the postscript output for your turnin and for reference as you identify the noise margins.

See Sections 17.5.13 and 15.3.2 of the `ngspice` manual for more details on the DC command.

Test Setup: Transient Analysis

For transient analysis, create a schematic that looks like:



- The second supply from the left is a VPulse. It is set with V1=0, V2=0.8V, TD=50ps, TR=10ps, TF=10ps, PW=250ps, Per=500ps. This is a pulse train that transitions between 0 (V1) and 0.8V (V2) with a 500ps period (Per) and a 250ps pulse width (PW). The rise time is 10ps (TR) as is the fall time (TF). The pulse is delayed 50ps (TD) from the start of the simulation.
- The other supplies are all set to 0.8V.
- To make sure we're measuring the typical behavior of the nand2 in a circuit, we measure across a nand2 that is both driven by a nand2 (not the artificial pulse generator) and loaded by nand2s. Here we use a standard "FO4" load—the load of 4 gate inputs. In the final stage, we create this by shorting together the two inputs to each of two nand2 gates. For the gate we are actually measuring (middle gate of middle row), we only switch one input, but we make sure the switching input is also loaded by 4 gate inputs.

Write out the SPICE Deck for this. Examine the SPICE Deck and identify the node names associated with the two nets marked "Measure Here".

Transient Analysis

Run `ngspice` on your netlist created. In SPICE, run the transient analysis command:

```
tran 1ps 1ns
```

This tells SPICE to run a simulation on your circuit for 1ns with a simulation granularity of 1ps.

You can now plot the result of your simulation by issuing a plot command. If the two nodes marked “Measure Here” are net@2 and net@3, then you can plot those two nodes with:

```
plot net@2 net@3
```

You can manipulate this plot as before. You will want to zoom in on the transitions and plot those for your handin and for determining propogation delays and rise and fall times. Note that you can measure voltage and delay differences on the plot by left-clicking and dragging the mouse (measurements show up in the output of the ngspice interaction window, not on the plot).

If you don’t see results you expect, you may need to plot other nodes to debug your circuit.

You can read more about the transient analysis command in the `ngspice manual` in Sections 17.5.77 and 15.3.9. You can read more about plotting in Section 17.4.43.

Plotting

You can plot currents as well as voltages. See the sections on the `.print` 15.6.2 and `.plot` 15.6.3 directives in the `ngspice manual` (ngspice-26). You will eventually want to use the `.measure` directive, which is described in section 15.4.

Naming

Do **not** use hyphens in any of the names you give to cells, signals, or ports. Electric will not complain. However, SPICE will misinterpret them as a minus sign and then fail to parse the lines in the SPICE deck. This has been a source of frustration for ESE3700 students in the past. Similarly, do **not** get fancy and try to put a plus (+) in a name.

The error message it gives in these cases is not the most clear. It will typically complain about there being too many nodes. This results from SPICE assuming the plus or minus sign splits the name you intended into two separate names, so it sees more names (of nodes) than it knows how to handle.

Software Tool Guides

In addition to all the resources above, there are helpful guides on our course webpage under the ”Tool Guides” heading: `https://www.seas.upenn.edu/~ese3700/#tools`

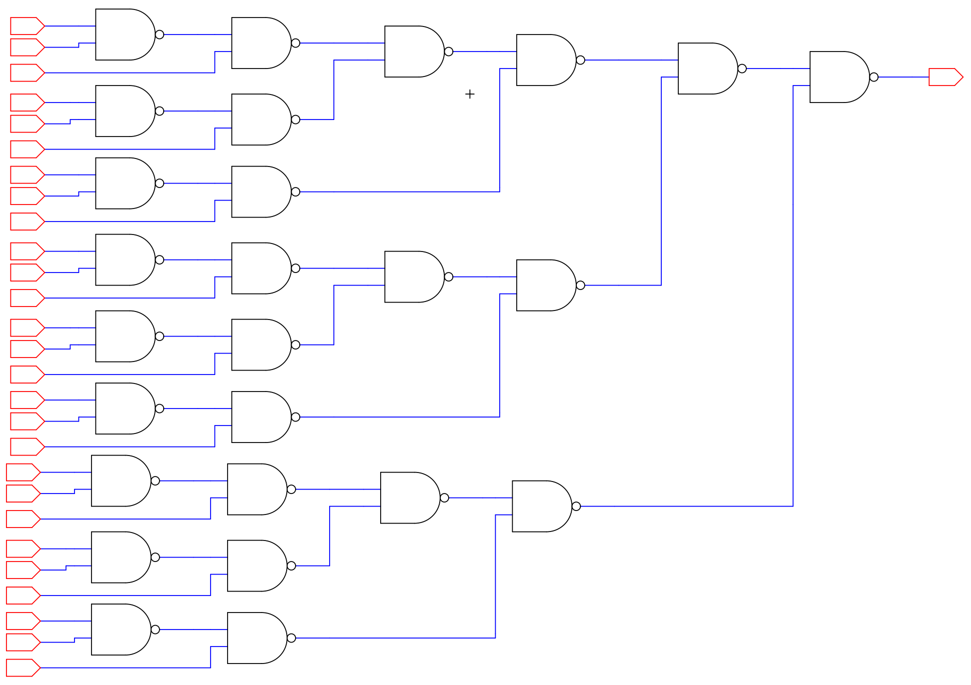
Problem R

Consider a "nand2" gate that has the continuous transfer function:

Function	When
$V_{out} = 1 - 0.25V_a \times V_b$	$V_a \times V_b < 0.2$
$V_{out} = 1.25 - 1.5V_a \times V_b$	$0.8 > V_a \times V_b > 0.2$
$V_{out} = 0.25 - 0.25V_a \times V_b$	$V_a \times V_b > 0.8$

where V_{out} is the voltage on the output of the gate when the input voltages are V_a and V_b.

1. For the circuit shown, what is the output voltage of the final nand2 when:
   (a) All the inputs are 1.0.
   (b) All the inputs are 0.95.



2. Identify the noise margins for this "nand2" gate (i.e., identify V_{OL}, V_{IL}, V_{OH}, V_{IH}, NM_L, and NM_H, for robust signal restoration).
   (Hint: consider worst-case degradation on both inputs.)
3. Building on your answer to part 2 and assuming all the inputs are 0.95, how much voltage drop could the circuit from part 1 tolerate on the wire between each pair of these nand2 gates and still produce a valid logic level at the final output of the composite nand2 circuit shown?
4. Lab 1 measurement (**Extra Credit**, Part 2): from the data received in your Lab 1 demo, characterize the performance of your nand2 gate

(a) Plot the voltage transfer characterstic (VTC), $V_{out} = f(V_{in})$, when the input is driven with a slow moving ramp waveform. Your axes should be V_{in} vs. V_{out} and should not include time.

(b) Identify workable noise margins for your measured gate from the gate-from- transistor lab. That is, identify V_{OL}, V_{IL}, V_{OH}, V_{IH}, NM_L, and NM_H, for robust signal regeneration.