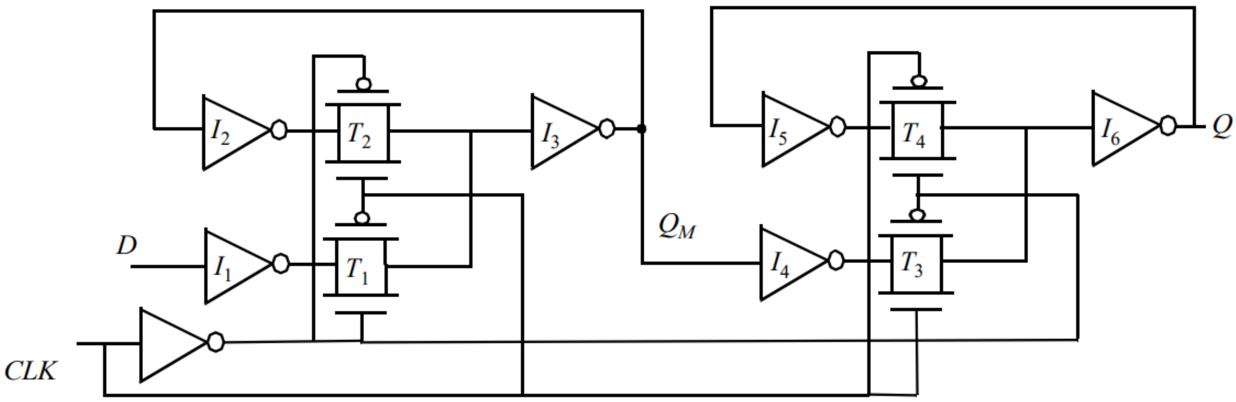
Lab 7: Flip Flop

Objective

The objective of this lab is to go through the full design flow of Master-Slave Based Edge triggered Register (Rising Edge Flip Flop), including frontend and backend flow (Design, Simulation, layout, DRC, LVS, Parasitic extraction and post layout simulation).

Theory

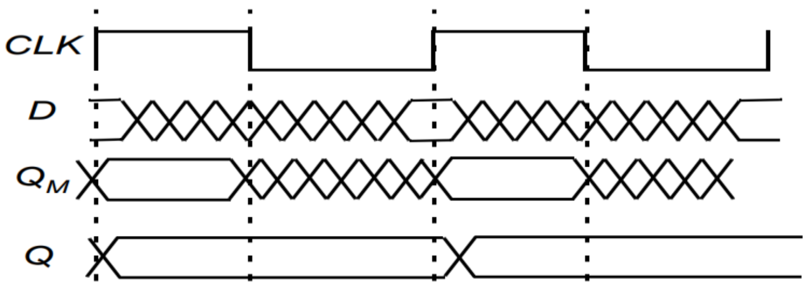
The most common approach for constructing an edge-triggered register (**Flip Flop**) is to use a master-slave configuration. The **positive edge-triggered Flip Flop** consists of cascading a negative latch (master stage) with a positive latch (slave stage). A **negative edge-triggered Flip Flop** can be constructed using the same principle by simply switching the order of the positive and negative latch (i.e., placing the positive latch first).



master-slave postive edge-triggered flip flop.

When **CLK is Law**, the master latch will be on transparent mode and $Q_m = D$ and the slave latch will be on the holding mode (hold old value of Q).

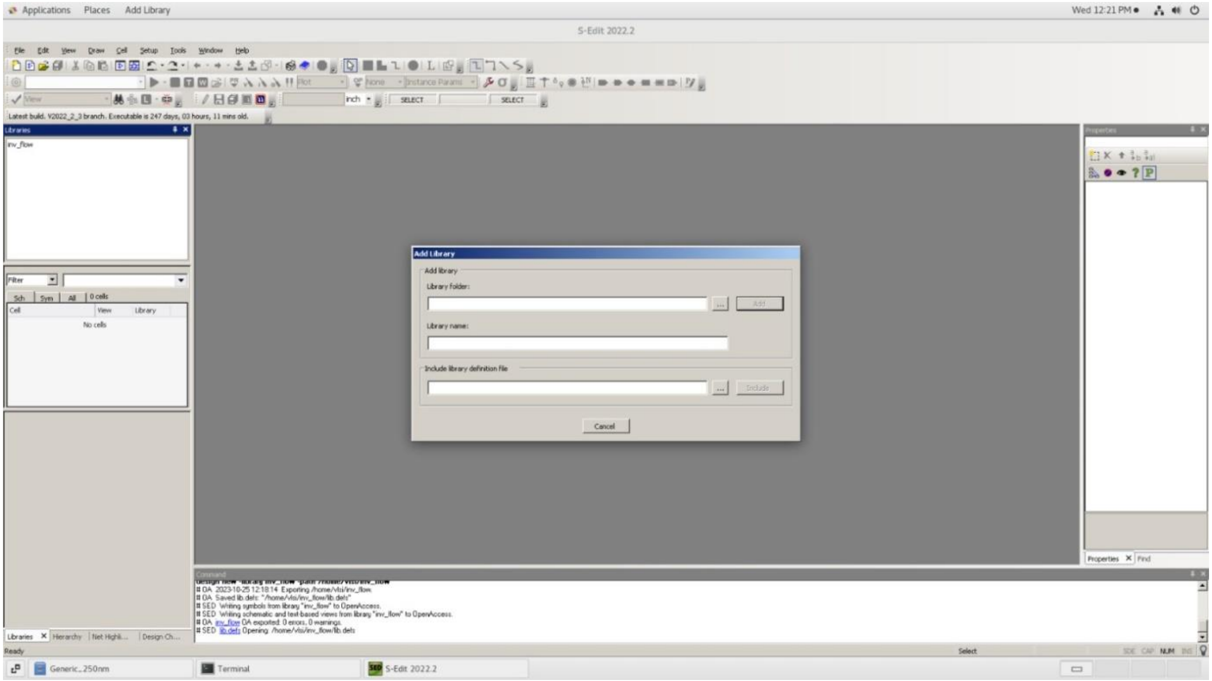
When **CLK is High**, the master latch will be on holding mode and Q_m will hold the old value, and the slave latch will be on the transparent mode $Q = Q_m$.



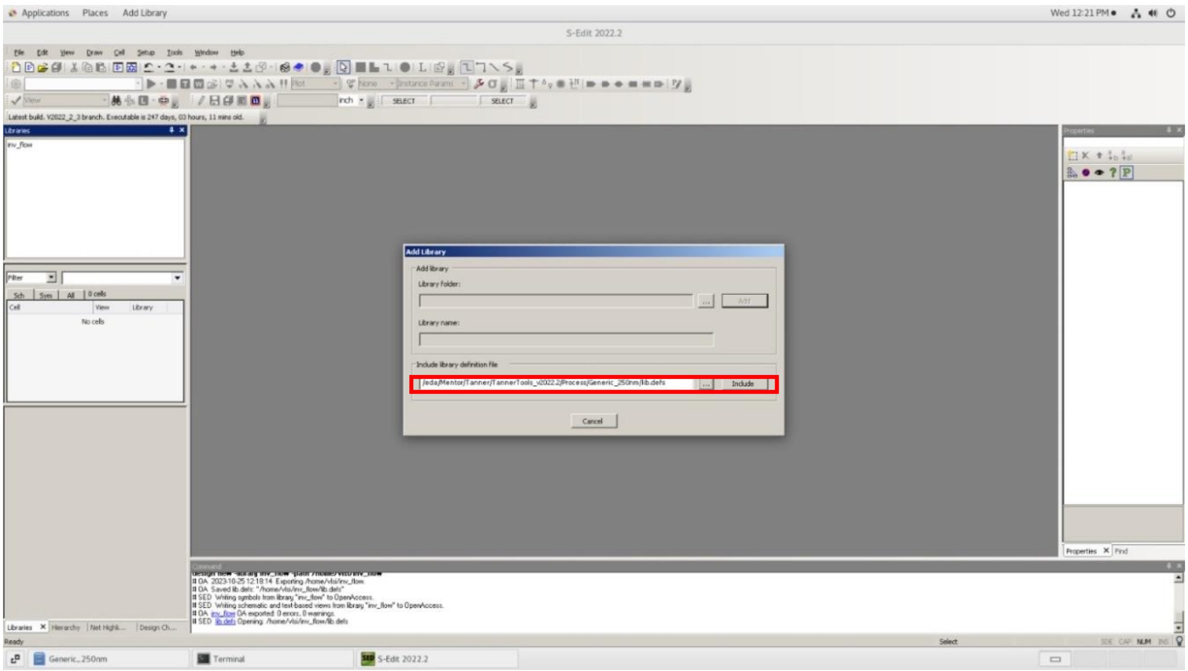
Frontend Flow

1. Include libraries

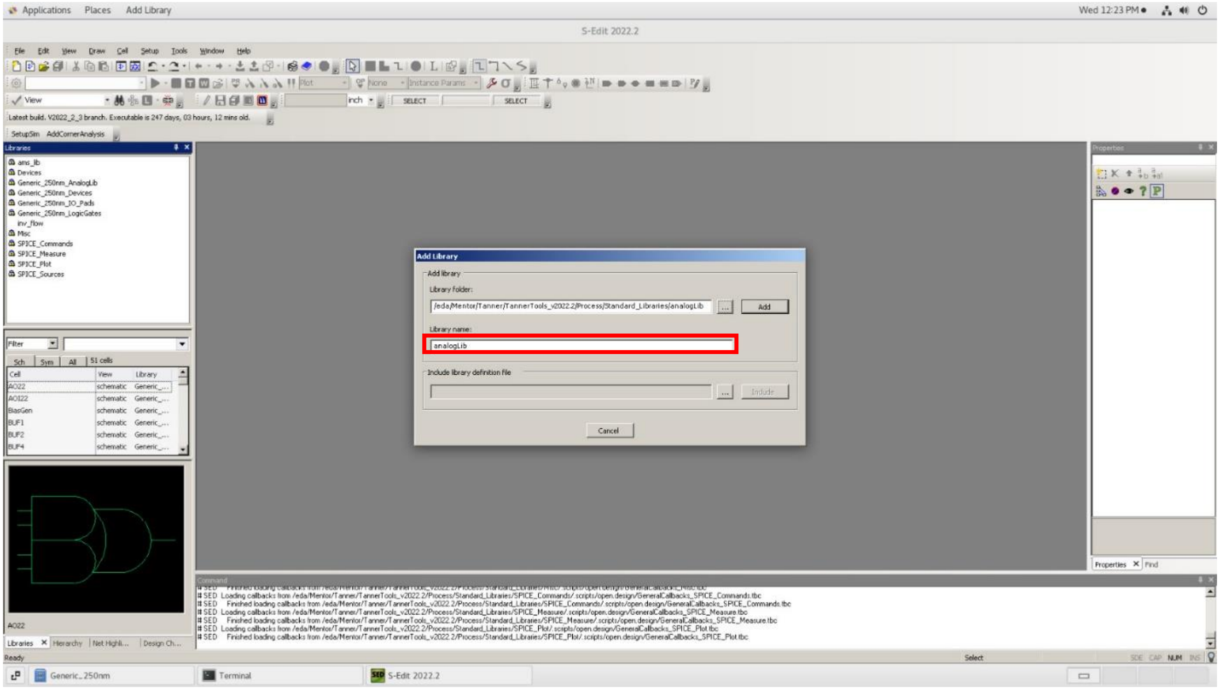
   Note: All Libraries used are from technology 2022.



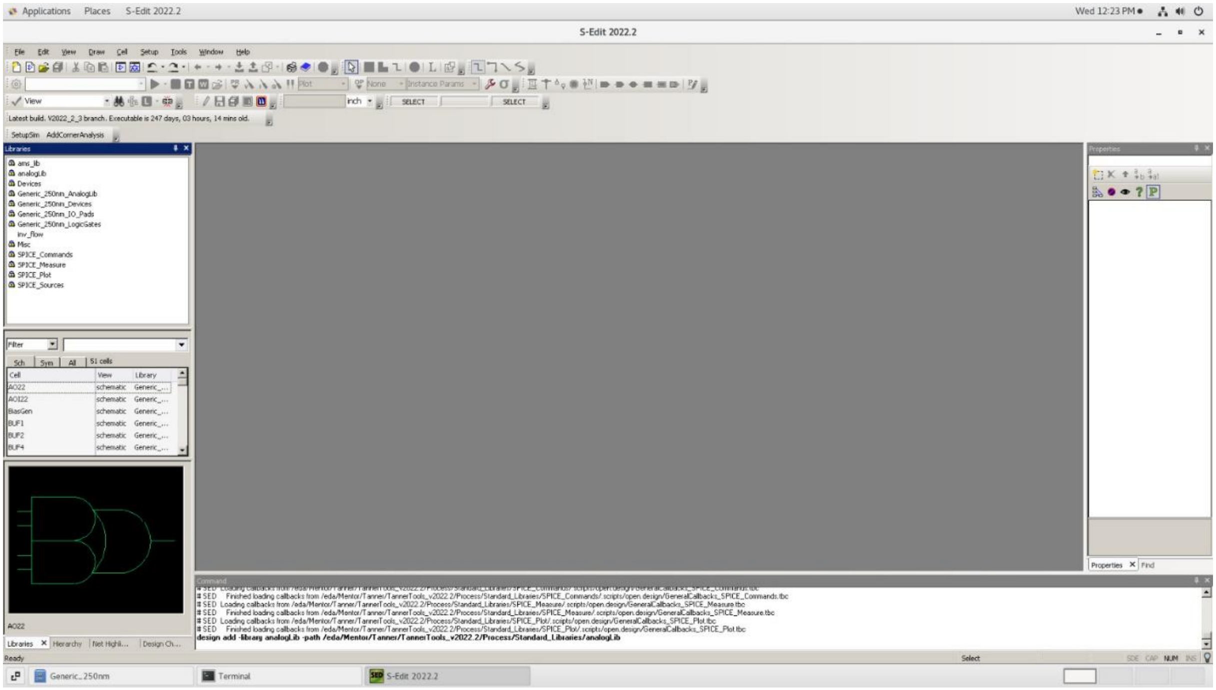
Include all libraries in one step ☺



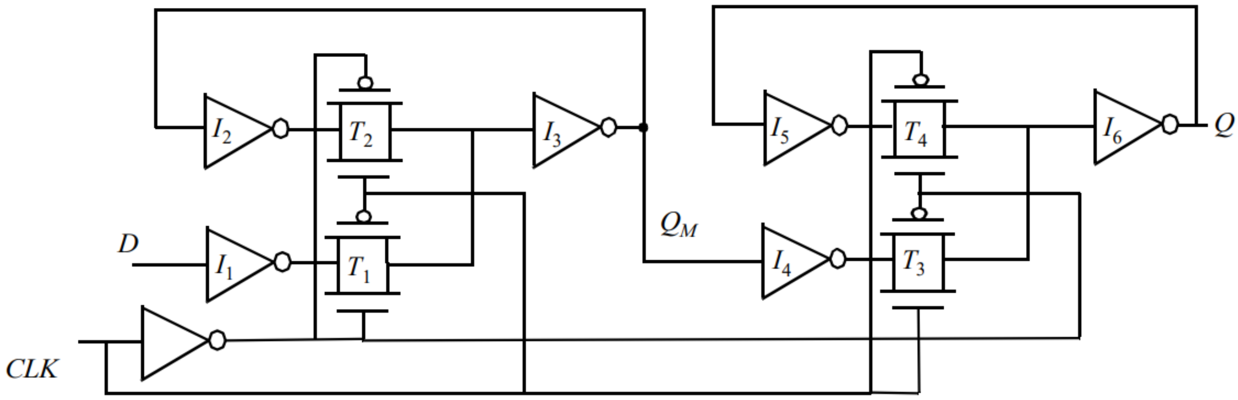
   You also need to add **<u>analog-lib</u>** from standard libraries.



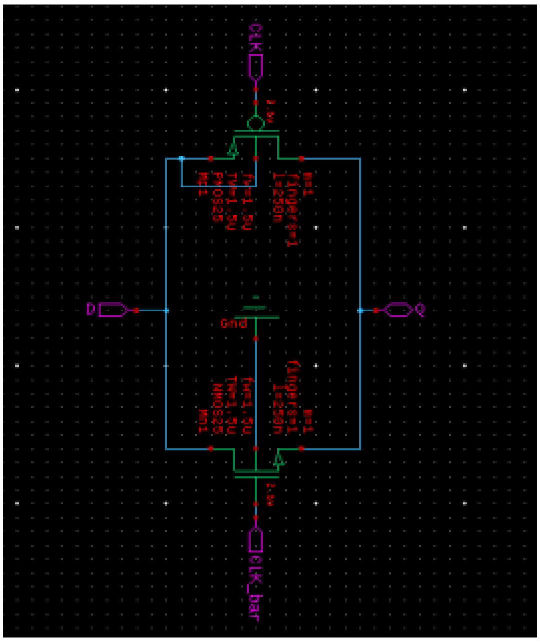
Now all done ☺



2. Design an Latch from scratch **(DON'T use generic gates)**

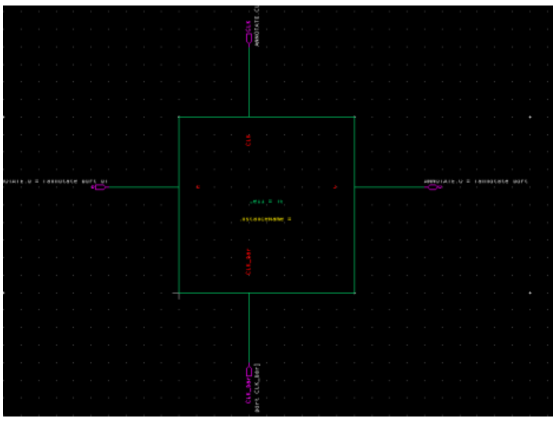


- **First**, you should design the transmission gate and create a symbol for it:

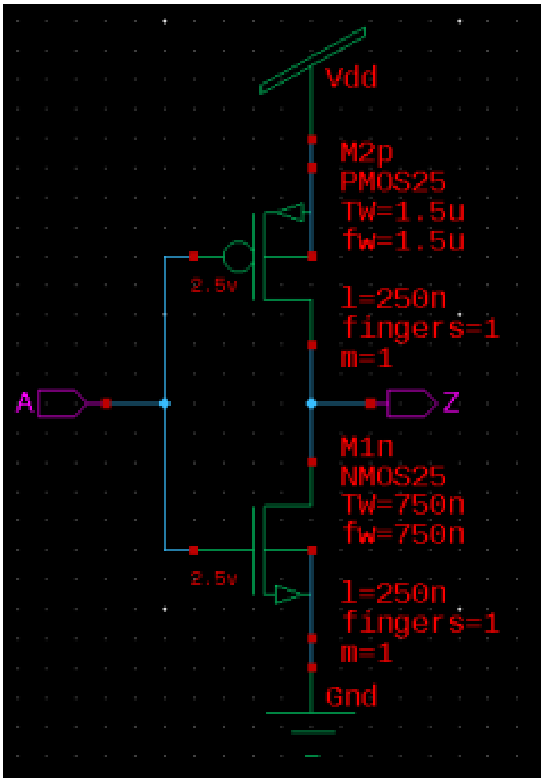


NOTE: **Don't include global ports** while generating symbol.

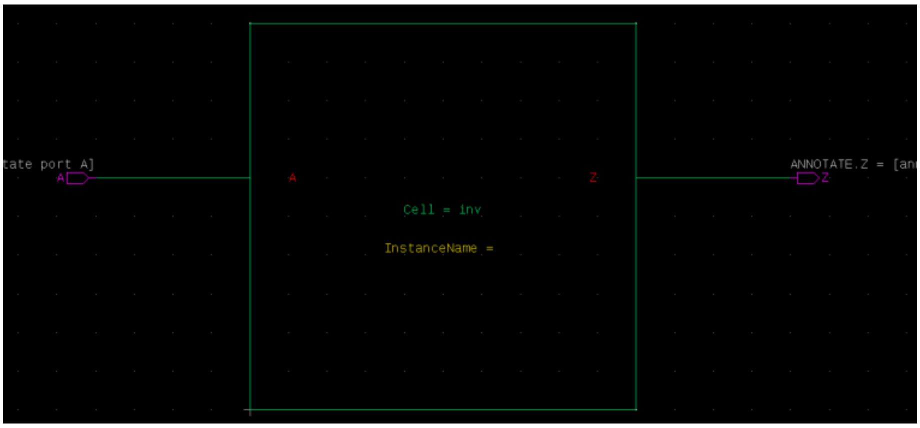
Hint: Use PMOS25x to connect bulk with source, on the other hand use NMOS25x but connect bulk to ground (to avoid errors in LVS because bulk pin by default connected to gnd in layout). **DON'T USE NMOS25 & PMOS25**.



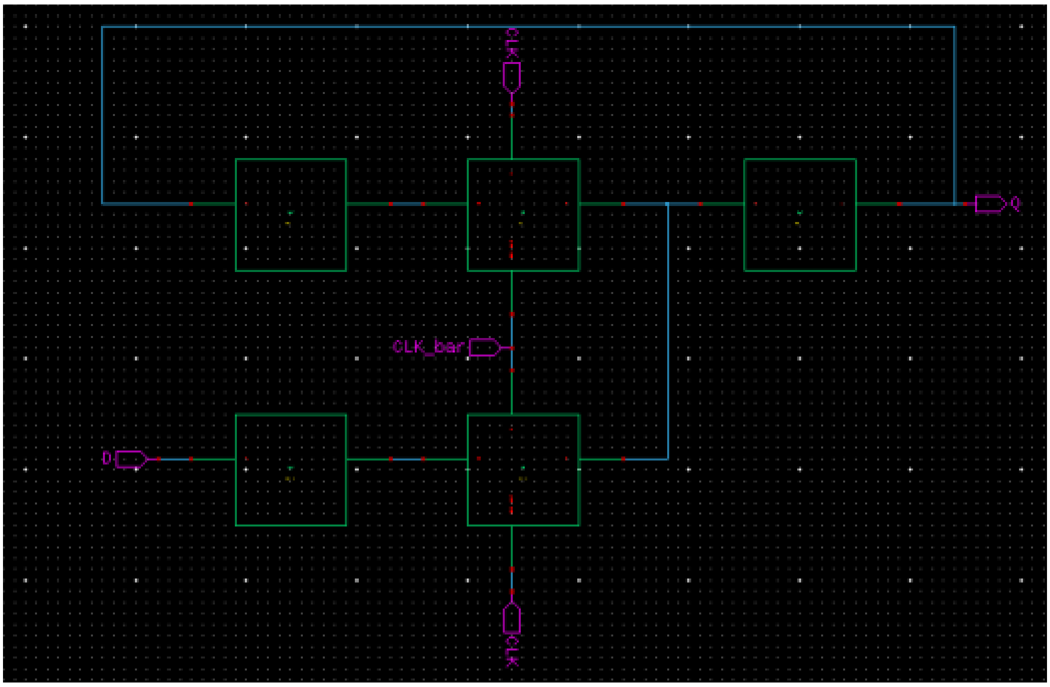
- **Second**, you should design the inverter from transistor level (or use the inverter from the previous lab) and create a symbol for it. (Avoid using generic gates)

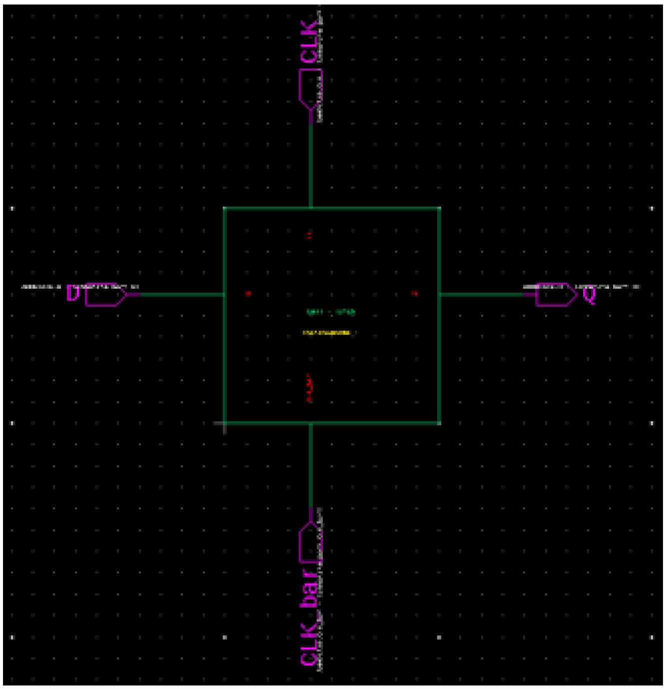


NOTE: Don't include global ports while generating symbol.

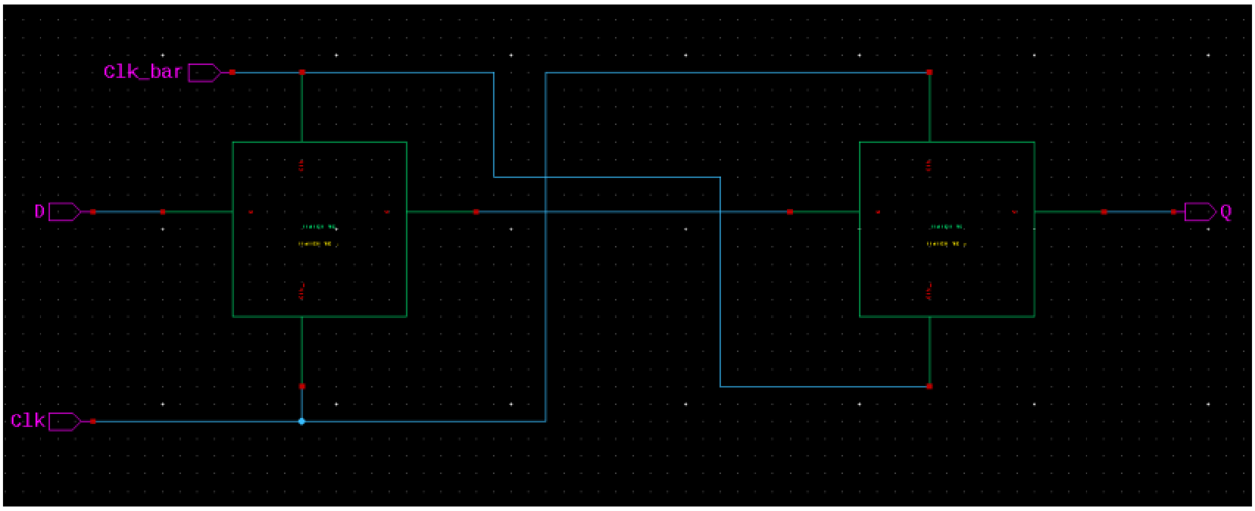


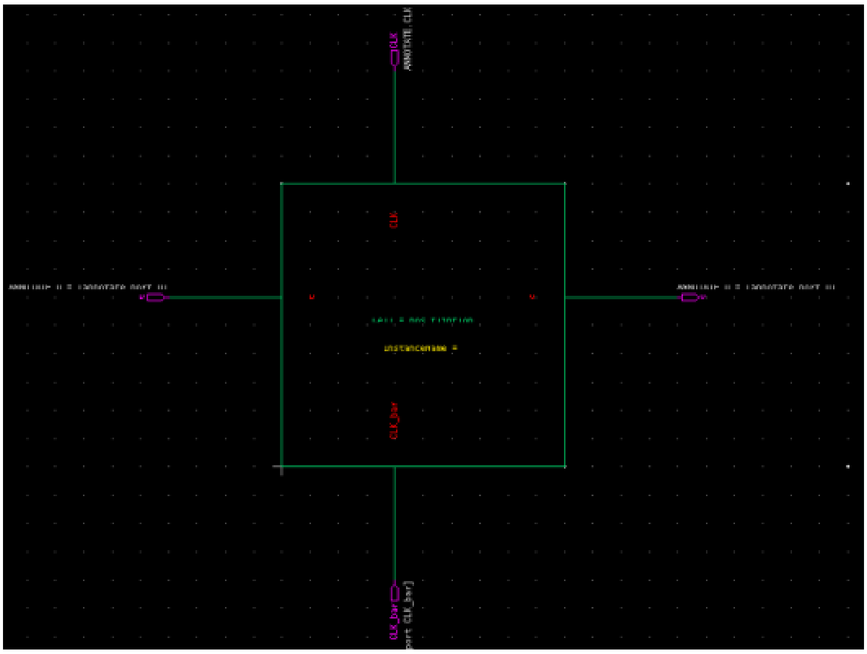
- **Third**, use the TG and inverter symbol to build your latch and create a symbol for it:



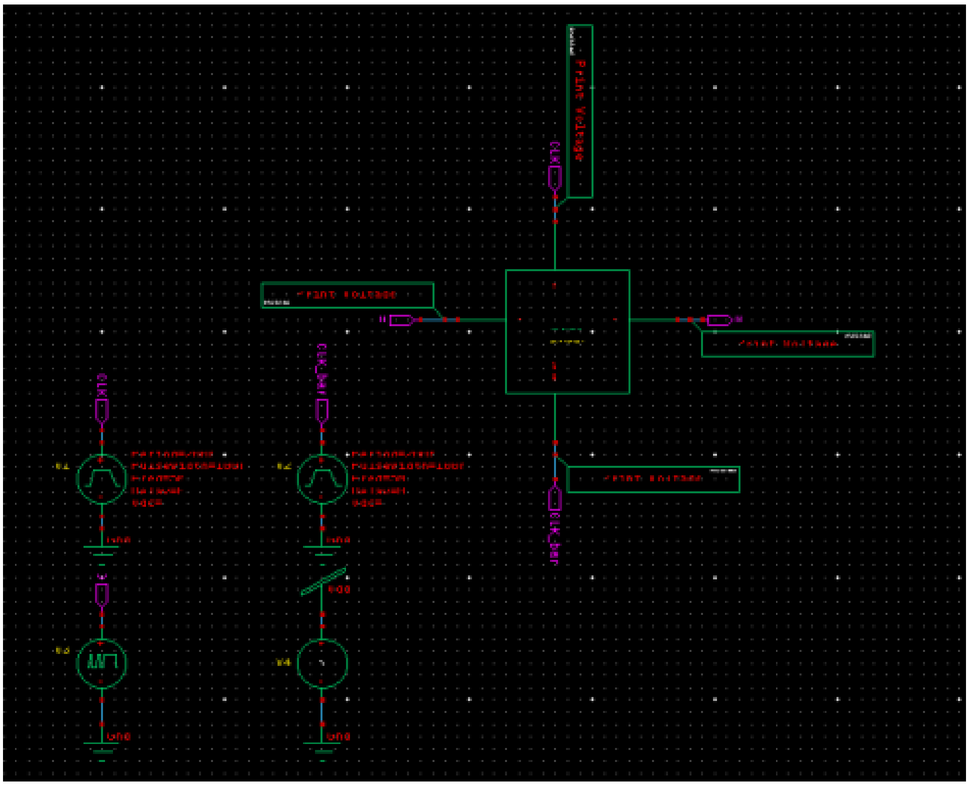


- **Finally**, use the TG, inverter and latch symbols to build your Flip Flop and create a symbol for it:





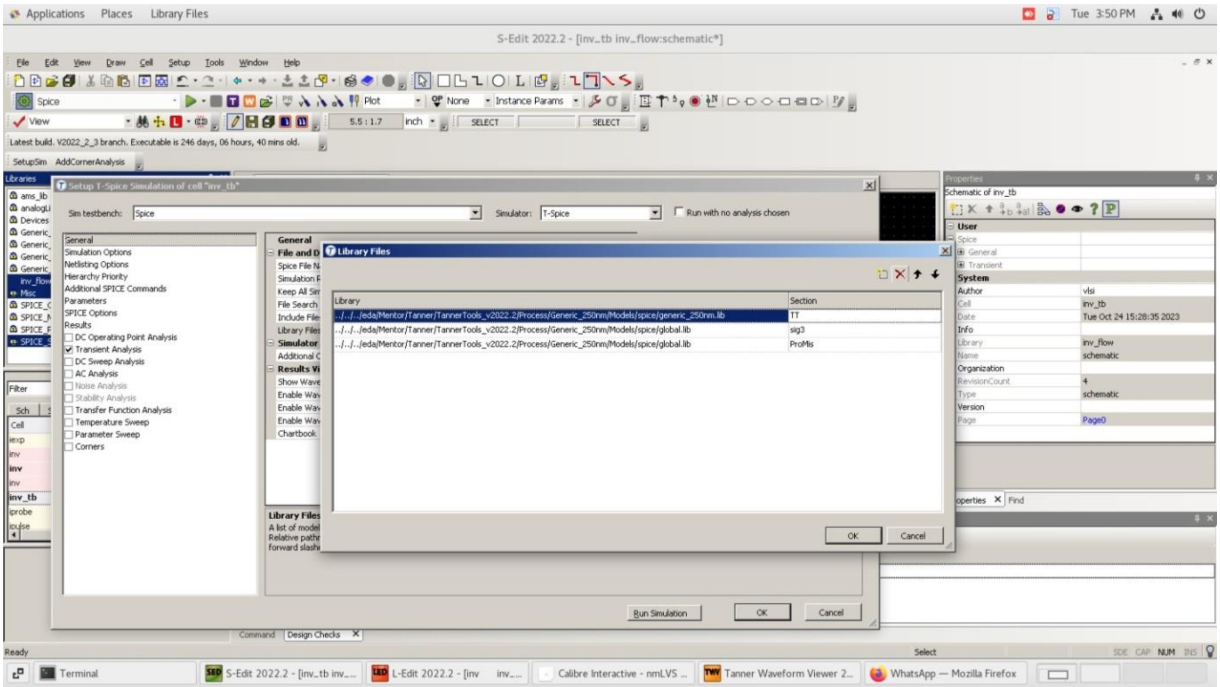
3. Test your design.

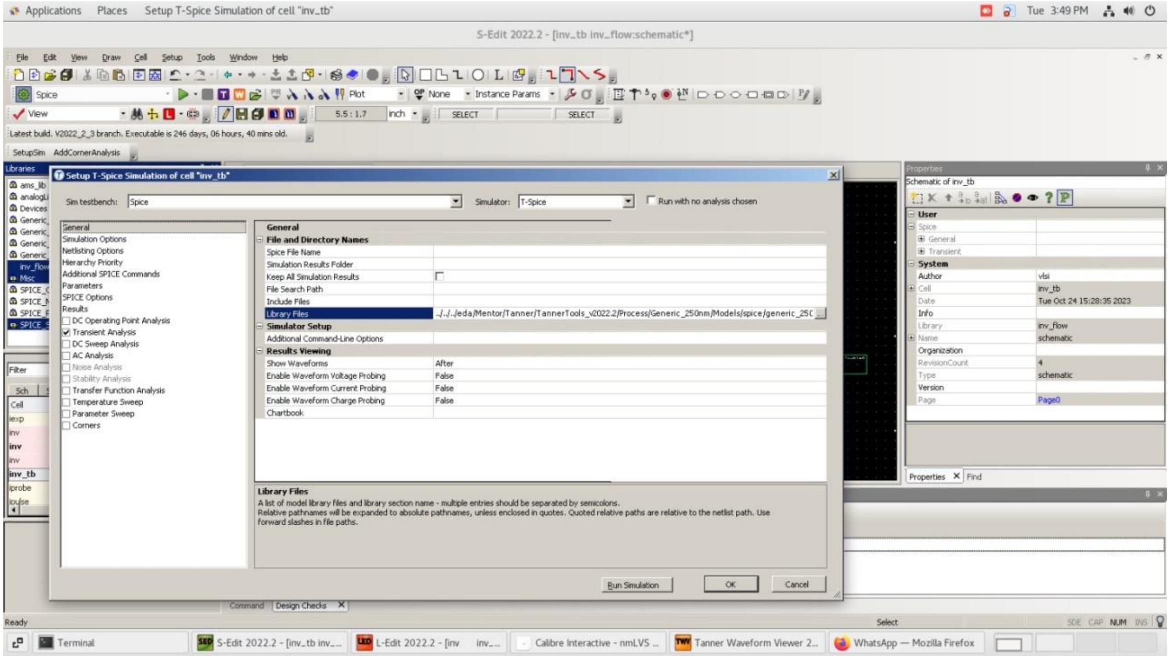


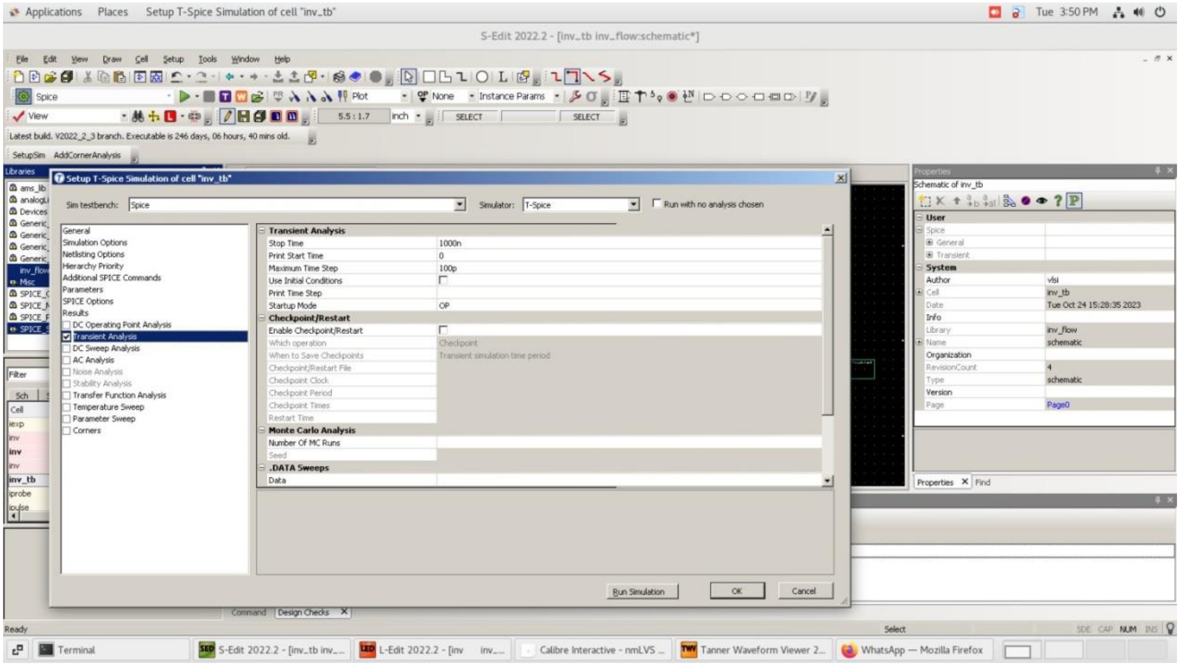
(Change the rise and fall time for all source to be zero)

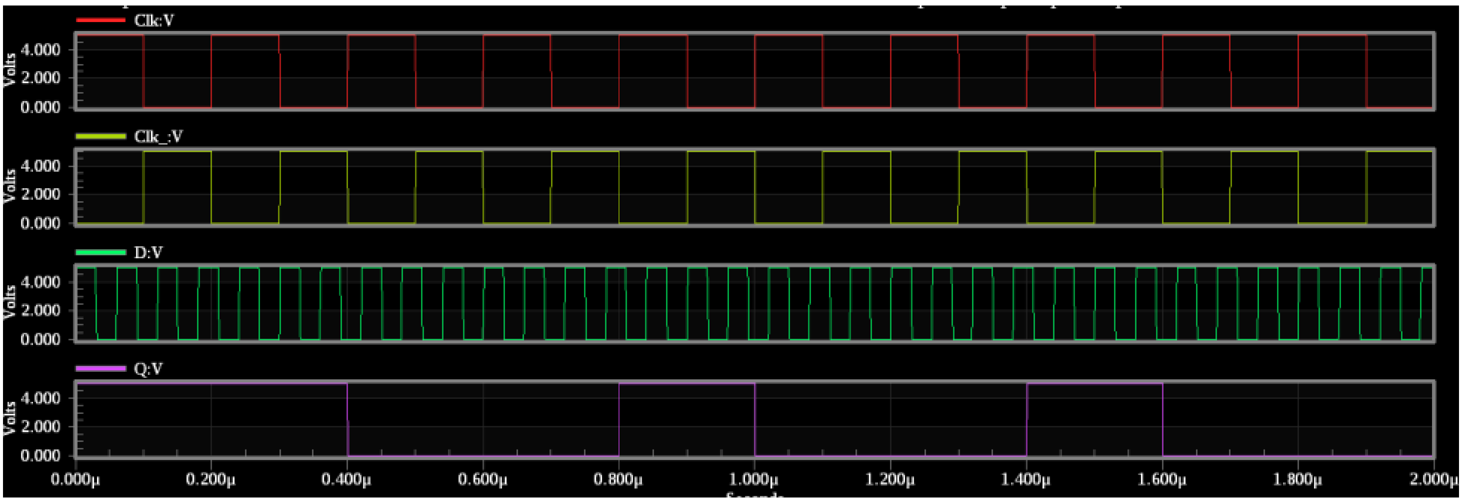
<u>**Hint**</u>: You can use Vpulse for CLK and CLK_bar, and Vbit for D. You can **invert** Vpulse polarity for for CLK & CLK_bar by **changing High and Low levels**.

4. Simulate




Setup T-Spice Simulation of cell "inv_tb"
S-Edit 2022.2 - [inv_tb inv_flow:schematic*]
Sim testbench: Spice
Simulator: T-Spice
Run with no analysis chosen
General
File and Directory Names
Library Files
Simulator Setup
Results Viewing
Show Waveforms After
Run Simulation
OK
Cancel


Setup T-Spice Simulation of cell "inv_tb"
S-Edit 2022.2 - [inv_tb inv_flow:schematic*]
Transient Analysis
Stop Time 1000n
Print Start Time 0
Maximum Time Step 100p
Startup Mode OP
Checkpoint/Restart
Monte Carlo Analysis
.DATA Sweeps
Run Simulation
OK
Cancel


Clk:V
Clk_:V
D:V
Q:V
Volts
4.000
2.000
0.000
0.000µ
0.200µ
0.400µ
0.600µ
0.800µ
1.000µ
1.200µ
1.400µ
1.600µ
1.800µ
2.000µ

Backend Flow

After Finishing your Schematic, now it is time to setup your Layout. The next step is to publish to **SDL** (schematic driven layout).

Important Notes Before Generating Layout:

To be able to **save VIAs** in L-edit follow the following steps before starting layout and make sure that your L-edit was closed:

1. In S-edit command window write:

   "tech report –library lab7 –attach Generic_250nm_Devices"

   Your library name

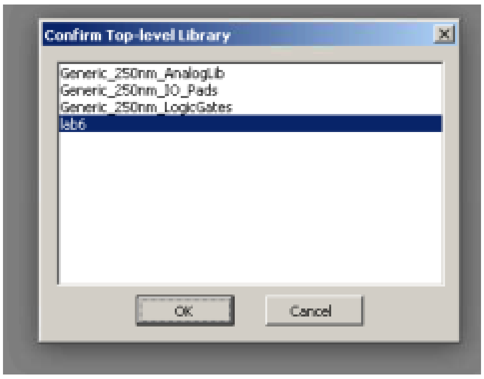
2. To ensure that it works well write in S-edit command window:

   "tech report -all"

   This command will appears

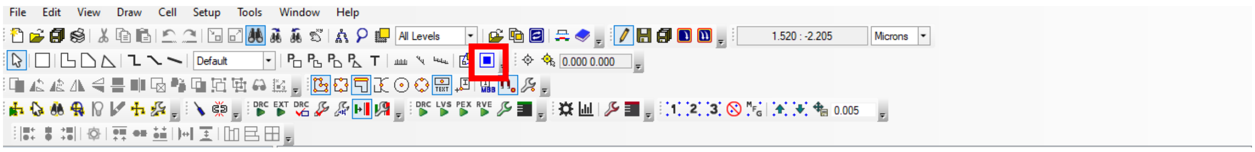
SED {lab7 attaches Generic_250nm_Devices} {Generic_250nm_Devices local} {Generic_250nm_LogicGates attaches Generic_250nm_Devices} {Generic_250nm_AnalogLib attaches Generic_250nm_Devices} {Generic_250nm_IO_Pads attaches Generic_250nm_Devices} {Misc local} {Devices local} {SPICE_Sources local} {SPICE_Commands local} {SPICE_Measure local} {SPICE_Plot local} {ams_lib local} {analogLib local}

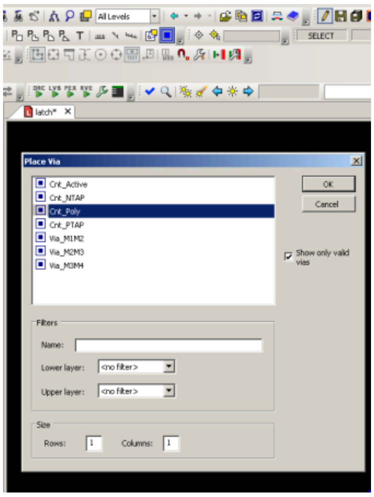
3. Open your library file location and remove **lib.defs.toplevel** if you found it.
4. Now open L-edit again and choose the **top library to be your library**, at this moment you can **purplish your SDL** ☺



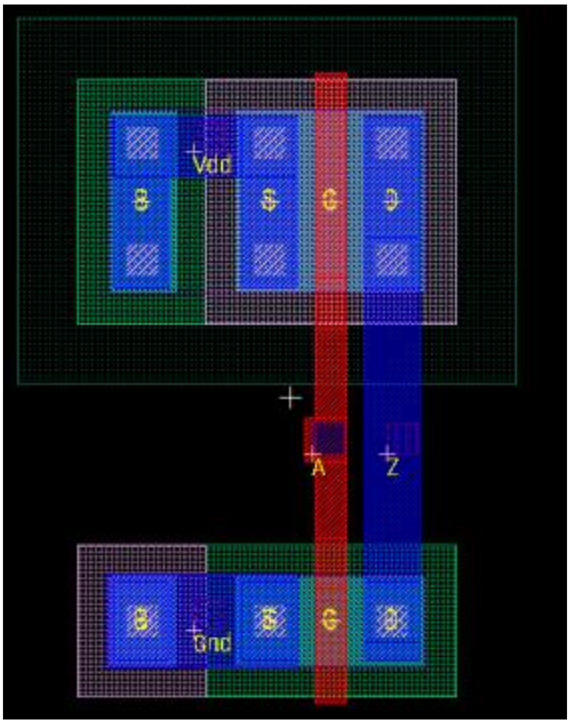
<u>Important Notes for Routing:</u>

- Manhattan routing is a **routing strategy**. You use **one dedicated layer for horizontal tracks** and **another layer for vertical tracks**. No horizontal tracks are allowed on the vertical layer, and no vertical traces are used on the horizontal layer.
- **Here assume that <u>metal 1 horizontal connections</u> , <u>metal 2 Vertical connections</u> and <u>metal 3 horizontal connections</u>.**
- To connect metal 1 with metal 2 you will need via **Via_M1M2**.
- To connect poly with metal 1 you will need via **Cnt_poly.**
- To connect poly with metal 2 you will need two vias **Cnt_poly & Via_M1M2**.
- To connect metal 2 with metal 3 you will need via **Via_M2M3**.





1. Generate a layout for inverter and route it:

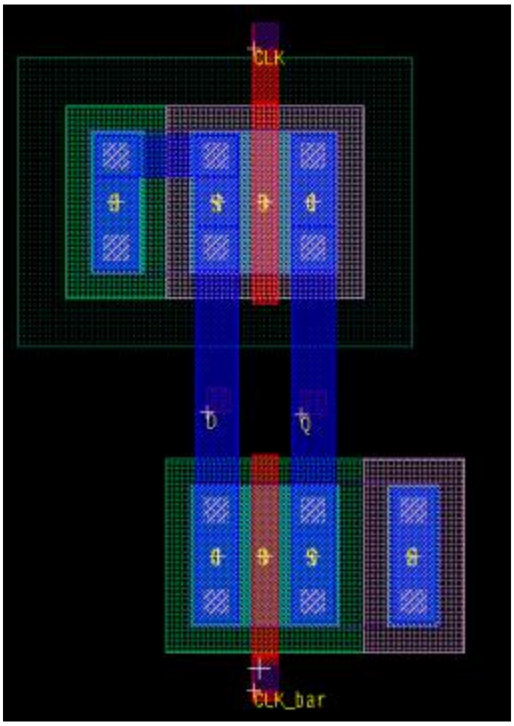


Routing Hints:

- Connect PMOS gate with NMOS gate using poly.
- Connect PMOS Drain with NMOS Drain using metal 1.
- Connect PMOS Source and Bulk with Vdd using metal 1.
- Connect NMOS Source and Bulk with Gnd using metal 1.
- Convert **Z, Vdd and Gnd** pins to **metal 1 pin** **(Ctrl + E)** and put them over metal 1.
- Convert **A** pin to **poly pin** **(Ctrl + E)** and put it over poly.

Don't forget to run DRC & LVS to validate your inverter ☺

2. Generate a layout for Transmission gate and route it:

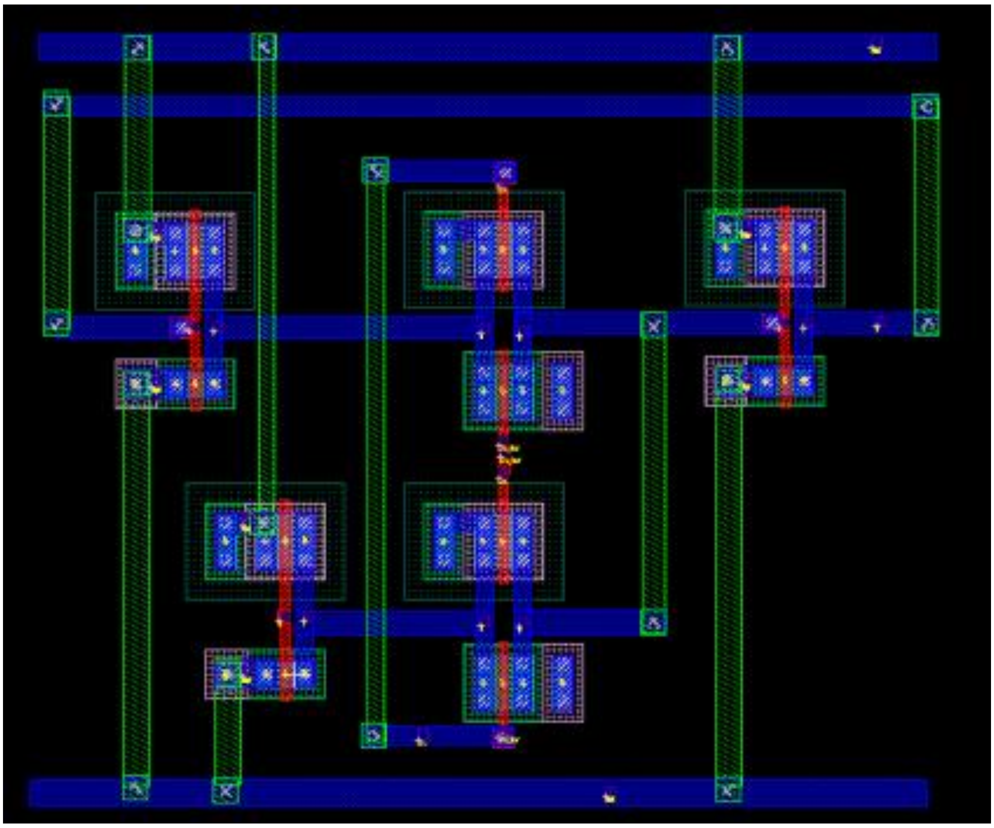


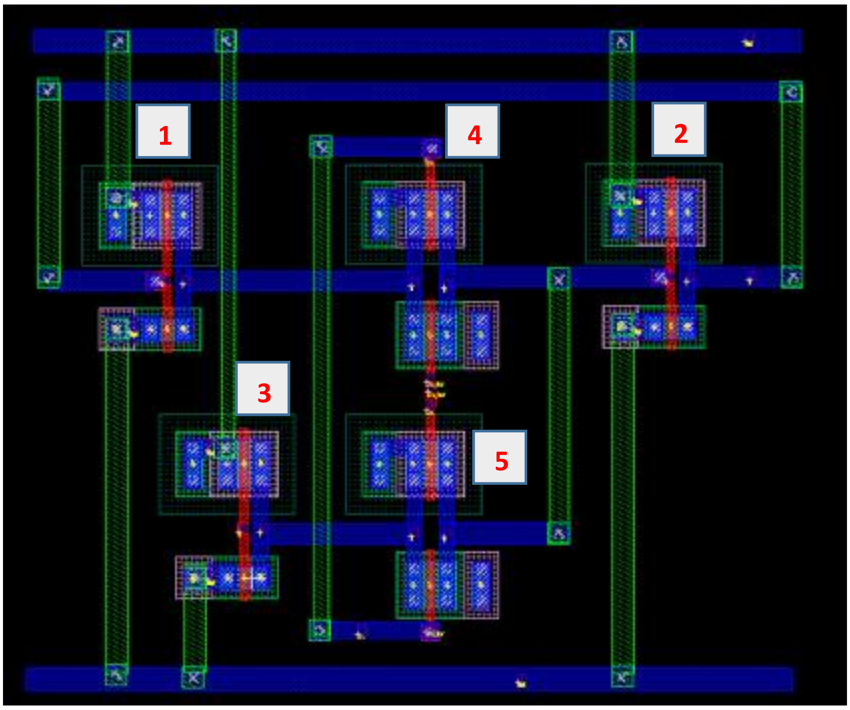
Routing Hints:

- Connect PMOS Source with NMOS Drain using metal 1.
- Connect PMOS Drain with NMOS Source using metal 1.
- Connect PMOS Source and Bulk using metal 1.
- **Don't connect** NMOS Source and Bulk with Gnd (fatal mistake because bulk grounded by default so it will ground source also).
- Convert **Q and D** pins to **metal 1 pin** **(Ctrl + E)** and put them over metal 1.
- Convert **Clk and Clk_bar** pin to **poly pin** **(Ctrl + E)** and extend some poly to put over it.

Don't forget to run DRC & LVS to validate your Transmission gate ☺

3. Generate a layout for all latch now and route it:



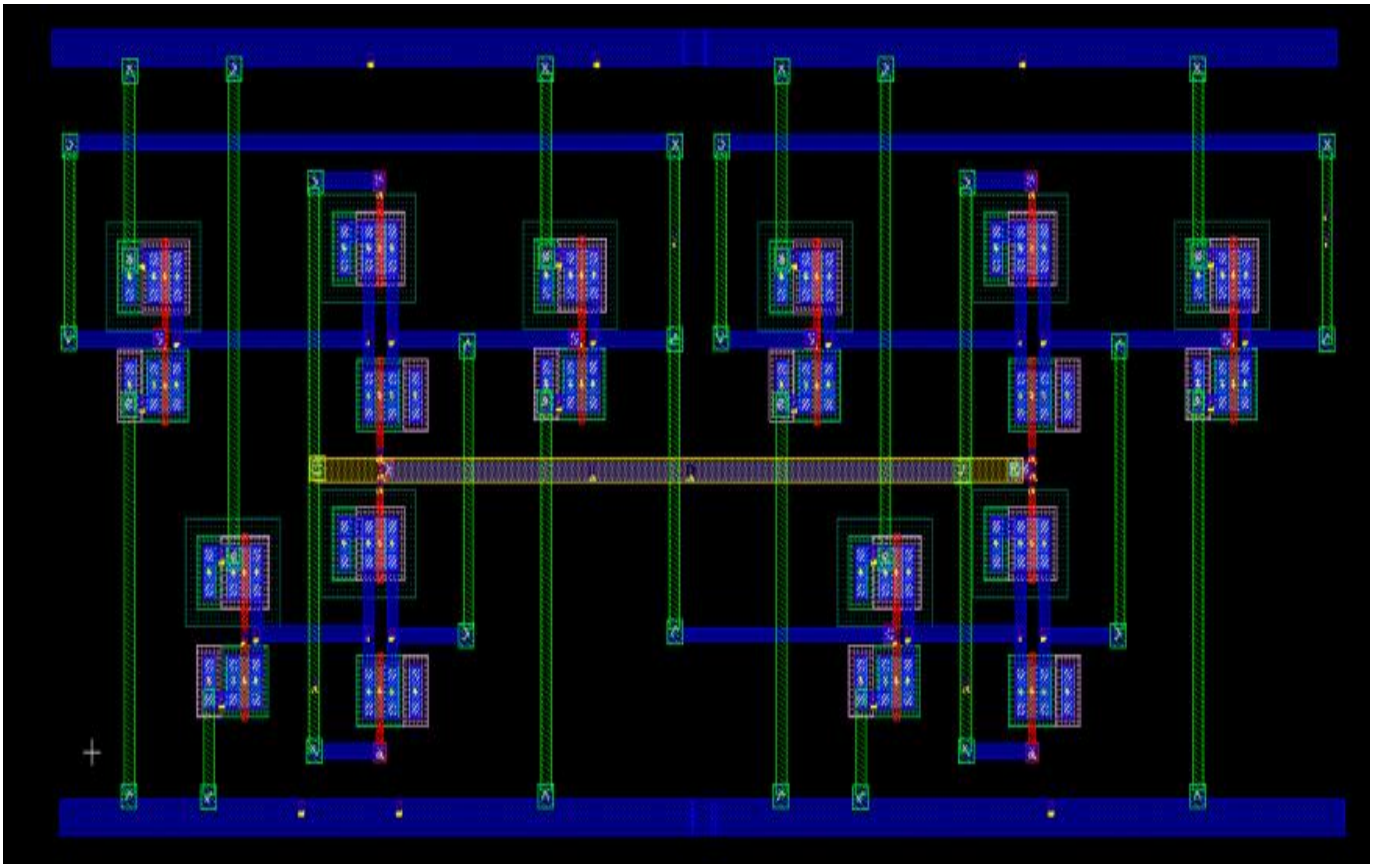


Routing Hints:

- Connect output of inverter 1 to input of Transmission gate 4.
- Connect output of Transmission gate 4 to input of inverter 2.
- Connect output of inverter 3 to input of Transmission gate 5.
- Connect output of Transmission gate 5 to output of Transmission gate 4.
- Connect Clk_bar pin of Transmission gate 4 to Clk pin of Transmission gate 5 (using poly) and connect both of them to latch Clk_bar pin after converting it to **poly pin** **(Ctrl + E).**
- Connect Clk pin of Transmission gate 4 to Clk_bar pin of Transmission gate 5 (using metal 1 & metal 2) and connect both of them to latch Clk_bar pin after converting it to **metal 1 pin** **(Ctrl + E).**
- Connect output of inverter 2 to input of inverter 1.
- Common Vdd for all inverters PMOS.
- Common Gnd for all inverters NMOS.

Don’t forget to run DRC & LVS to validate your Latch ☺

4. Generate a layout for all Flip Flop now and route it:



Routing Hints:

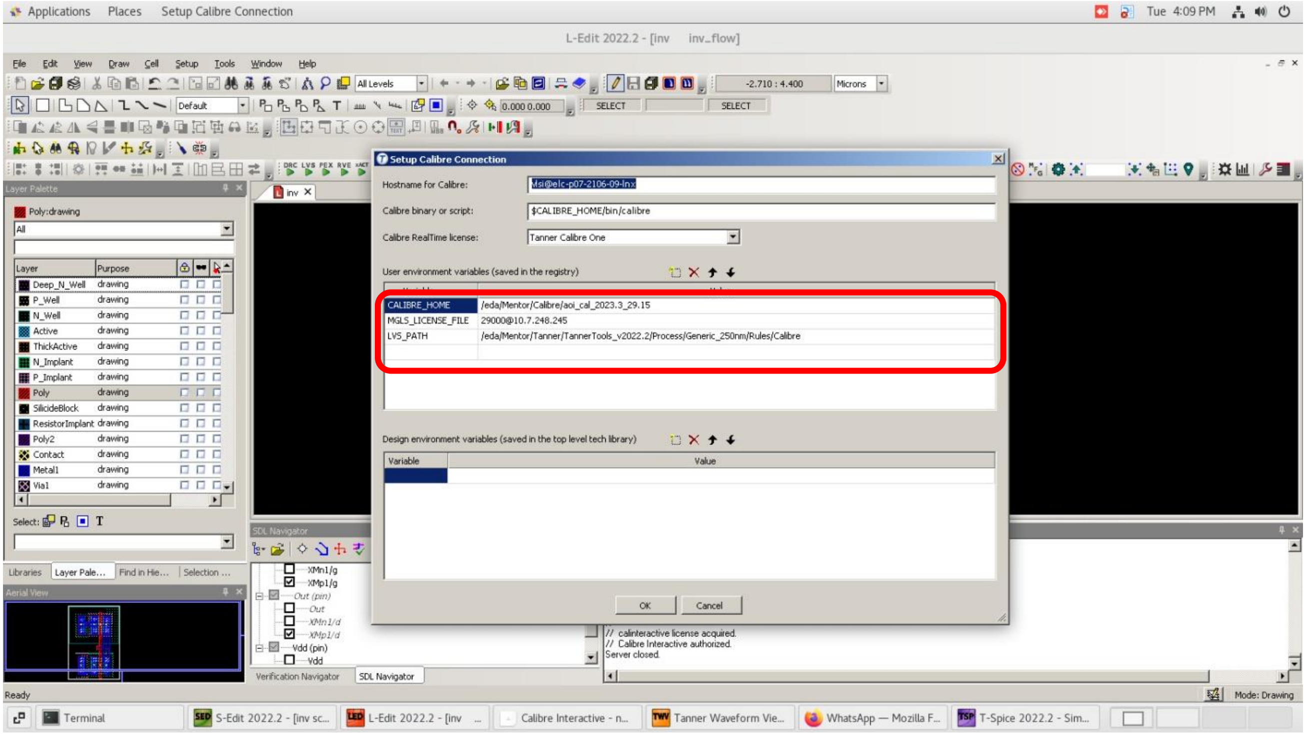
- Connect the output of the master latch with the input of the slave latch.
- Connect the CLK of the master latch with the CLK_bar of the slave latch.
- Connect the CLK_bar of the master latch with the CLK of the slave latch.

  Hint: you can use metal 3 for those connections

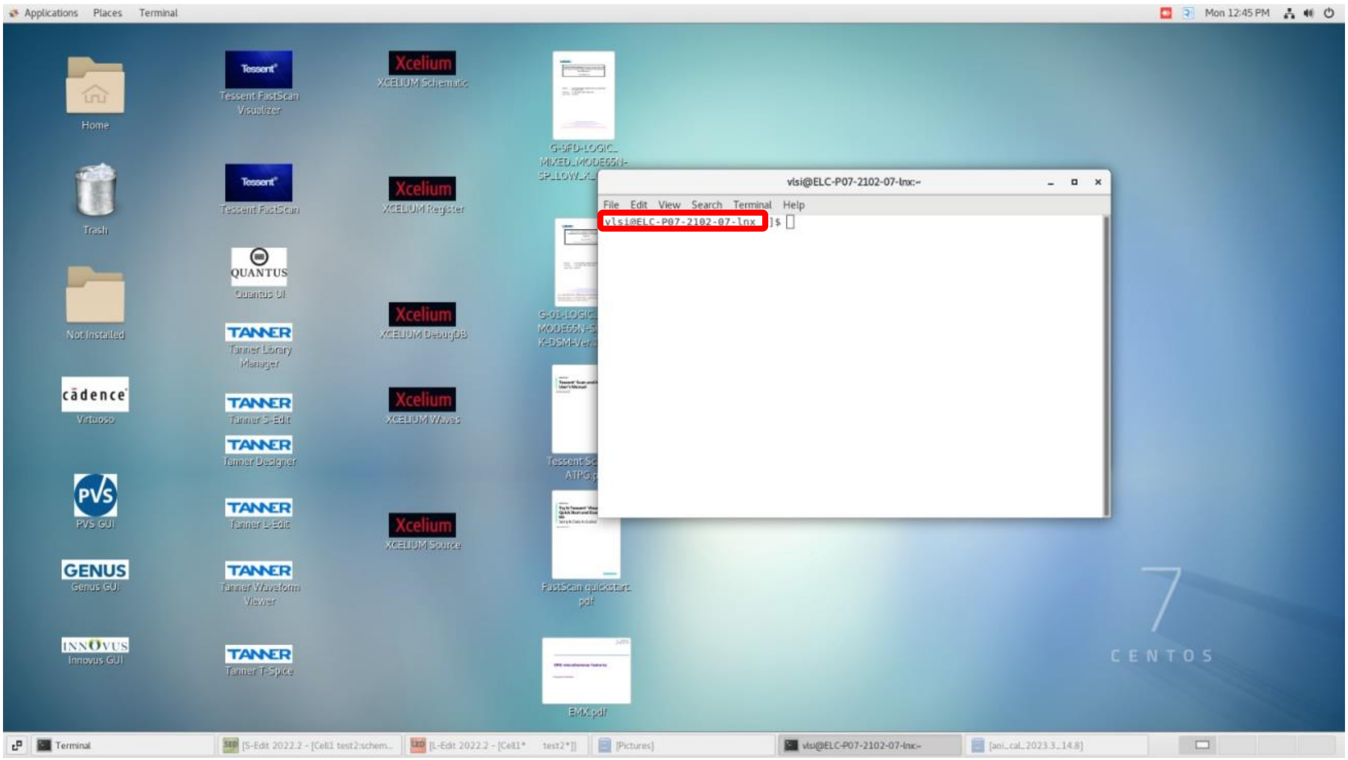
- Connect the common Vdd of the master latch with the common Vdd of the slave latch.
- Connect the common Gnd of the master latch with the common Gnd of the slave latch.

Calibre Setup

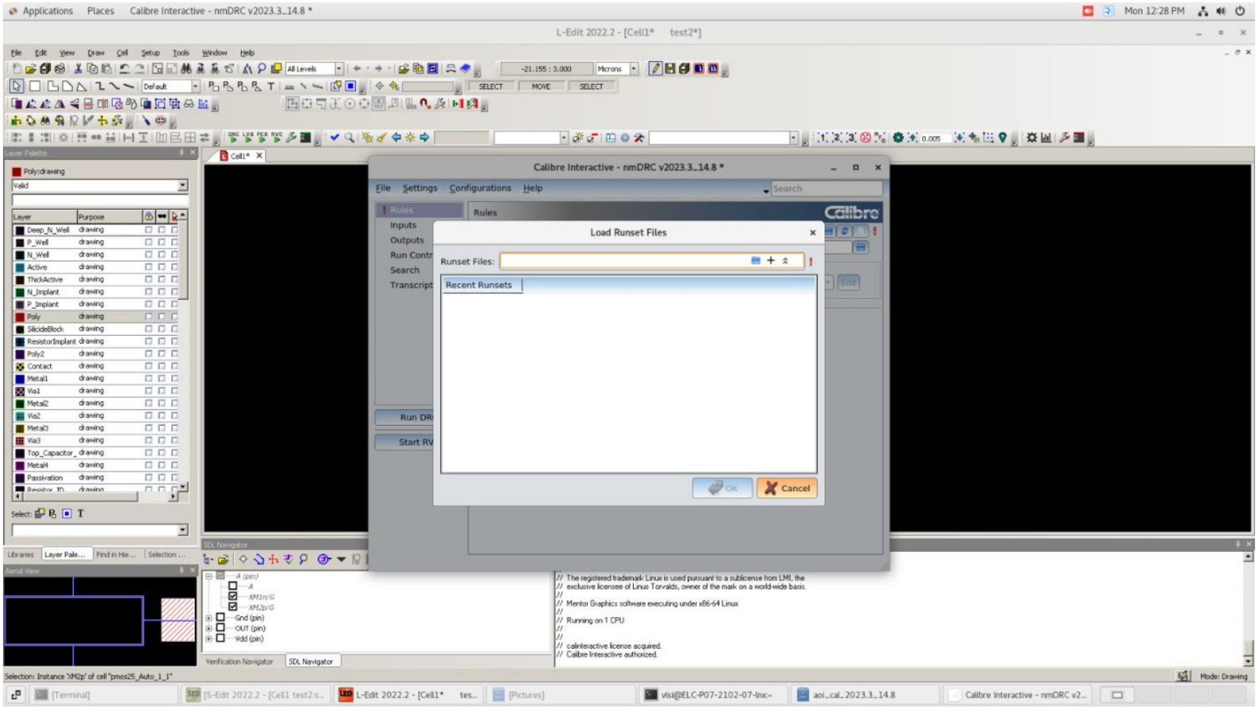
Setup Calibre connection **VERY IMPORTANT STEP**:



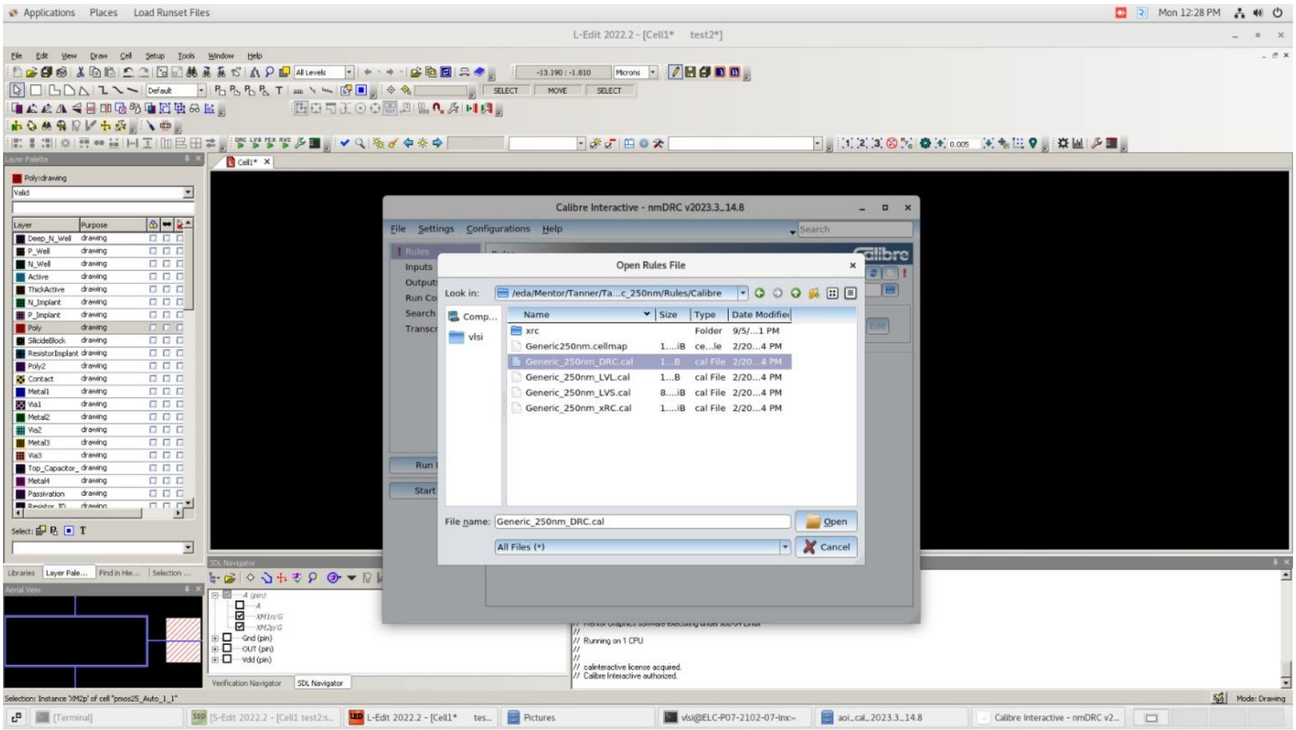
- Note: write **MGLS_LISENCE_FILE** instead of **MGLS_LICENSE_FILE.**
- Note: you can find Hostname for Calibre from your tanner location just open it in the terminal of the device.



DRC Setup

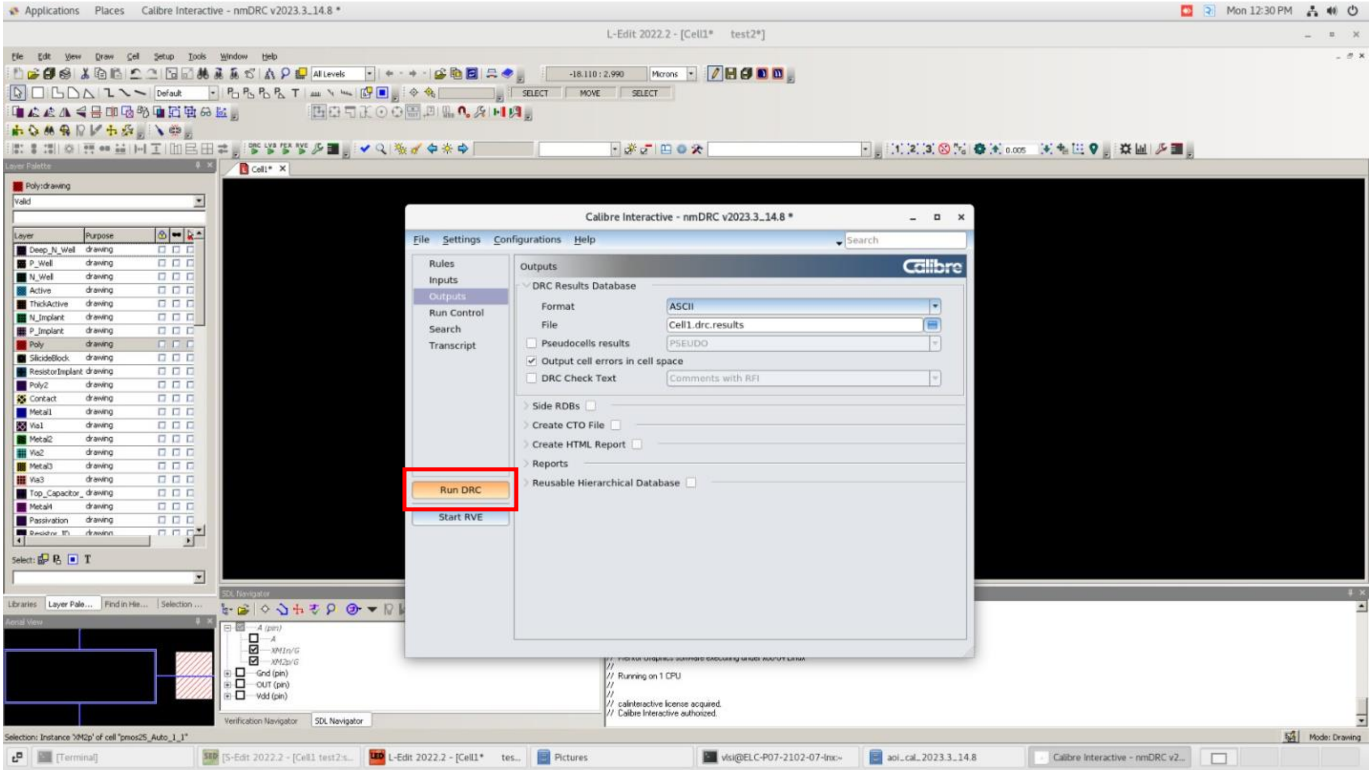


Cancel the first window appears.



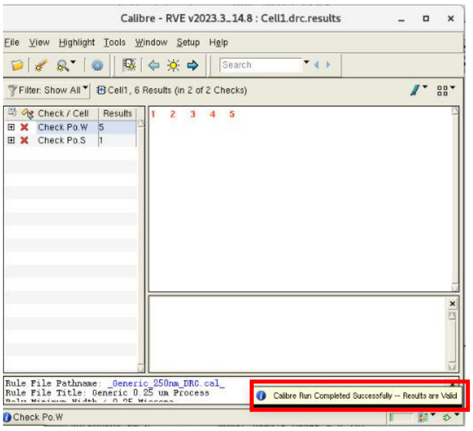
Add DRC library from Process file.
“/eda/Mentor/Tanner/TannerTools_v2022.2/Process/Generic_250nm/Rules/Calibre/Generic_250nm_DRC.cal”

Use default setting ……



Run DRC

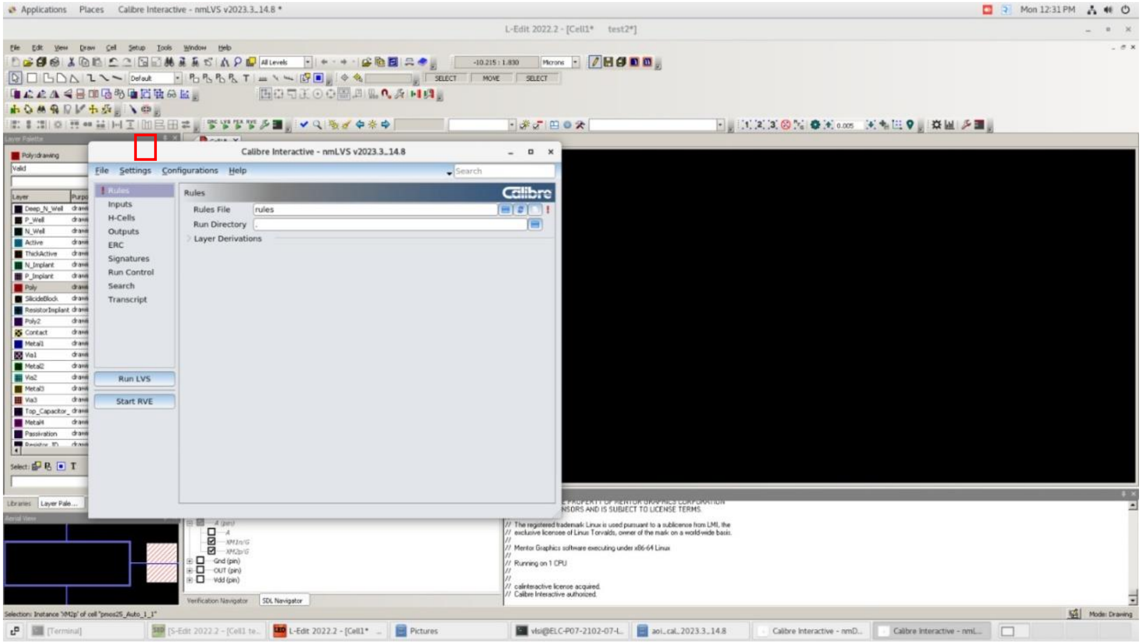
Result view (RVE) window will open



Fix all errors & **DRC** is Done Now ☺

<u>LVS SETUP</u>

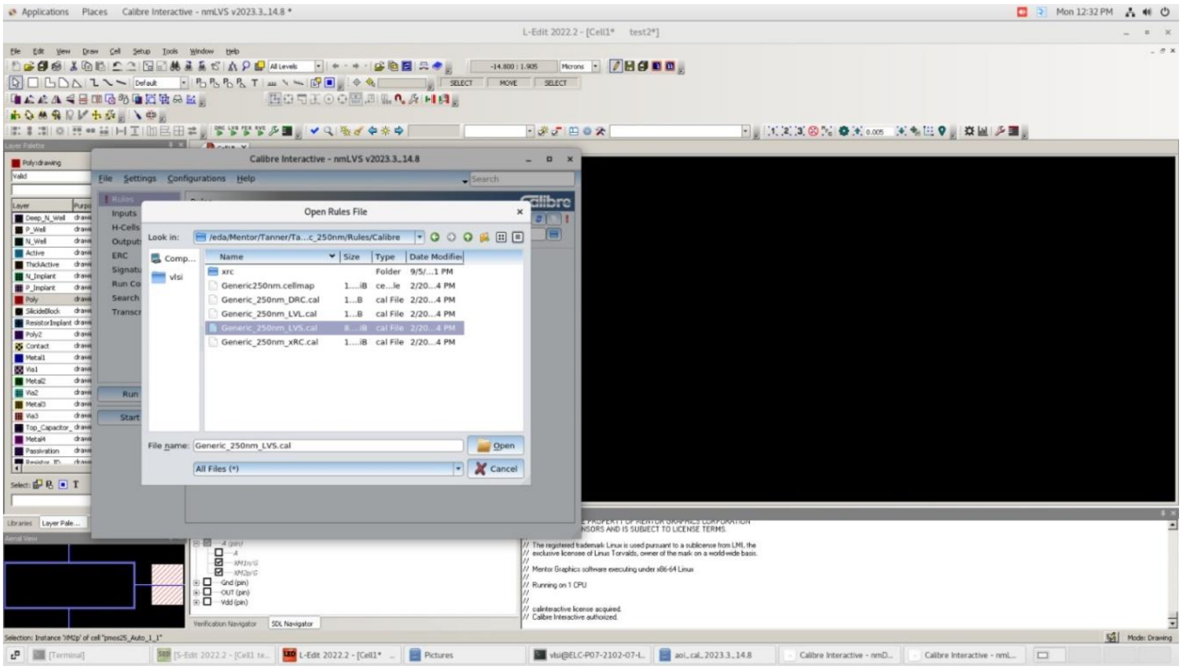
Run LVS



Cancel the first window appears.

Then add LVS library from Process file:

"/eda/Mentor/Tanner/TannerTools_v2022.2/Process/Generic_250nm/Rules/Calibre/Generic_250nm_LVS.cal"



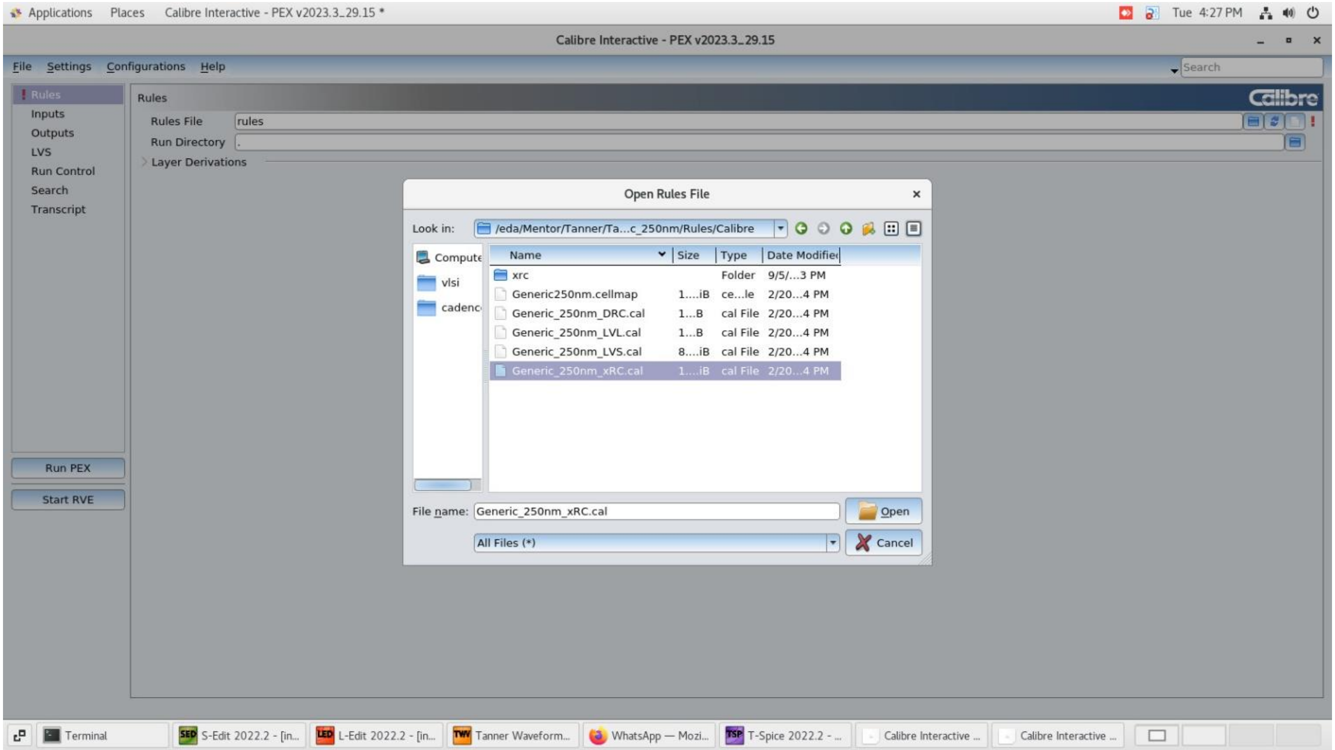
Run LVS

Fix all errors & LVS is Done Now ☺

Extraction SETUP

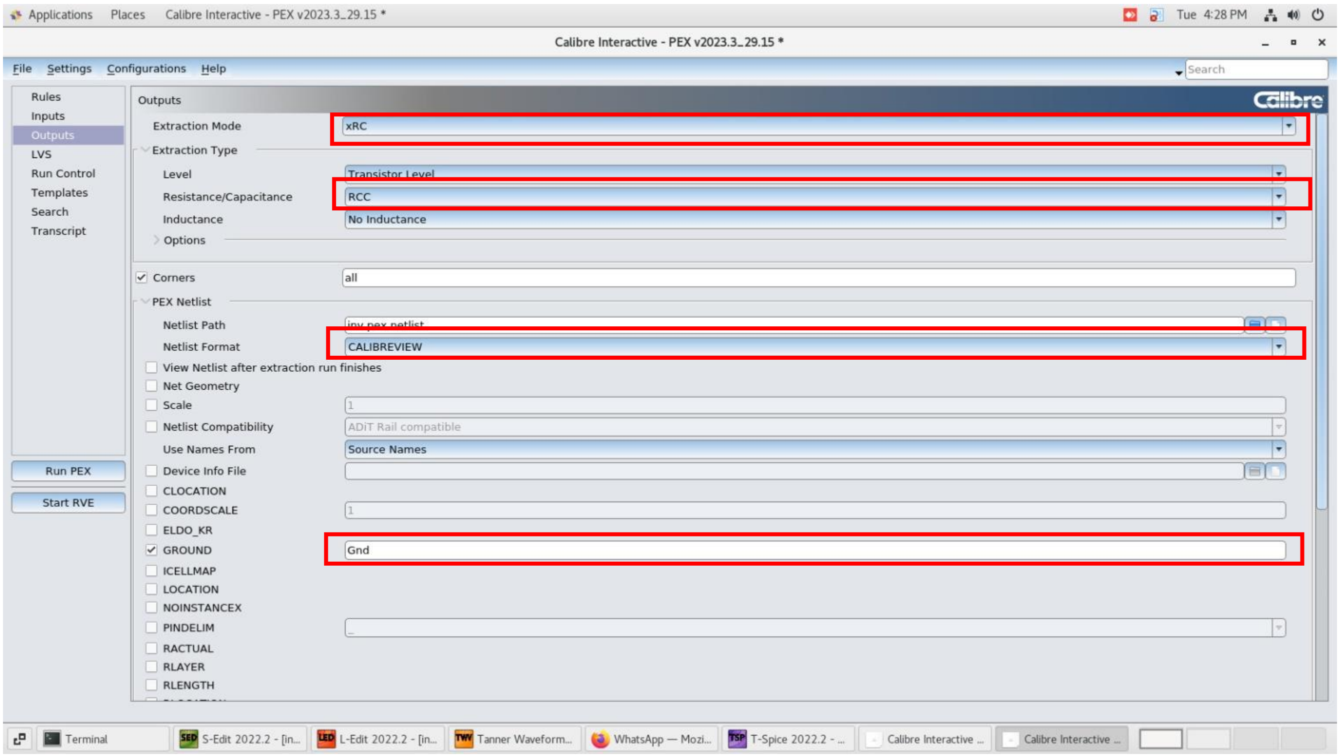
Run **PEX**

Cancel the first window appears.

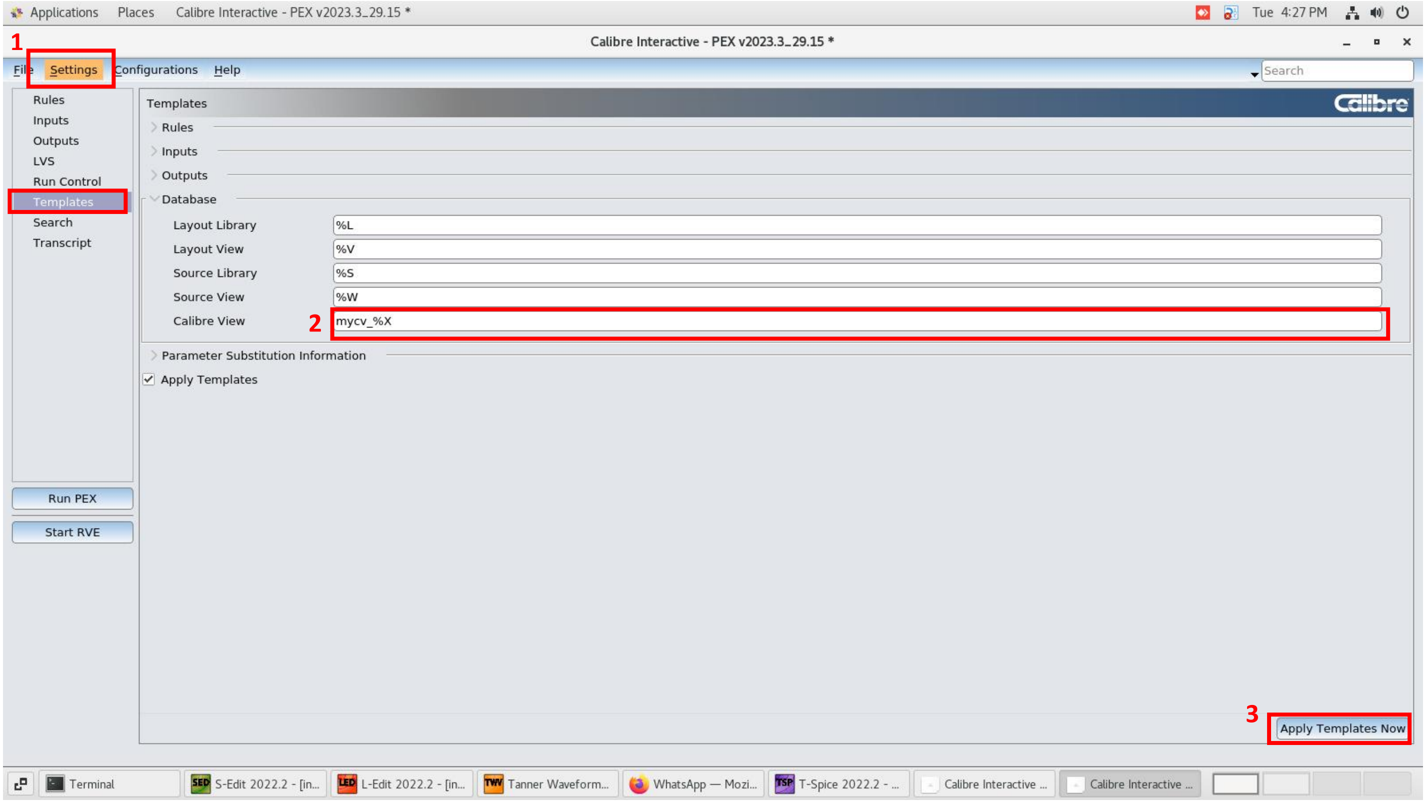


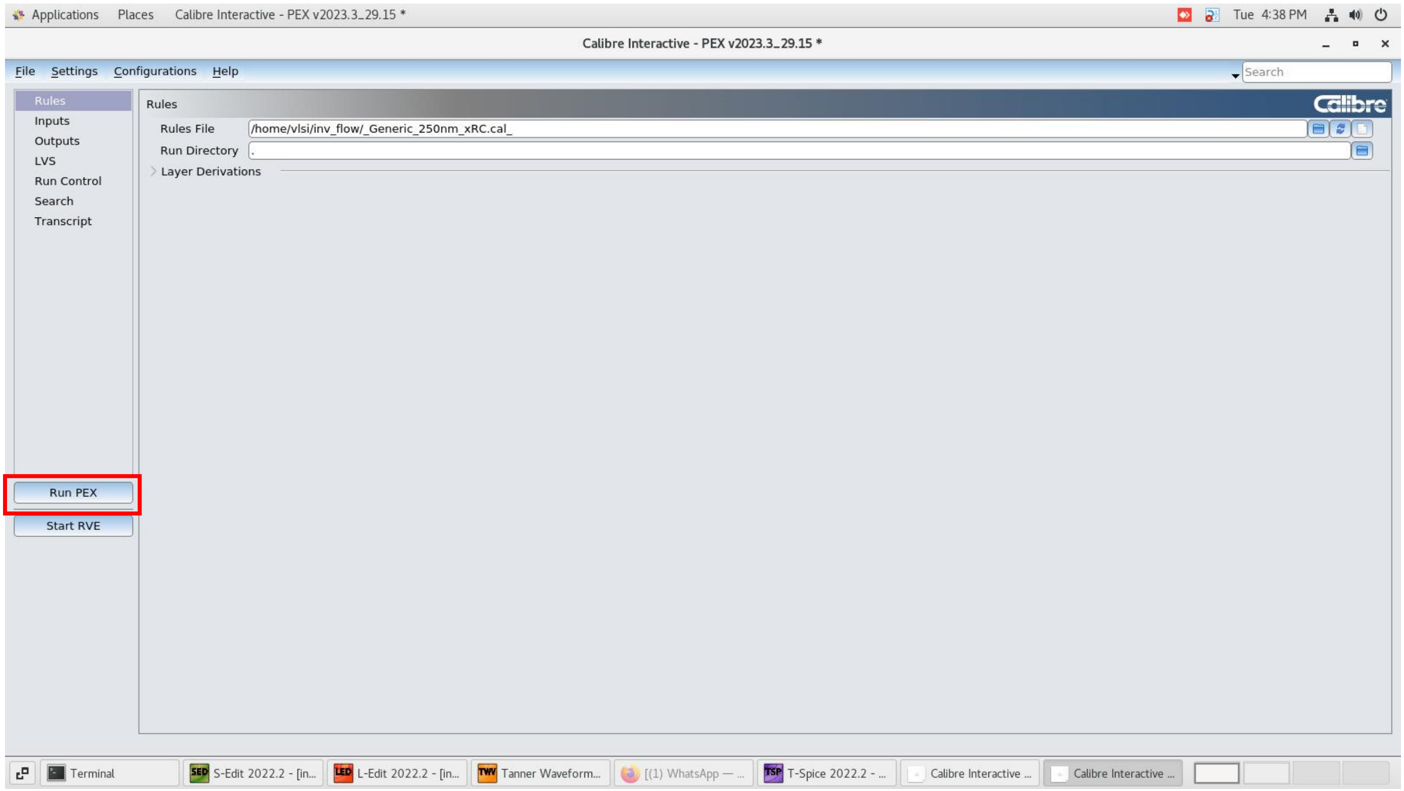
Then add PEX library from Process file.

"/eda/Mentor/Tanner/TannerTools_v2022.2/Process/Generic_250nm/Rules/Calibre/Generic_250nm_xRC.cal"



If Templates not appears show it from setting >> show pages

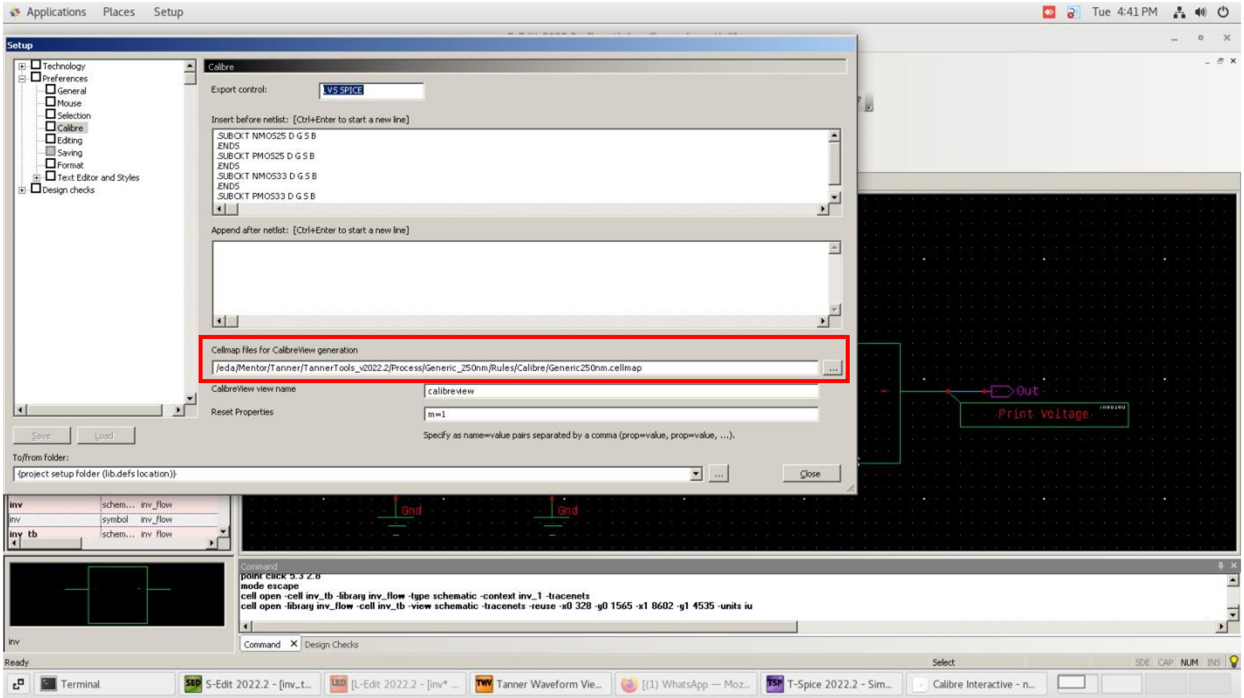




Post Layout Simulation

Open S-edit

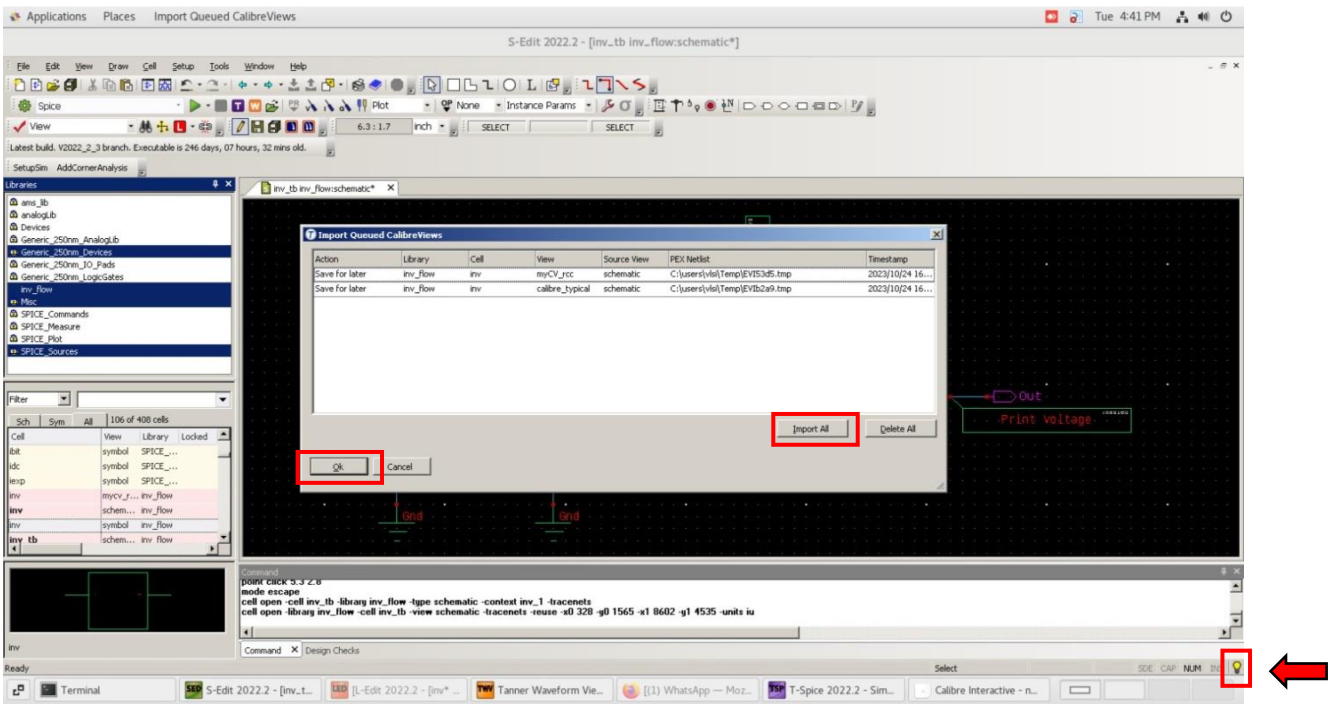
Setup >> Preferences >> General >> Calibre



Remove the old cellmap file and add the new path:

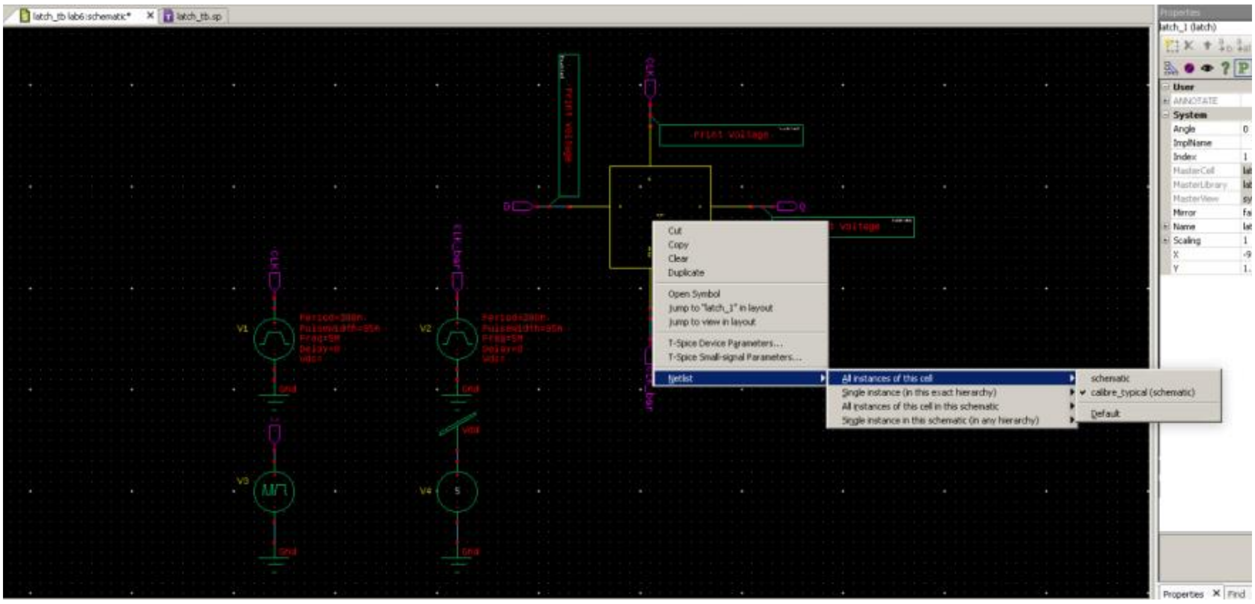
“/eda/Mentor/Tanner/TannerTools_v2022.2/Process/Generic_250nm/Rules/Calibre/ Generic_250nm.cellmap”

Press the **lamp** in the **bottom right**, import calibre view and press ok



Now right click in your symbol (long click) and choose….

Netlist >> All instance of this cell >> Choose schematic or mycv_rcc.



Now open your extracted view contains all parasitics and add Vdd & Gnd symbol then save and r-simulate .