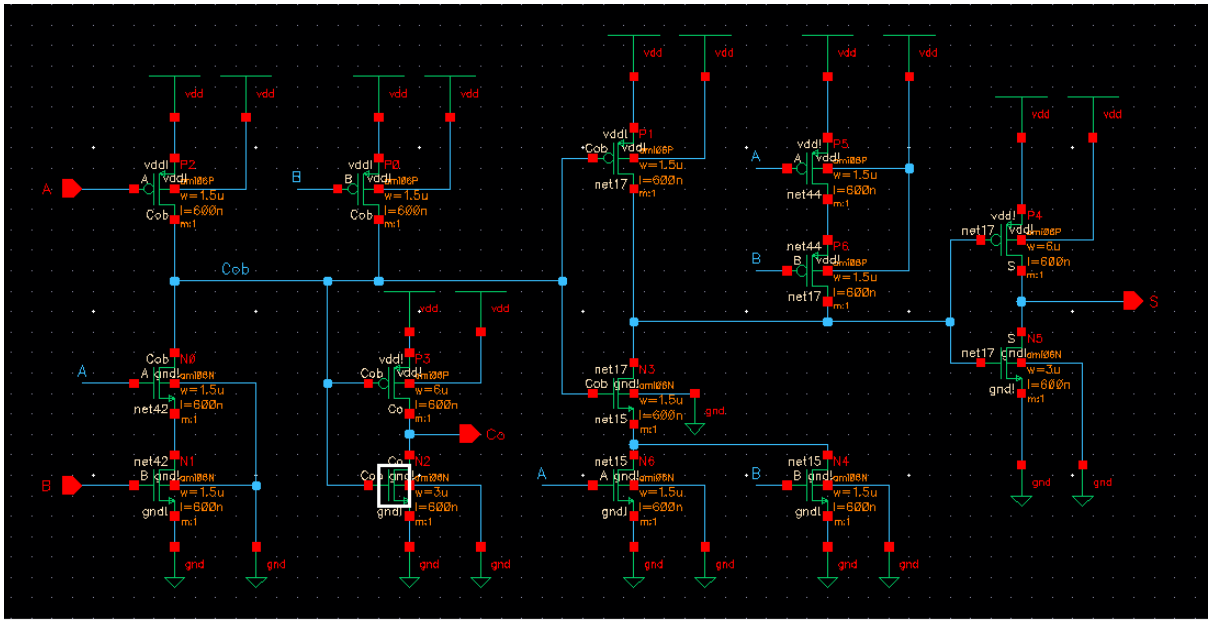
# ECE520: Lab 11 Half Adder, Full adder cell layouts

Name: Arijit Sengupta, ID: 001441748

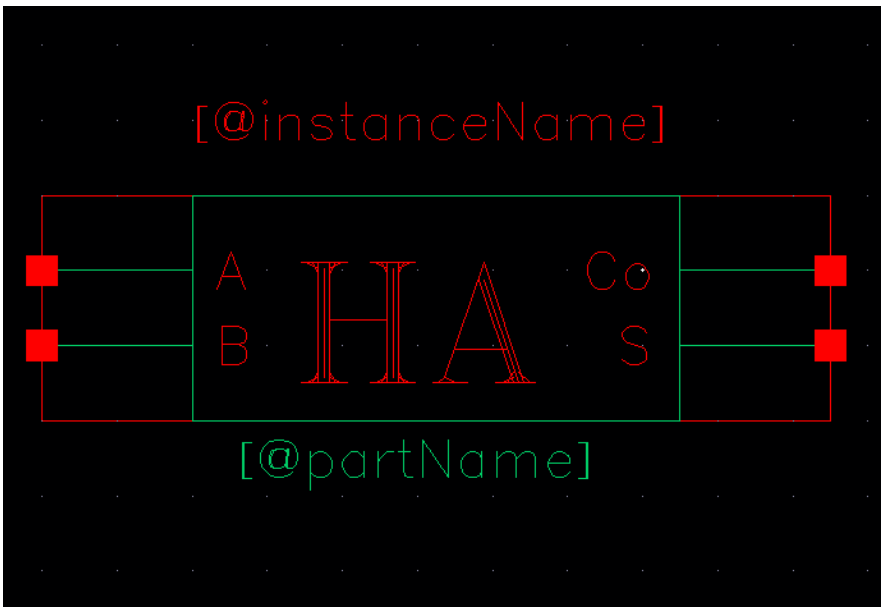
**1) Designing a HAX1 (size 1 Half Adder) cell**




HAX1 Schematic

**HAX1 (internal connections):**
pmos W/L = (1.5µm) / (0.6µm) = 2.5 W/L=2.5. This is a size 1 pmos.
nmos W/L = (1.5µm) / (0.6µm) = 2.5 W/L=2.5. This is a size 1 nmos

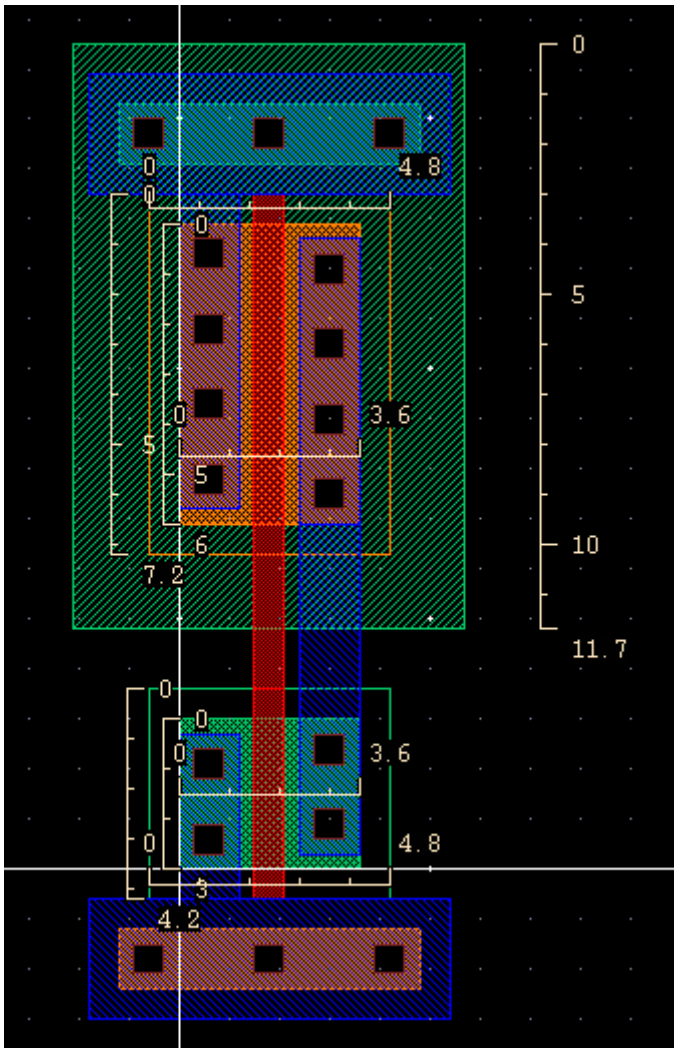
**HAX1 (output-exposed transistors):**
pmos W/L = (6.0µm) / (0.6µm) = 10 W/L=10. This is a size 4 pmos.
nmos W/L = (3.0µm) / (0.6µm) = 5 W/L=5. This is a size 2 nmos.




HAX1 Symbol

Our starting point for HAX1 is INVX2, so we copy the INVX2 layout into HAX1.

We delete all of the pins: A Y vdd! gnd! We will put the pins back in at the end of the design. We also delete the A input pad (metal1, poly, cc combination).



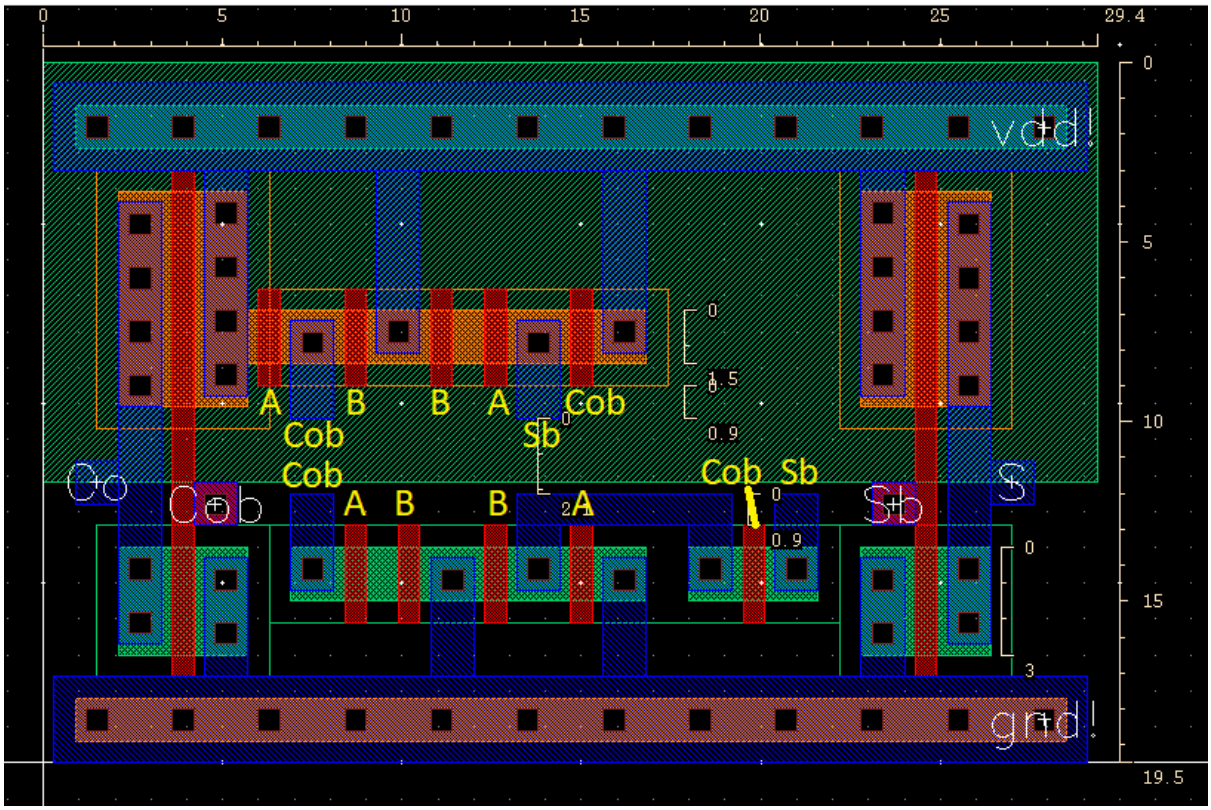

The floor plan is shown below with every transistor placed and every obvious connection made.

We pushed the final (buffered) inverters to the outside of the cell, so they don't interfere with the complicated internal routing. Looking at the inverted outputs (Cob and Sb, which are Co' and S'), it can be seen that they have very few connections to the insides of the cell. This is why pushing them to the sides is a good idea to avoid congestion during routing inside.

S and Co output pins are very easy to place due to the way we positioned the final buffering inverters. We made these output pins using metal1, because they are outside the cell and it is very easy to access them directly using metal1, without resorting to metal2.

We put output pins on Co, S and put inputOutput pins on vdd! and gnd! and run DRC. It gives no errors.

```
DRC started.......Sun Dec  8 15:58:31 2019
    completed ....Sun Dec  8 15:58:31 2019
    CPU TIME = 00:00:00  TOTAL TIME = 00:00:00
*********   Summary of rule violations for cell "HAX1 layout"   *********
   Total errors found: 0
```




Cob metal1 connection on the bottom left and Sb metal1 connection on the bottom right were very easy to make. A, B input pins are a little trickier. Because of the A B B A configuration,we will route the B-B connection as a U shape above the pmos transistors and place a metal1-poly pad there and take the B input right there.

This will not leave enough room to make a U connection for the A-A. However, we can place a metal1-poly pad above the A line. In this way, we have the A and B inputs of this HAX1 cell.

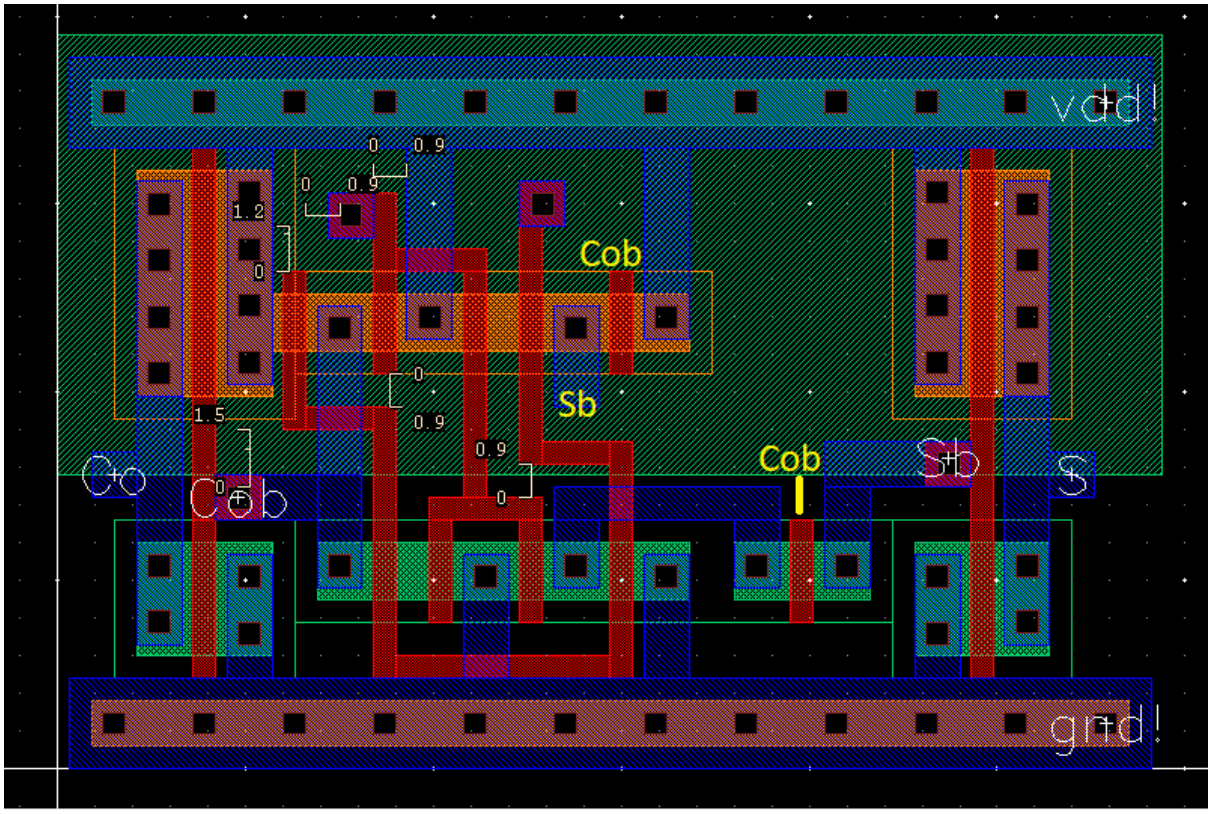
One interesting note about the B pin (metal1-poly) is that, as I marked below, it has to obey three DRC rules:

I) It must be 0.9µm (3λ) away from any other poly (this is the minimum-poly-poly-distance rule)

II) It must be 0.9µm (3λ) away from any metal1 line (this is the minimum-metal1-metal1-distance rule)

III) Any poly line must be 1.5µm (5λ) away from a poly-metal1 contact (this is the minimum-poly-poly contact spacing rule).

All that is left is connecting Sb, Cob, Cob and placing input pins on A, B inputs.




Note about A, B inputs: Currently they are made from a 1.2µm x 1.2µm (4λ x 4λ) poly on top of a 4λ x 4λ metal1 with a small 2λ x 2λ poly-metal1 contact. If we look around it carefully, we will realize that we do not have a metal1 options to draw these pins. Because the surrounding area of these A, B inputs is trapped with metal1, the only way to route into these A, B inputs pins is metal2.

So, we will draw a 4λ x 4λ on top of metal1 and place a 2λ x 2λ via to connect metal1-metal2. In other words, the A, B input pins are being taken out through poly -> cc -> metal1 -> via -> metal2. It is a giant oil drilling rig structure that connects metal2 all the way down to poly.

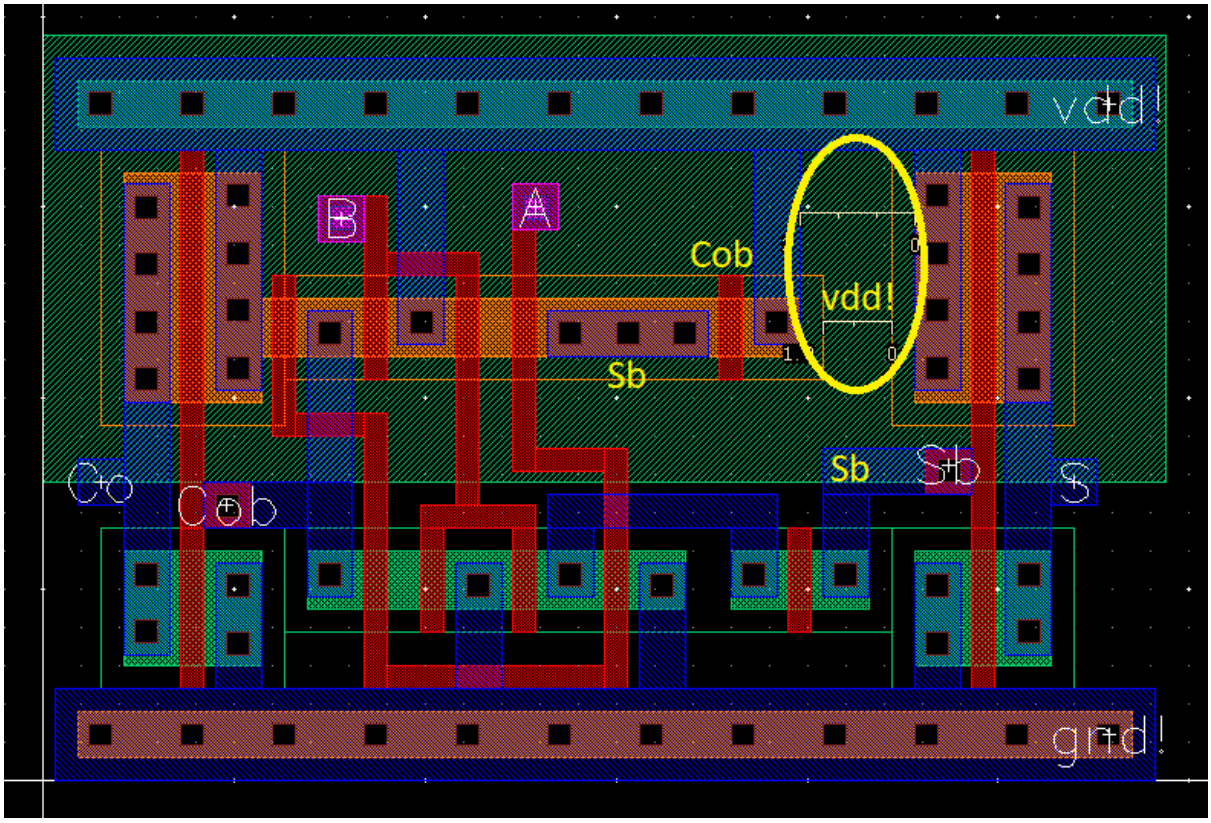
All we have left is the Sb and Cob connections. If we look carefully, we will observe that we need to push the pmos transistor to the right. For this, we will keep everything intact, but will add more pactive contacts in the Drain (D) connection (which is the connection that will eventually connect to Sb through metal1). We can, then, connect the Cob poly lines and run the Sb metal1 connection to the right and connect it to Sb below.

Contact spacing rule is 1.5µm (5λ). So, we can push it 5λ or 10λ. If we push as far as 15λ, the vdd! lines and contacts will get close and we will get DRC errors.

Below, it is shown the scenario when we have pushed the far right pmos to the right by 10λ. Although we created a lot of space there, we still would like to push it more to the right.

Verify -> DRC gives no errors, so we're good.

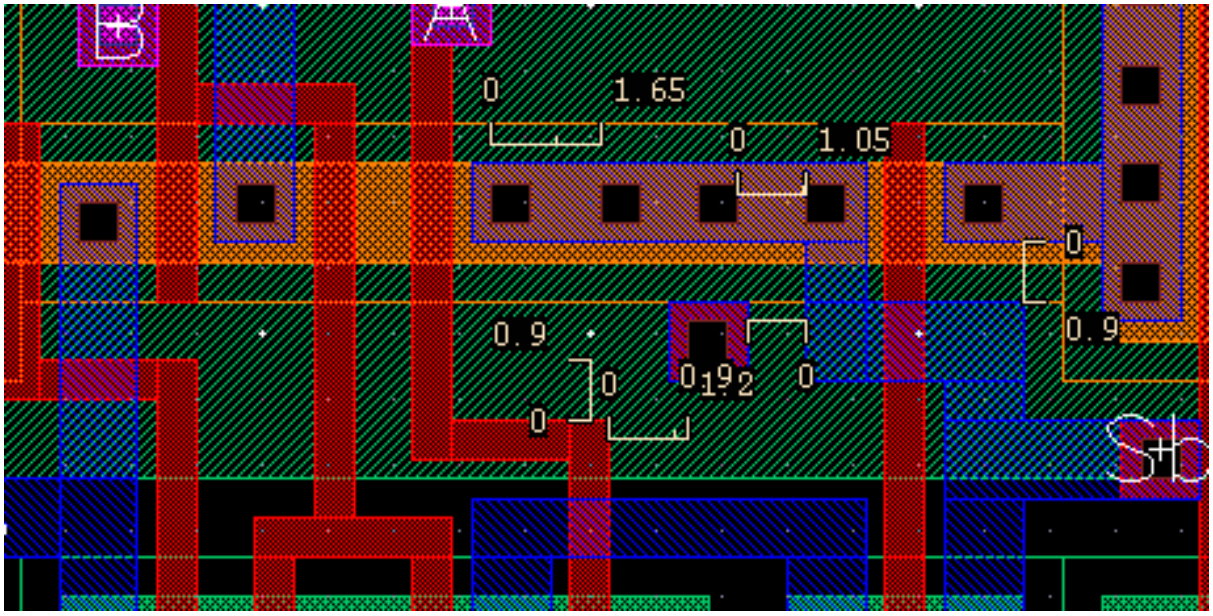
```
DRC started.......Sun Dec  8 16:50:13 2019
    completed ....Sun Dec  8 16:50:13 2019
    CPU TIME = 00:00:00  TOTAL TIME = 00:00:00
*********   Summary of rule violations for cell "HAX1 layout"   *********
   Total errors found: 0
```




It took a little bit of work to do the stretching to the right. First, we pushed the far right pmos a lot to the right, so the vertical Cob poly connection would be straight. This allowed me to put a poly-metal1-cc pad in the middle, because we are perfectly clear from the poly line on the left of it (this is another example of the diagonal being $5\lambda$=1.5$\mu m$), evident from the rulers below.

We had to space the contacts a little farther from each other (they are 3.5λ more apart from each other as compared to 3λ). This creates a weird situation on the right side vdd! connection. We can only put a single contact there but that's fine.

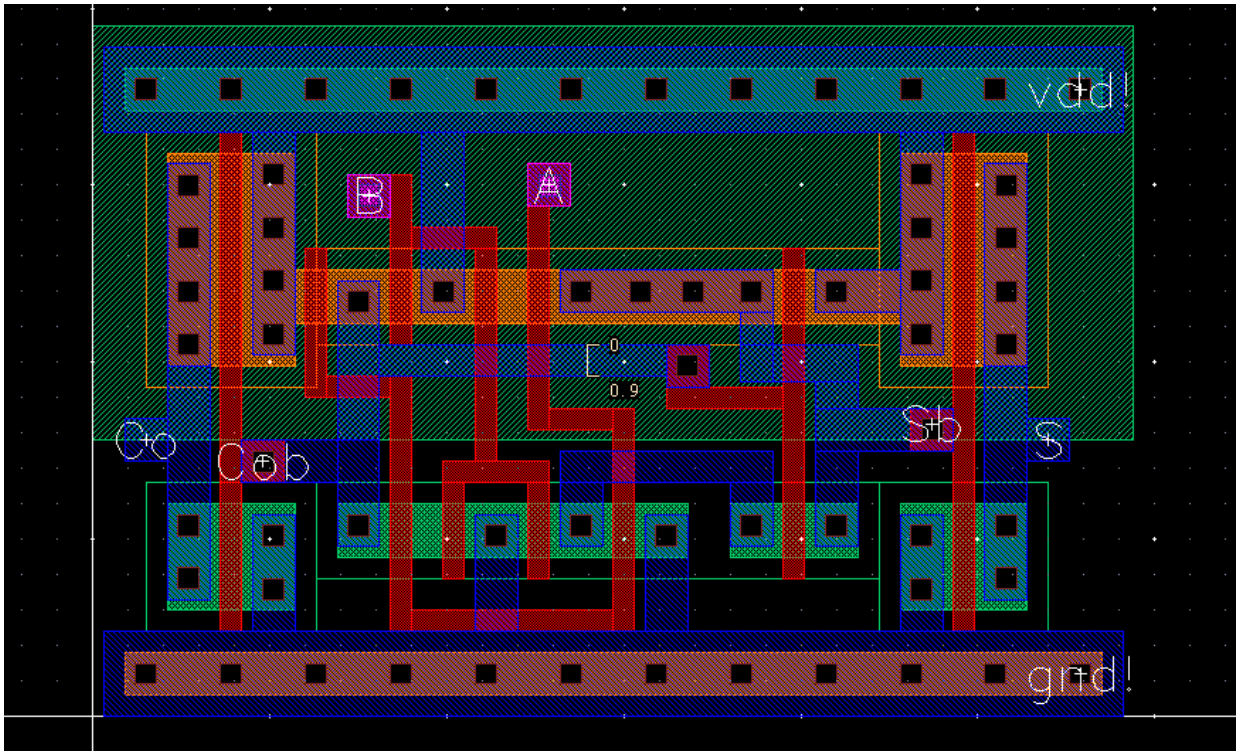
We pushed the 4-contact metal1 up a λ to stay clear of the poly-metal1-cc pad below.



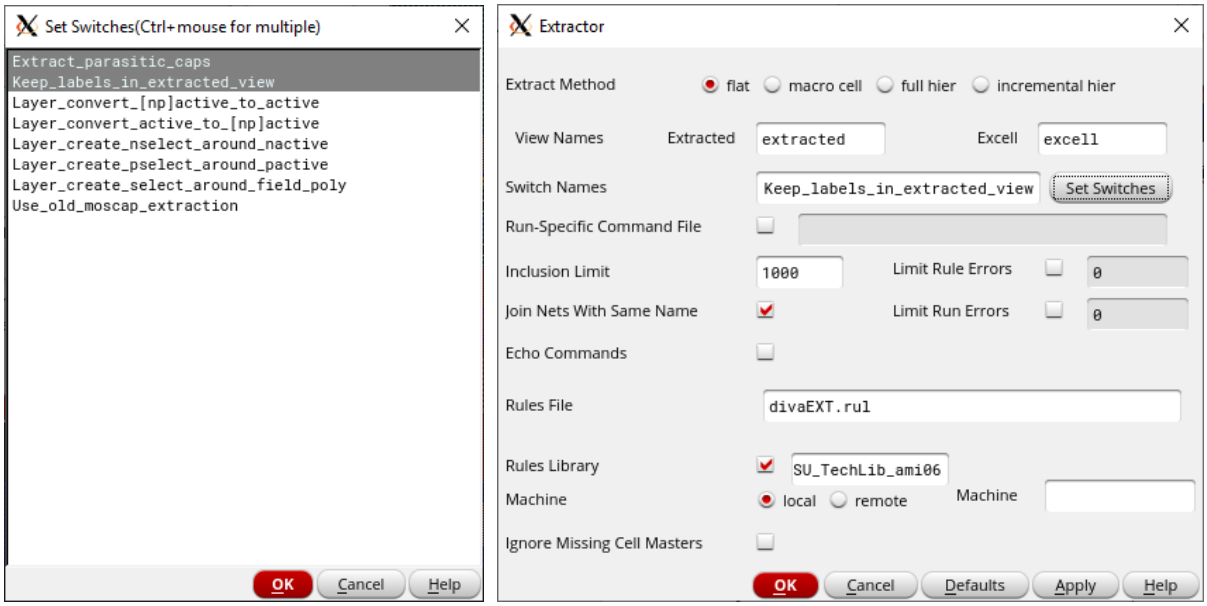
This is the final HAX1 layout. Verify -> DRC. No errors.

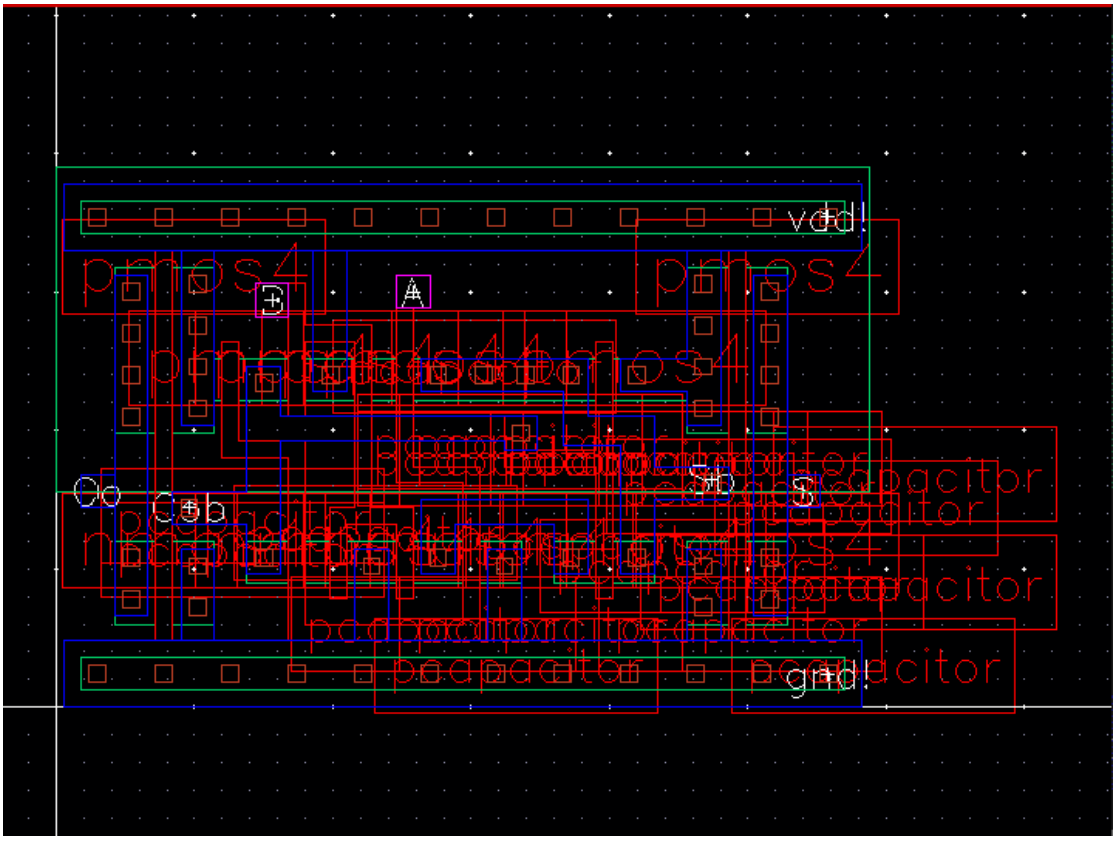
A, B input pins are metal2. S, Co output pins are metal1.

```
DRC started.......Sun Dec  8 17:15:14 2019
    completed ....Sun Dec  8 17:15:14 2019
    CPU TIME = 00:00:00  TOTAL TIME = 00:00:00
*********   Summary of rule violations for cell "HAX1 layout"   *********
   Total errors found: 0
```

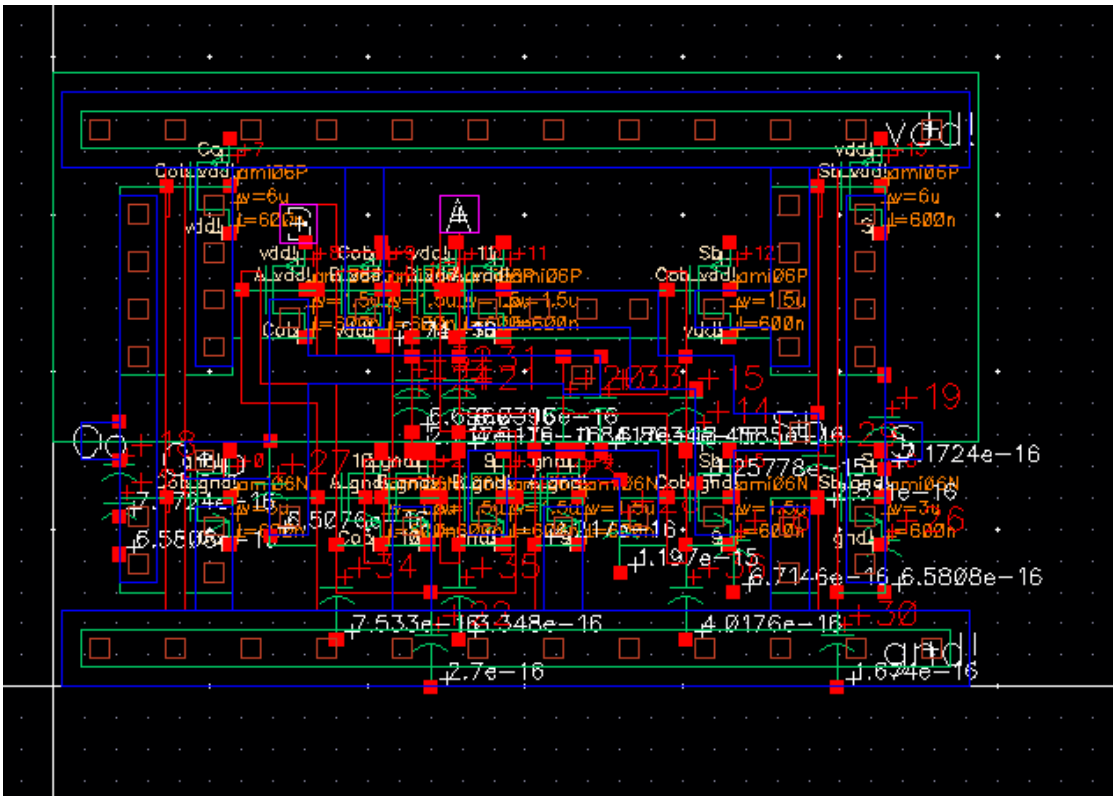



Next, Verify -> Extract. We use switches Extract_parasitic_caps Keep_labels_in_extracted_view.






HAX1 Extracted view (Top Level)



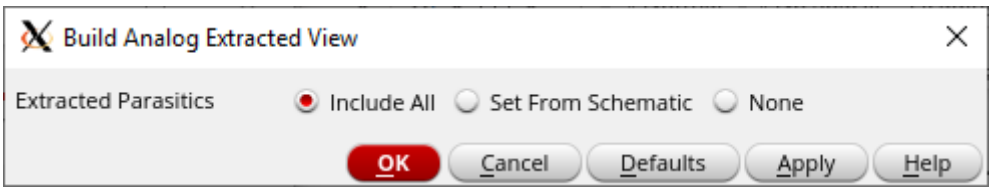

HAX1 Extracted view (Lower Level)

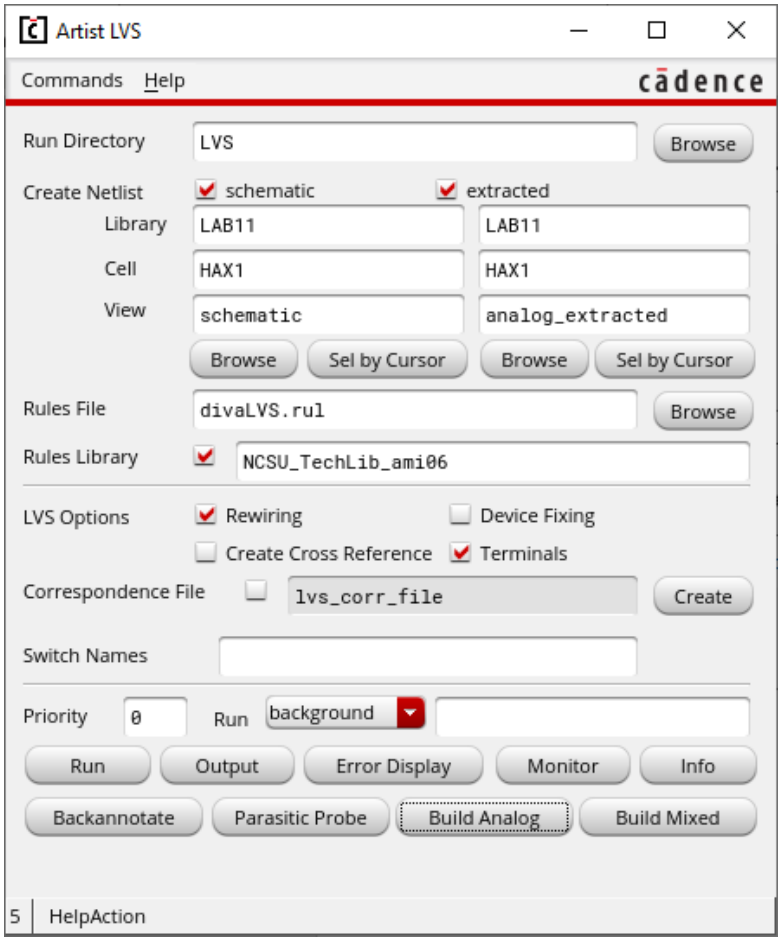
After getting the extracted view, we do Verify -> LVS. We compare HAX1 schematic to HAX1 extracted. Netlists match.

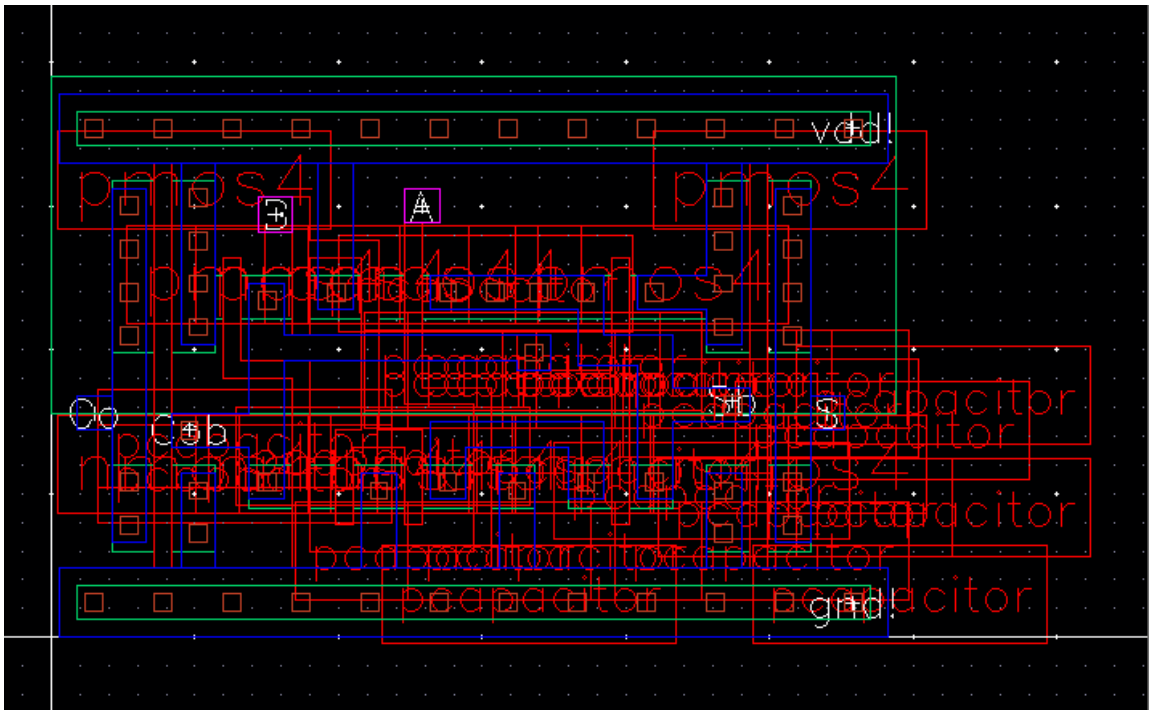


We change the second view to analog_extracted. Build Analog. Include all.

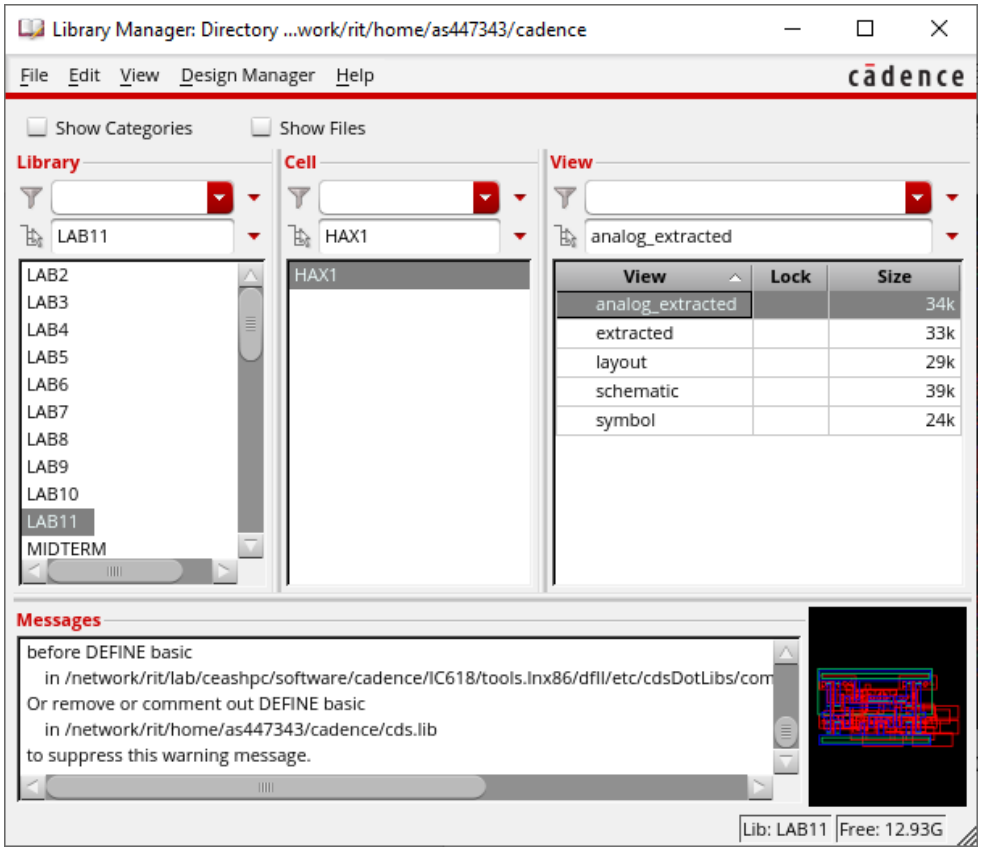
After this last step passes, we now have 5 views for HAX1. We are ready to simulate.






HAX1 Analog Extracted view




**2) Determining the FO4 Delay of HAX1 (size 1 Half Adder) cell**

In our LAB11 library, we create a test bench named **TBhax**. We want to measure the FO4 (fanout of 4) delay of the HAX1 cell using this test bench.

The HAX1 cell has two inputs, so, we will use two vpulse generators to stimulate its inputs, exactly as before. For vpulse generators:

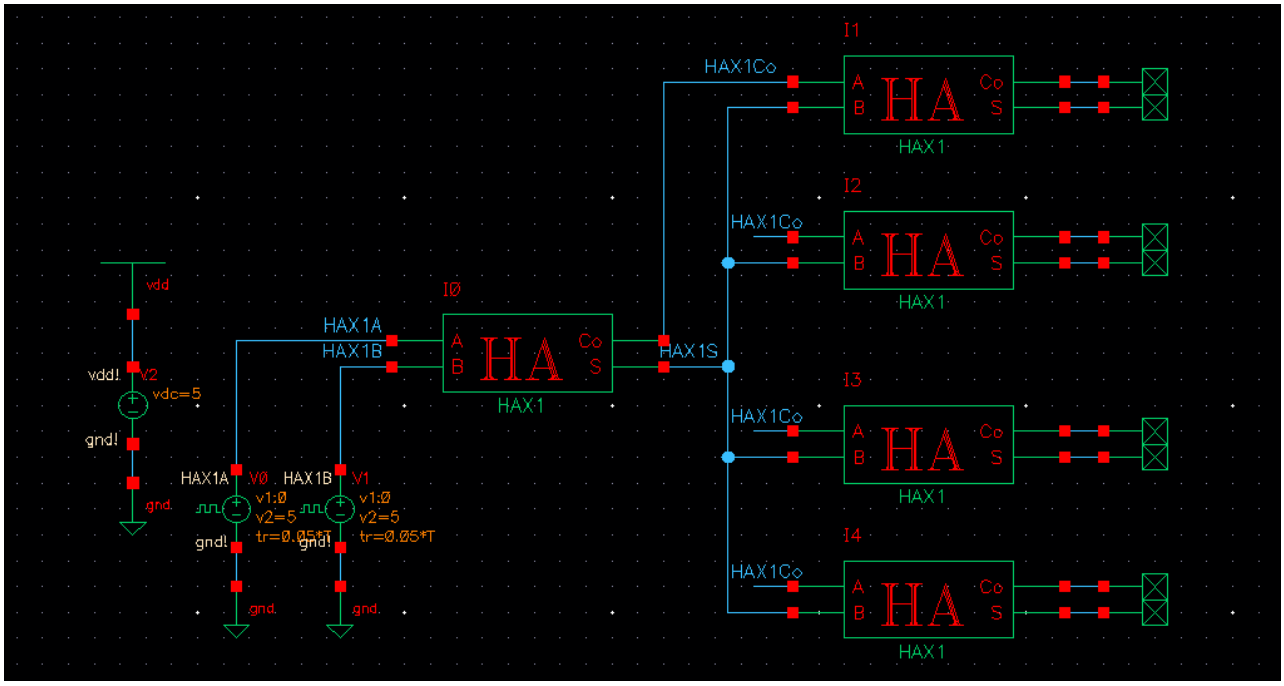
First vpulse: Delay=0 Period=T Rise Time=0.05*T Fall Time=0.05*T Pulse Width=0.45*T

Second vpulse: Delay=0 Period=2*T Rise Time=0.05*T Fall Time=0.05*T Pulse Width=0.95*T

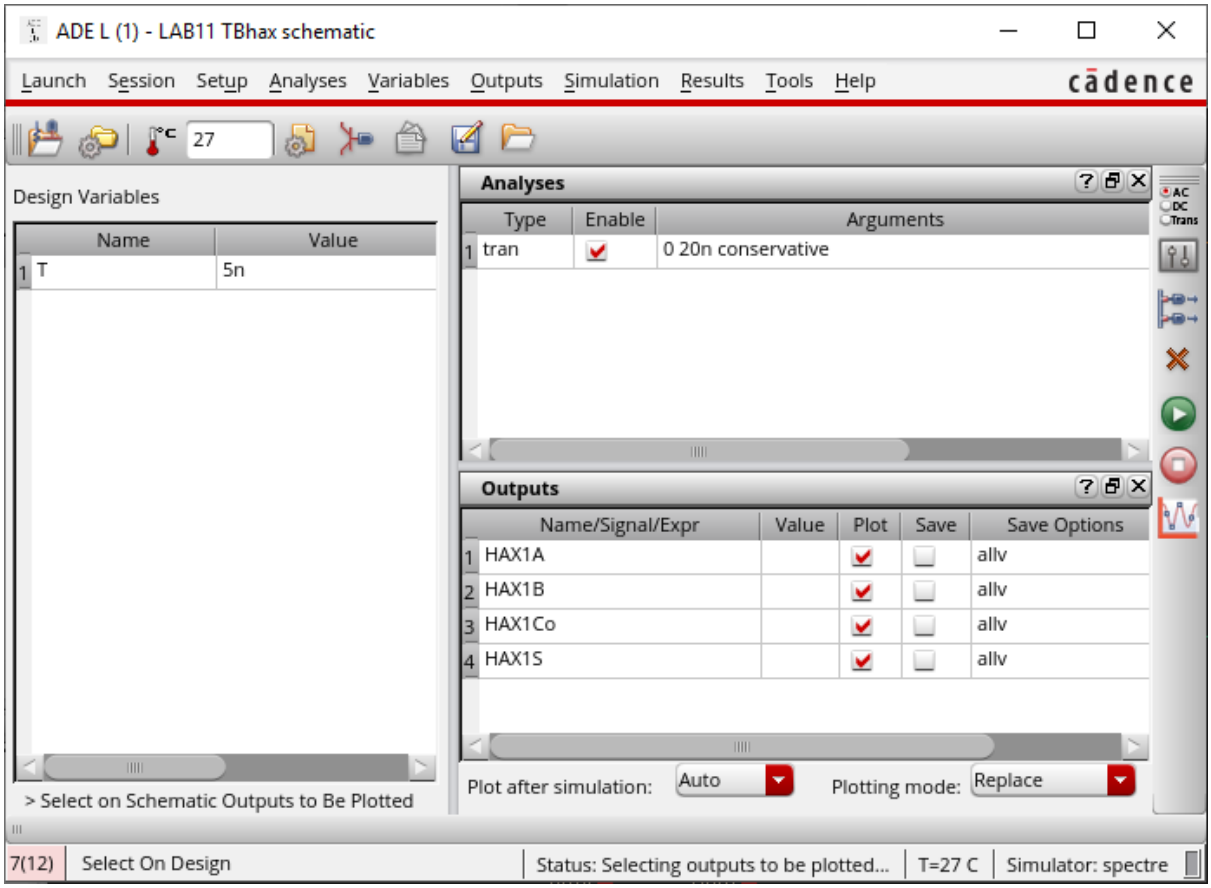
There is no cap device in this test bench. To load both outputs (S, Co) of the HAX1, we will use 4 identical copies of the same exact cell as load (hence the term fanout-of-4, or FO4, which means that the load is 4 identical copies of itself).

Because the outputs of the loads are not connecting to anything, ADE L will think that it is an open circuit. In order to avoid getting an error, we will use an element named noConn (from the basic library), which has this symbol:

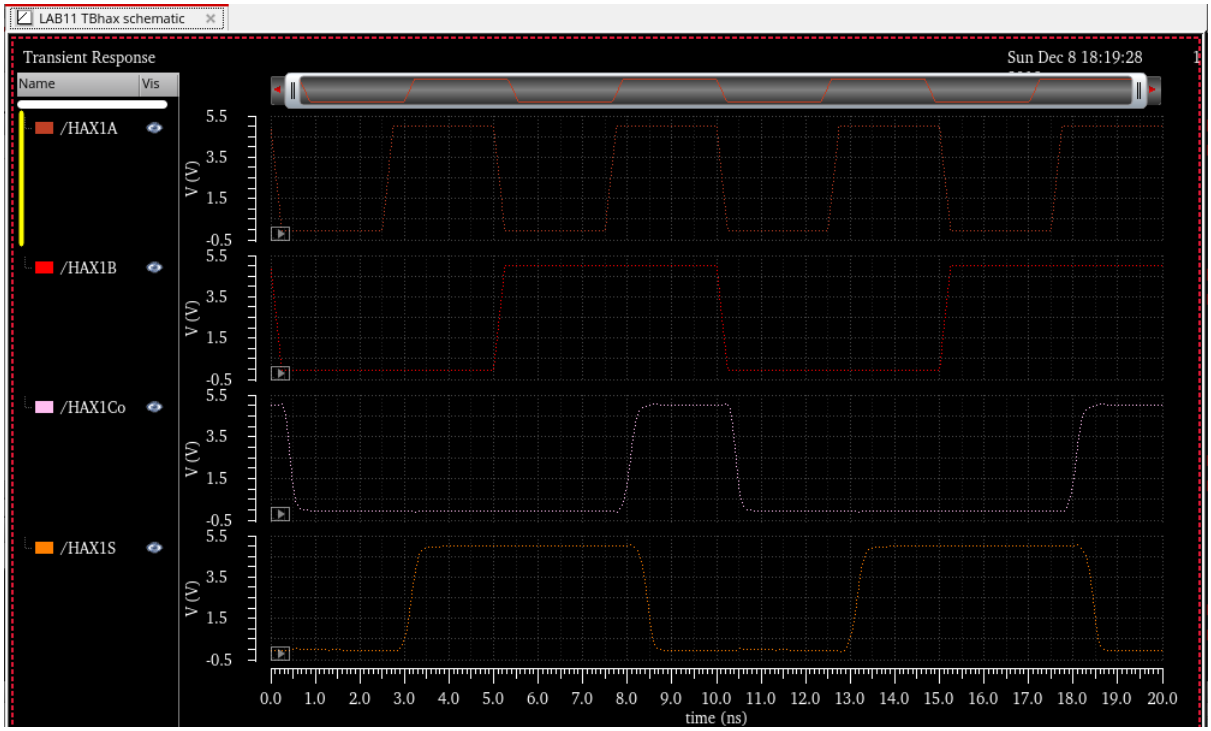
Finished test bench is shown below:




TBhax schematic




TBhax schematic ADE L



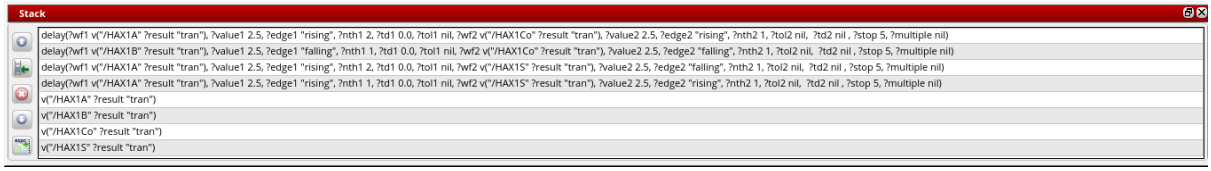

TBhax schematic waveform (200MHz without parasitics)

$t_{pd}$(S|AB:00 -> 10) = 529.6E-12 = **529.6ps**

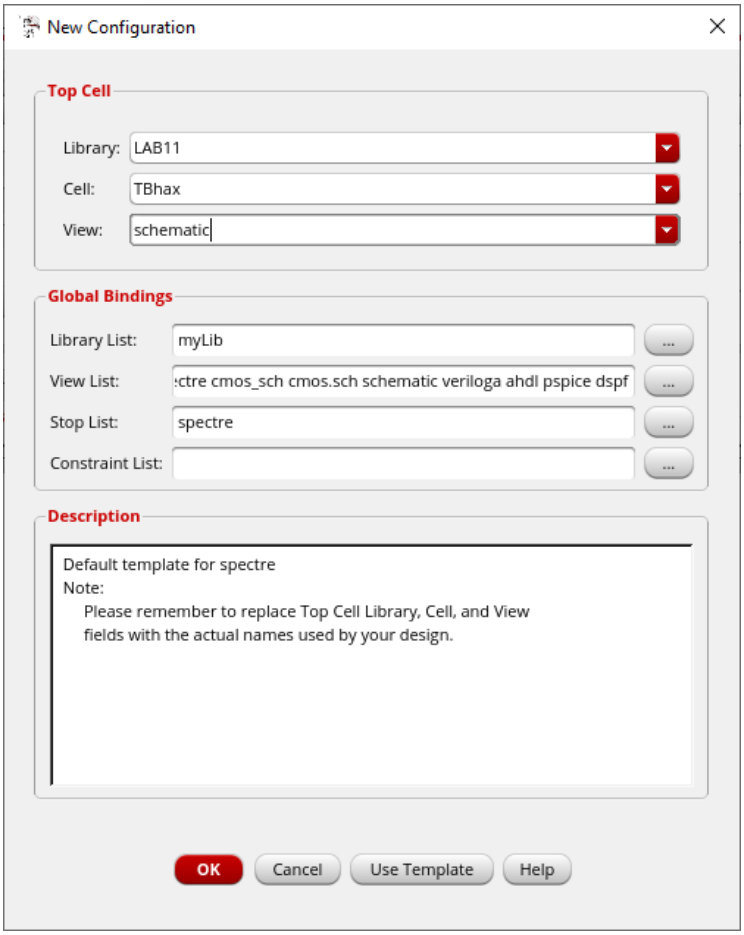
$t_{pd}$(S|AB:01 -> 11) = 821.9E-12 = **821.9ps**

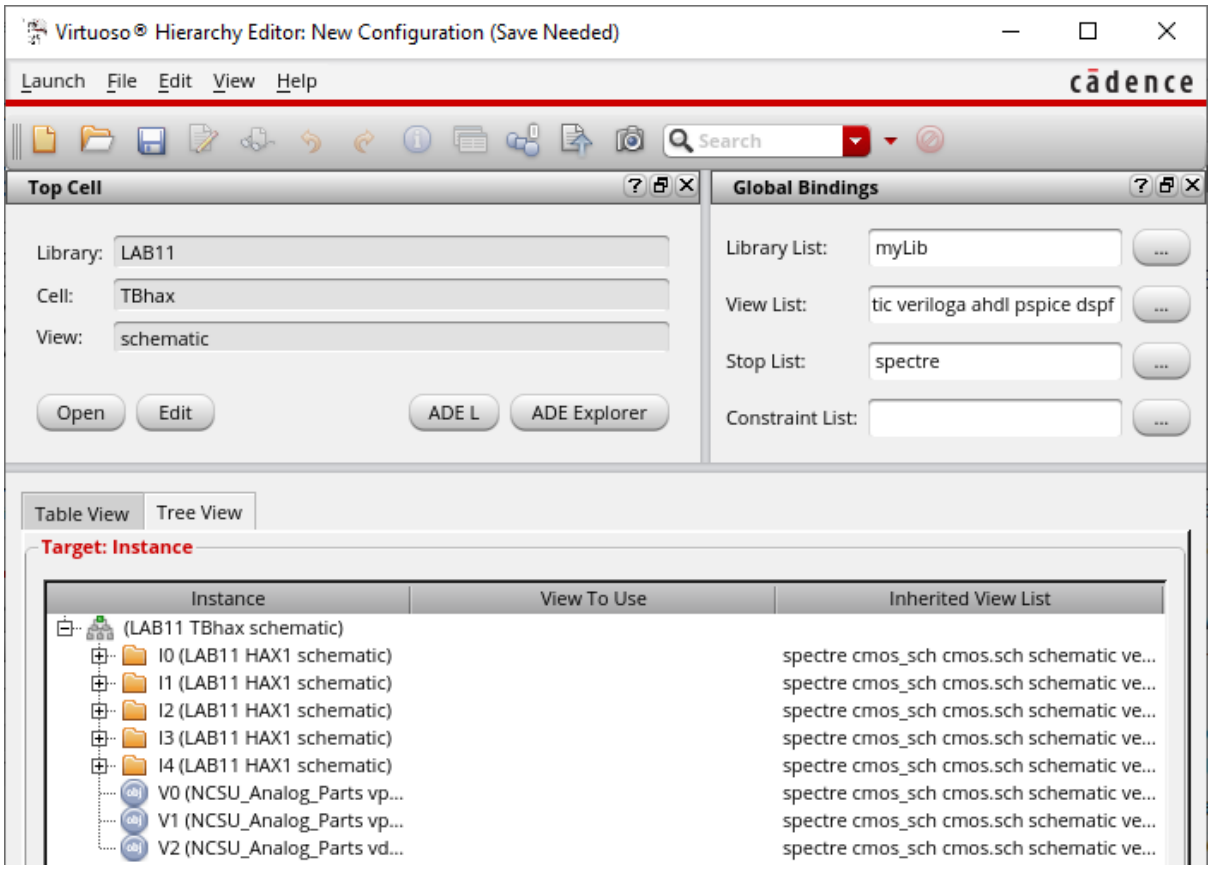
$t_{pd}$(Co|AB:11 -> 00) = 312.4E-12 = **312.4ps**

$t_{pd}$(Co|AB:01 -> 11) = 443.6E-12 = **443.6ps**



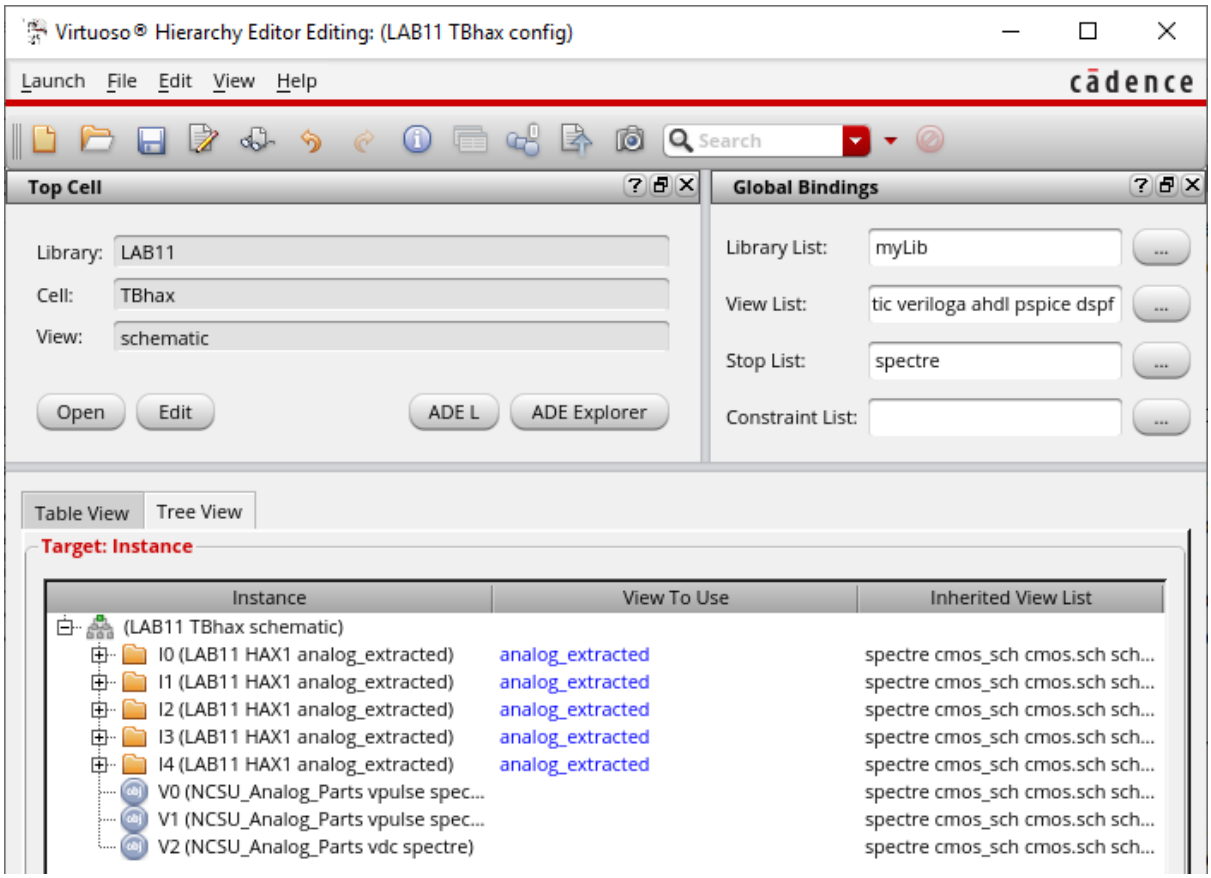

Now, we'll simulate it using parasitics.

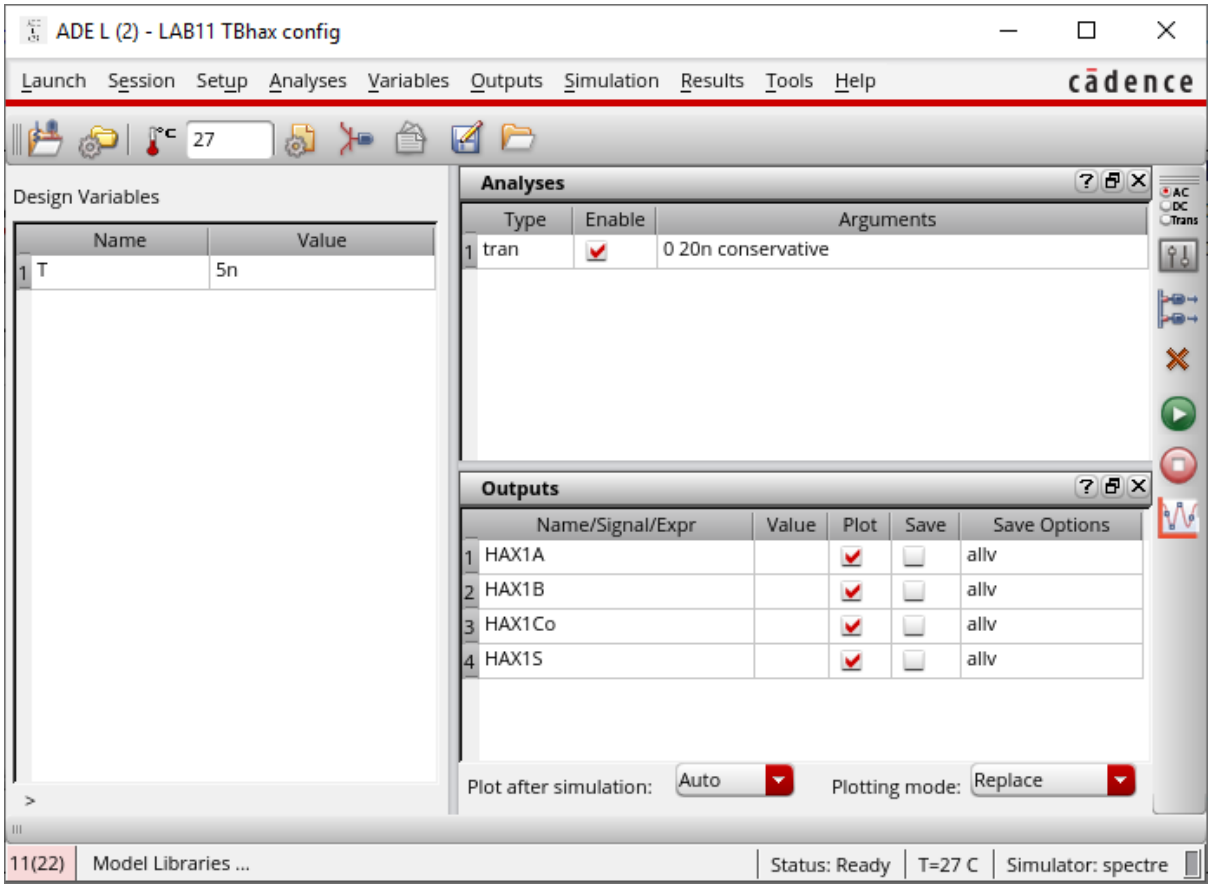

New Configuration
Top Cell
Library: LAB11
Cell: TBhax
View: schematic
Global Bindings
Library List: myLib
View List: ctre cmos_sch cmos.sch schematic veriloga ahdl pspice dspf
Stop List: spectre
Constraint List:
Description
Default template for spectre
Note:
Please remember to replace Top Cell Library, Cell, and View
fields with the actual names used by your design.
OK
Cancel
Use Template
Help


Virtuoso® Hierarchy Editor: New Configuration (Save Needed)
Launch File Edit View Help
cadence
Search
Top Cell
Library: LAB11
Cell: TBhax
View: schematic
Open
Edit
ADE L
ADE Explorer
Global Bindings
Library List: myLib
View List: tic veriloga ahdl pspice dspf
Stop List: spectre
Constraint List:
Table View
Tree View
Target: Instance
Instance
View To Use
Inherited View List
(LAB11 TBhax schematic)
I0 (LAB11 HAX1 schematic) spectre cmos_sch cmos.sch schematic ve...
I1 (LAB11 HAX1 schematic) spectre cmos_sch cmos.sch schematic ve...
I2 (LAB11 HAX1 schematic) spectre cmos_sch cmos.sch schematic ve...
I3 (LAB11 HAX1 schematic) spectre cmos_sch cmos.sch schematic ve...
I4 (LAB11 HAX1 schematic) spectre cmos_sch cmos.sch schematic ve...
V0 (NCSU_Analog_Parts vp... spectre cmos_sch cmos.sch schematic ve...
V1 (NCSU_Analog_Parts vp... spectre cmos_sch cmos.sch schematic ve...
V2 (NCSU_Analog_Parts vd... spectre cmos_sch cmos.sch schematic ve...

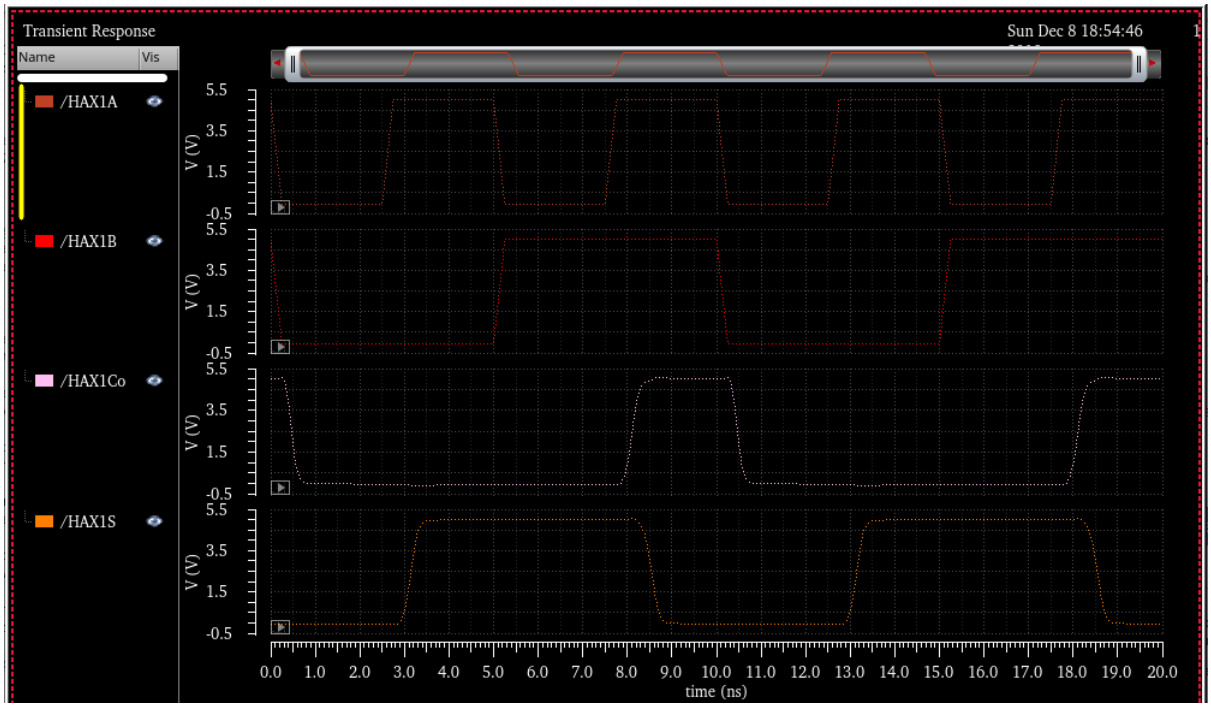
There are 5 instances of the HAX1 cell you just created a layout for. We choose Instance View = analog_extracted for every single one of them.

This means that not only our device under test (DUT) will include parasitics, but also the other 4 that are being used as a load for the DUT.






TBhax config ADE L




TBhax config waveform (200MHz with parasitics)

$t_{pd}$(S|AB:00 -> 10) = 519.8E-12 = **519.8ps**

$t_{pd}$(S|AB:01 -> 11) = 921.8E-12 = **921.8ps**

$t_{pd}$(Co|AB:11 -> 00) = 348.0E-12 = **348ps**

$t_{pd}$(Co|AB:01 -> 11) = 483.8E-12 = **483.8ps**

Here is the comparison between the schematic-only and layout-parasitics based simulation:

| **FO4 Delay** | **HAX1 schematic-only** | **HAX1 using analog_extracted view (layout parasitics included)** |
|---|---|---|
| **$t_{pd}$(S\|AB:00 -> 10)** | 529.6ps | 519.8ps |
| **$t_{pd}$(S\|AB:01 -> 11)** | 821.9ps | 921.8ps |
| **$t_{pd}$(Co\|AB:11 -> 00)** | 312.4ps | 348ps |
| **$t_{pd}$(Co\|AB:01 -> 11)** | 443.6ps | 483.8ps |
| **S output** | **821.9ps** | **921.8ps** |
| **Co output** | **443.6ps** | **483.8ps** |
| **HAX1 cell** | **821.9ps** | **921.8ps** |

This is a more realistic scenario, where the incorporation of the parasitics results in a worse performance.

The FO4 delay from input to Co is 483.8ps.

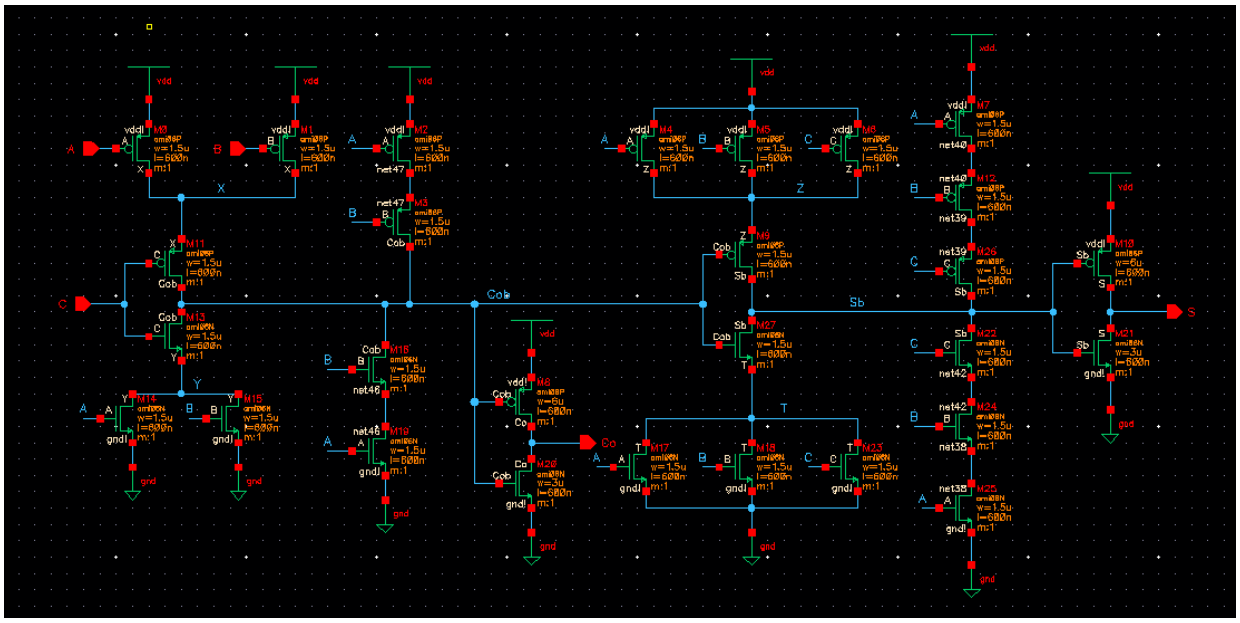
The FO4 delay from input to S is 921.8ps.

The FO4 delay of the HAX1 cell is 921.8ps (worst of the two). This is the worst-case time for the change at the inputs to reach any one of the outputs.

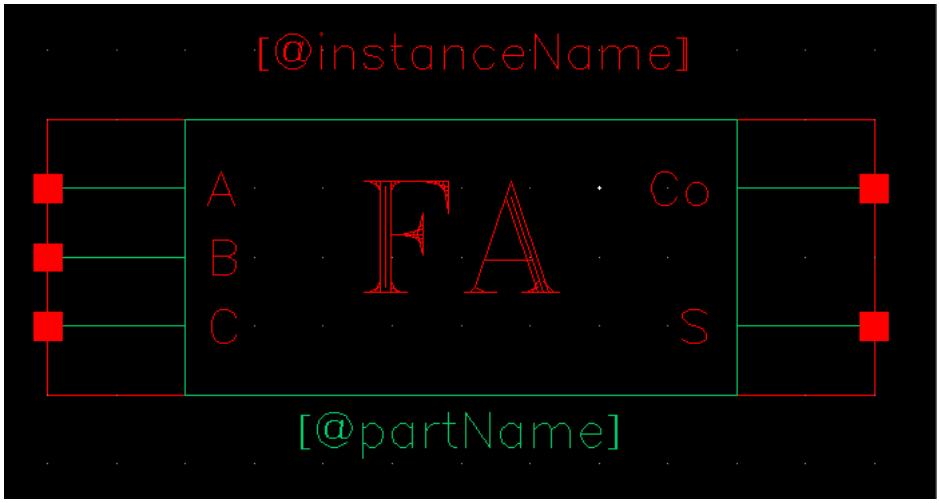
**3) Designing a FAX1 (size 1 Full Adder) cell**

We first draw the schematic for the FAX1 cell. Create a FAX1 cell schematic in your LAB11 library. We will use nmos and pmos this time. They are good enough. We don't want to clutter up the schematic.

The "hidden" nodes X, Y, Z, T, which are just internal connections. Cob and Sb are also internal, but they are different.



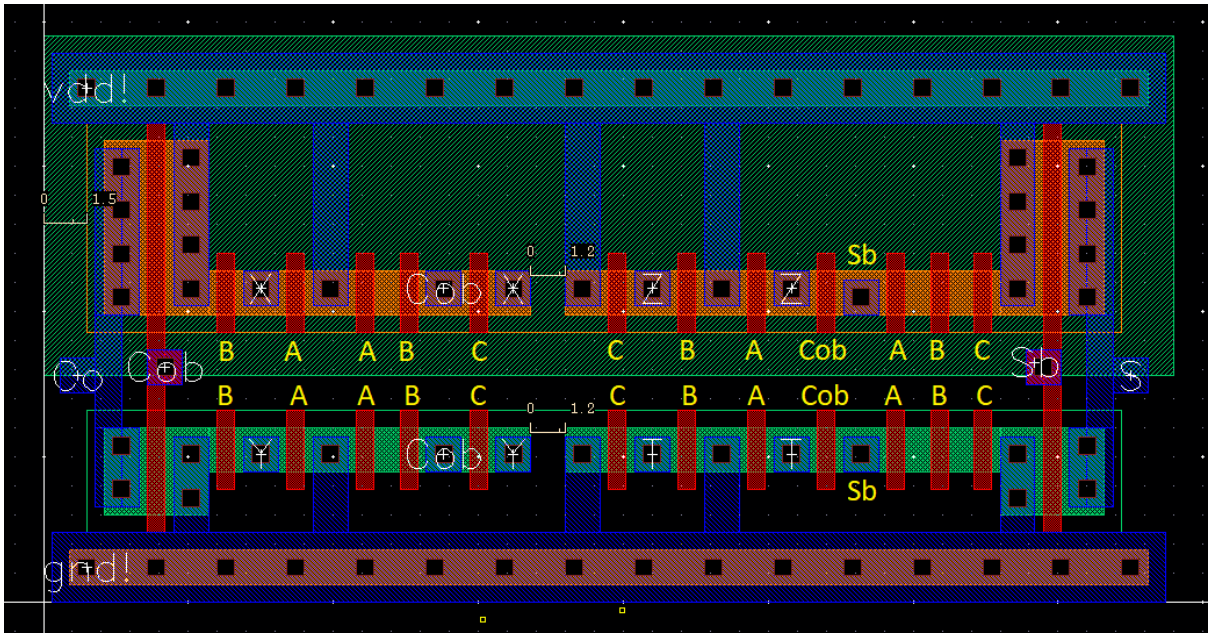
FAX1 Schematic




FAX1 Symbol

When in the layout editor, delete the A, B pins and labels. However, keep the S, Co pins and labels. Additionally, keep the poly-metal1-cc pads you placed on Cob, Sb.

We move the gnd! and vdd! pins to the left edge of the cell. Since we always extend the cell to the right, placing the vdd! and gnd! pins on the left will eliminate the need to delete them and remember them and re-create them.




We run DRC and get no errors!

```
DRC started.......Sun Dec  8 21:03:33 2019
    completed ....Sun Dec  8 21:03:33 2019
    CPU TIME = 00:00:00  TOTAL TIME = 00:00:00
*********   Summary of rule violations for cell "FAX1 layout"   *********
   Total errors found: 0
```

The beauty of this schematic is that the A, B, C input connections and many other connections are ridiculously symmetric. So, the only thing that makes sense is to run them vertically.

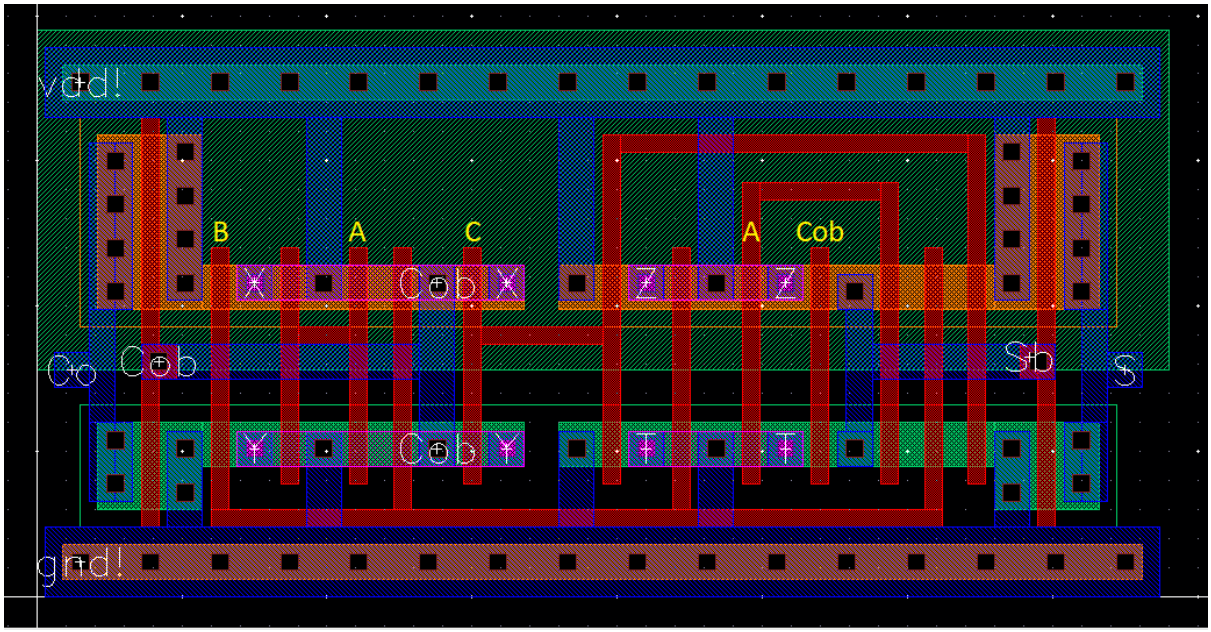
We had to connect the B inputs through the area below the nmos transistors.

The hidden internal nodes that are labelled as X, Y, Z, T in the schematic (and layout) do not connect to anything else other than each other. So, they are a perfect candidate to connect them using metal2. To connect X-X, I put a 1.2µm x 1.2µm (4λ x 4λ) metal2 on top of the metal1 and connected metal1-metal2 using a via. I ran horizontal metal2 lines to connect them. We did the same thing for Y-Y, Z-Z, and T-T connections.

We also connected the two Cob metal1's vertically and ran it to the left, where there is already a metal1 connection for Cob. We also connected two Sb lines vertically using metal1 and ran it to the metal1 Sb connection on the right.

All that is left is placing A, B, C input pads, which we know must be made from metal2. Also, the Cob poly line must be connected.

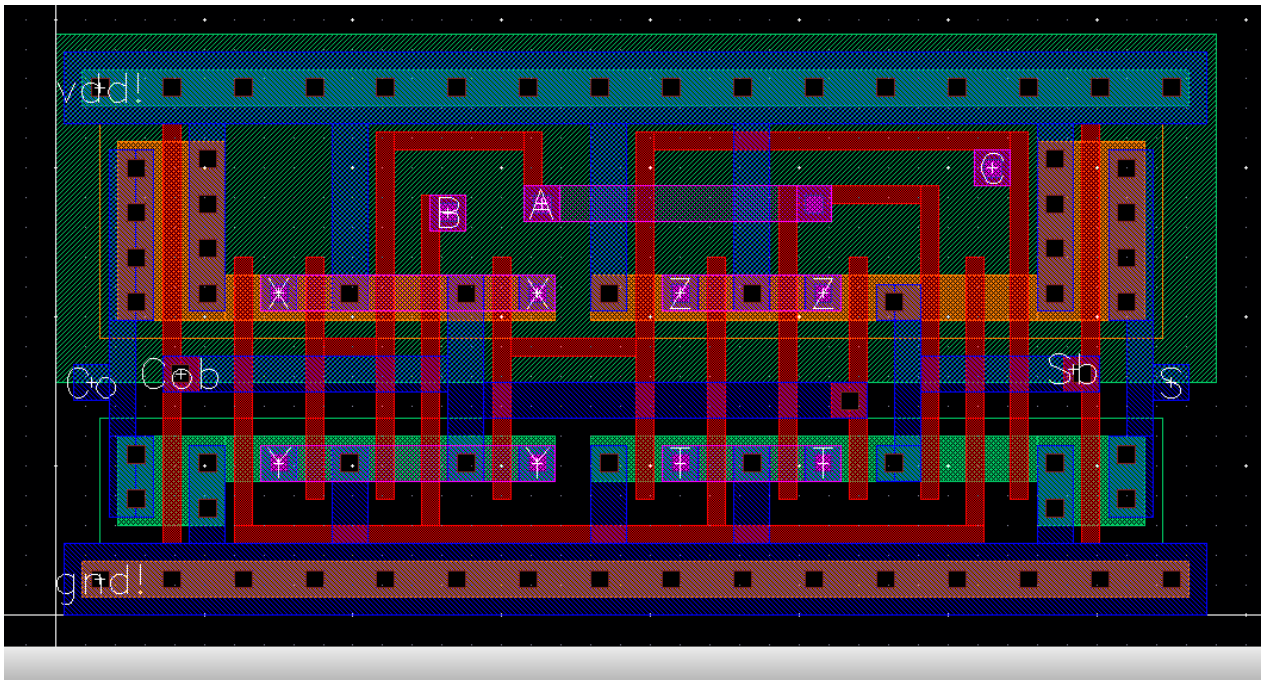
By connecting the X-X, Y-Y, Z-Z, T-T, and all Cob and Sb lines, we run DRC again and get no errors.




Now, all that is left is the input pads. The poly lines are real badly resistive, so, if you can keep them short, that's great. However, if they must be long like the ones shown above, one possible way to reduce the resistance is to place the pads in the middle of the inputs.

There is also the fact that the A pole lines have two separate segments, which must be connected.

The final layout is shown below. A, B, C input pins must be taken using metal2 and S, Co pins are metal1 the way they are now. However, with a via and metal2, they can also be taken using metal2.

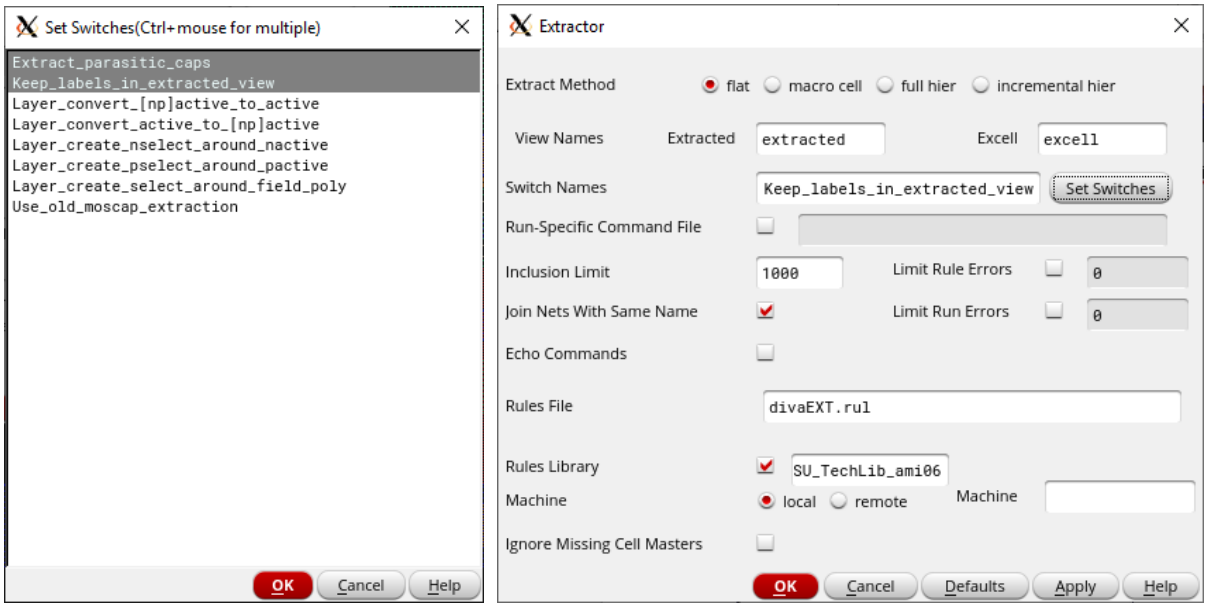


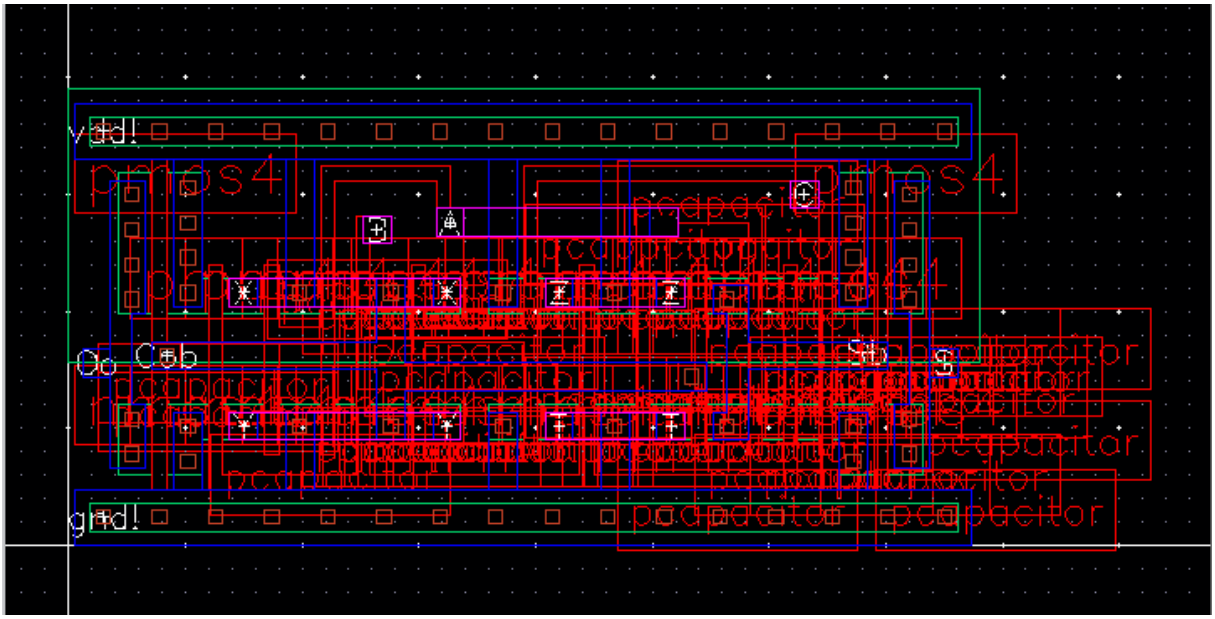

FAX1 Layout

We run DRC and get no errors.

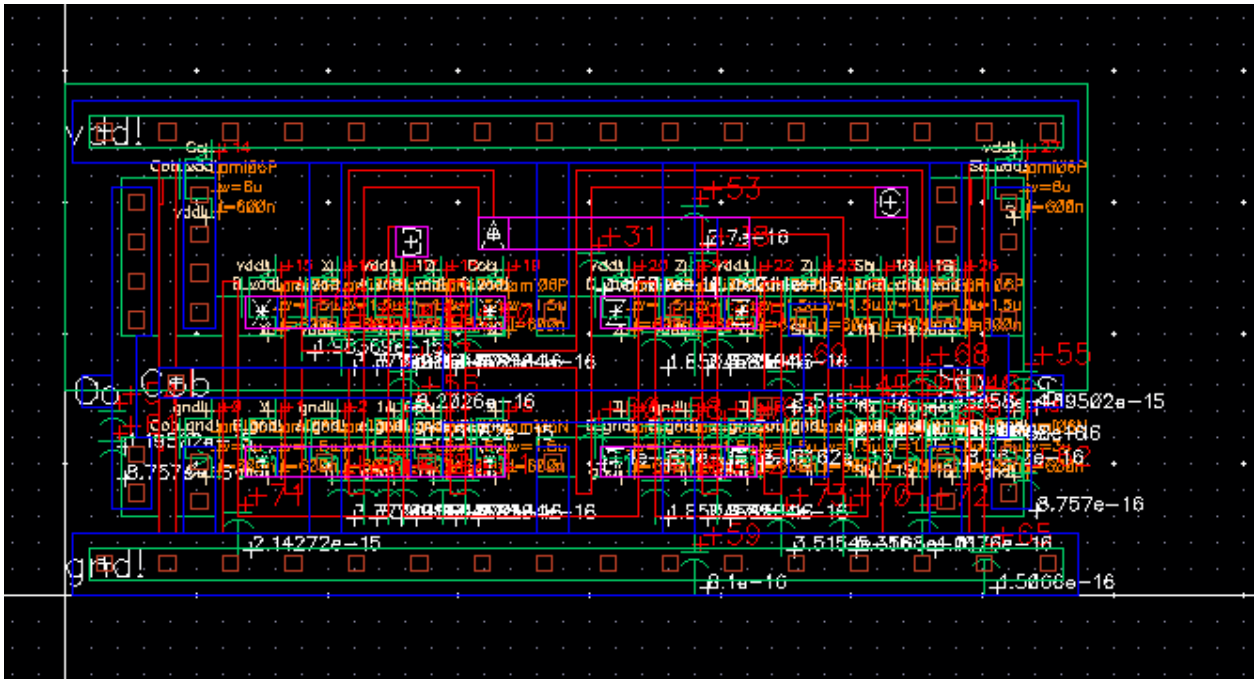
```
DRC started.......Sun Dec  8 22:21:26 2019
    completed ....Sun Dec  8 22:21:26 2019
    CPU TIME = 00:00:00  TOTAL TIME = 00:00:00
*********   Summary of rule violations for cell "FAX1 layout"   *********
   Total errors found: 0
```

Next, Verify -> Extract. We use switches Extract_parasitic_caps Keep_labels_in_extracted_view.



FAX1 Extracted view (Top Level)

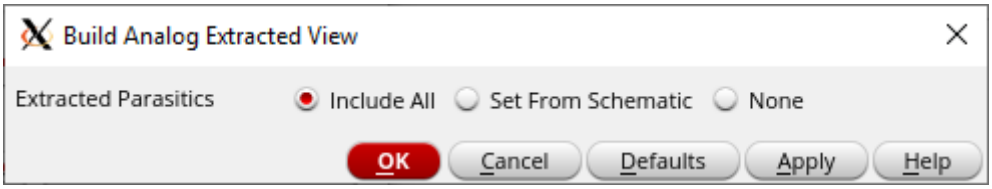


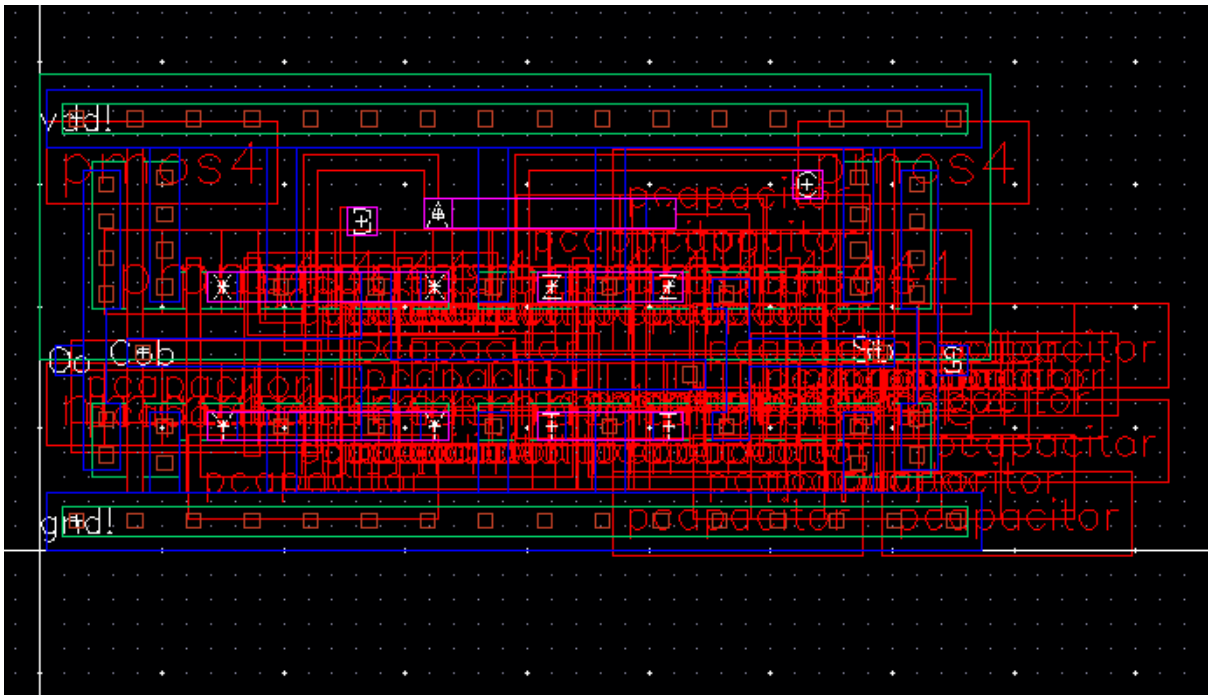

FAX1 Extracted view (Lower Level)

After getting the extracted view, we do Verify -> LVS. We compare FAX1 schematic to FAX1 extracted. Netlists match.

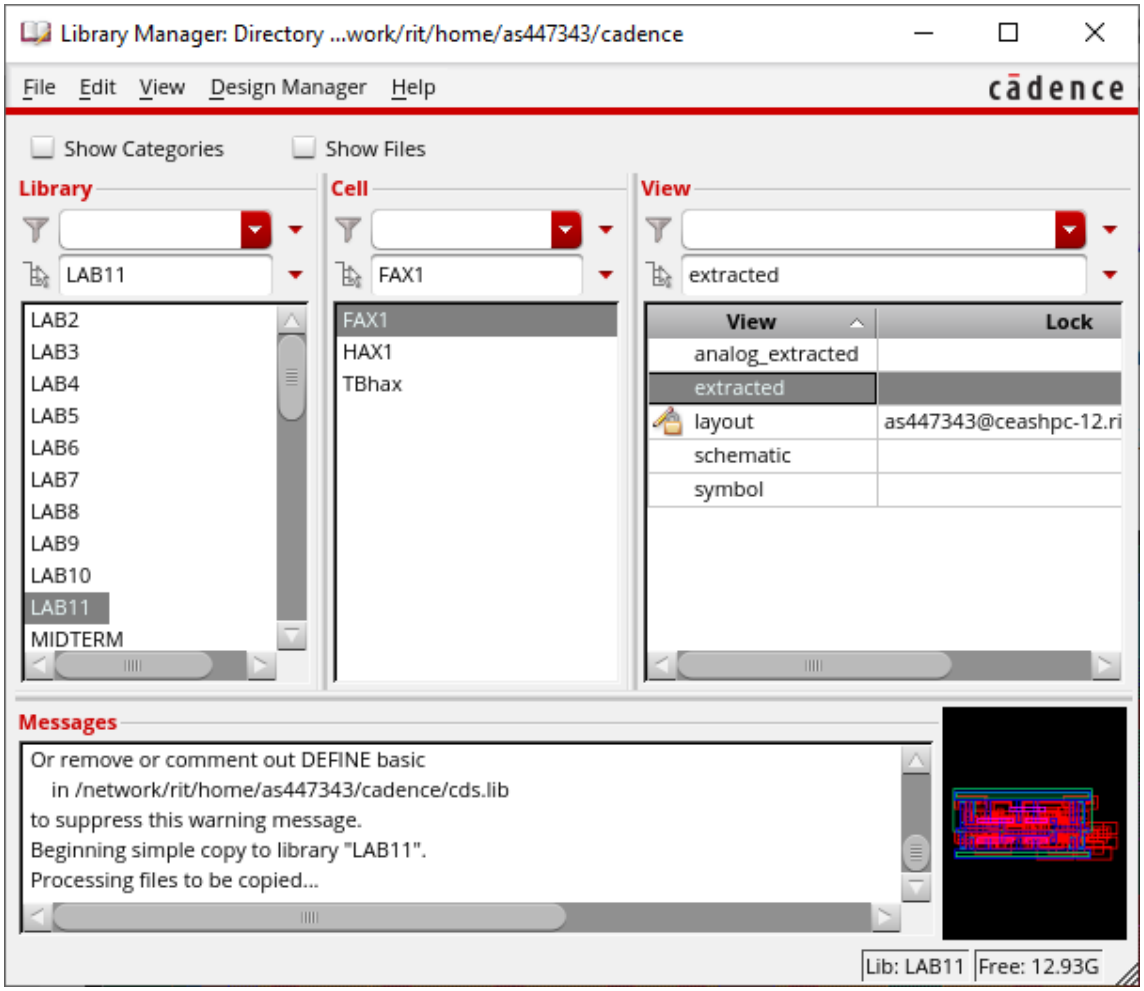

We change the second view to analog_extracted. Build Analog. Include all.

After this last step passes, we now have 5 views for FAX1. We are ready to simulate.




FAX1 Analog Extracted view




**4) Determining the FO4 Delay of FAX1 (size 1 Full Adder) cell**

We create a test bench named TBfax to measure the FO4 (fanout of 4) delay of the FAX1 cell using this test bench. So, the load to this cell will be 4 identical copies of itself. Because the cell has 3 inputs and 2 outputs, we will connect one of the inputs (B input) to logic 1 (vdd!) and will use the other two inputs as load to the 4 outputs.

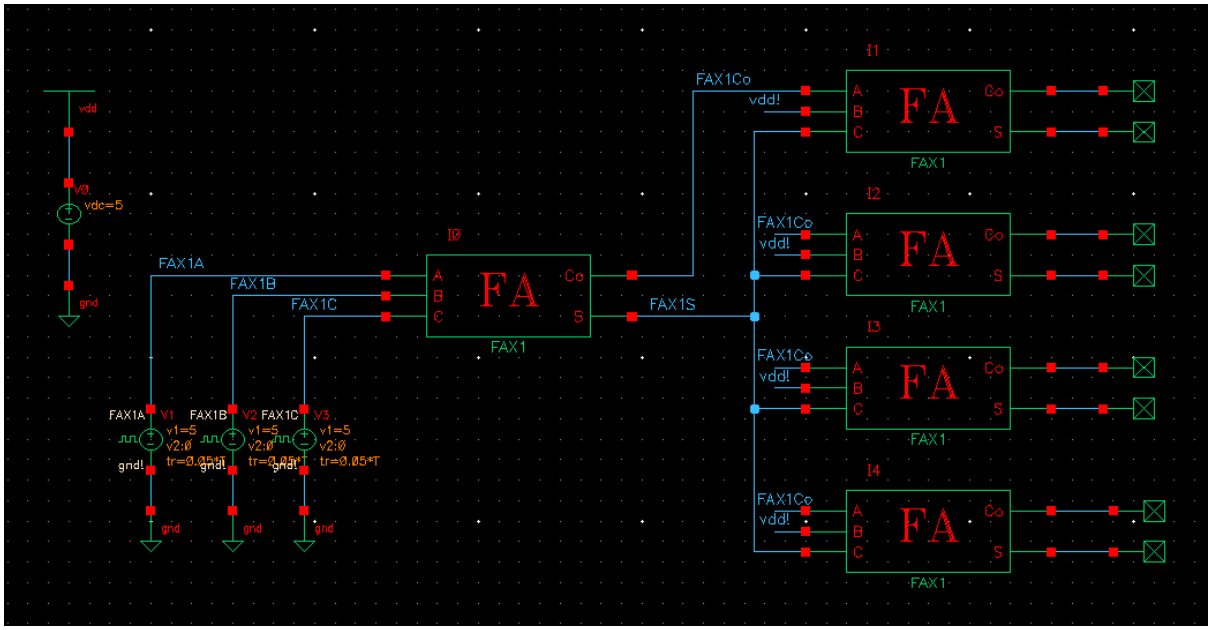
The FAX1 cell has two inputs, so, we will use three vpulse generators to stimulate its inputs, exactly as before. For vpulse generators:

First vpulse: Delay=0 Period=T Rise Time=0.05*T Fall Time=0.05*T Pulse Width=0.45*T

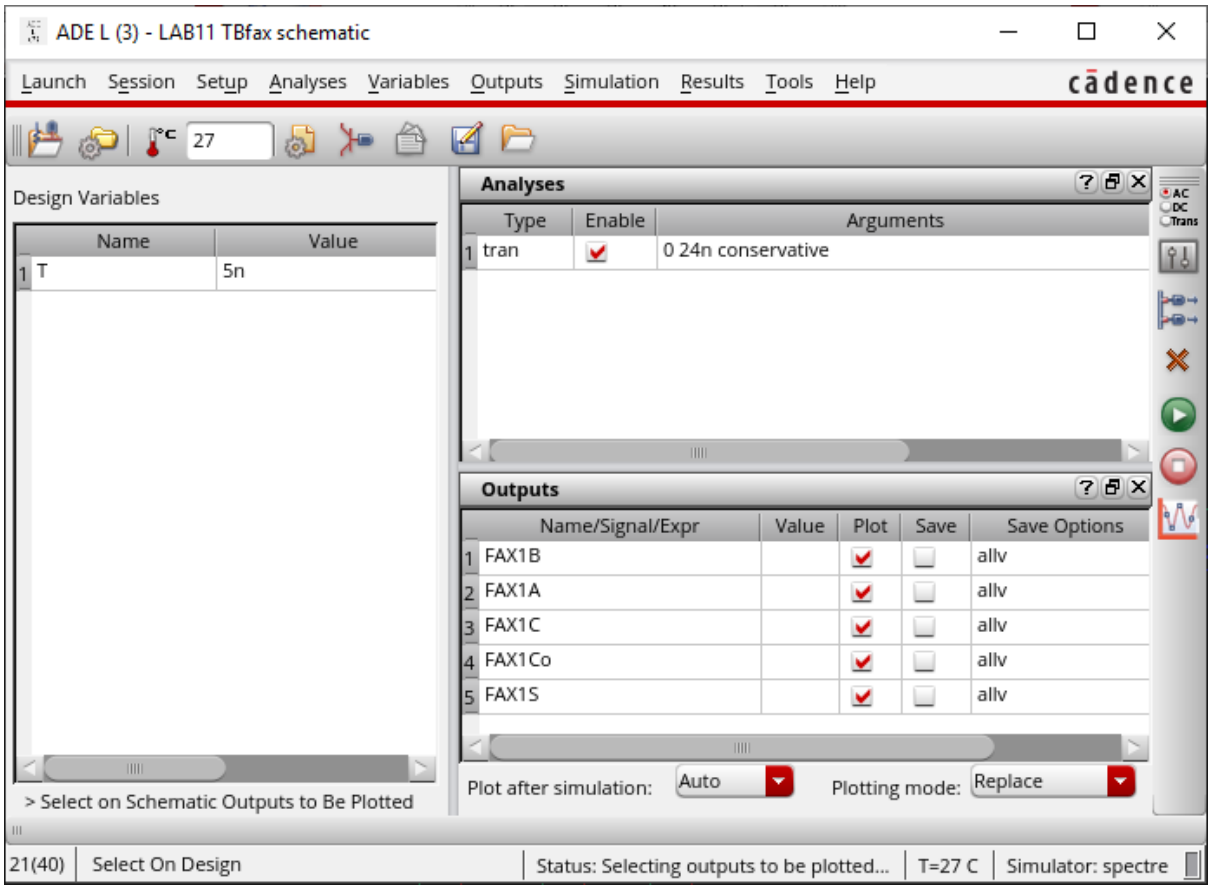
Second vpulse: Delay=0 Period=2*T Rise Time=0.05*T Fall Time=0.05*T Pulse Width=0.95*T

Third vpulse: Delay=0 Period=4*T Rise Time=0.05*T Fall Time=0.05*T Pulse Width=1.95*T

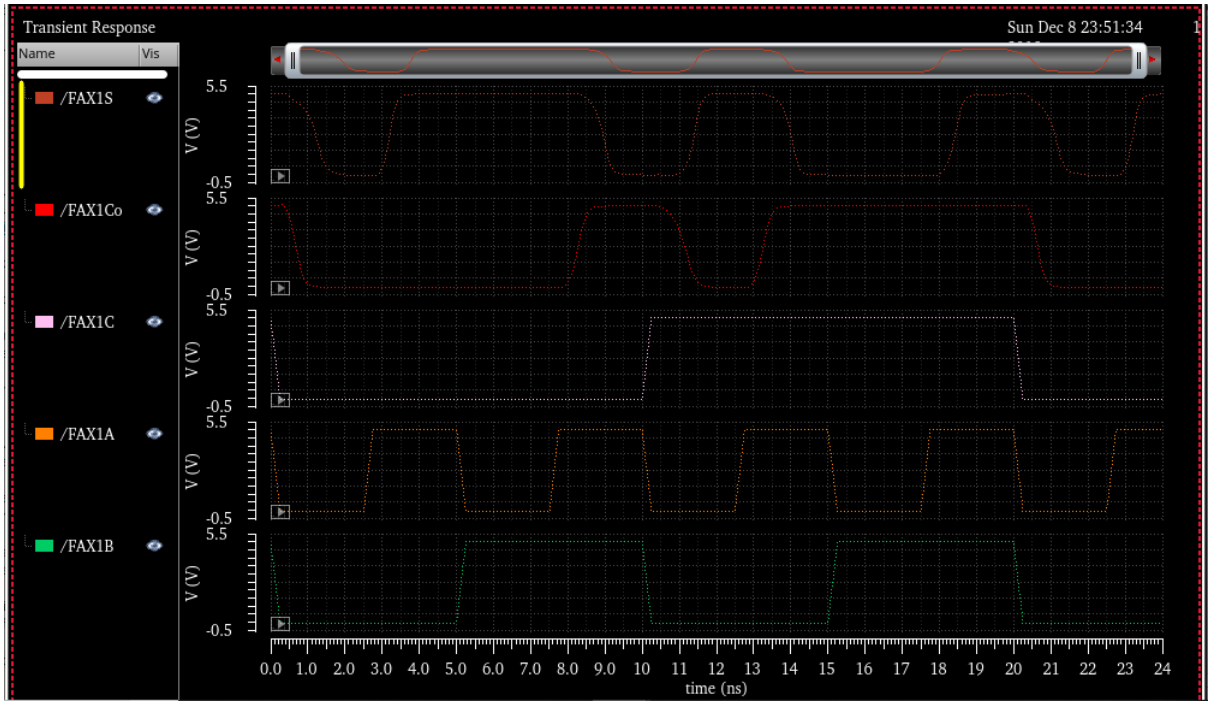
Finished test bench is shown below:




TBfax schematic




TBfax schematic ADE L




TBfax schematic output (200MHz without parasitics)

The transition propagation delays of the S and Co outputs are:

Schematic-only simulation of FAX1:

$t_{pd}$(S|ABC:000 -> 100) = 576.6E-12 = **576.6ps**

$t_{pd}$(S|ABC:001 -> 101) = 1.379E-9 = **1379ps**

$t_{pd}$(S|ABC:010 -> 110) = 1.32E-9 = **1320ps**

$t_{pd}$(S|ABC:011 -> 111) = 798.3E-12 = **798.3ps**

$t_{pd}$(S|ABC:110 -> 001) = 1.276E-9 = **1276ps**

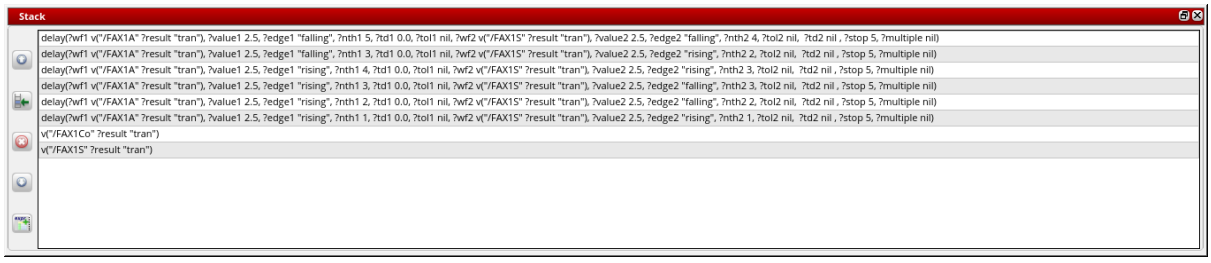
$t_{pd}$(S|ABC:111 -> 000) = 978.2E-12 = **978.2ps**

$t_{pd}$(Co|ABC:001 -> 101) = 631.1E-12 = **631.1ps**

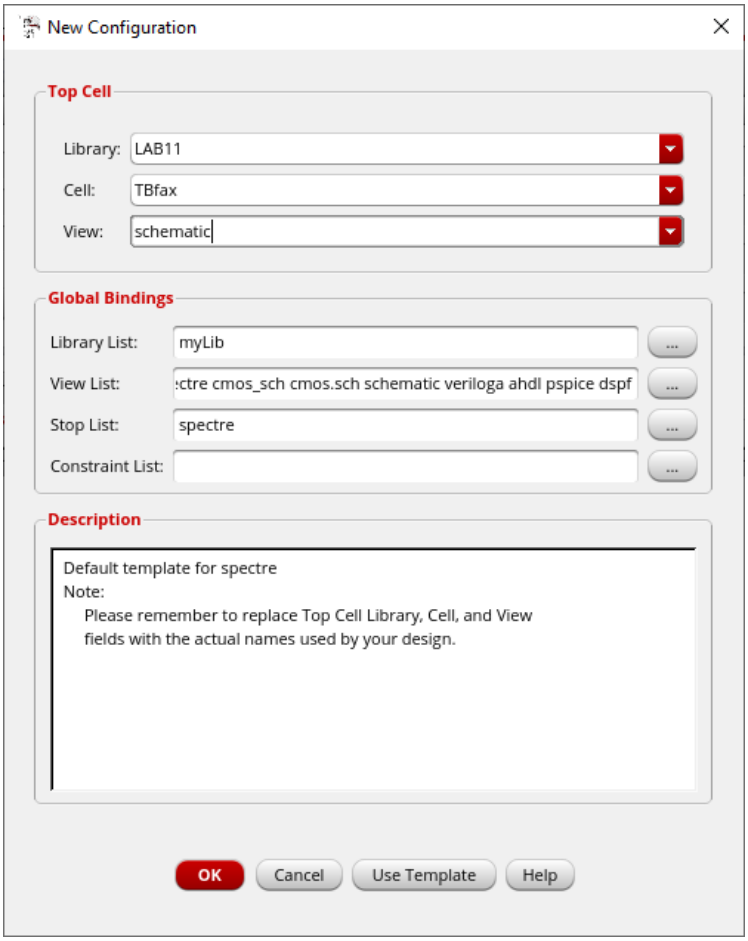
$t_{pd}$(Co|ABC:010 -> 110) = 647.6E-12 = **647.6ps**

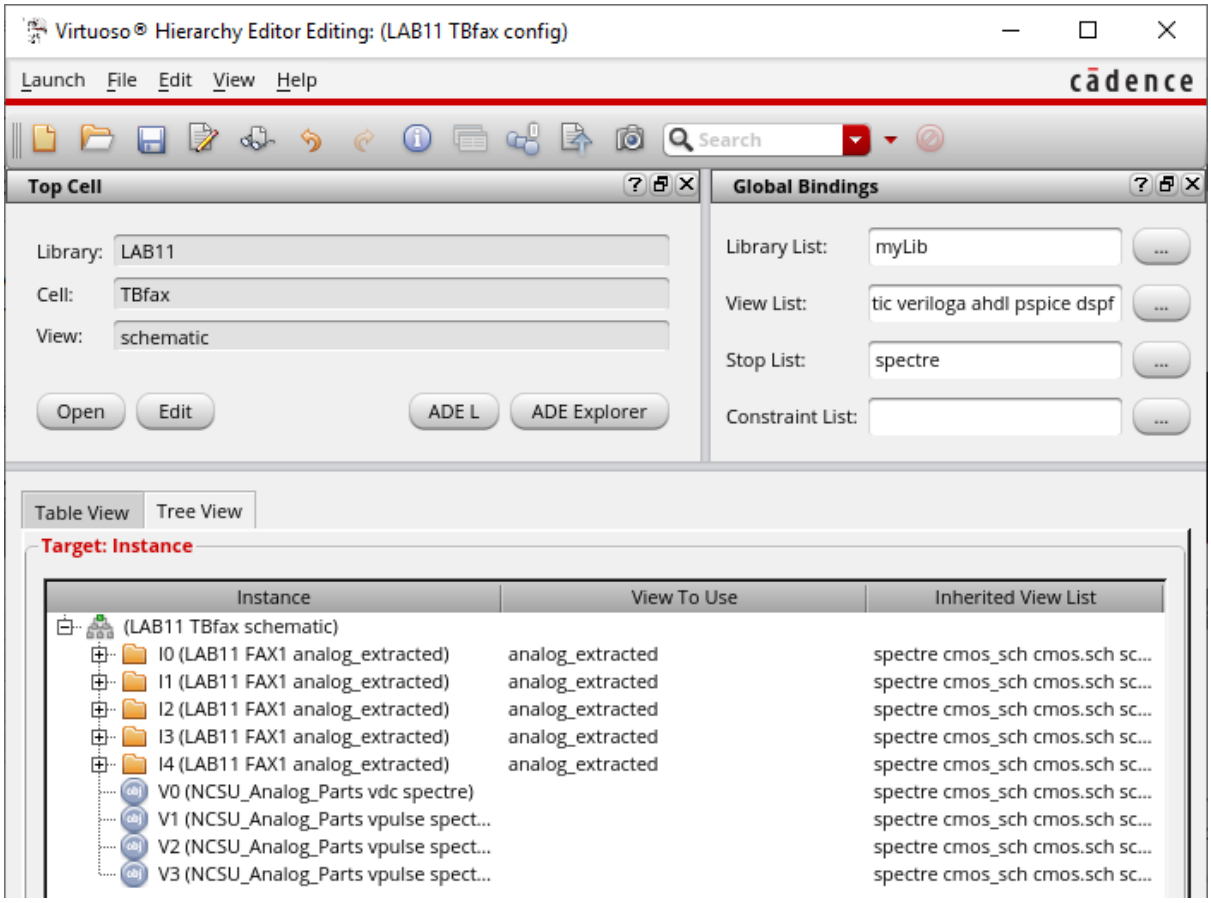
$t_{pd}$(Co|ABC:110 -> 001) = 1.023E-9 = **1023ps**

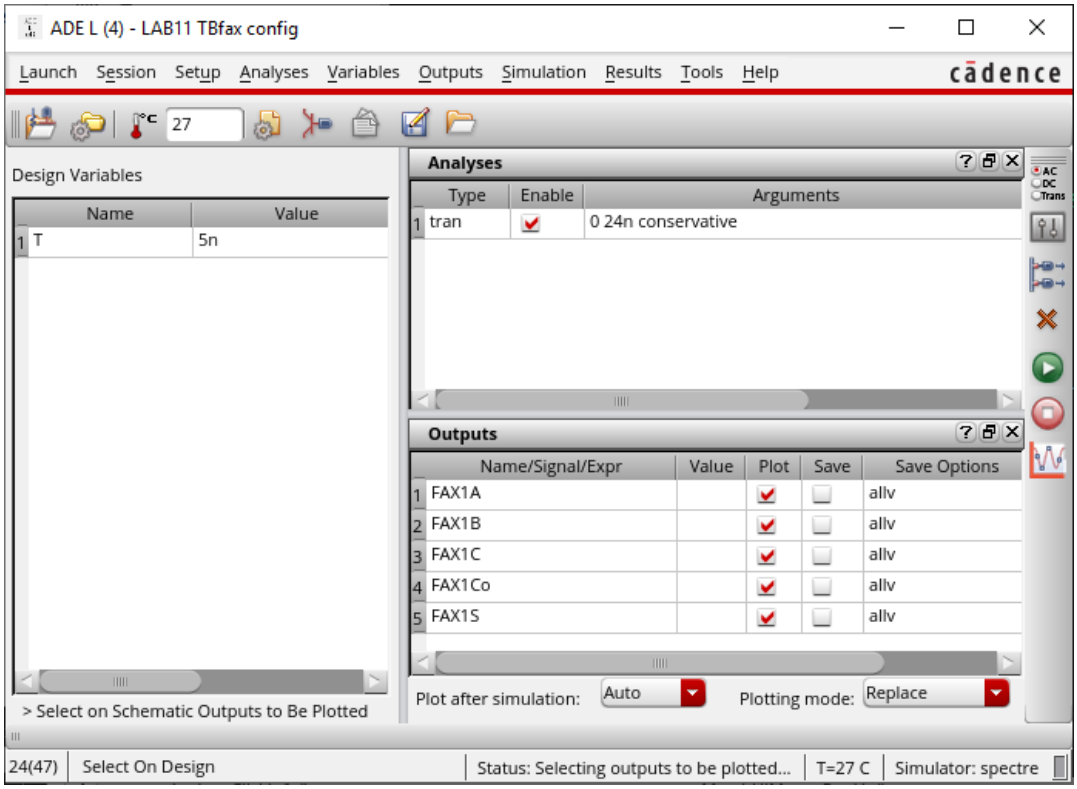
$t_{pd}$(Co|ABC:111 -> 000) = 518.0E-12 = **518ps**



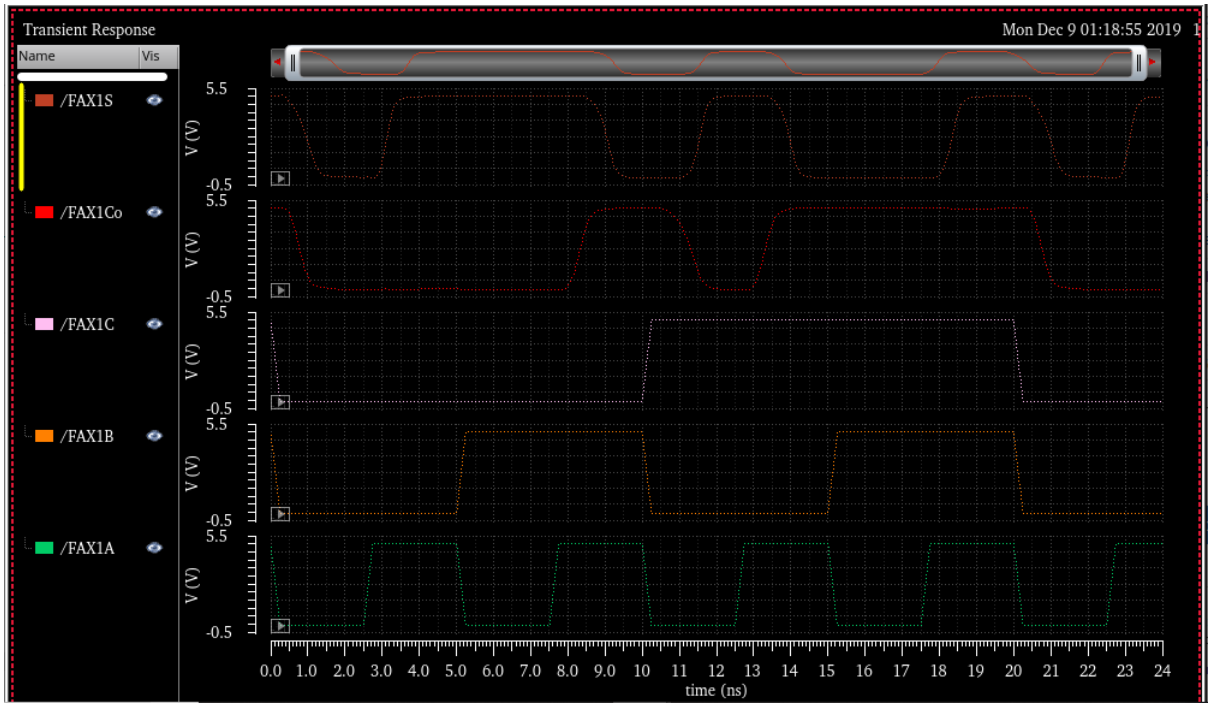

We'll compare these results to layout-based results, including parasitics. For this, we will create a config view for TBfax.


New Configuration
Top Cell
Library: LAB11
Cell: TBfax
View: schematic
Global Bindings
Library List: myLib
View List: :ctre cmos_sch cmos.sch schematic veriloga ahdl pspice dspf
Stop List: spectre
Constraint List:
Description
Default template for spectre
Note:
Please remember to replace Top Cell Library, Cell, and View
fields with the actual names used by your design.
OK
Cancel
Use Template
Help


Virtuoso® Hierarchy Editor Editing: (LAB11 TBfax config)
Launch File Edit View Help
cadence
Search
Top Cell
Library: LAB11
Cell: TBfax
View: schematic
Open
Edit
ADE L
ADE Explorer
Global Bindings
Library List: myLib
View List: tic veriloga ahdl pspice dspf
Stop List: spectre
Constraint List:
Table View
Tree View
Target: Instance
Instance
View To Use
Inherited View List
(LAB11 TBfax schematic)
I0 (LAB11 FAX1 analog_extracted) analog_extracted spectre cmos_sch cmos.sch sc...
I1 (LAB11 FAX1 analog_extracted) analog_extracted spectre cmos_sch cmos.sch sc...
I2 (LAB11 FAX1 analog_extracted) analog_extracted spectre cmos_sch cmos.sch sc...
I3 (LAB11 FAX1 analog_extracted) analog_extracted spectre cmos_sch cmos.sch sc...
I4 (LAB11 FAX1 analog_extracted) analog_extracted spectre cmos_sch cmos.sch sc...
V0 (NCSU_Analog_Parts vdc spectre) spectre cmos_sch cmos.sch sc...
V1 (NCSU_Analog_Parts vpulse spect... spectre cmos_sch cmos.sch sc...
V2 (NCSU_Analog_Parts vpulse spect... spectre cmos_sch cmos.sch sc...
V3 (NCSU_Analog_Parts vpulse spect... spectre cmos_sch cmos.sch sc...




FAX1 config ADE L




TBfax config output (200MHz with parasitics)

The transition propagation delays of the S and Co outputs are:

Layout-based simulation of FAX1 (including parasitics):

$t_{pd}$(S|ABC:000 -> 100) = 504.3E-12 = **504.3ps**

$t_{pd}$(S|ABC:001 -> 101) = 1.333E-9 = **1333ps**

$t_{pd}$(S|ABC:010 -> 110) = 1.42E-9 = **1420ps**

$t_{pd}$(S|ABC:011 -> 111) = 718.3E-12 = **718.3ps**

$t_{pd}$(S|ABC:110 -> 001) = 1.424E-9 = **1424ps**

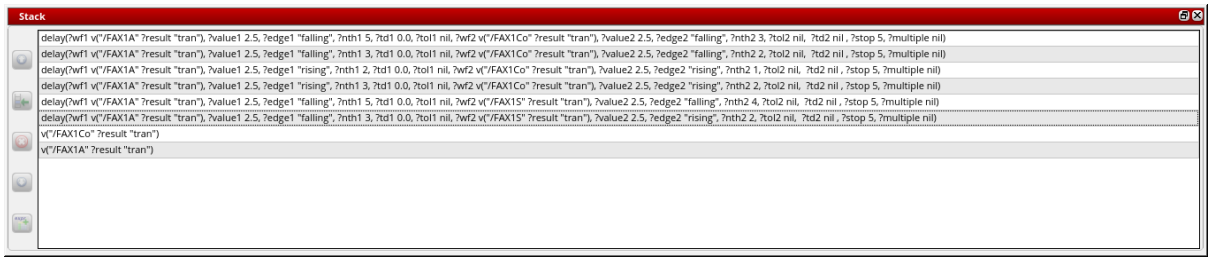
$t_{pd}$(S|ABC:111 -> 000) = 835.2E-12 = **835.2ps**

$t_{pd}$(Co|ABC:001 -> 101) = 683.9E-12 = **683.9ps**

$t_{pd}$(Co|ABC:010 -> 110) = 706.2E-12 = **706.2ps**

$t_{pd}$(Co|ABC:110 -> 001) = 1.171E-9 = **1171ps**

$t_{pd}$(Co|ABC:111 -> 000) = 635.5E-12 = **635.5ps**




Here is the comparison between the schematic-only and layout-parasitics based simulation for FAX1:

| FO4 Delay | HAX1 schematic-only | HAX1 using analog_extracted view (layout parasitics included) |
|---|---|---|
| **$t_{pd}$(S\|ABC:000 -> 100)** | 576.6ps | 504.3ps |
| **$t_{pd}$(S\|ABC:001 -> 101)** | 1379ps | 1333ps |
| **$t_{pd}$(S\|ABC:010 -> 110)** | 1320ps | 1420ps |
| **$t_{pd}$(S\|ABC:011 -> 111)** | 798.3ps | 718.3ps |
| **$t_{pd}$(S\|ABC:110 -> 001)** | 1276ps | 1424ps |
| **$t_{pd}$(S\|ABC:111 -> 000)** | 978.2ps | 835.2ps |
| **$t_{pd}$(Co\|ABC:001 -> 101)** | 631.1ps | 683.9ps |
| **$t_{pd}$(Co\|ABC:010 -> 110)** | 647.6ps | 706.2ps |
| **$t_{pd}$(Co\|ABC:110 -> 001)** | 1023ps | 1171ps |
| **$t_{pd}$(Co\|ABC:111 -> 000)** | 518ps | 635.5ps |
| **S output** | **1379ps** | **1424ps** |
| **Co output** | **1023ps** | **1171ps** |
| **FAX1 cell** | **1379ps** | **1424ps** |

Although the layout-based simulation results are not always worse, the overall FAX1 cell performance ended up being slightly worse than the schematic-only simulation.

The worst-case input transition FO4 delay for FAX1 ended up being 1.424ns, based on the layout-based results whereas for the HAX1, FO4 delay was 0.922ns. This is because the FAX1 layout is almost double the size of the HAX1 and hence the parasitic elements present in FAX1 is much more in number, and their individual is also greater due to longer poly connections between the inputs and outputs.