EE307 Digital Electronics and Integrated Circuits

HW3 - Assigned: 1/22/18 Due: 1/26/18

SOLUTIONS: 41 points total

This week's homework is about entering designs and measuring things (and working towards a datapath/ALU unit for a bigger design that we'll be creating over the quarter).

This week's homework is about entering designs and measuring things (and working towards a datapath/ALU unit for a bigger design that we'll be creating over the quarter).
This homework might be easier to do electronically since there are a bunch of screen captures…
If you do the on-line submission, please submit a pdf.

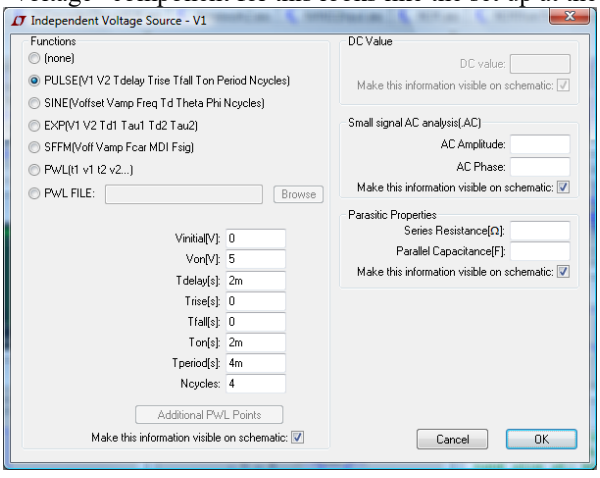
READING part 1: New ways to set inputs

How to set inputs to cover truth table inputs:
In this homework and future homework you'll need to test your circuits. That means setting their inputs and watching their outputs. So far you've selected a "voltage" component, right clicked right on the component, selected "Advanced", clicked on PULSE, and set Vinitial, Von, and Ton to get a single pulse that stays on for Ton and then goes to Vinitial for the rest of the time. But, to completely test a digital circuit, you need to test every entry in a truth table. Here's a truth table for a 3-input XOR gate:

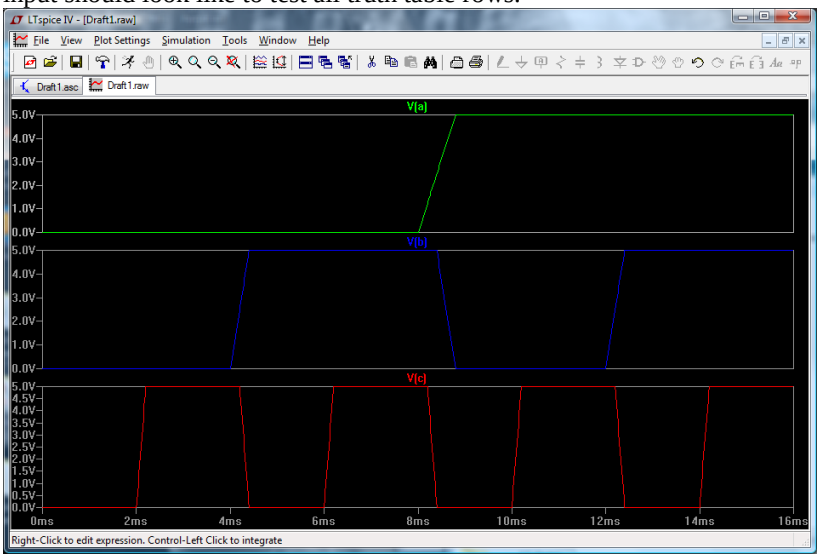
INPUT			OUTPUT
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

I'll show you five ways to do that here:

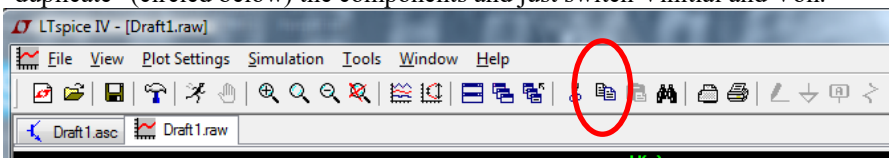
<u>Way 1 to set inputs:</u> Repeated pulses
If you look at the "C" input column, you see that it alternates from 0 to 1 for every new line. "B" alternates every two rows. "A" alternates every 4 rows. Let's look at "C" for a second. Let's say you want the simulation to stay at each row for 2ms. (You've seen that the transitions happen much quicker than that so selecting 1ms or 0.5ms would work just fine too but for this example I'll use 2ms). "C" stays at o for 2ms, then goes to 5V for 2ms, then goes back to 0 for 2ms more, then back to 5v, etc etc… To make "C" do this you can repeat a pulse. The set up on the "voltage" component for this looks like the set up at the top of the next page.



To do the entire truth table you need multiple sources all set to a different delays and different Tons. Here's what the input should look like to test all truth table rows:

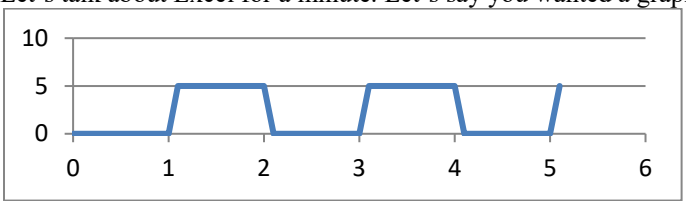


If you need the inverted signals then you can use an inverter (we'll add a symbol to your inverter in one of the next pages so you can simply just drop your own inverter in to a schematic to create the negated signal) or you can also "duplicate" (circled below) the components and just switch Vinitial and Von.



<u>Way 2 to set inputs:</u> Piece-wise linear

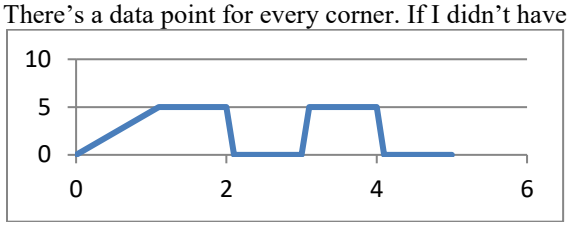
Let's talk about Excel for a minute. Let's say you wanted a graph that looked like:



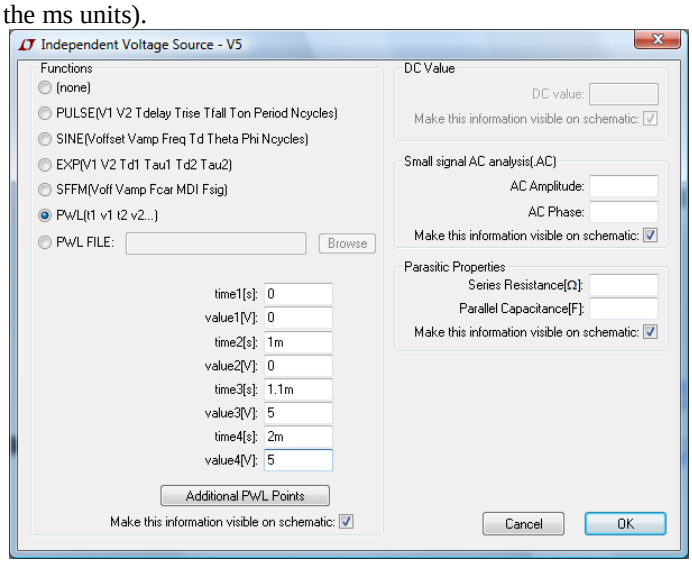
The data would look like:

0	0
1	0
1.1	5
2	5
2.1	0
3	0
3.1	5
4	5
4.1	0
5	0
5.1	5

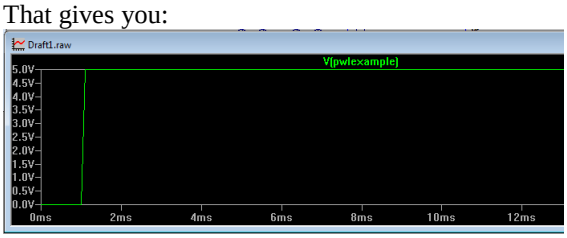
There's a data point for every corner. If I didn't have the point (1,0) the graph would look like:



So to get a graph in Excel, you need a point at every corner. Understanding how to get a shape by specifying points in Excel is really all you need to know about to specify a custom input in LTSpice. PWL is a way to enter a custom signal of any shape where you enter the points just like you would in Excel. To enter a custom shape, again drop in a "voltage" component, right click on it and select Advanced. Then select PWL (t1 v1 t2 v2…). I've entered the same points I used in the Excel example above into the textboxes from 0ms to 2ms. (In the excel example I didn't include the ms units).



That gives you:



Which isn't enough points. To add more points click on "Additional PWL Points" and you'll be able to add as many points as you'd like.

Way 3 to set inputs: Use .op to specify input through text specification.
You can enter supply through the .op icon. The example I did above can be entered as:

```
V5 PWLexample 0 PWL(0 0 1m 0 1.1m 5 2m 5)
```

You can also enter:

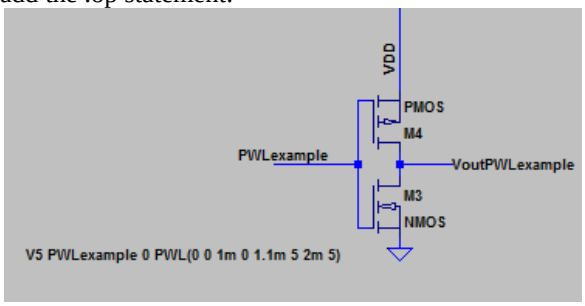
```
V5 PWLexample 0 PWL REPEAT FOR 5 (0 0 1m 0 1.1m 5 2m 5) ENDREPEAT
```

To repeat a pattern over and over or:

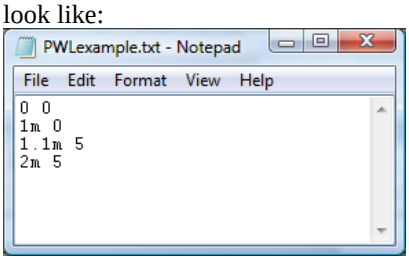
```
V5 PWLexample 0 PWL REPEAT FOREVER (0 0 1m 0 1.1m 5 2m 5) ENDREPEAT
```

To repeat forever.

These are in place of the “voltage” component. You need to delete the “voltage” component from the schematic and add the .op statement:



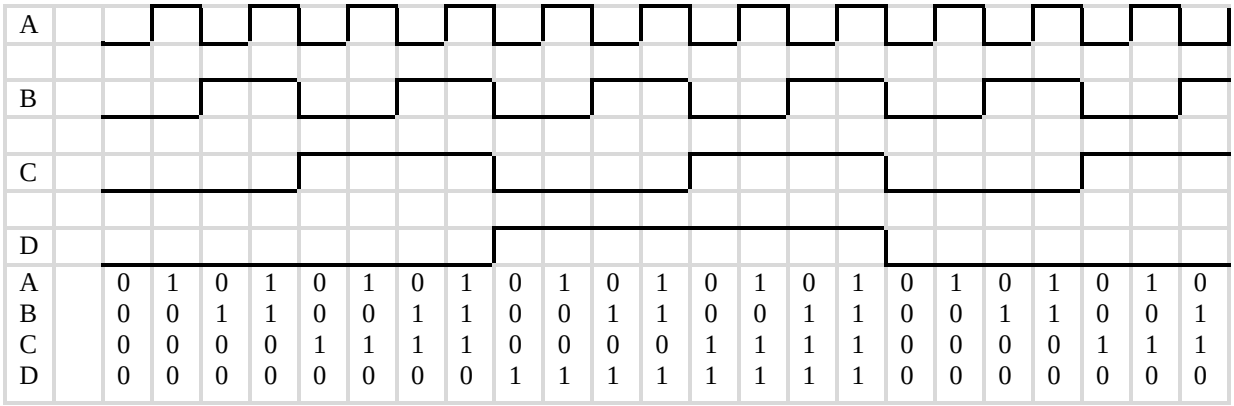
Way 4 to set inputs: Put points into a file
You can also create a file and, instead of using “PWL (t1 v1 t2 v2 etc)”, you can use “PWL FILE: “. Use an editor like notepad that doesn’t do formatting keep from having invisible characters in the file. Save as text. The file would look like:



When you select “PWL FILE” it will let you browse so you can select the file. This is nice to use if you have a particular waveform that you might use again in the future or on many sources. You can just put it in a file and attach it to any “voltage” source that you want to behave that way.

READING part 2: How to use text to specify voltage sources

A truth table specifies all possible inputs on the eft side of the table and the output for each input on the right side of the table. The only way to completely test a combinational logic circuit is to give it ALL truth table inputs and see if all the outputs match. To give a circuit all input combinations as a truth table does, you have some options. The easiest way is to imitate the truth table and make the LSB of the input change twice as fast as the second bit and have the second bit change twice as fast as the third and so on and son on. That would look like:

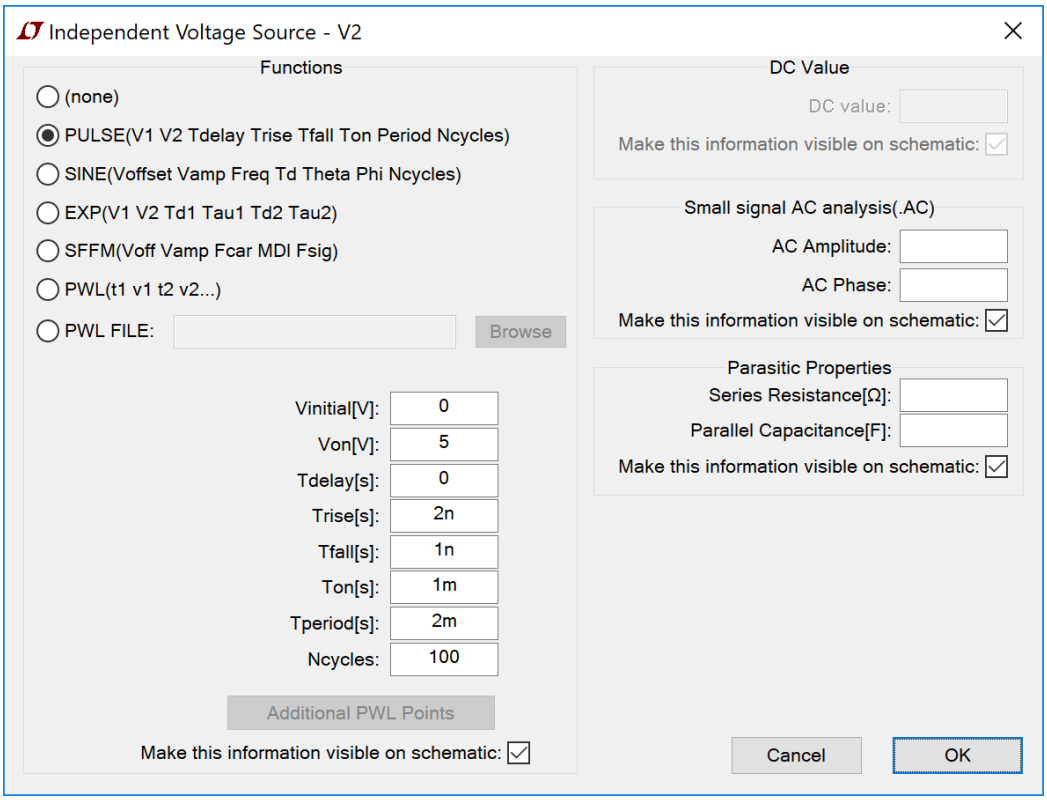


A		0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
B		0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1
C		0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1
D		0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0

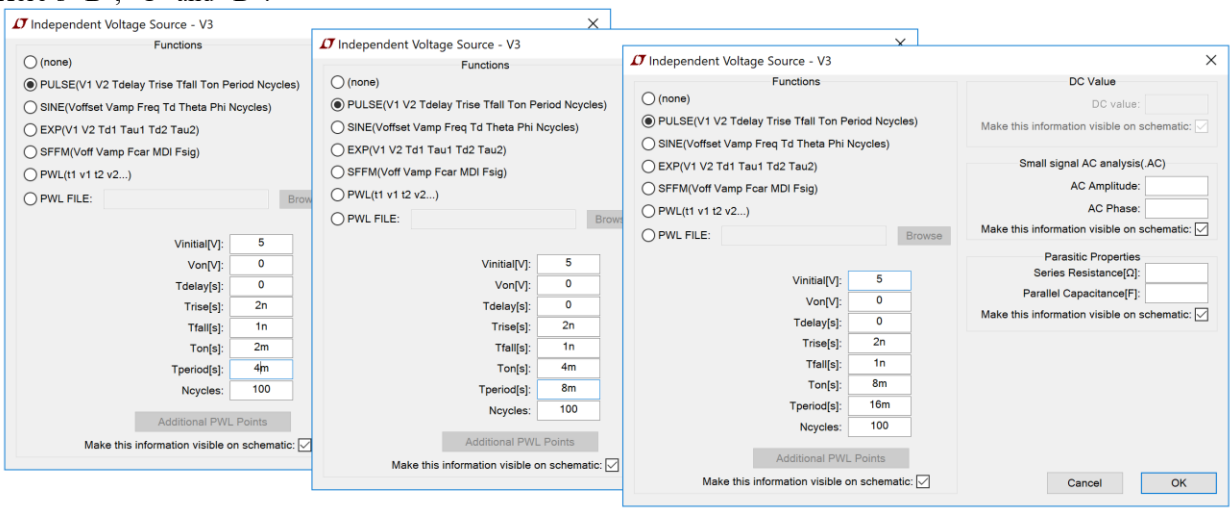
If you look at each successive column you can see that they are counting up (if you consider “A” as the LSB). You can also see that “B” is half the frequency as “A”, “C” is half the frequency of “B”, and “C” is half the frequency of “D”. That means the period doubles from “A” to “B”, “B” to “C” and “C” to “D”.

There are multiple ways to set up this input:

1. You can drop a “voltage” component in the schematic and then, through the “Independent voltage source” setting window, set the pulse information. Here’s the specification for “A”:



Here’s “B”, “C” and “D”.



Note that the period and ON time double each source.

2. OR you can add the following text through the .op icon:

```
VA A 0 PULSE(0 5 0 2n 1n 1m 2m)
VB B 0 PULSE(0 5 0 2n 1n 2m 4m)
VC C 0 PULSE(0 5 0 2n 1n 4m 8m)
VD D 0 PULSE(0 5 0 2n 1n 8m 16m)
```

You no longer need voltage sources if you use this technique. The wires into your circuit need to be named A, B, C and D.

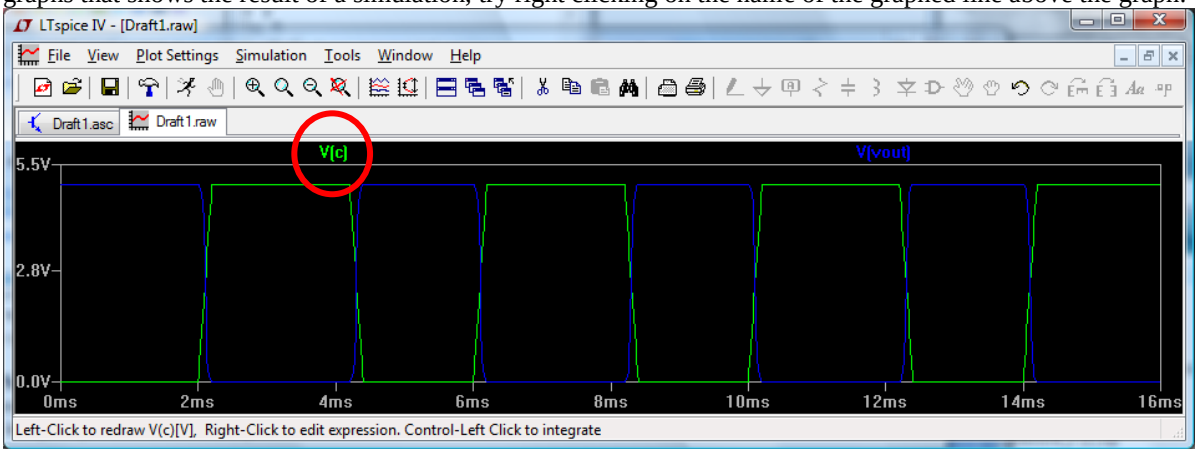
3. Or you can put the above text into a file and then use the .op icon to ".inc" that file. Here's what would go in the file:

```
VA A 0 PULSE(0 5 0 2n 1n 1m 2m)
VB B 0 PULSE(0 5 0 2n 1n 2m 4m)
VC C 0 PULSE(0 5 0 2n 1n 4m 8m)
VD D 0 PULSE(0 5 0 2n 1n 8m 16m)
```

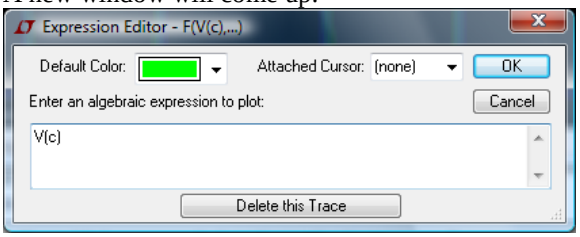
The advantage of putting the sources into a file is that that each time you need a truth table input all you need to do is to include this file. It ca be re-used as long as you name the wires A, B, C, and D.

READING part 3: Using cursors

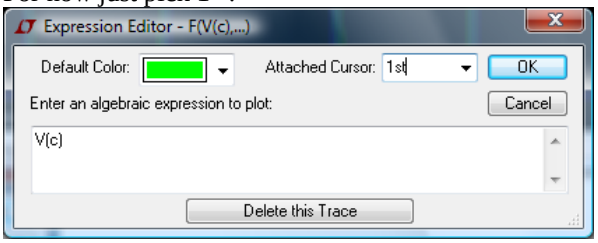
So now you've seen some new ways to set inputs. Next we'll look at some ways to read outputs.
If you are in my lab you've seen some ways to use cursors. I'll describe those and one or two more.
There are a number of shortcuts to turn cursors on but I'll go with just one technique here. That is: On one of your graphs that shows the result of a simulation, try right clicking on the name of the graphed line above the graph:



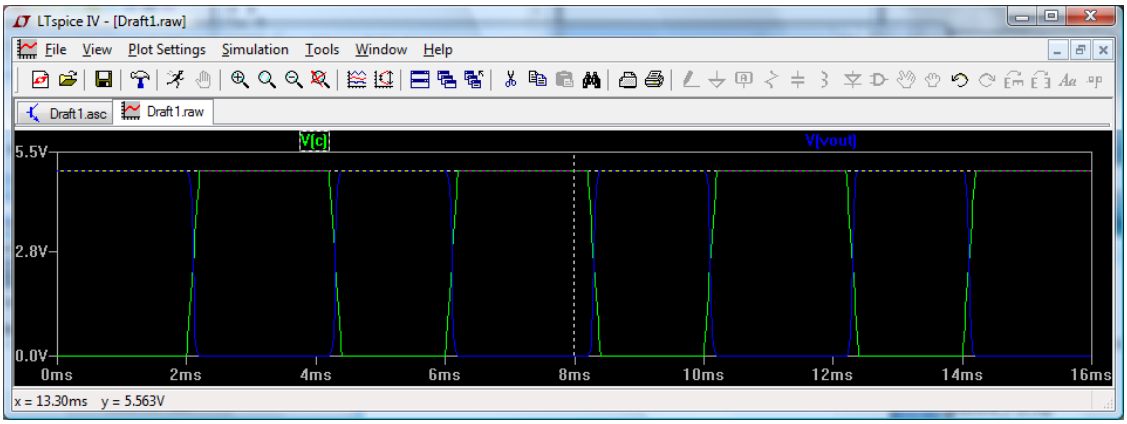
A new window will come up:



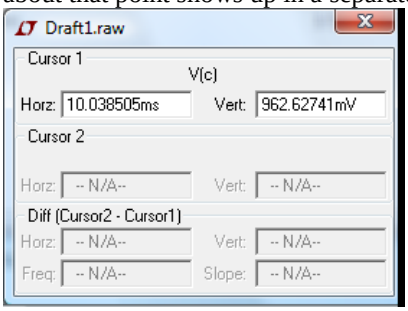
There are two pulldown lists. One let's you pick a new color for your graph and the other lets you turn on cursors. For now just pick 1st :



Click OK and you'll see some new whitish lines on your graph:

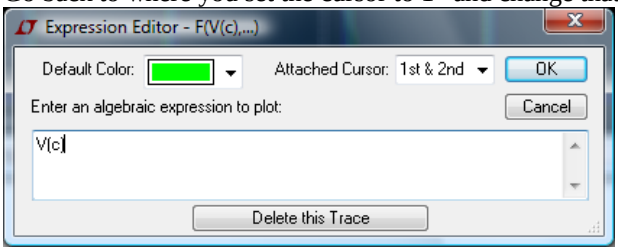


If you bring your mouse cursor over the line and move it around you should find a point where the cursor turns into a "1". If you left click you'll see the lines move around. Also notice that where the horizontal and vertical white line crosses is always on whatever signal you just set the attached cursors to 1st on. As you move the cursor, information about that point shows up in a separate window:

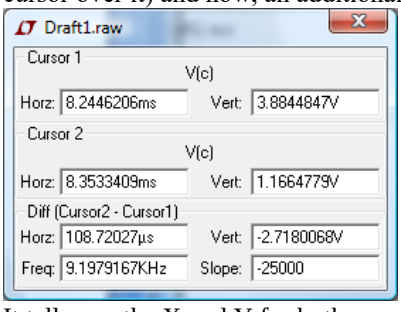


This tells you about what X and Y are.

Go back to where you set the cursor to 1st and change that to 1st and 2nd :

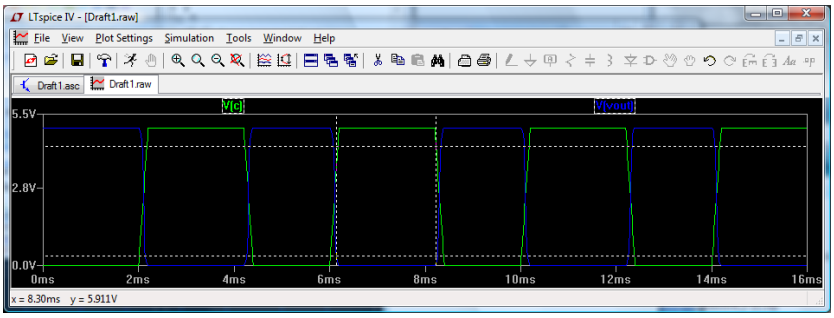


Click OK and now you'll have the original cursor (1st – The one that showed a "1" when you moved the mouse cursor over it) and now, an additional, cursor "2". The little information window now has some new values it shows:

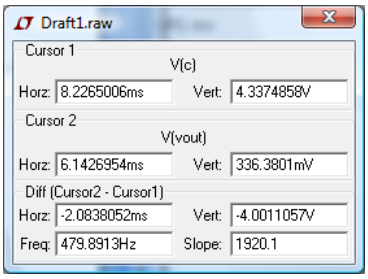


It tells you the X and Y for both cursors, the difference in the Xs of cursor 1 and Cursor 2, the difference in the Ys of cursor 1 and Cursor 2, and also the slope between the two cursor points. Slope is important for analog signals but, in this class, you'll often need a difference in time and that's where this is useful.

Both of the cursor usages I just showed you were done on a single signal. You can also set one cursor on one signal and the other cursor of another signal. To do that, go to one signal and set 1st and then go to the other signal and select 2nd :



Now one cursor will follow one line and the other cursor will follow the other. The information window will give the same information that it did when both cursors were on the same signal.



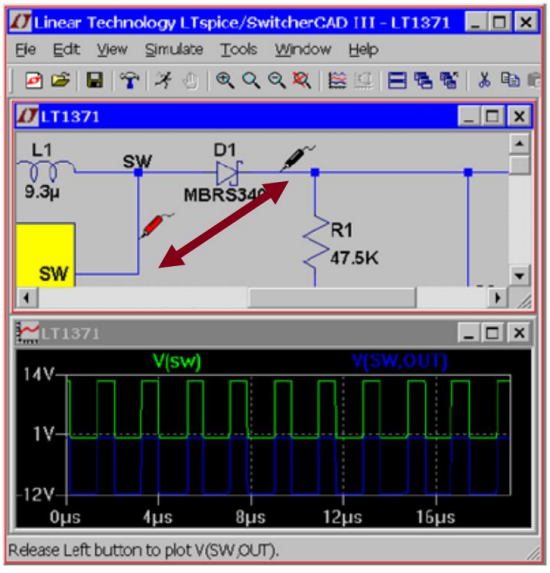
When you start to measure time differences between two points, this will be quite useful.

You can also take the difference between two nodes by:

Voltage Differences Across Nodes

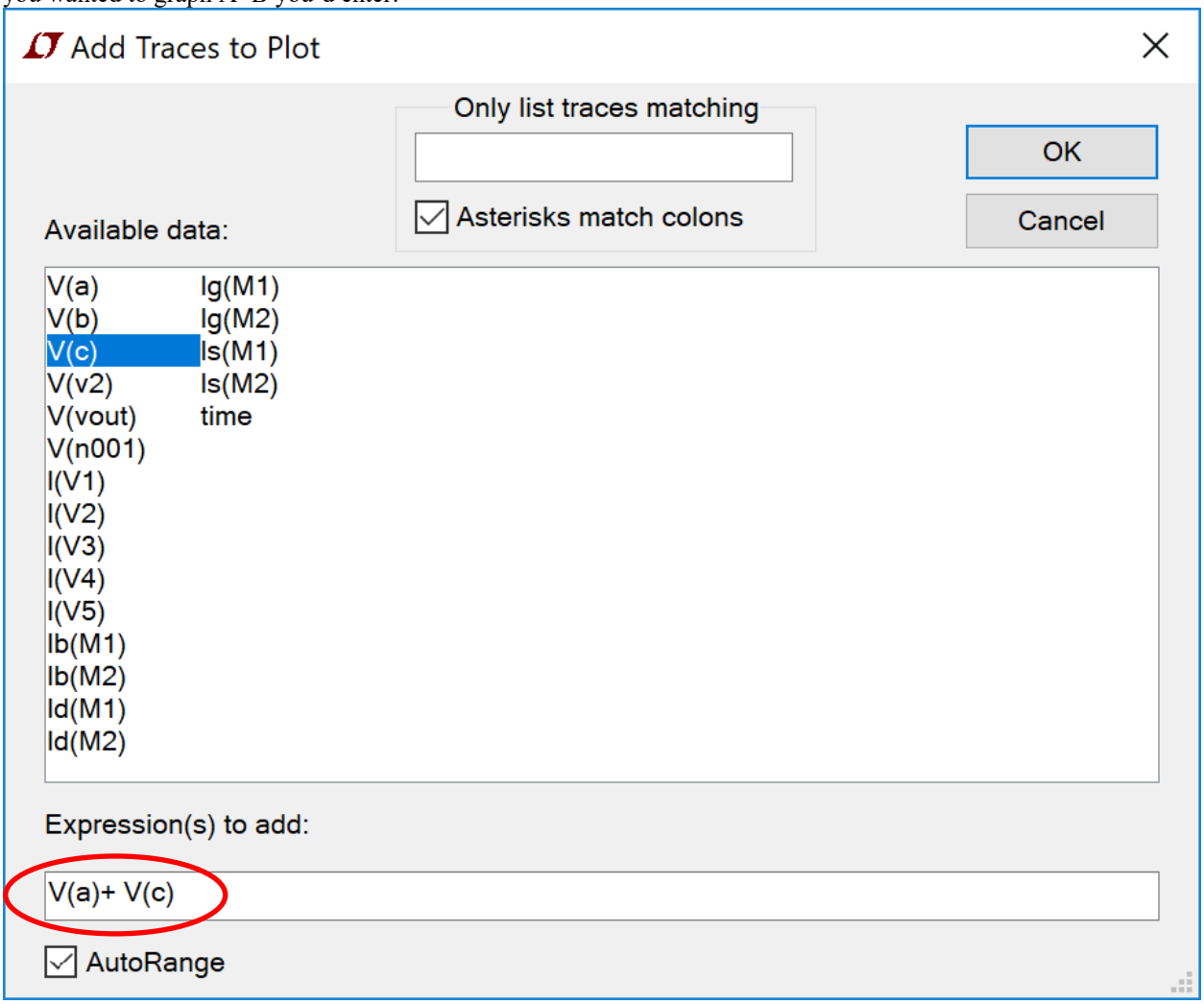
- Left click and hold on one node and drag the mouse to another node
  - Red voltage probe at the first node
  - Black probe on the second

Differential voltages are displayed in the waveform viewer



(http://cds.linear.com/docs/en/software-and-simulation/LTspiceGettingStartedGuide.pdf)

You can do math by right clicking on graph and selecting “Add traces”, and entering an equation. For example, if you wanted to graph A+B you’d enter:



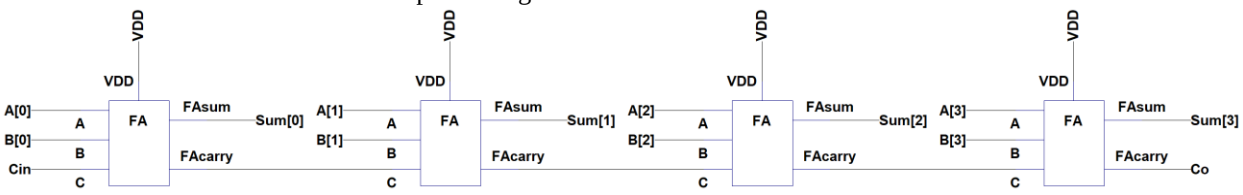
Finally on to the actual homework!!!!!!!!!

1. Read the reading for this homework (pages 1-9 of this handout) and, when answering questions, on LTSpice graphs:
    a. Use multiple panes to make it easier for the reader and I to see what you've done:
    b. Right click on graph.
    c. Select "Add Plot Pane".
    d. Click on a pane to select it (nothing visible changes). Then click on wire in schematic to add it to selected pane.
    e. If you want to move signals to a different pane, drag the NAME of the wire to a new pane and drop it in the graph area.
    f. To use less ink, go to Tools → Color Preferences
    g. In the Select Item area, select background.
    h. Slide all three slider to the right.
    i. OK.

    . The answer is OK!

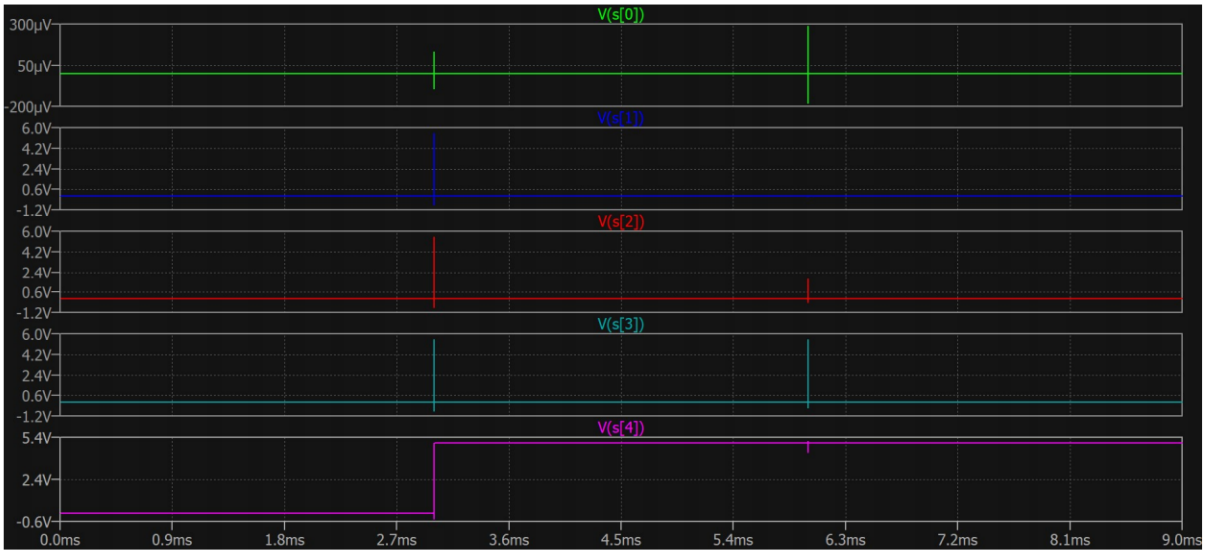
    ANSWER: OK!!!
2. Build a 4-bit ripple carry adder using the gates from last week and creating any new gates that might be needed using CMOS logic. The correct design technique would be to test each gate first (smallest piece of design) and then put them together into the next bigger piece. Then, after testing those put them into the next bigger block… Create a file using the sources at the top of page 6 of this handout and use them as inputs to test your adder. Include a screen capture of the inputs and outputs (sum bits) so the grader can see the results of the simulation clearly. Note that buses can be used. Simulate for 0000+0000, 1111+0001, and 1000+1000.
   (http://ltwiki.org/index.php5?title=Undocumented_LTspice#Bussing_of_Connections_and_Components_.28BUS_shorthand_notation.29)
   ANSWER: (8pts for showing it works. Some people missed email on what inputs to use so just check a couple of values)

Cin on the 0th bit is GNDed in the complete design:



Simulations:

0000 + 0000 = 0000 1111 + 0001 = 10000 1000 + 1000 = 10000

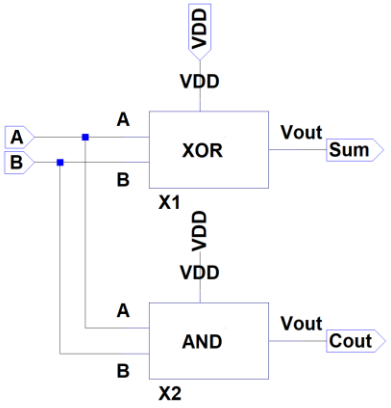


3. Multiplier:
    a. Watch the array multiplier videos and then build one using your AND gates, HAs and FAs. Test three values: All zeros, all 1s and 0111 x 0110. Show a screen capture of the three results. The Multiplier takes in two 4-bit numbers to multiply. Note that buses can be used. (http://ltwiki.org/index.php5?title=Undocumented_LTspice#Bussing_of_Connections_and_Components_.28BUS_shorthand_notation.29)

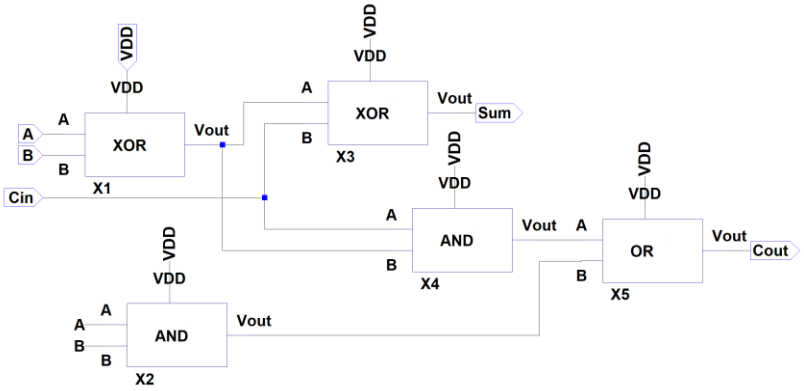
ANSWER: (14 pts)

Here's a pretty multiplier turned in by a classmate:

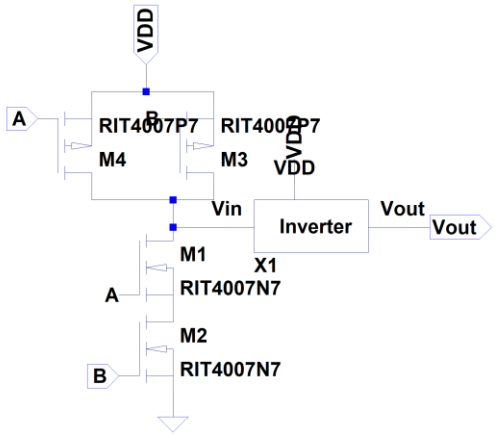
Half-adder:



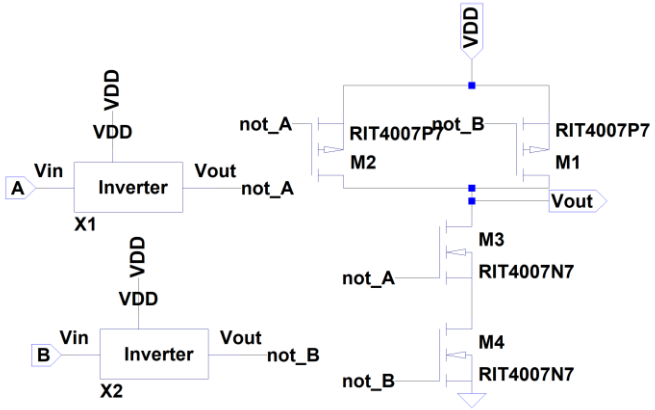
Full adder:



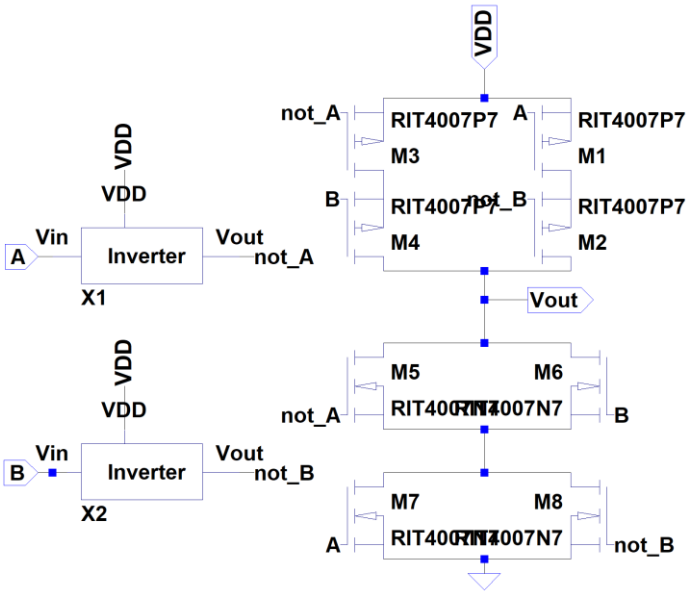
AND:



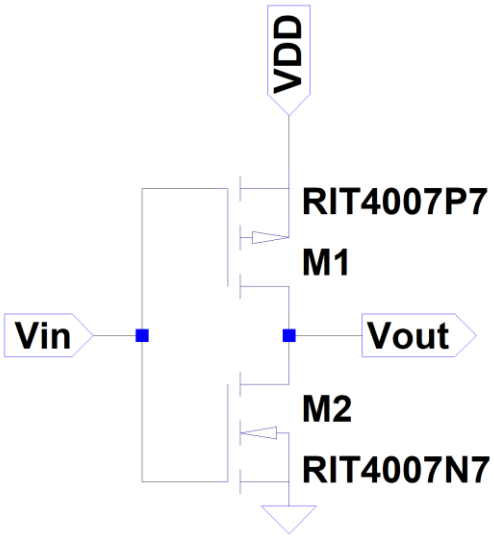
OR:



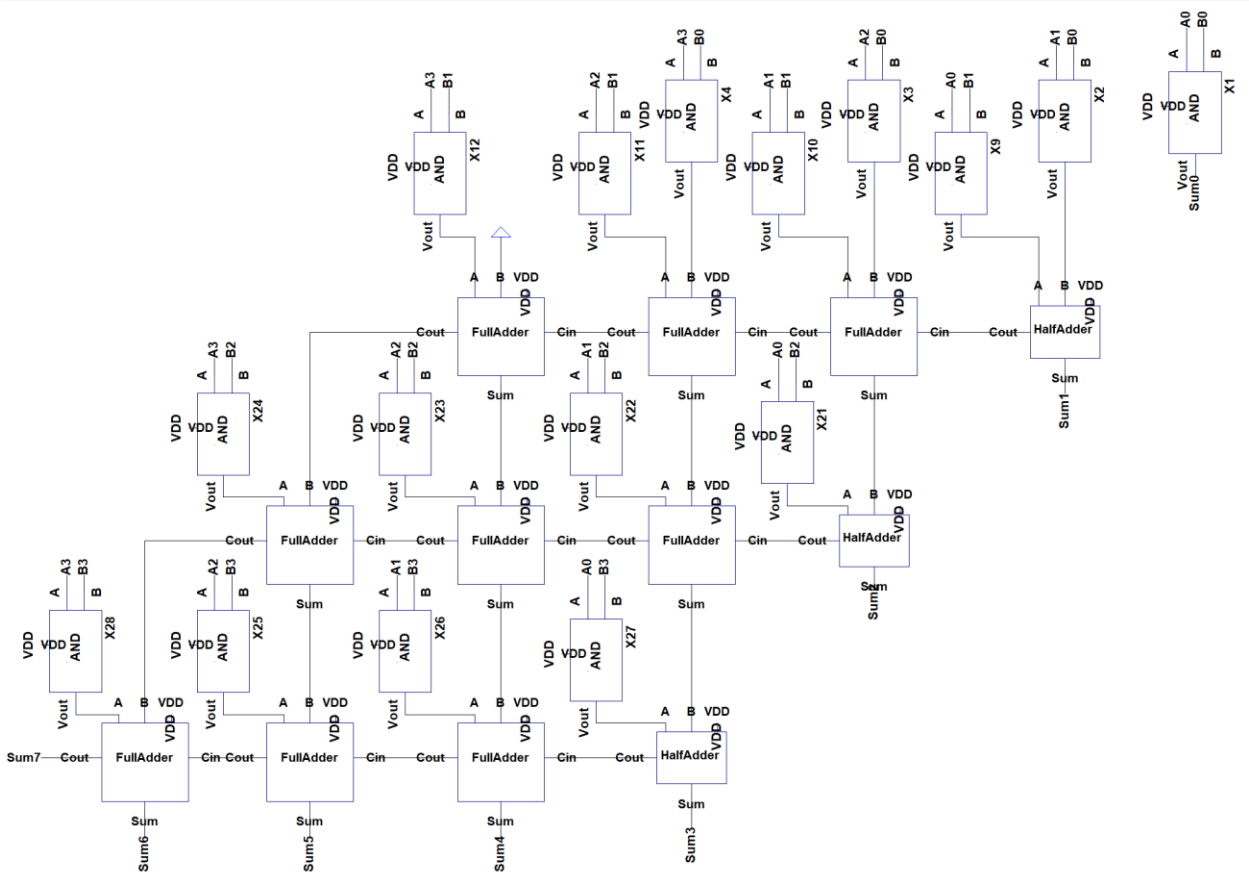
XOR:



INV:



Full multiplier:



Simulations stolen from another student (actually a team):

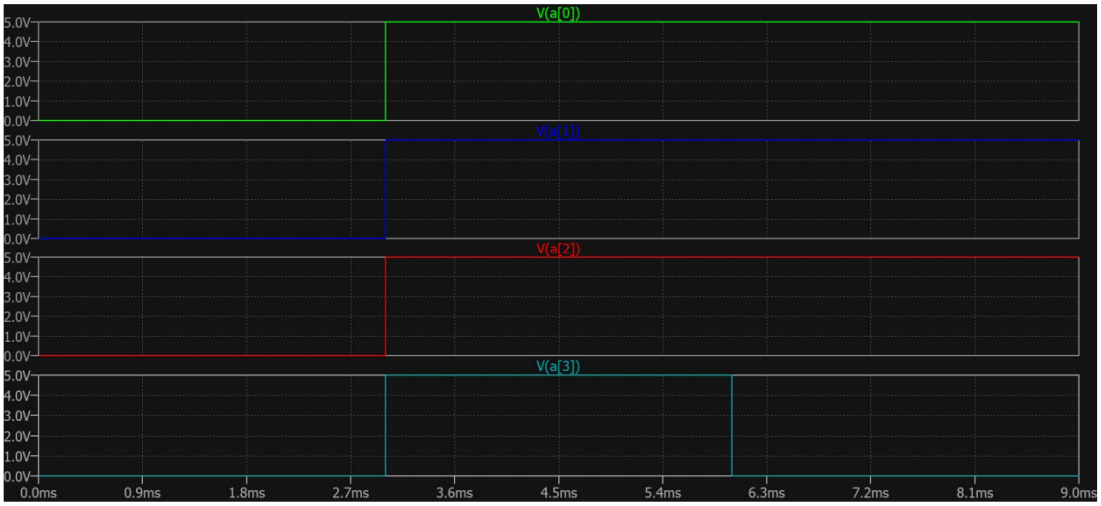


Figure 5 – Multiplier 4 bit A input waveform.

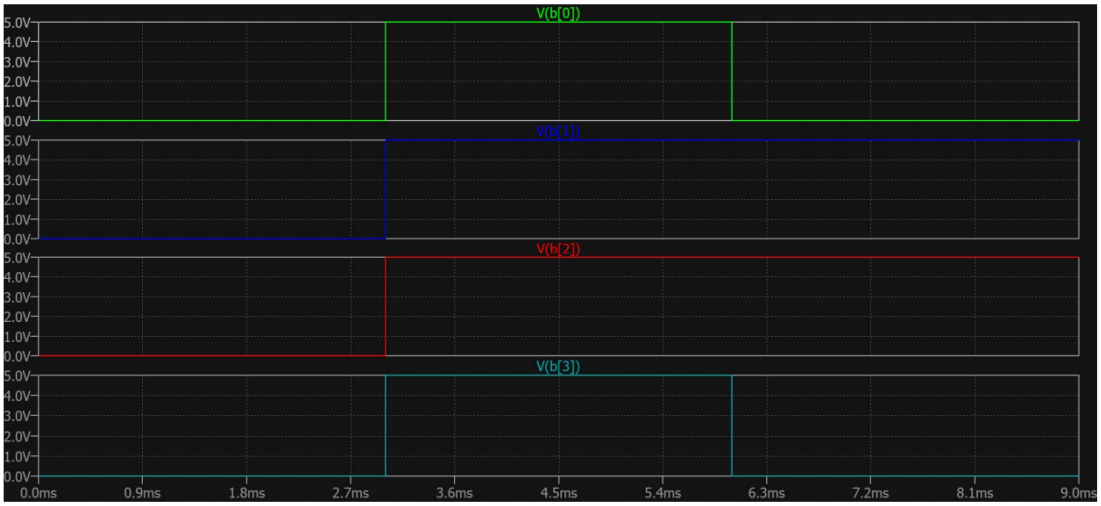


Figure 6 – Multiplier 4 bit B input waveform.

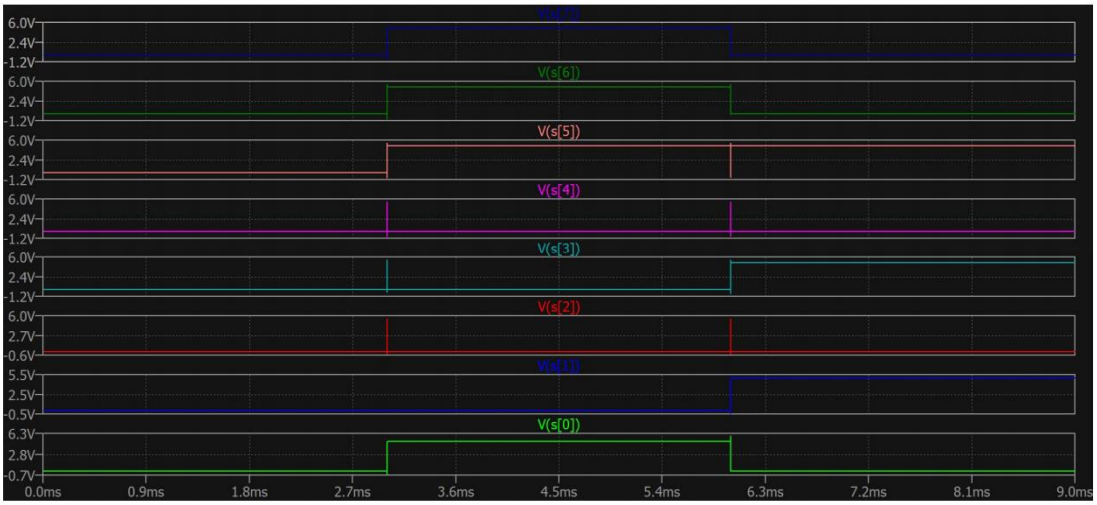
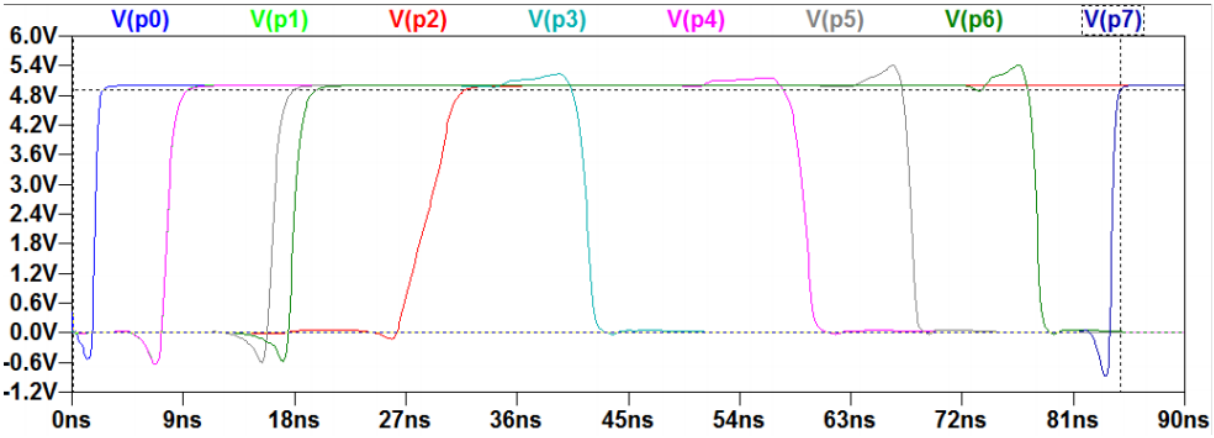


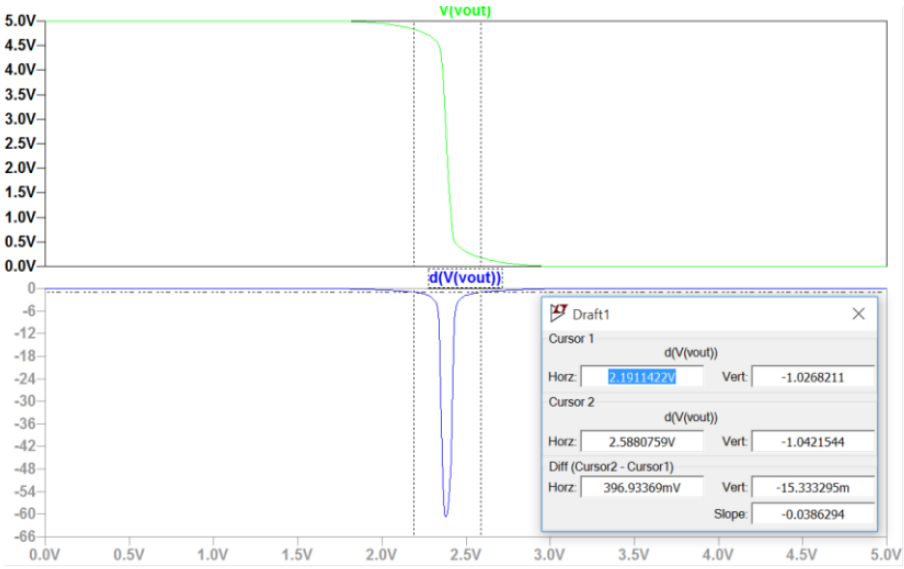
Figure 7 – Multiplier S output waveform.

b. What inputs give the worst delay?
ANSWER: (4pts)
There were two models used and they give different delays. Also, if you designed any of your gates differently, they might have a different delay but 1111 x 1111 should give the worst delay. I see a homework that is well documented and shows that that 0111 x 0110 is the slowest. If well documented full points but if no graphs shown and the answer 0111 x 0110 is given, no points.

c. What is that delay? Note that you'll need to set all the inputs to zero and then set them to your input with a fast rise time to accurately measure your delay.
ANSWER: (4pts)
Acceptable answers are in the range of 10s of ns to ~200ns. If the delay is in the ms there may be a couple of causes. 1) The rise and fall times for the inputs are left as 0. They should be changed to 1n. 2) The default LTSpice model is still associated with a transistor someplace. They all should have a model connected to them. Half points for a delay in ms assuming model left as NMOS and/or PMOS someplace.
One person checked all inputs and found the worst. Absolutely full points for that. Slowest for the models he used and inputs 1111 x 1001:



4. Metric measurements:
   a. Process variation happens when transistor are doped differently (concentration and/or depth), or have a slightly different size. These are unintentional differences and can affect the transistors' VT. How might process variation affect the noise margins?
   ANSWER: (3pts. Comments on VT give full points)
   Doping affects VT. VT affects VIL and VIH. When those move they directly change the noise margin. If you are using an RTL inverter, doping variations will change the R value and therefore the VOL. VOL changes the NML. All of these circuits can be flipped and implemented with PMOS and then a change in R would change VOH and NMH.
   b. Find the noise margins of your NAND gate.
   ANSWER: (4pts. Full points if approach looks good. Numbers should be close to these)
   An example. Values may be different depending on your model, inputs selected, and design (though this design is pretty much the same for everyone). Stolen from the same person I stole the picture in the next answer (I especially like the touch where he used the derivative):

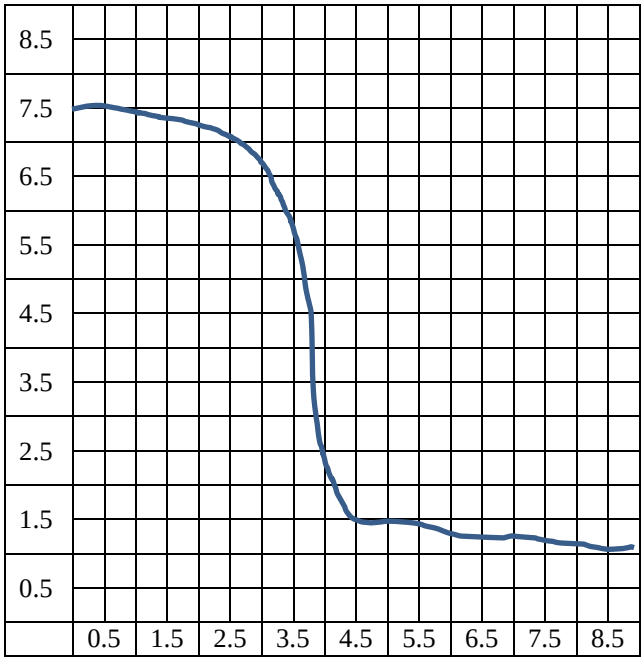


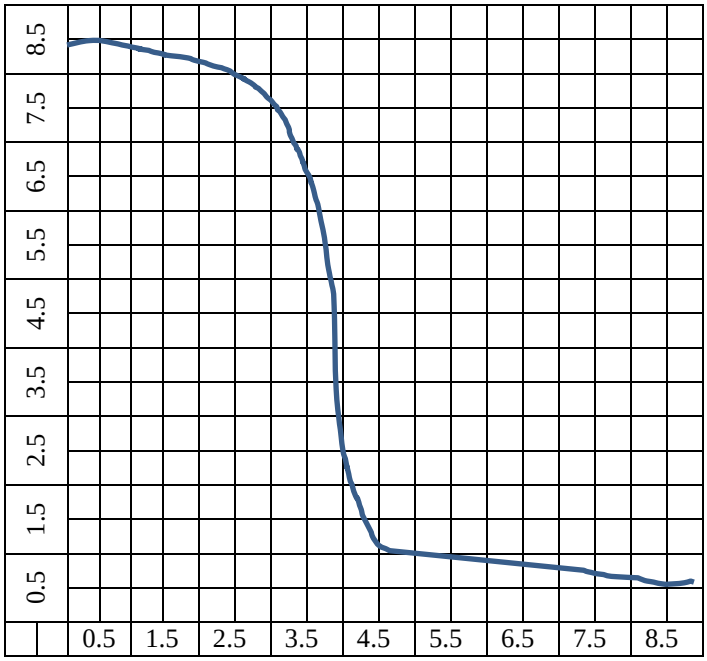
Using cursors, VIL = 2.191V and VOH = 2.588V. Because my NAND is CMOS and we are rail to rail, V_{OUTMAX} = 5V and V_{OUTMIN} = 0V. Thus,

$$NML = VIL - V_{OUTMIN} = 2.191V - 0 = \underline{\mathbf{2.191V}}$$

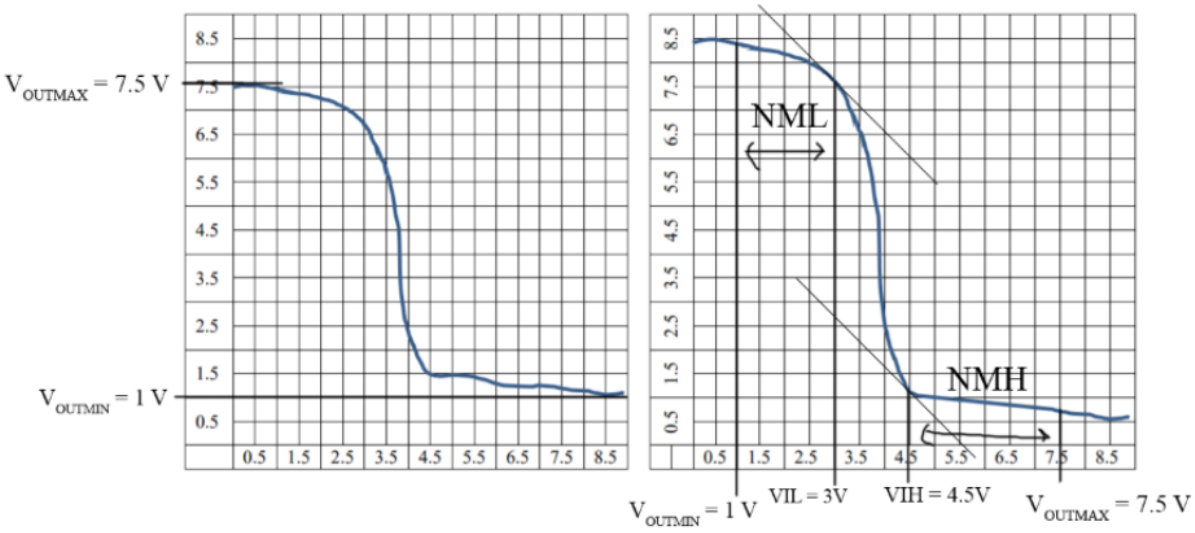
$$NMH = V_{OUTMAX} - VIH = 5V - 2.588V = \underline{\mathbf{2.412V}}.$$

c. What are the noise margins when circuit 1's output is the input to circuit 2.





ANSWER: (4pts)
Stolen from yet another classmate:



$$NML = VIL - V_{OUTMIN} = 3V - 1V = \underline{\mathbf{2V}}$$

$$NMH = V_{OUTMAX} - VIH = 7.5V - 4.5V = \underline{\mathbf{3V}}.$$