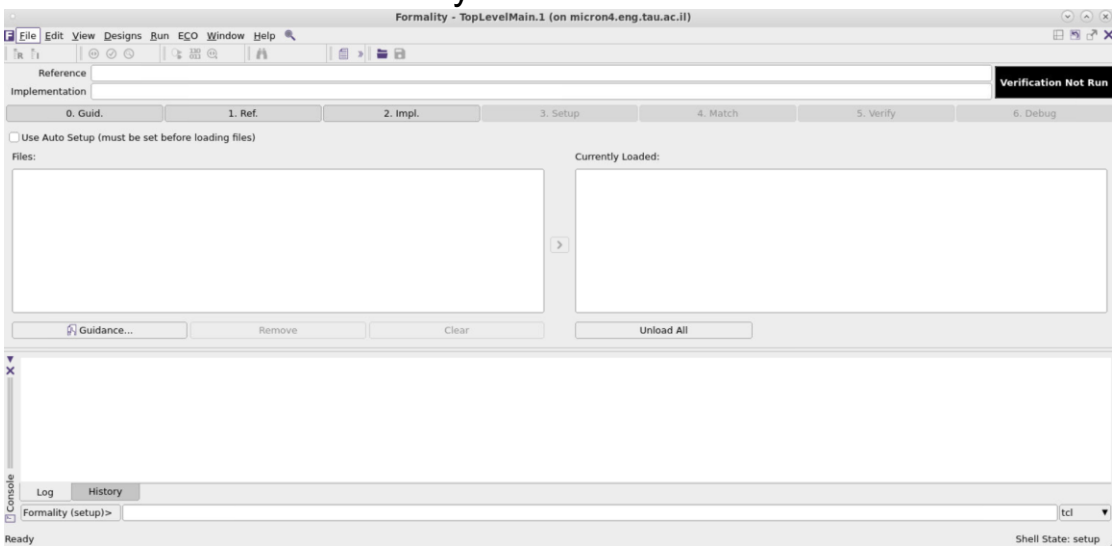
Guide for formality from Synopsys

formality refers to the process of formal verification, which ensures that two representations of a design (e.g., RTL vs. gate-level netlist) are functionally equivalent without requiring simulation.

first go the work directory, for example :

```
RISCV1.dlib/ RISCV.dlib/
[foqara@micron-x01 ws]$ cd riscv
```

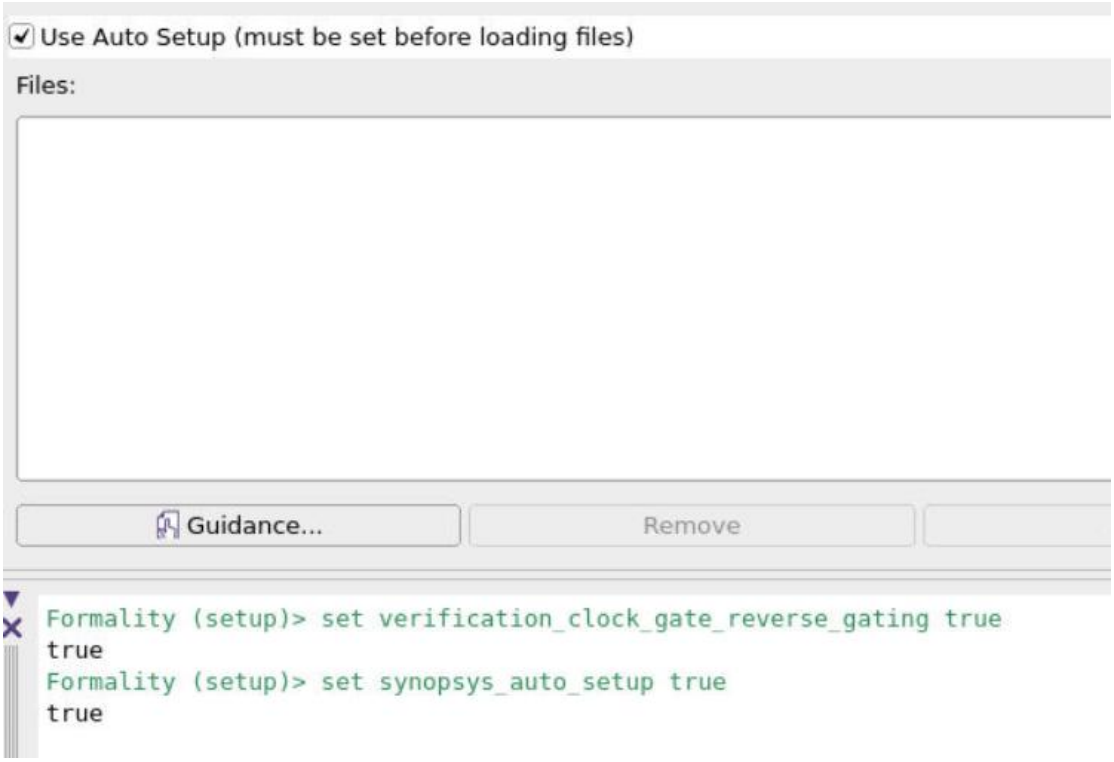
After that, write in the terminal : fm_shell -gui .
this line will run the formality :



Now we should to chose the clock gating that we have :
we use :
set verification_clock_gate_reverse_gating true

there are also :
set verification_clock_gate_hold_mode
set verification_clock_gate_edge_analysis

```
Formality (setup)> set verification_clock_gate_reverse_gating true
true
```

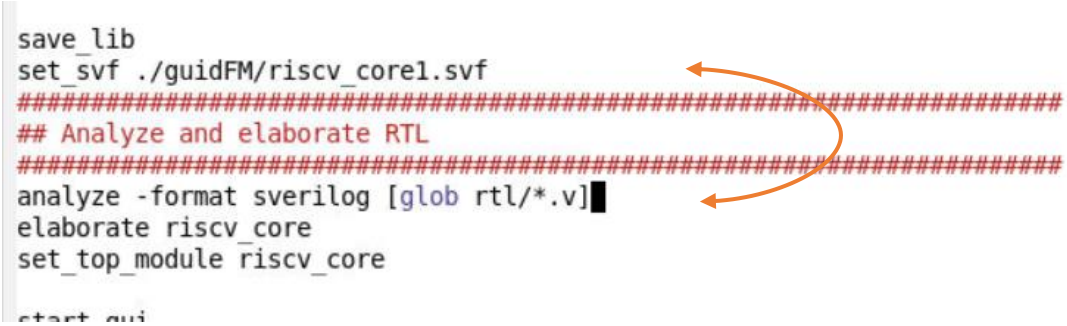


Click on a “Auto setup”

now we need the Guidance file that tell the tool what cells we have and It helps Formality overcome potential mismatches and verification challenges by providing hints, constraints, or directives
.
so to build this file you can use :
set_svf ./{WORK DIR}/{name}.svf

for example we use :
set_svf ./guidFM/riscv_core1.svf

** you should to put this line before the analyze of the rtl



We need this file to compare between the original rtl and the netlist after the syntheses, so we will save and close the file after the syntheses function.

but before it we will make the netlist file that we will use to compare :

file mkdir ./netlist

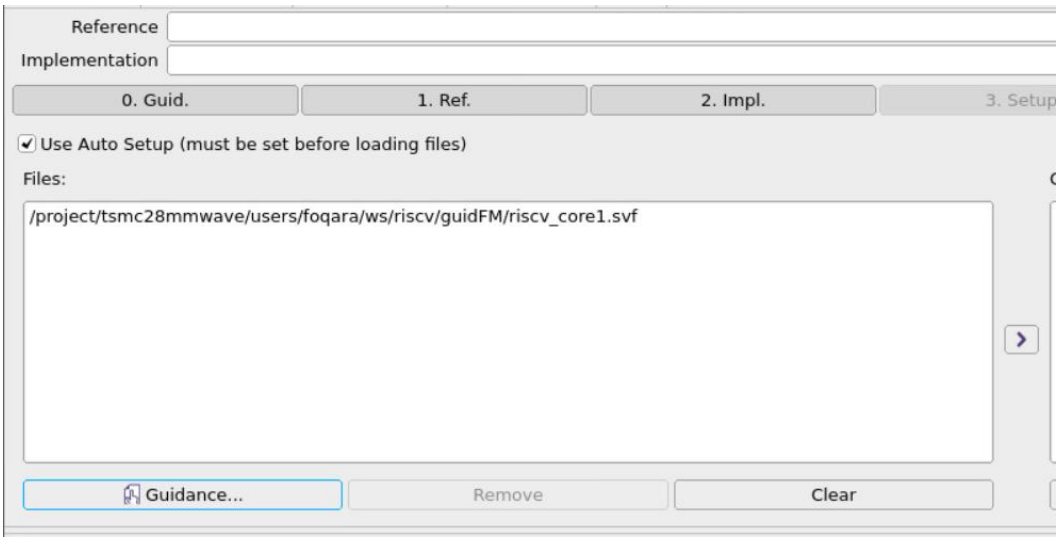
write_verilog -hierarchy all ./netlist/riscv_core_syn.v

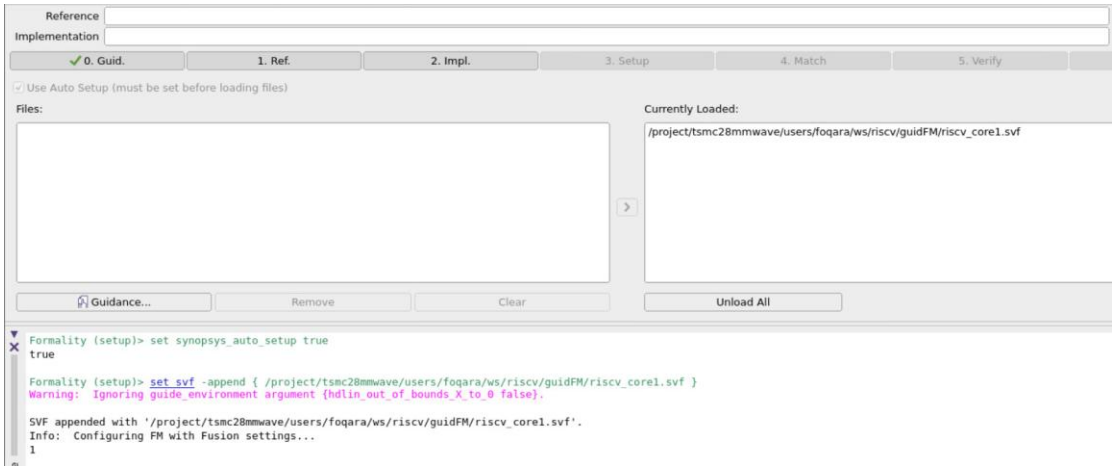
and to save the svf file :
set_svf -off

```
# final_opto stage
compile_fusion -from final_opto -to final_opto

check_legality

##########################################################
## Export Netlist for Formality
##########################################################
file mkdir ./netlist
write_verilog -hierarchy all ./netlist/riscv_core_syn.v

##########################################################
## Save SVF for Formality
##########################################################
set_svf -off
##########################################################
```

so now we have the netlist file and the guidance.

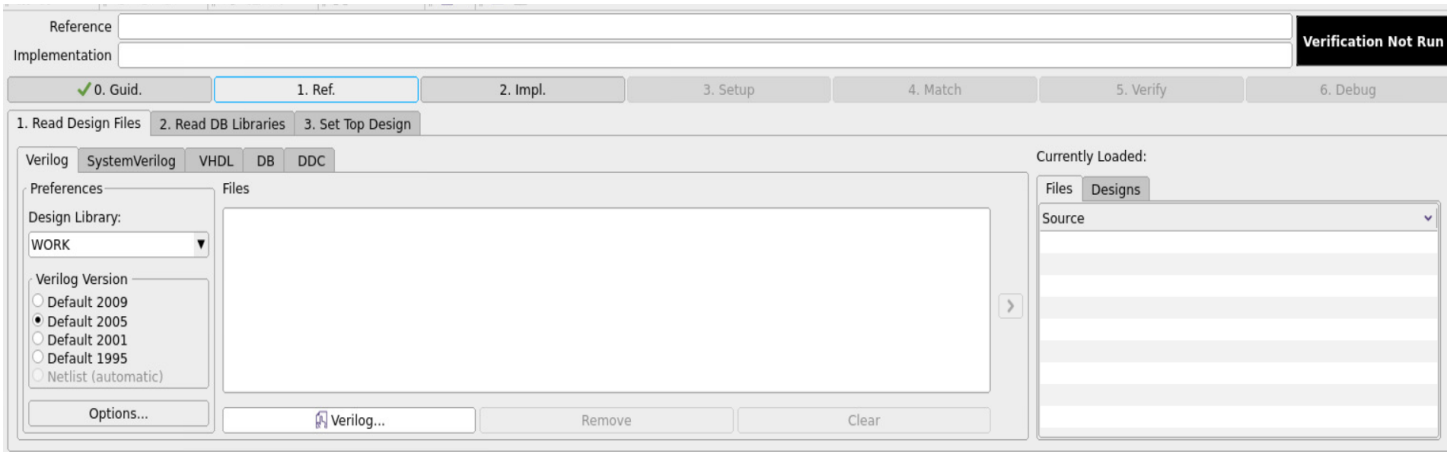
upload your guidance file in stage 0 :



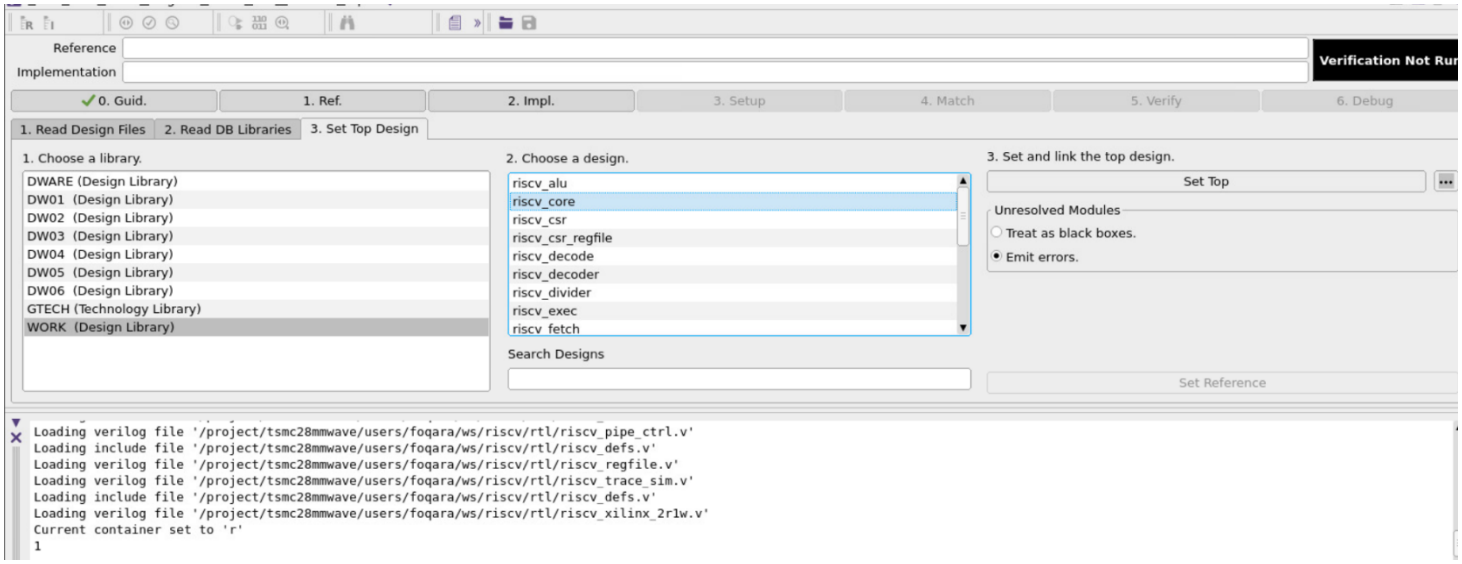


stage 1 :

Upload the rtl code .

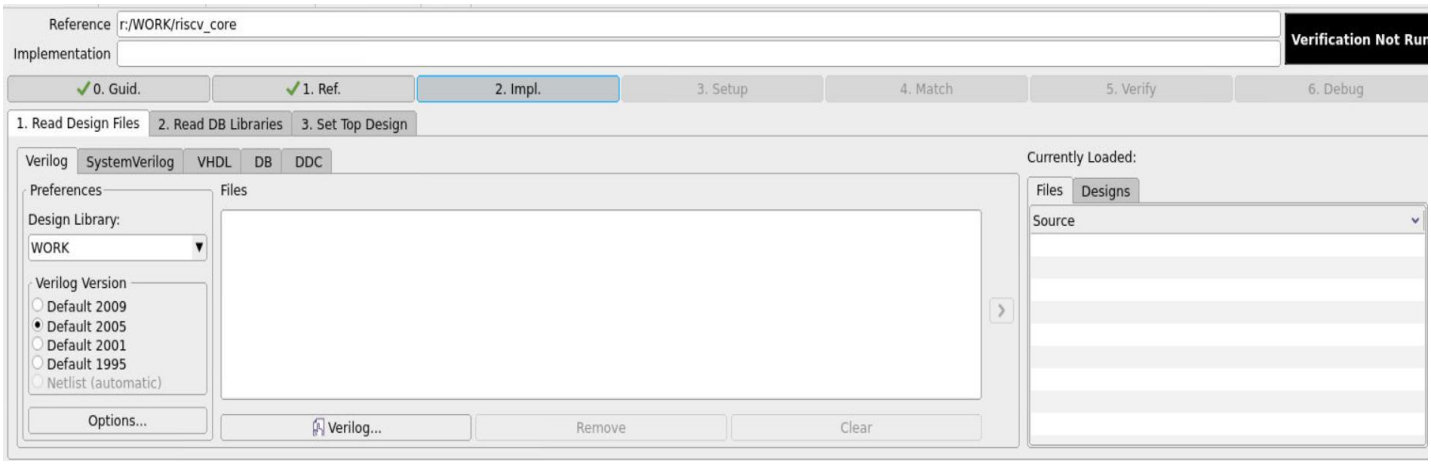


Set your top module.

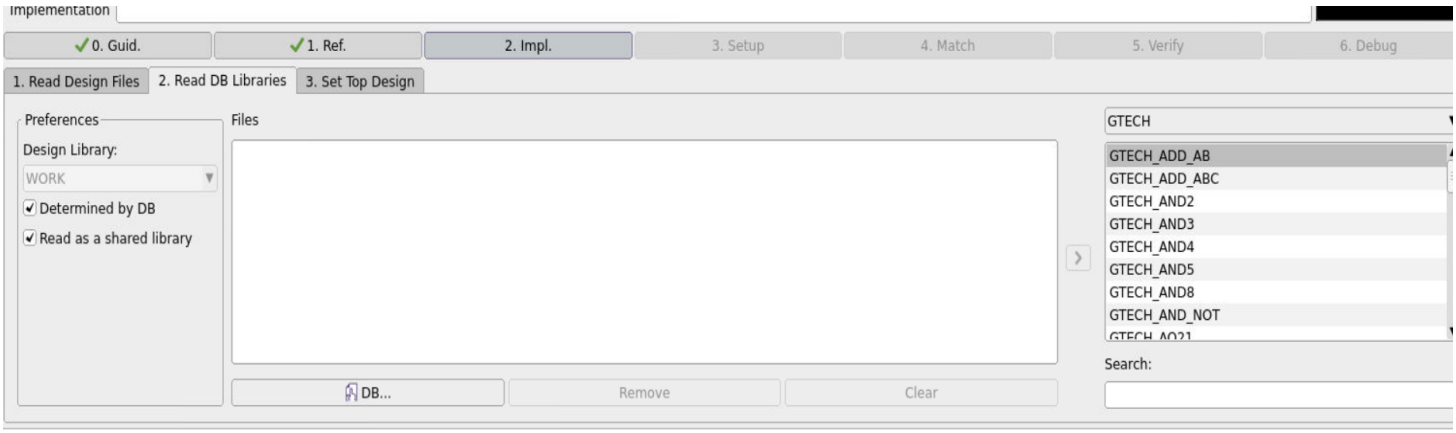


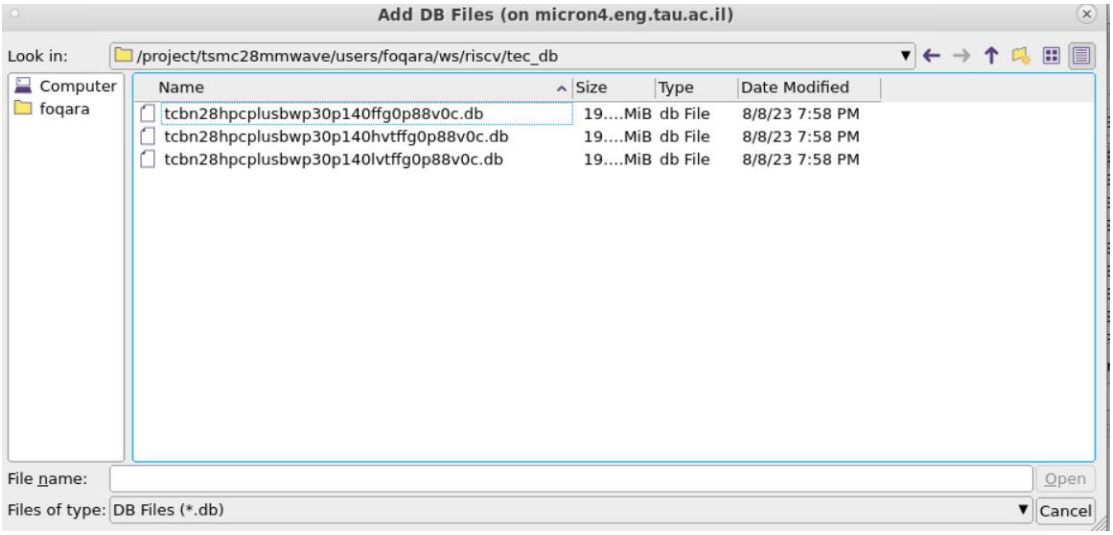
Stage 2 :

Upload the netlist file .

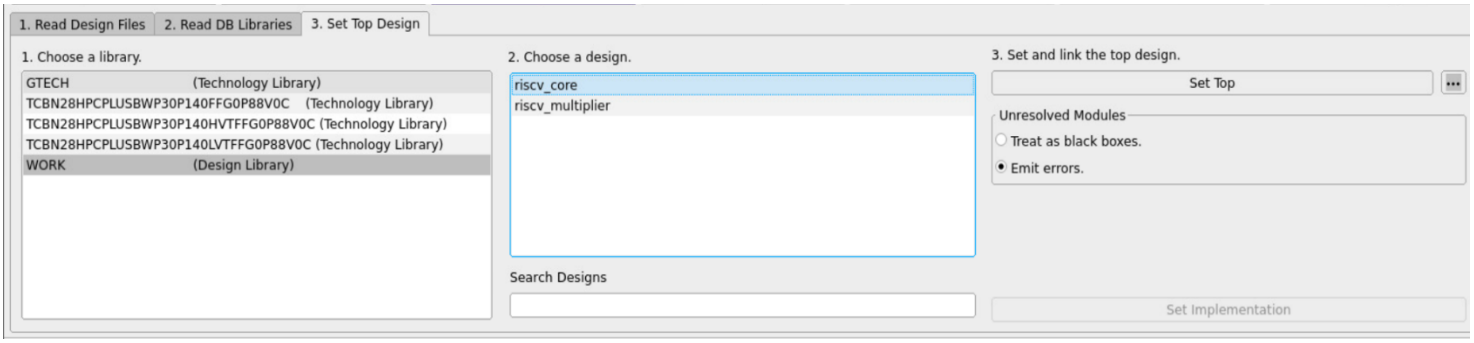


Put the DB files :



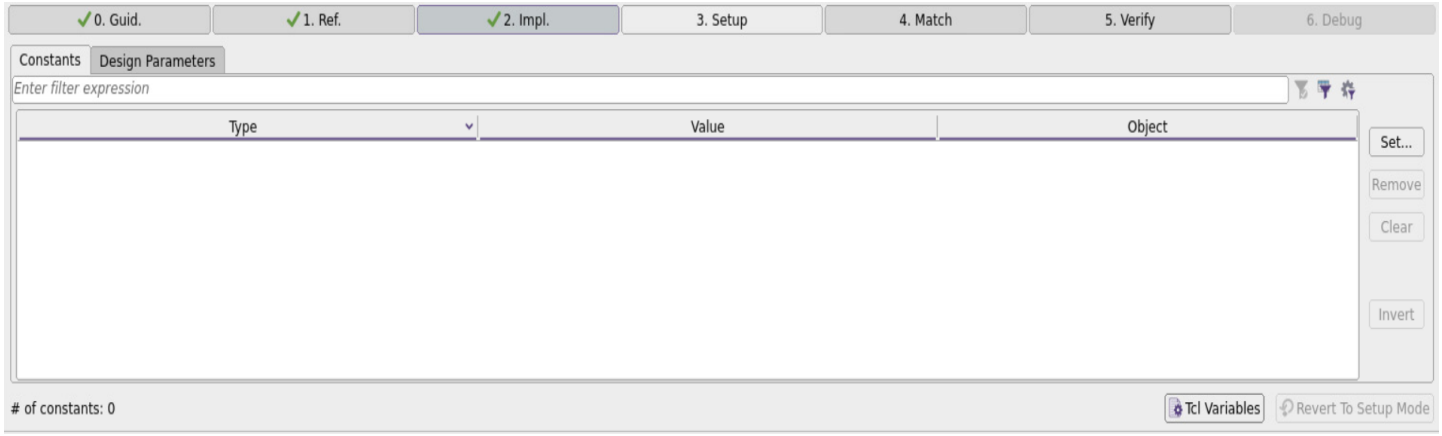


Set the top design :



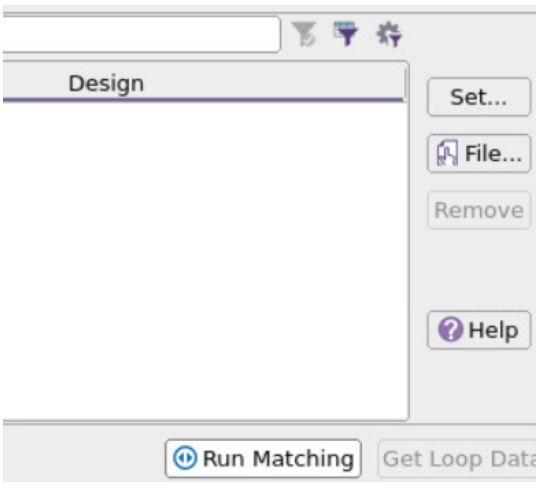
Stage 3 :

This stage can help us to define the constrains in the DFT if we use constrains there, we do formality after syntheses so no to define any thing.

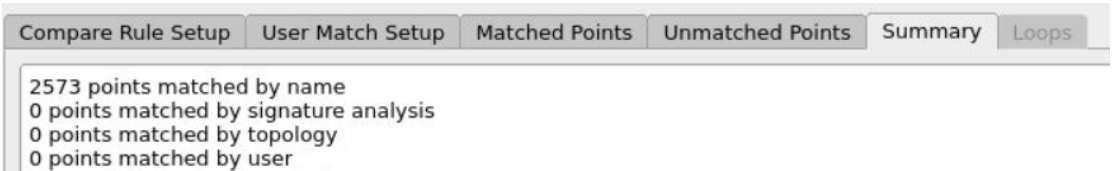


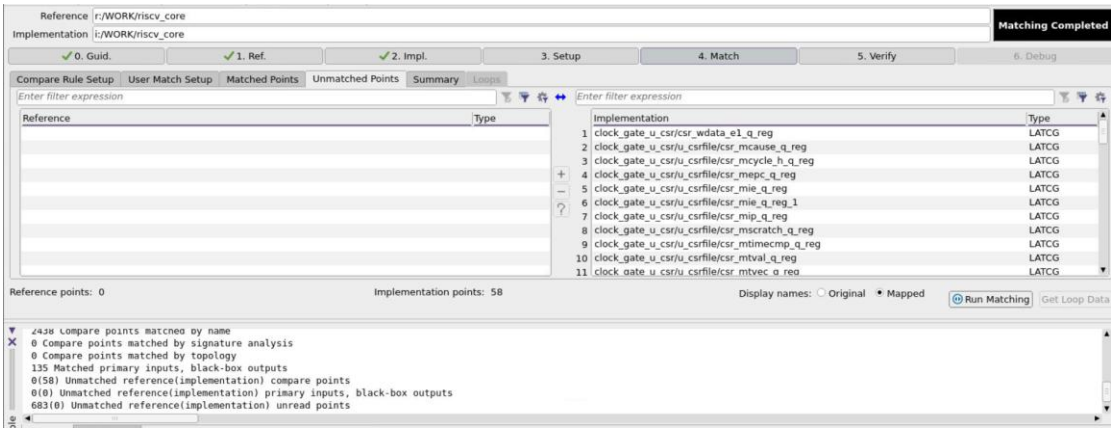
Stage 4 :

Run matching :



In stage 4 you have summary of the matching and unmatched and unmatched points :

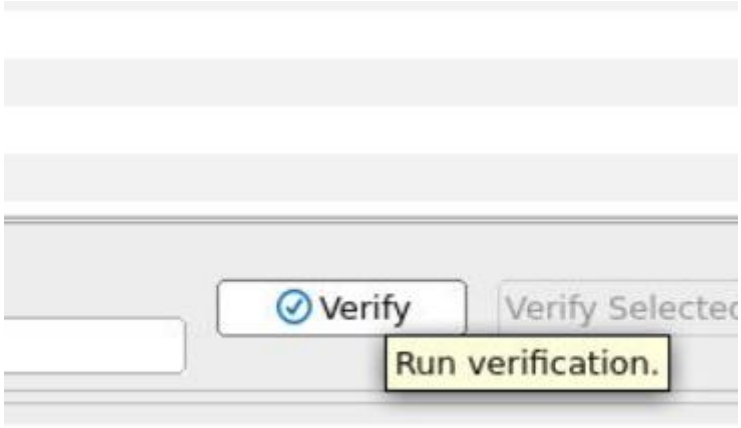


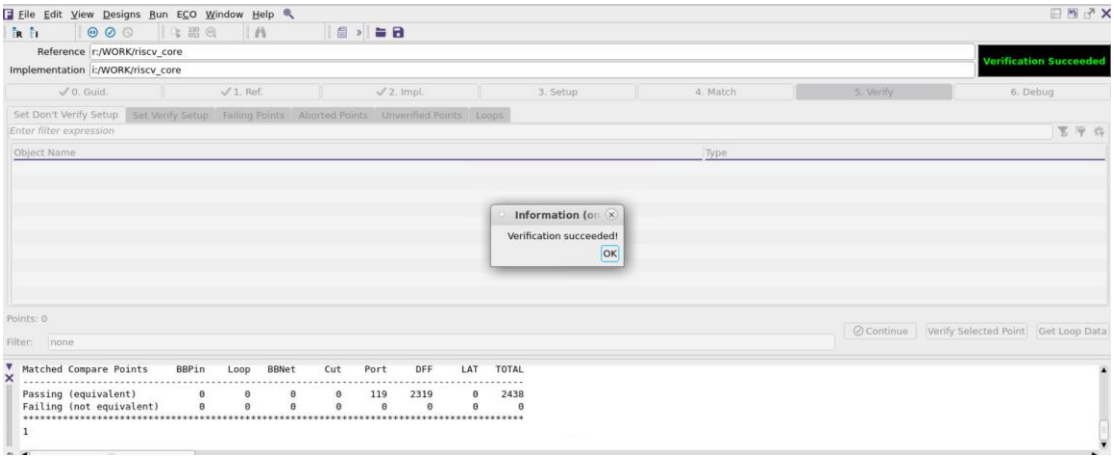


So we see that the unmatched points its clock_gate, so we can move on because we set in stage 0 line that will solve that in the verification .

Stage 5:

Click on verify .

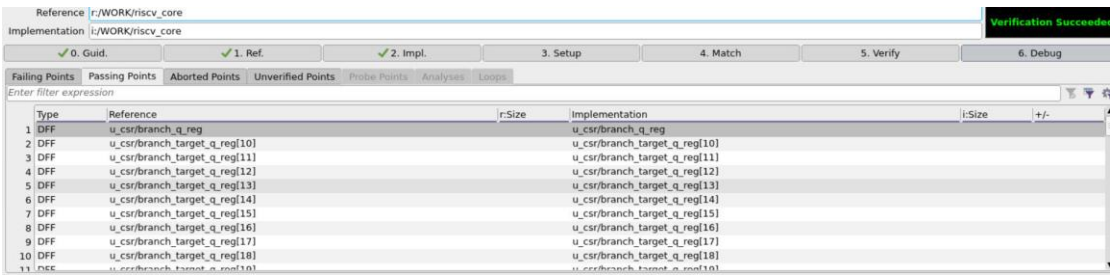




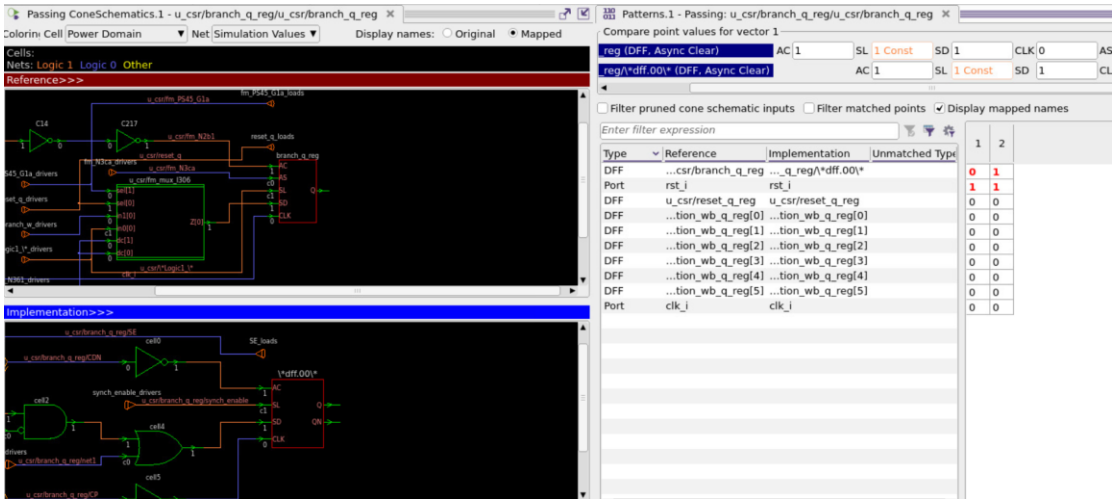
Now we have it!!!

Stage 6 :

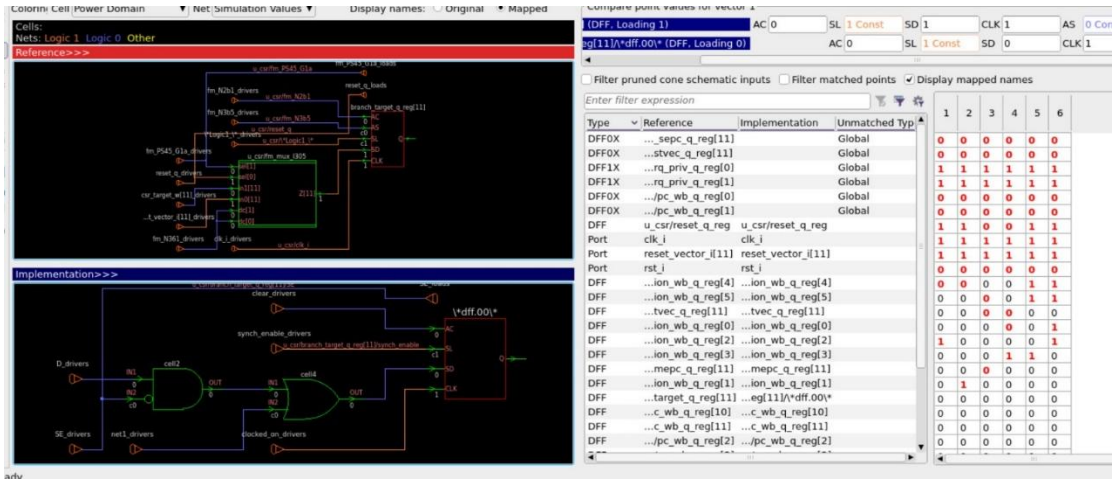
Stage 6 is a debug, that have many tools that can help you to know where the error.



If you click twice in any component you will have another screen that show how the design look in the original rtl and in the netlist after the syntheses.



If you have a global unmatched type that mean that you build the svf file wrong.



You can also to expand the blocks .

This drive link have many files from synopses that explain how to run your formality and how to fix advanced problem.

https://drive.google.com/drive/folders/1S8u4vYmcDT1XTIuADV_w59Q-vAEbbIfA?usp=sharing