PSS algorithm:

1.Get started: Begin the program execution.

2.Declare the required variables and take input streams.

3.Generate X-Reference sequence: Generate the X-Reference sequence based on the predefined logic.

4.Wait for the input stream (n_id value): Pause the program and wait for the user to input the value for n_id.

5.After receiving the n_id value, based on conditions generate the data.

6.After completion of data generation, send the data: Send the generated data to the intended recipient or location as required.

7.The sent data is our PSS sequence: Confirm that the data sent in the previous step is the desired PSS (Primary Synchronization Sequence) sequence.

SSS algorithm:

1. Get started: Begin the program execution.

2.Declare the required variables and take input streams.

3.Generate X_0-Reference and X_1-Reference sequences.Generate the X-Reference sequences based on the predefined logic.

4.Wait for the input stream (n_id value): Pause the program and wait for the user to input the valu for n_id.

5.After receiving the n_id value, based on conditions generate the data.

6.After completion of data generation, send the data: Send the generated data to the intended recipient or location as required.

7.The sent data is our SSS sequence: Confirm that the data sent in the previous step is the desired SSS (Secondary Synchronization Sequence) sequence.

PSS IMPLEMENTATION

1.Open vitis HLS then the window on figure 1 will be visible. Click create project and click on create project.Fill in the project name and the project location in their designated fields and then click next.

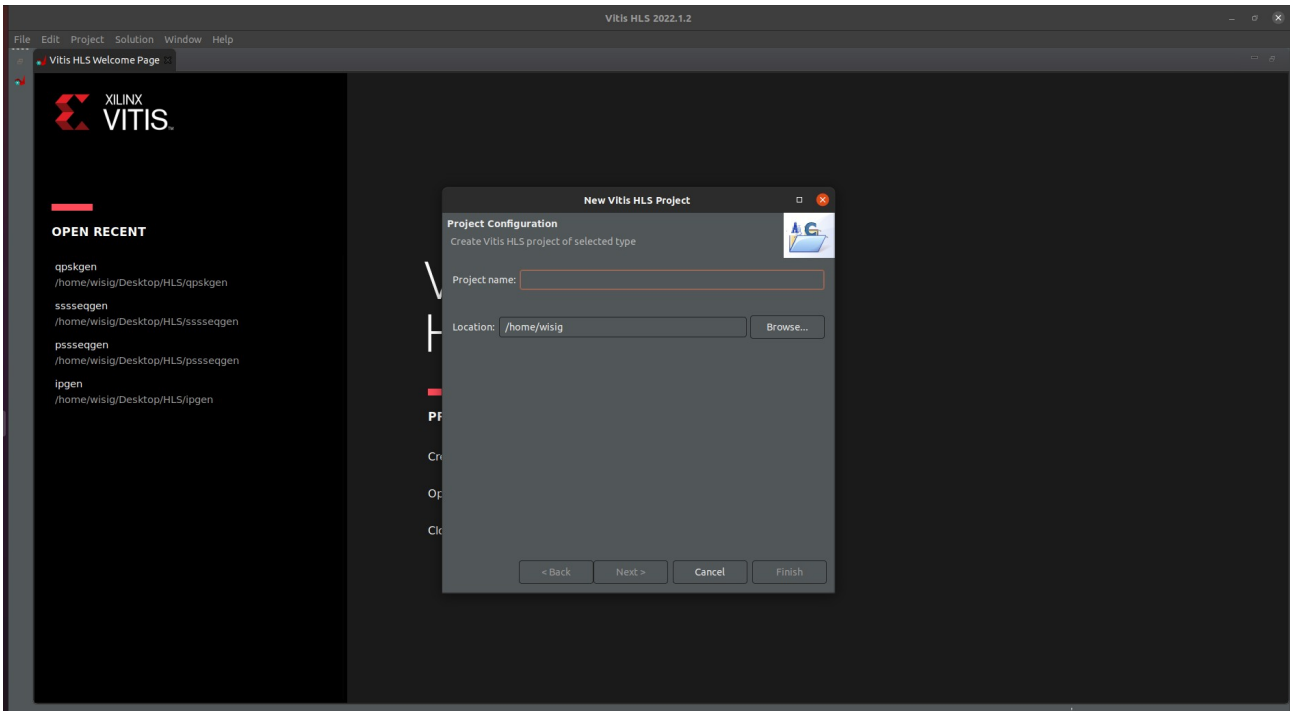
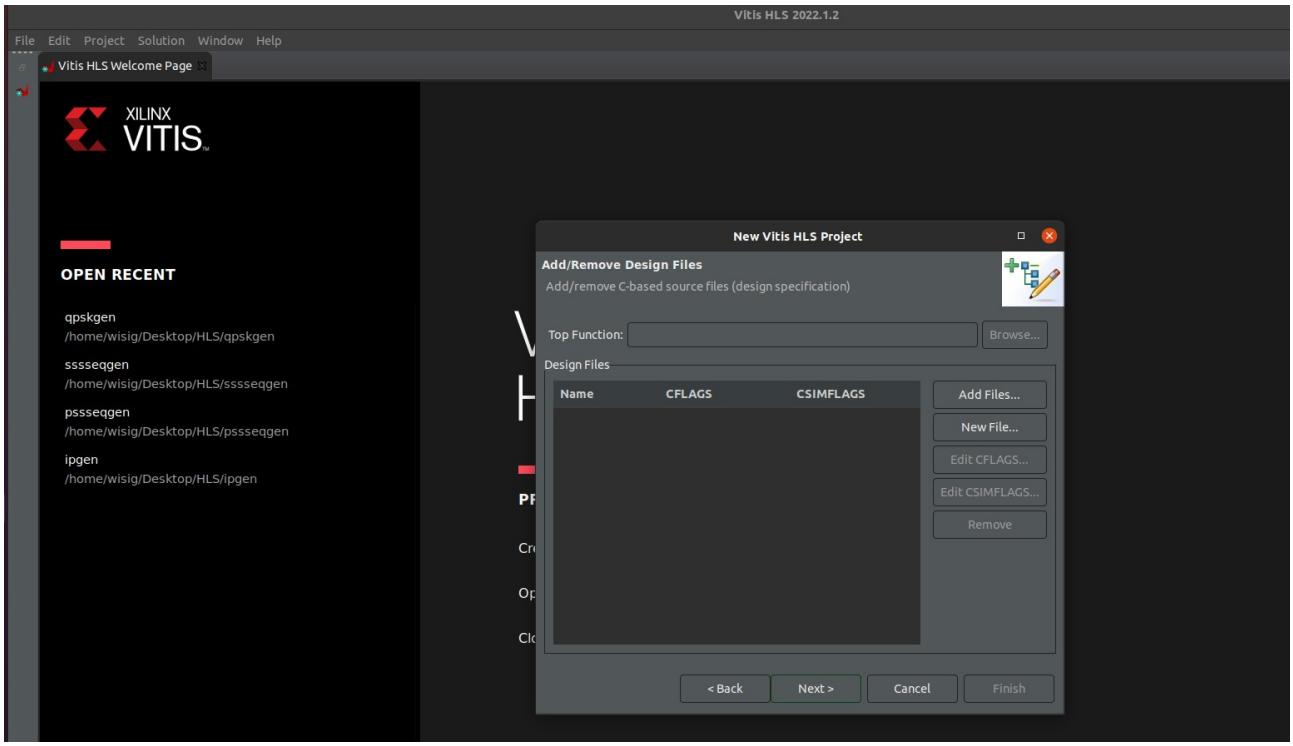
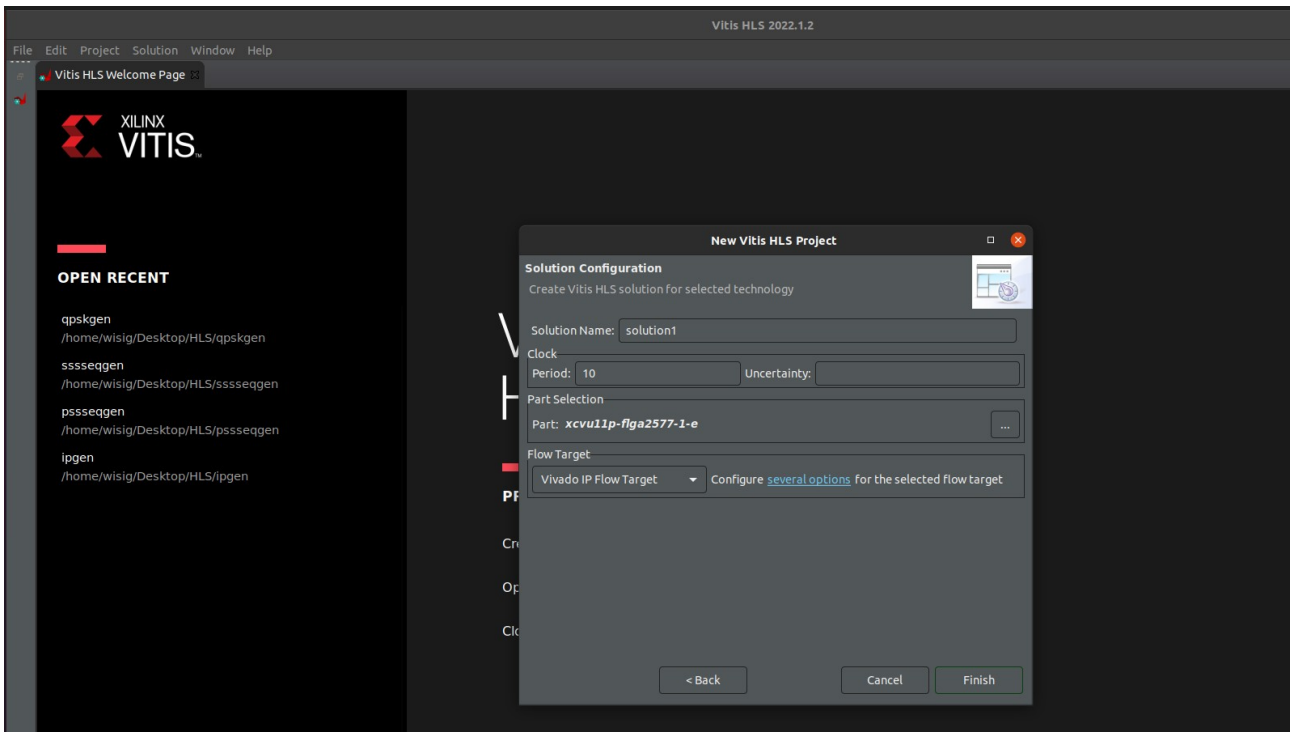


Figure1- vitis HLS opening.

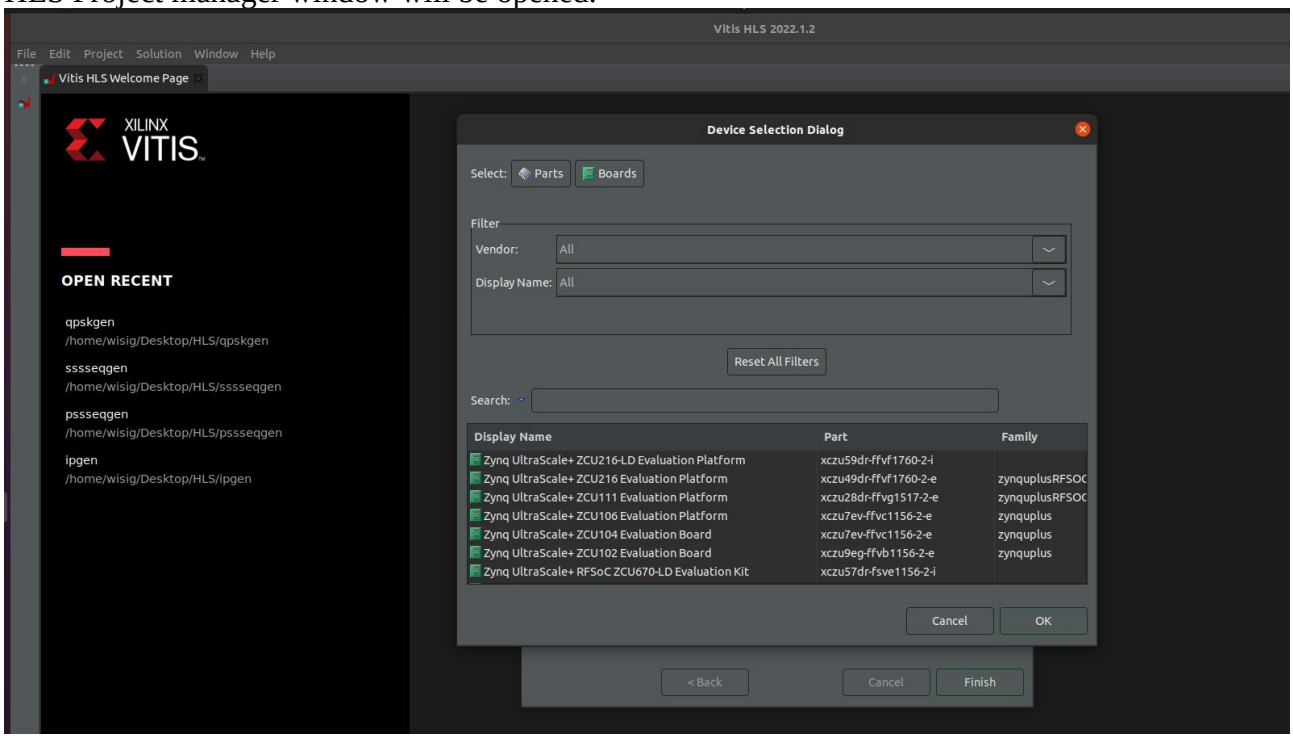
2. Upload your file (.cpp) by clicking on Addfiles option and also give your Top function and then click next.

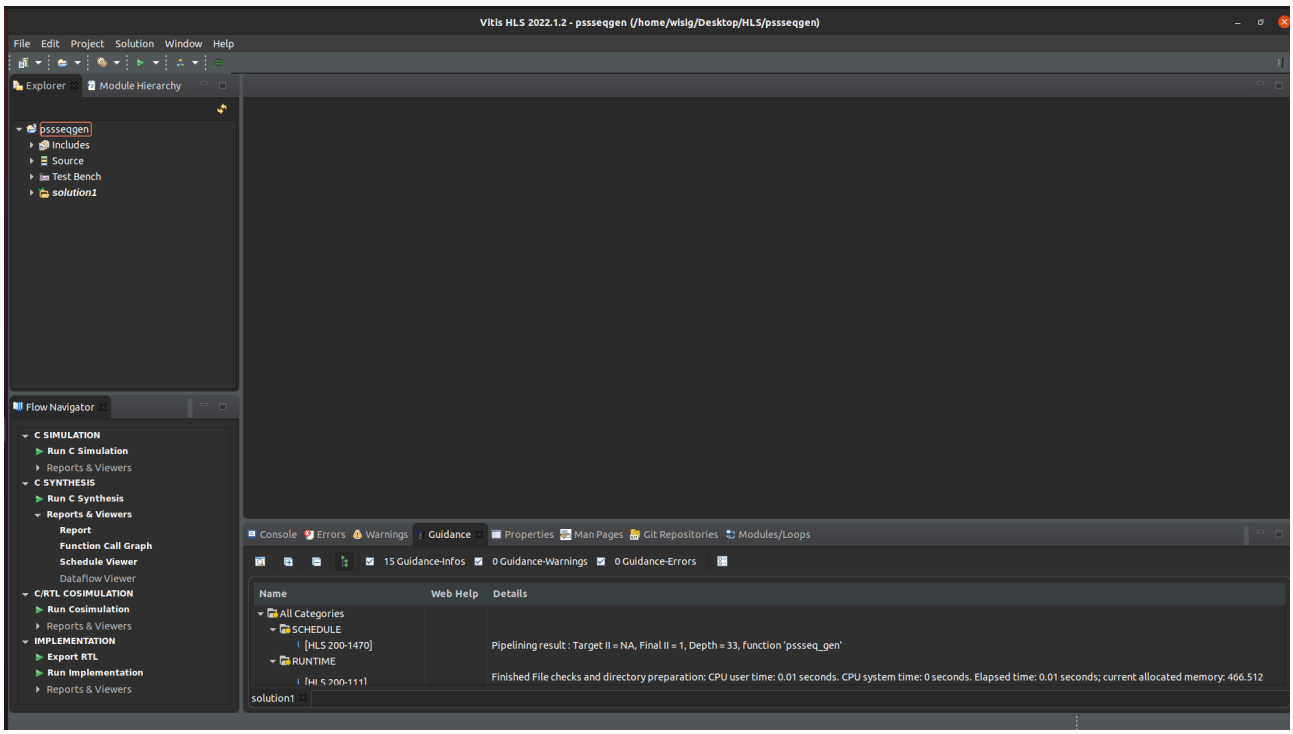


3. Give solution Name and clock period and click on part option.

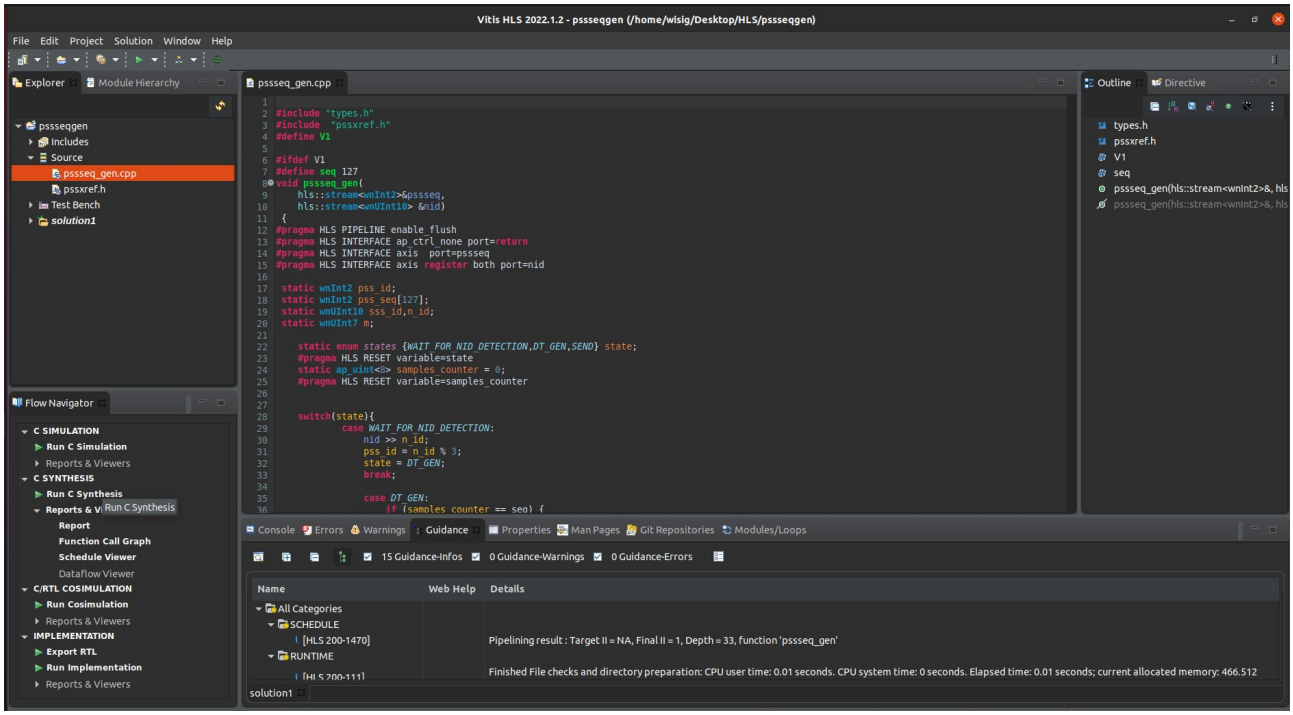


4.After clicking part option below dialog box will open. In that select the board you want to work on. Then click next and then click on finish. Now, a vitis HLS project has been created and the vitis HLS Project manager window will be opened.

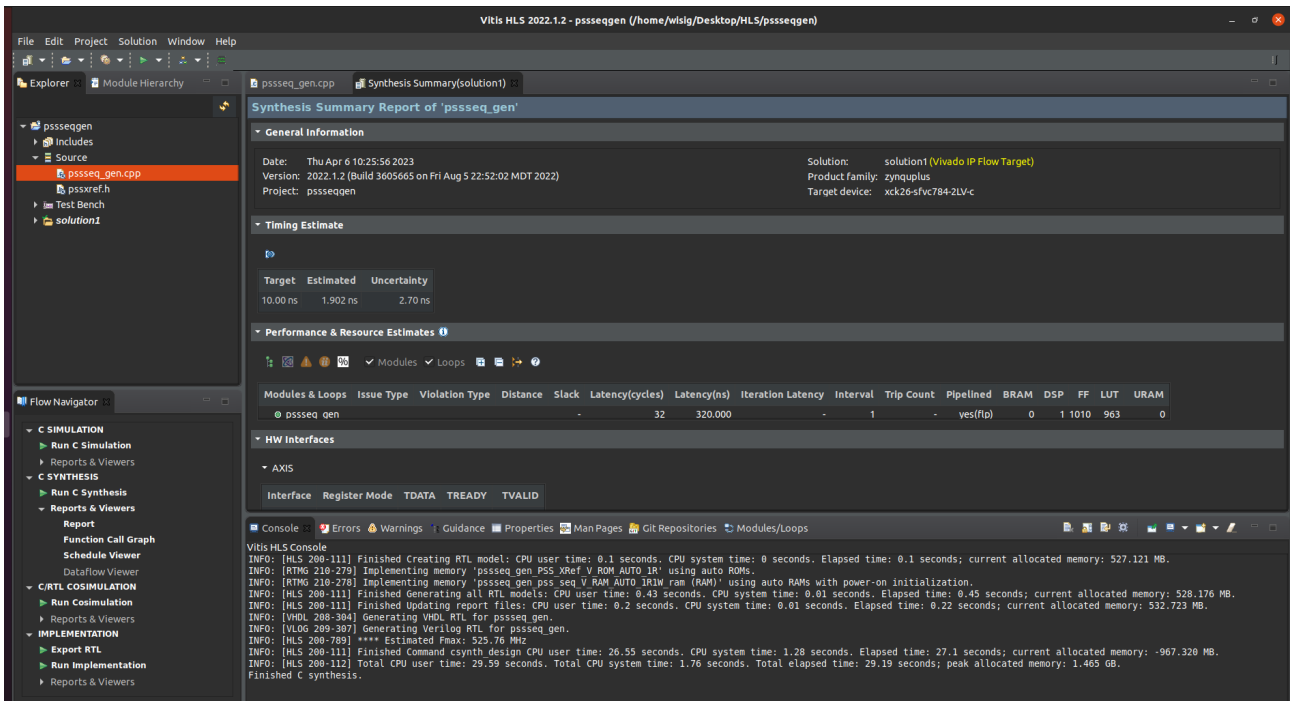




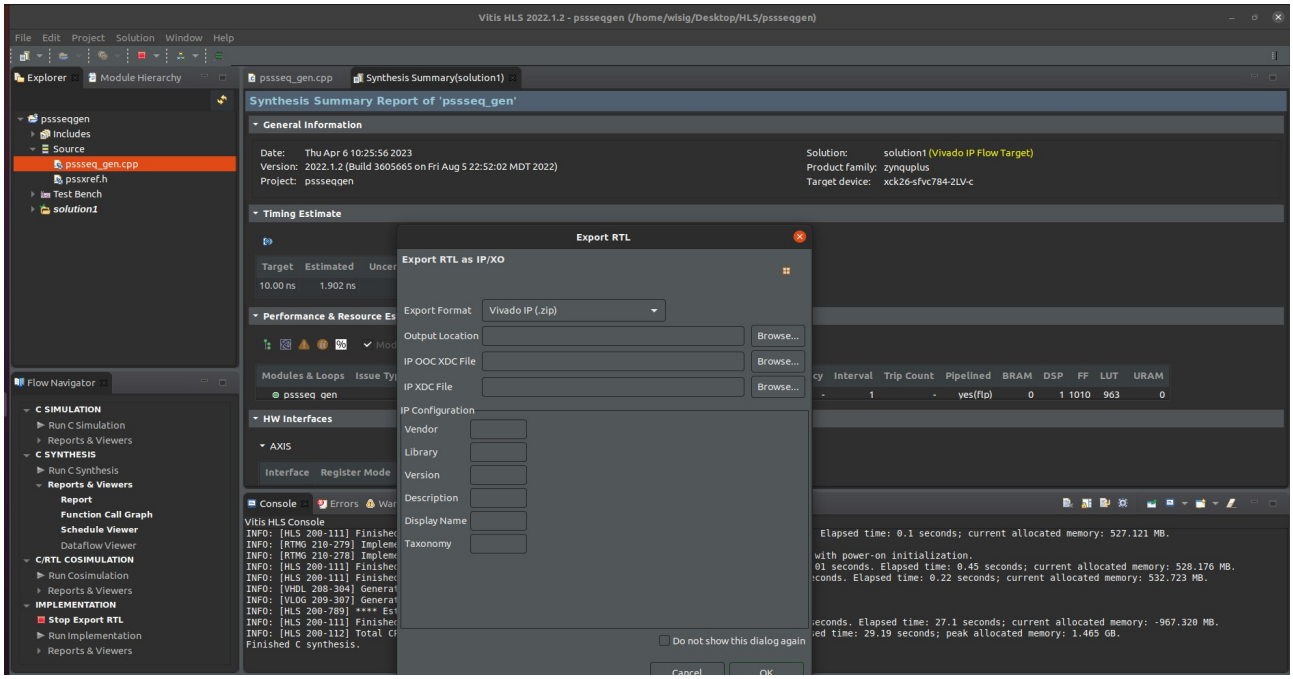
5. click on source then the uploaded files will be shown there. Select particular file to open.After opening the file click on ‘Run C Synthesis’.



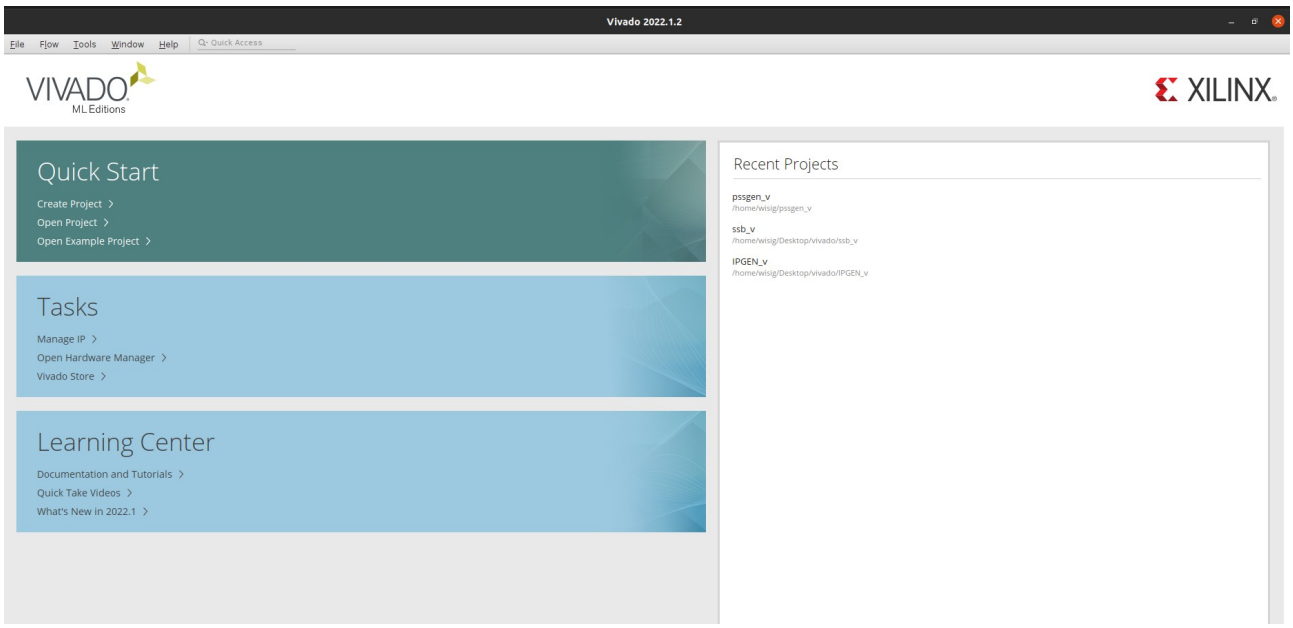
6. After that Wait until the C synthesis is completed on the console. After completing , C synthesis review will be shown on the screen.Analize the review and click on Export RTL.



7. Then an export RTL block will be open as shown in the below figure.After that click on ok. RTL exporting will be started and then wait for until Export RTL is fineshed in console window.

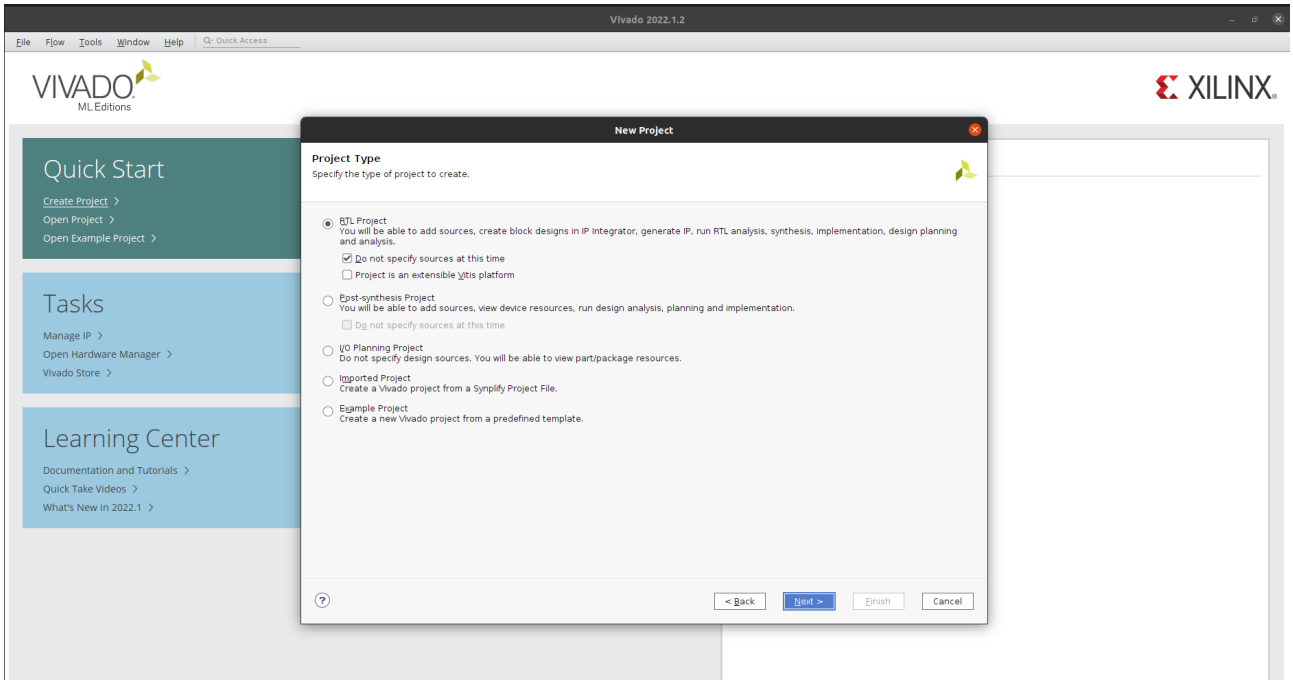


8. After Exporting RTL, Open Vivado then the window on image 1 will be visible. Click create project and click on create project.

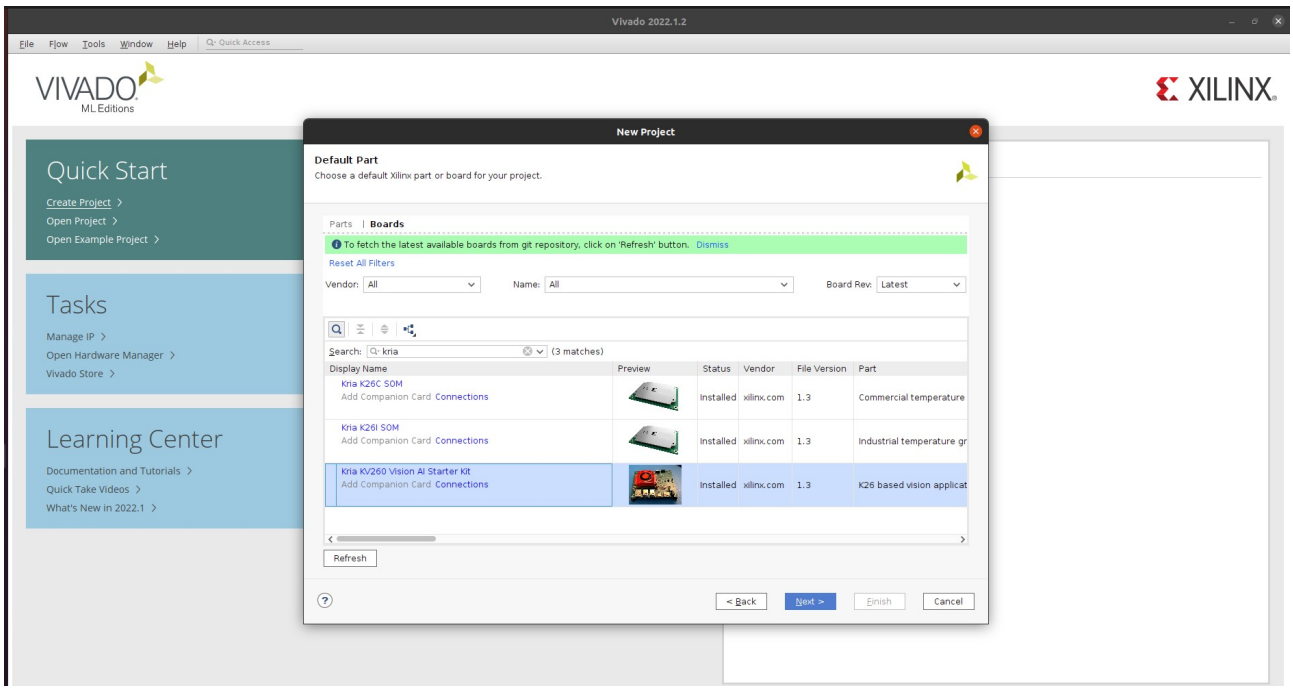


Img.1

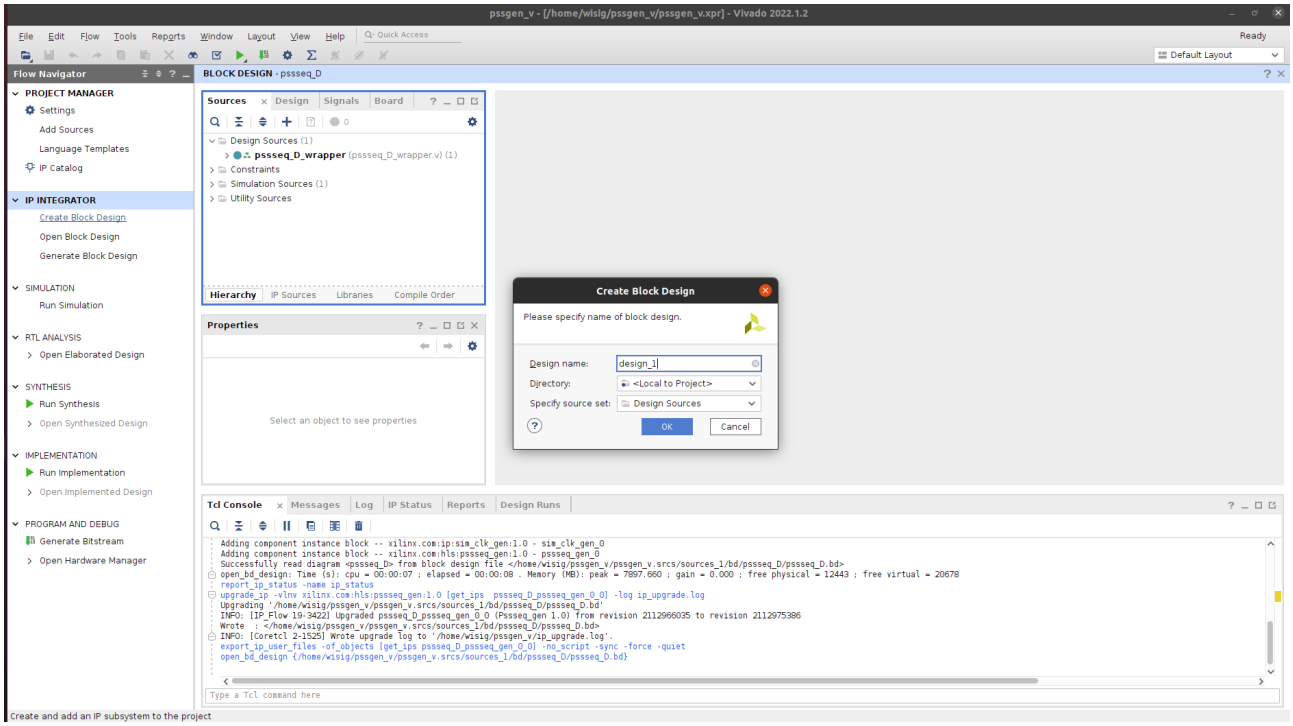
9.Fill in the project name and the project location in their designated fields and then click next. Then the window in figure 2 will be visible. Select RTL Project and tick the first check box as shown in the image and click next.



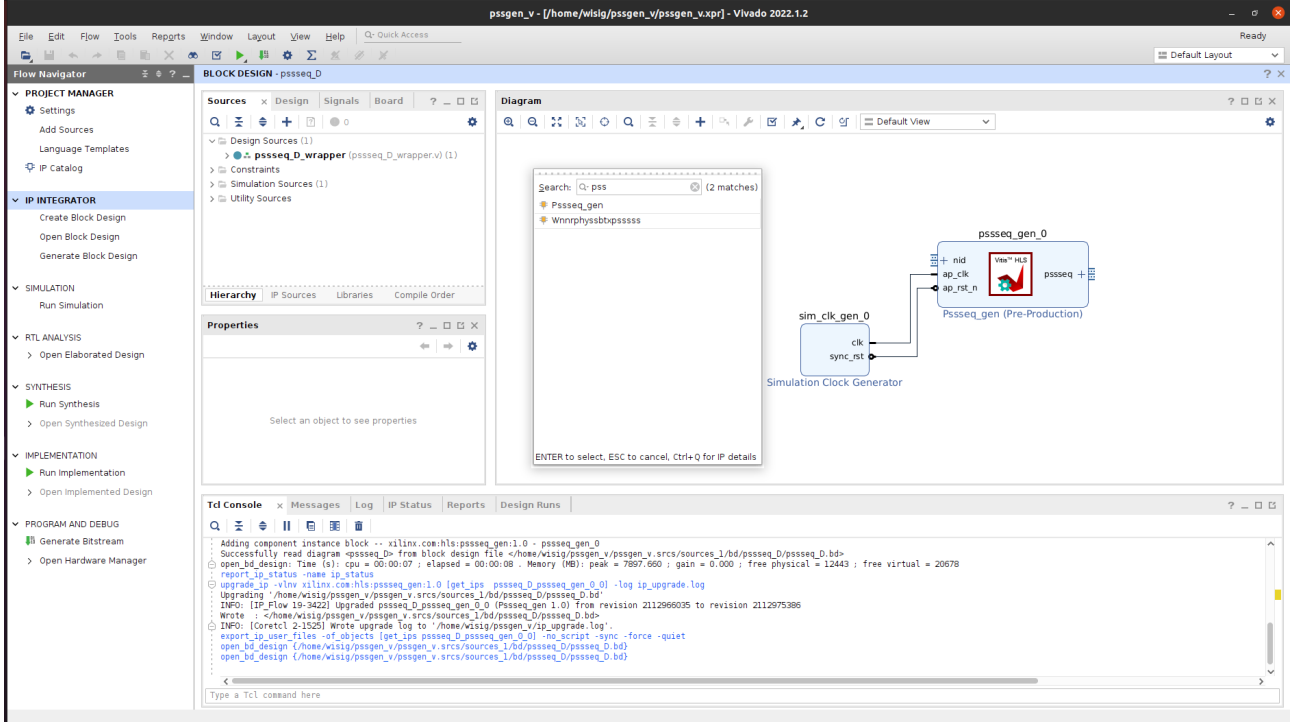
10.Then select the board you want to work on.After clicking on next, a summary of all our choices in the created project will be shown. Review the summary and the click on finish. Now, a Vivado project has been created and the Vivado Project manager window will be opened.



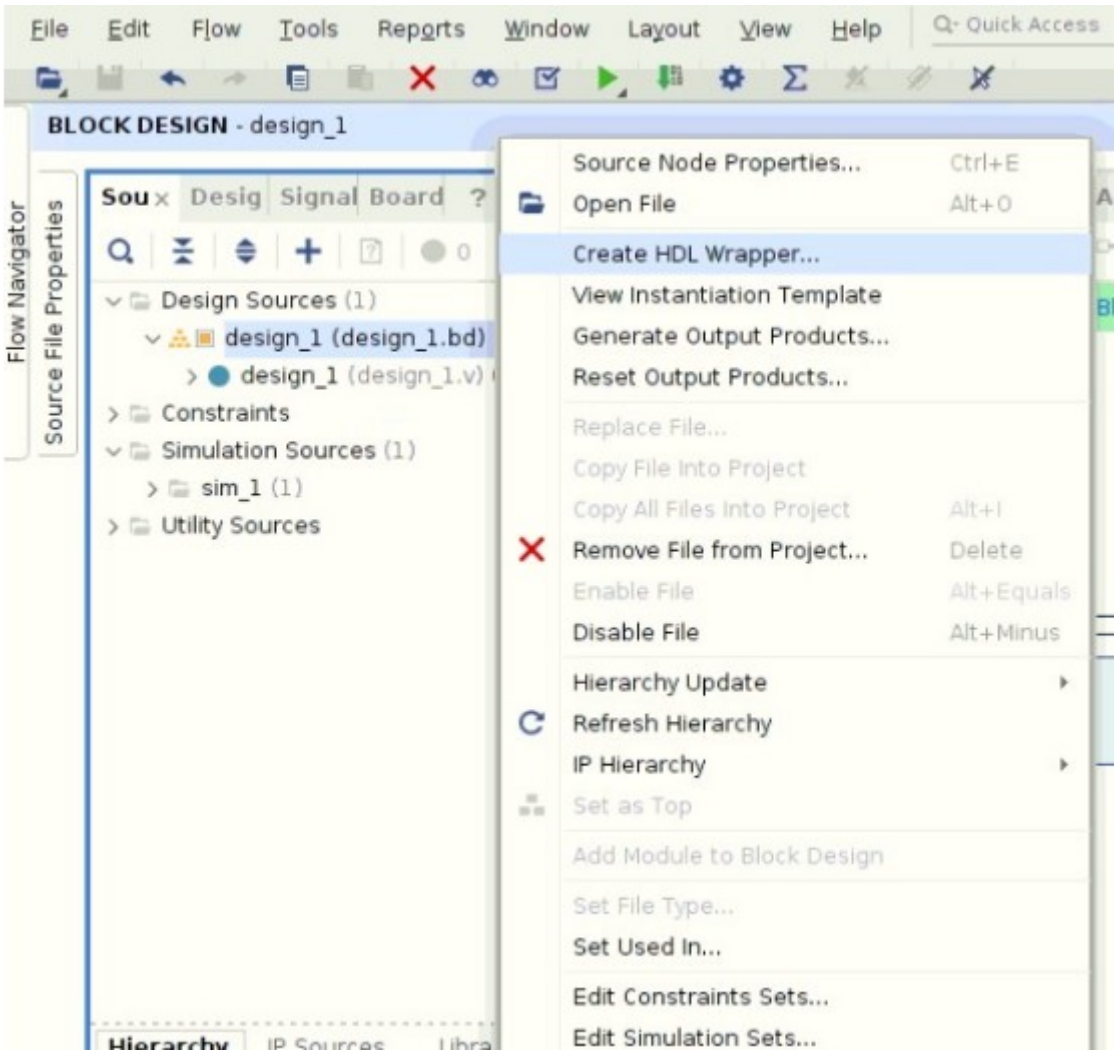
11.Now Click on create block design. Give a name to the design and save it in the following window.



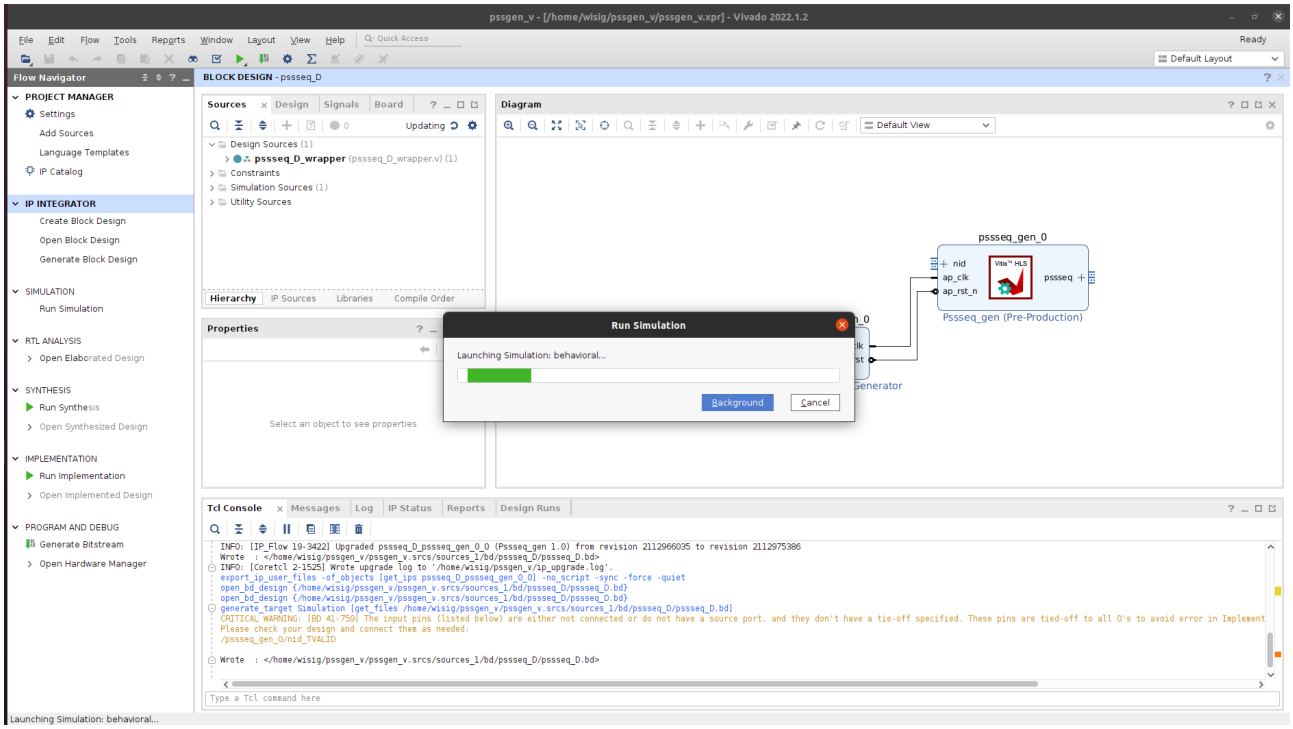
12. Click on the “ +’” sign or use the shortcut ctrl + I to open the menu to select the required IP from the available list.
Now make the following connections
The written IP is conected to simulation clock Generator[clock to clock and reset to reset].



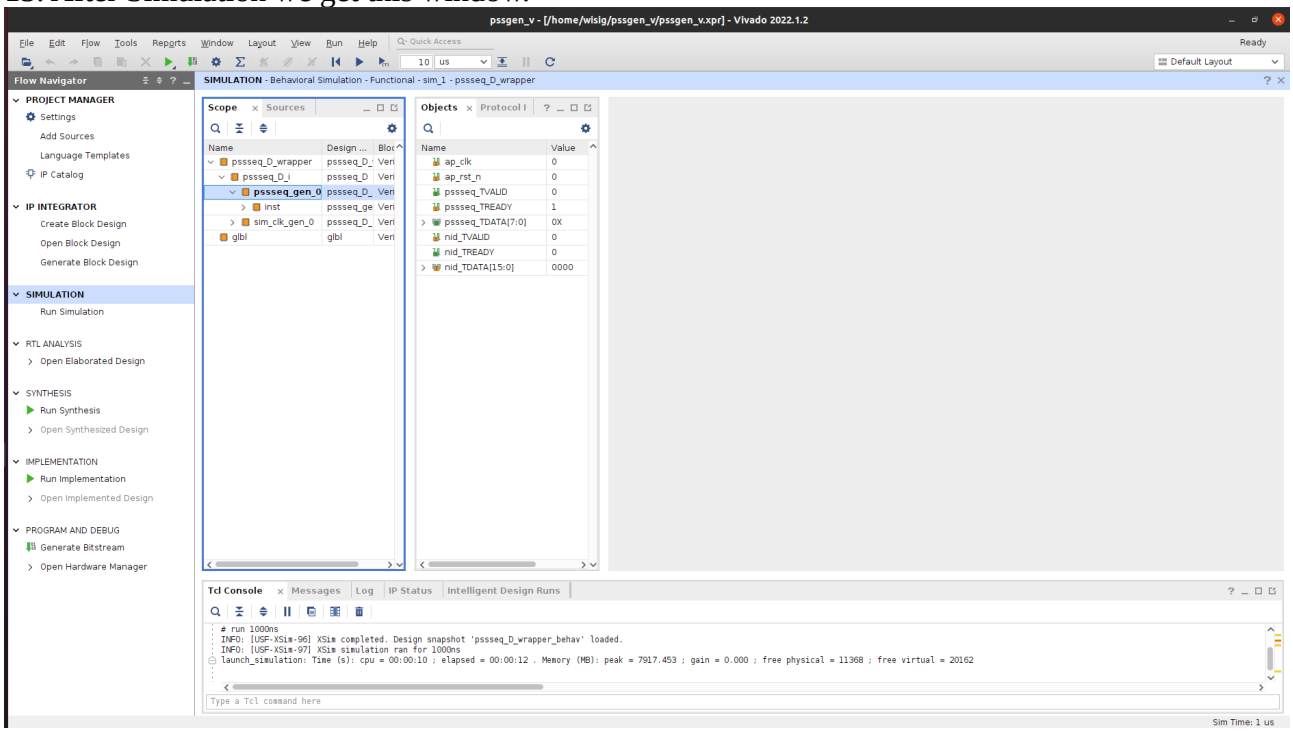
13.Save the design and then click on validate the design (highlighted in red). If there are no errors in the design, we should generate a wrapper for the block design. To do so, look in the sources window of the screen. Expand the design sources and right click on the block diagram (file with .bd extension) and select “create HDL wrapper”. In the next window, select the option “let Vivado manage wrapper and auto-update” to let Vivado create a wrapper module for you and then press ok.



14.After this click on Run Simulation,as shown in the below figure.

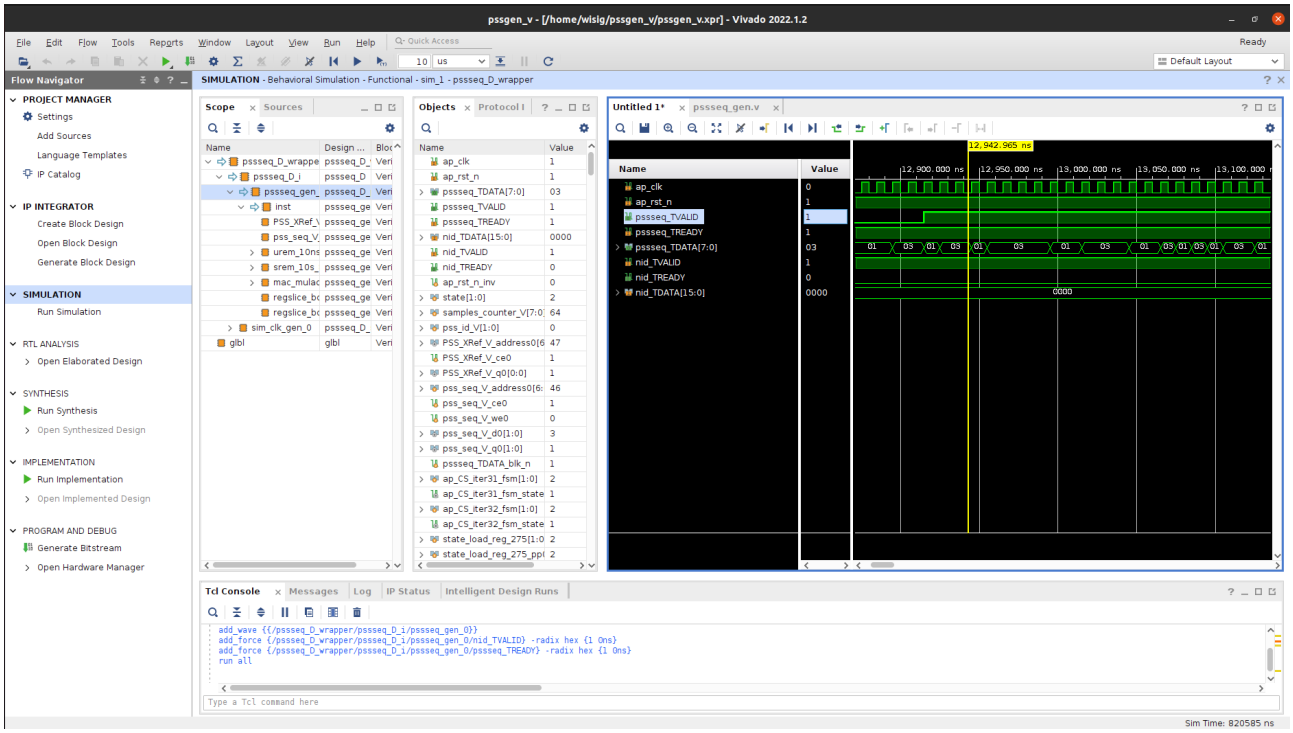


15. After Simulation we get this window.



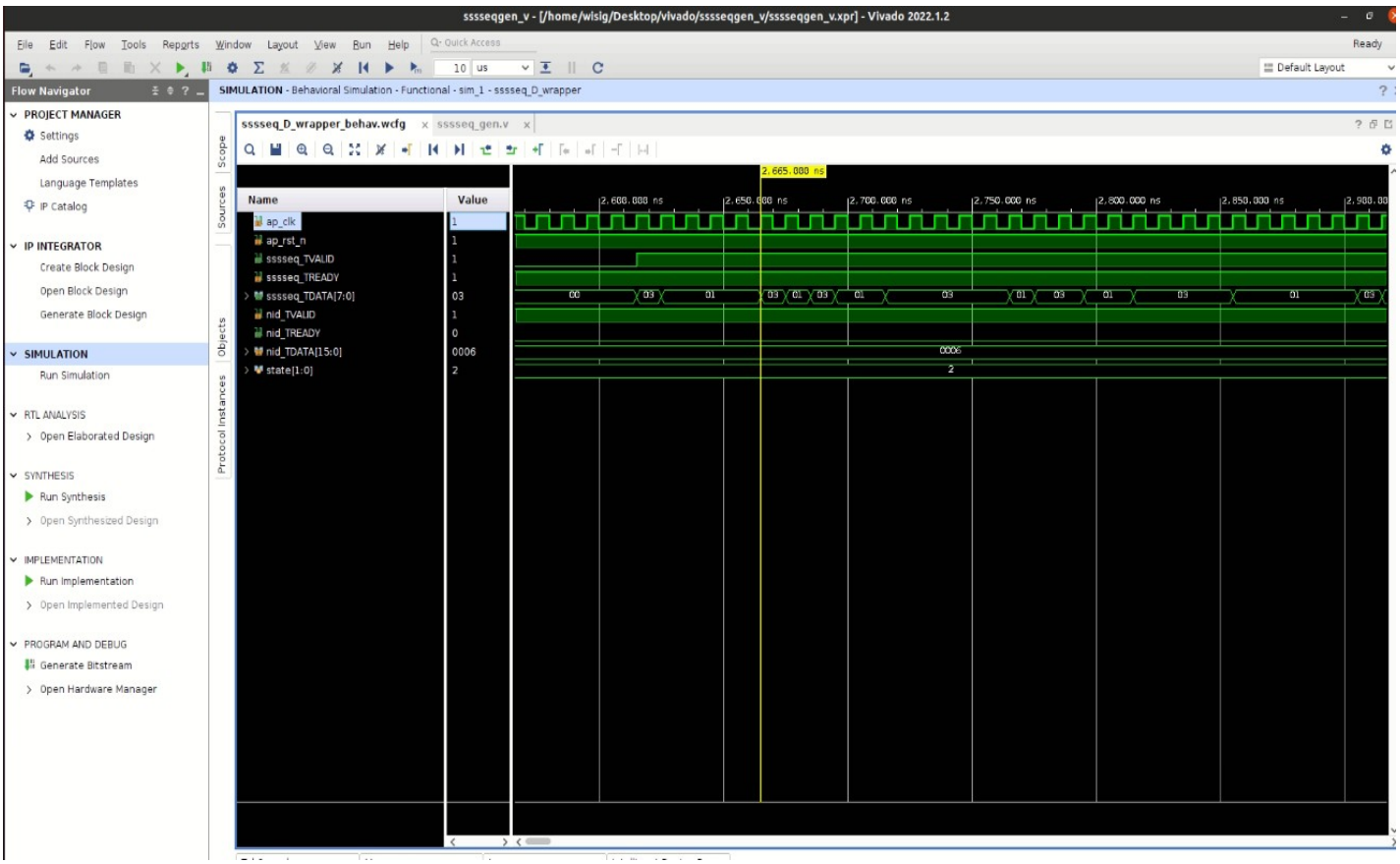
15.After simulation,add to wave window what you want for results verifing.

PSS Sequence RESULTS:



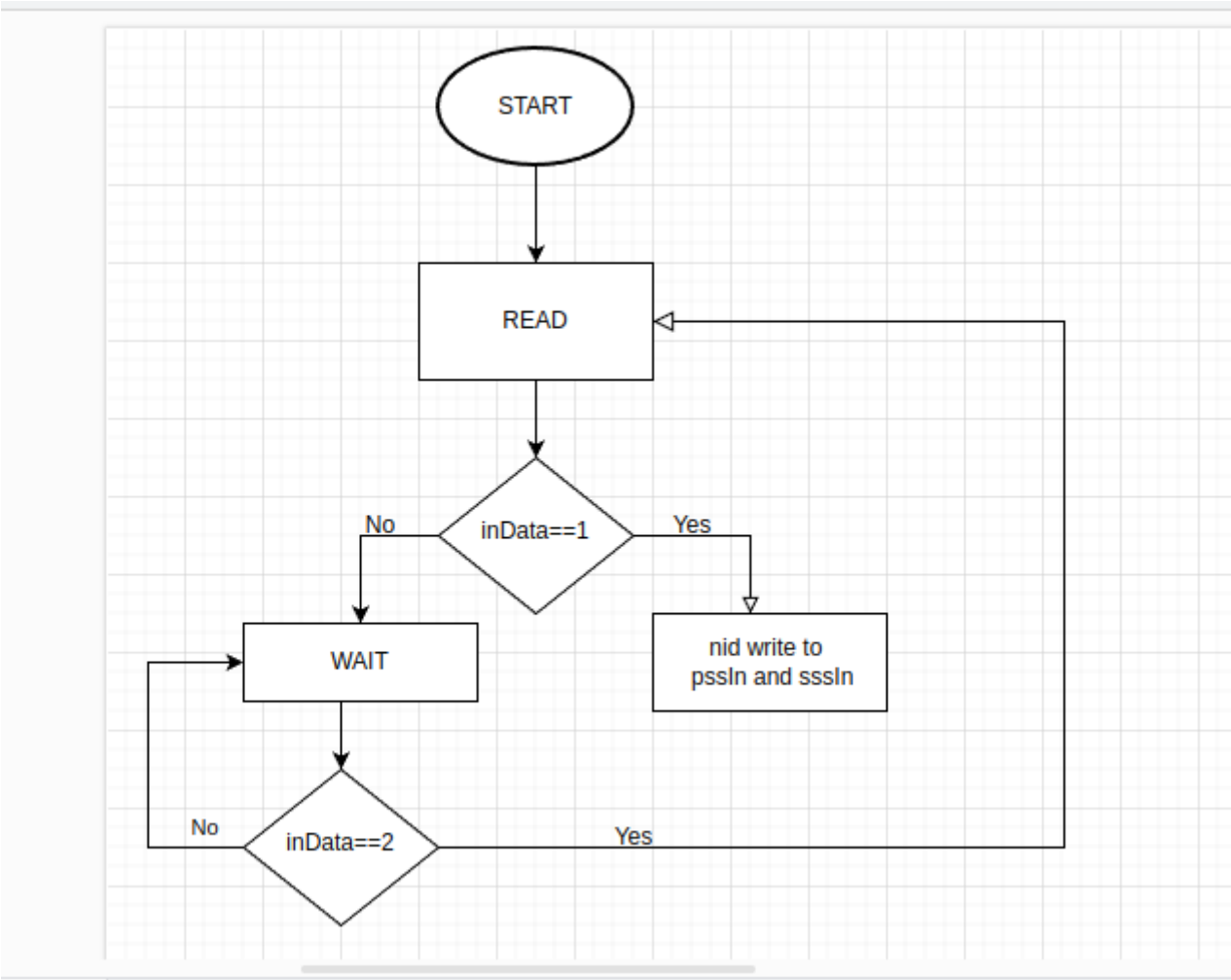
> same above procedure is Done for both SSS and QPSK generation and verified results.

SSS Sequence RESULTS:

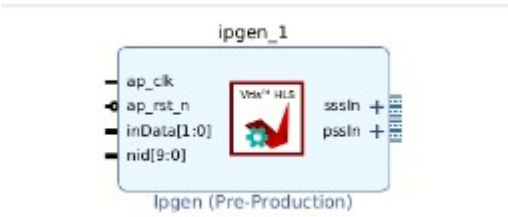


>I have written code to implement one IP it acts as controller to check the results of PSS and SSS on board.

> Here is the flowchart for the implementation of the IP.



> Below shown the generated IP.



> To implement a chain, I will create a Vivado project and proceed with the following steps:

1. Open Vivado then the window on figure 1 will be visible. Click create project and click on create project.

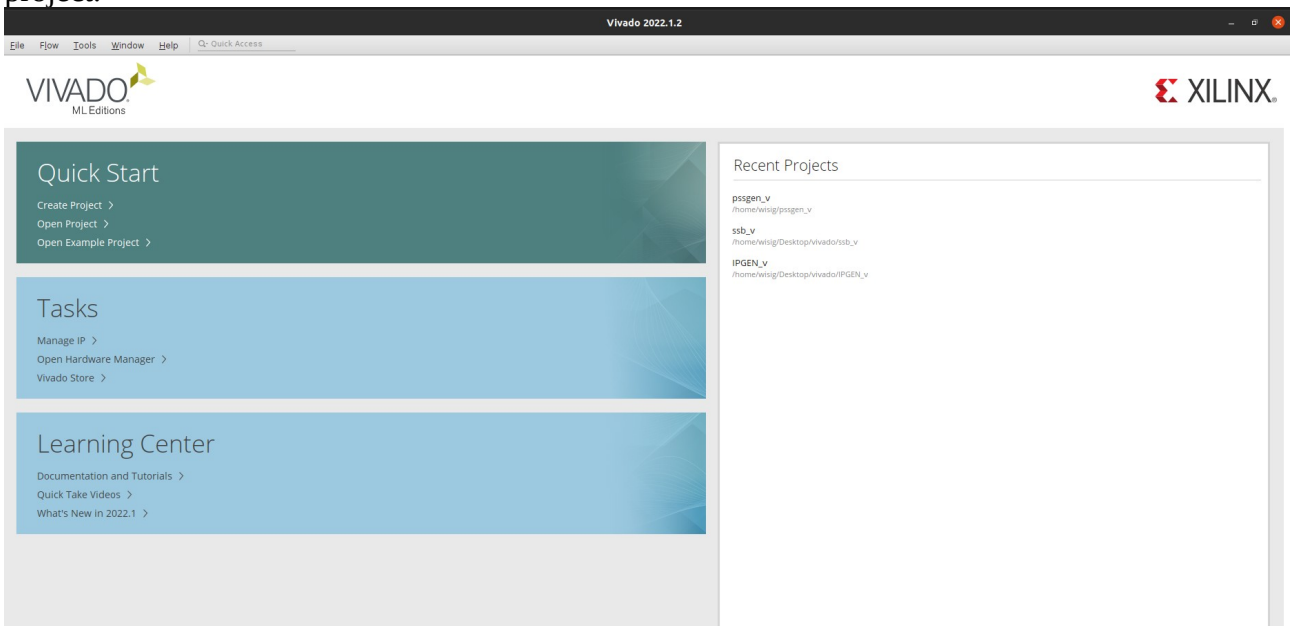


figure.1

2.Fill in the project name and the project location in their designated fields and then click next. Then the window in figure 2 will be visible. Select RTL Project and tick the first check box as shown in the image and click next.

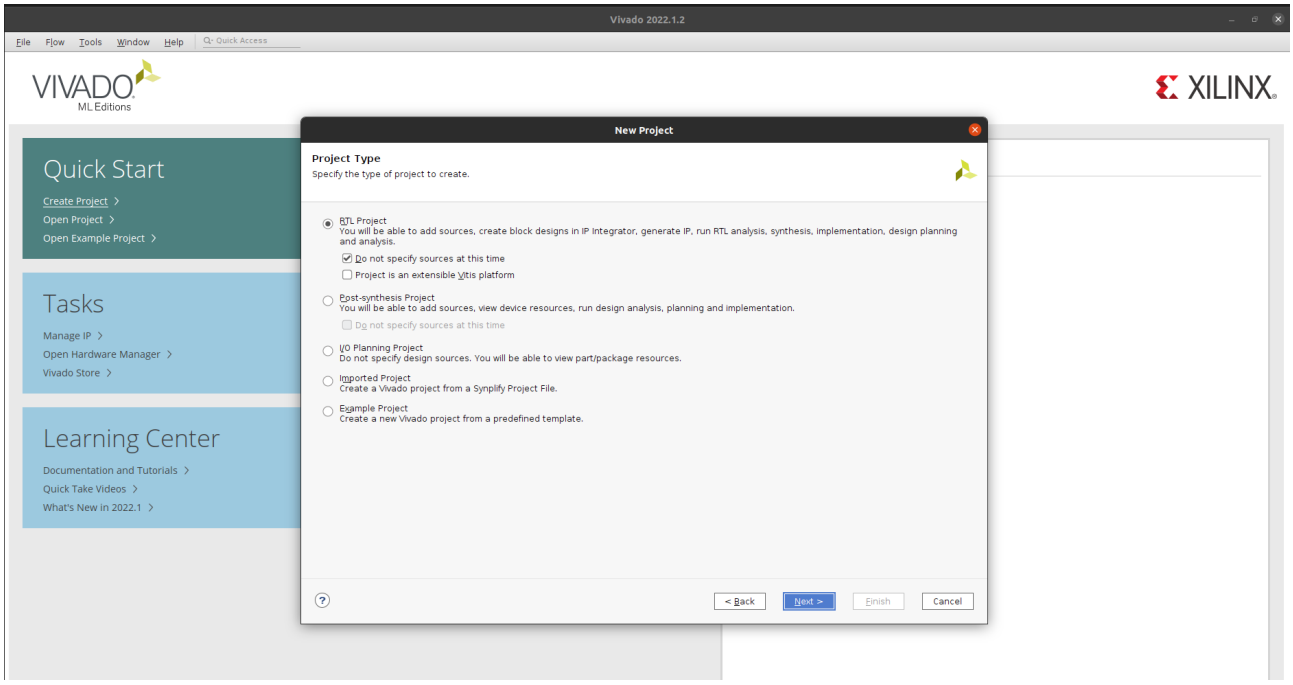
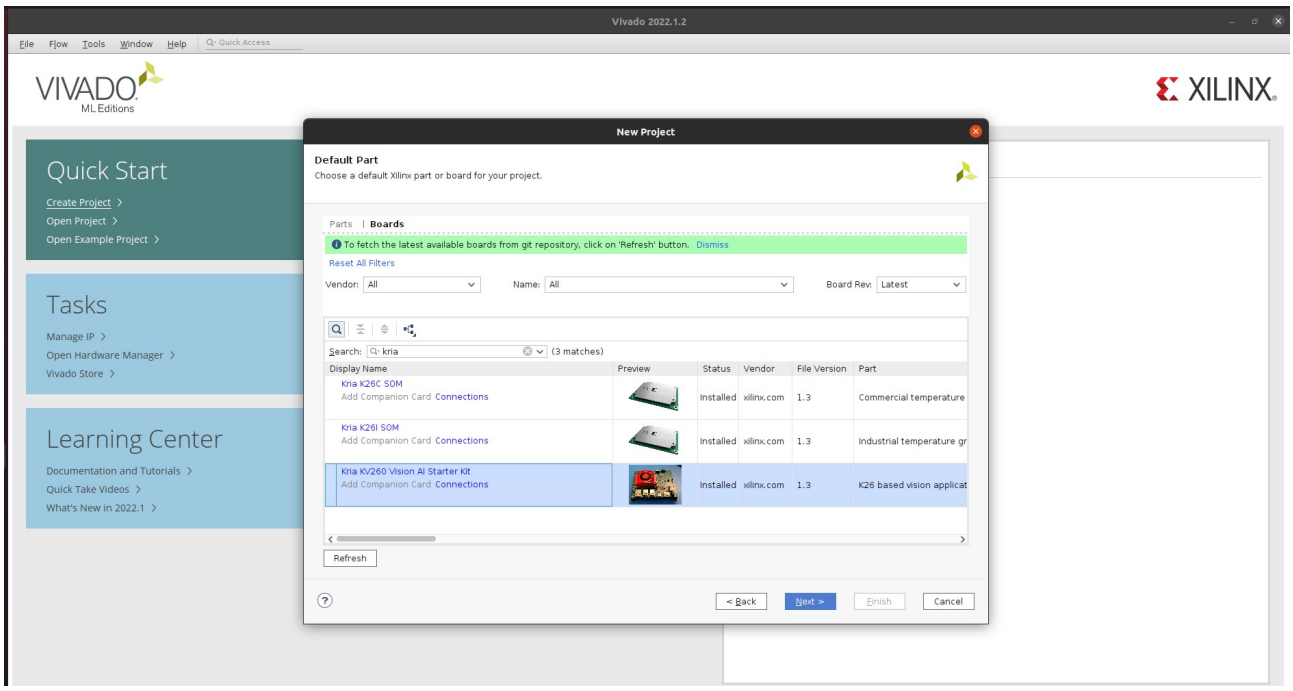
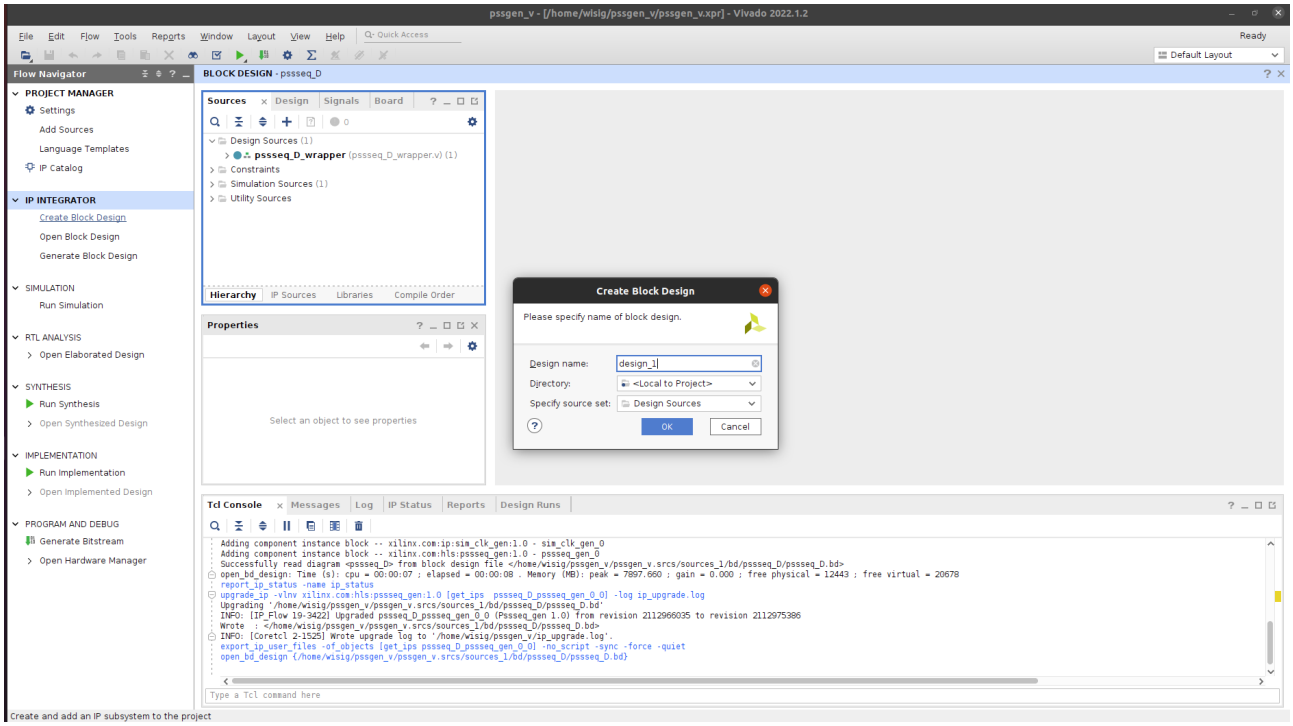


Figure.2

3.Then select the board you want to work on.After clicking on next, a summary of all our choices in the created project will be shown. Review the summary and the click on finish. Now, a Vivado project has been created and the Vivado Project manager window will be opened.



4.Now Click on create block design. Give a name to the design and save it in the following window.



5. Click on the “ +’” sign or use the shortcut ctrl + I to open the menu to select the required IP from the available list.

Now make the following connections

(i)In the zynq IP, connect the pl_clk0 to clk_in1 which input clock in clocking wizard.

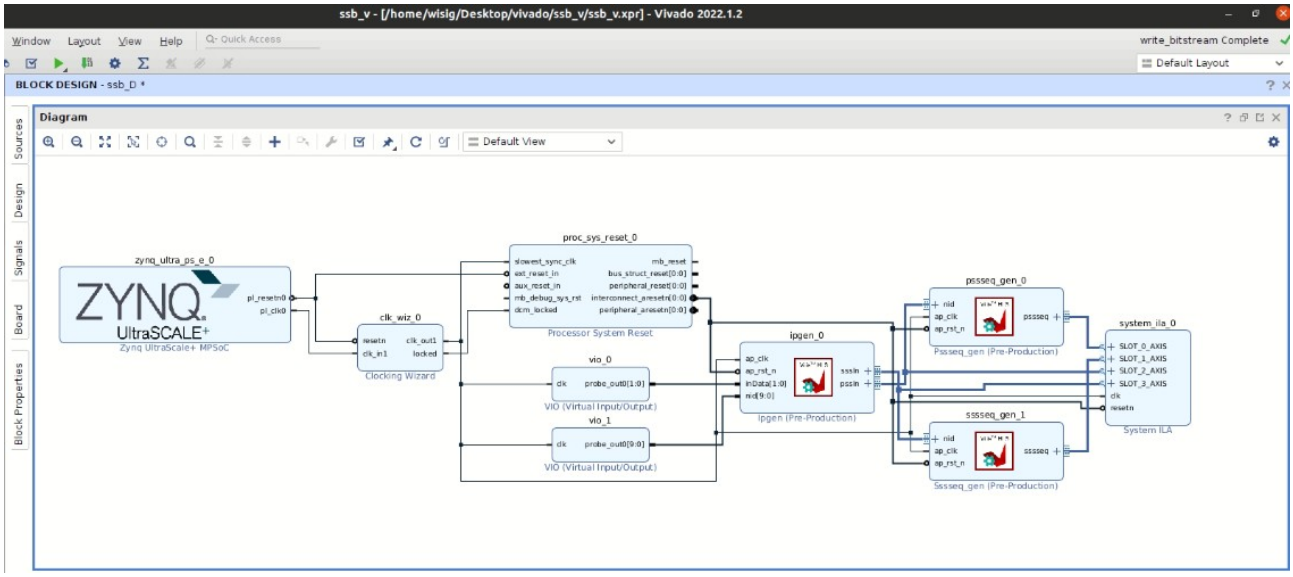
(ii)In the clocking wizard, connect the clk_out1 to slowest_sync_clk port and connect the locked to dcm_locked of the processor system reset block.

(iii)In the processor system reset block

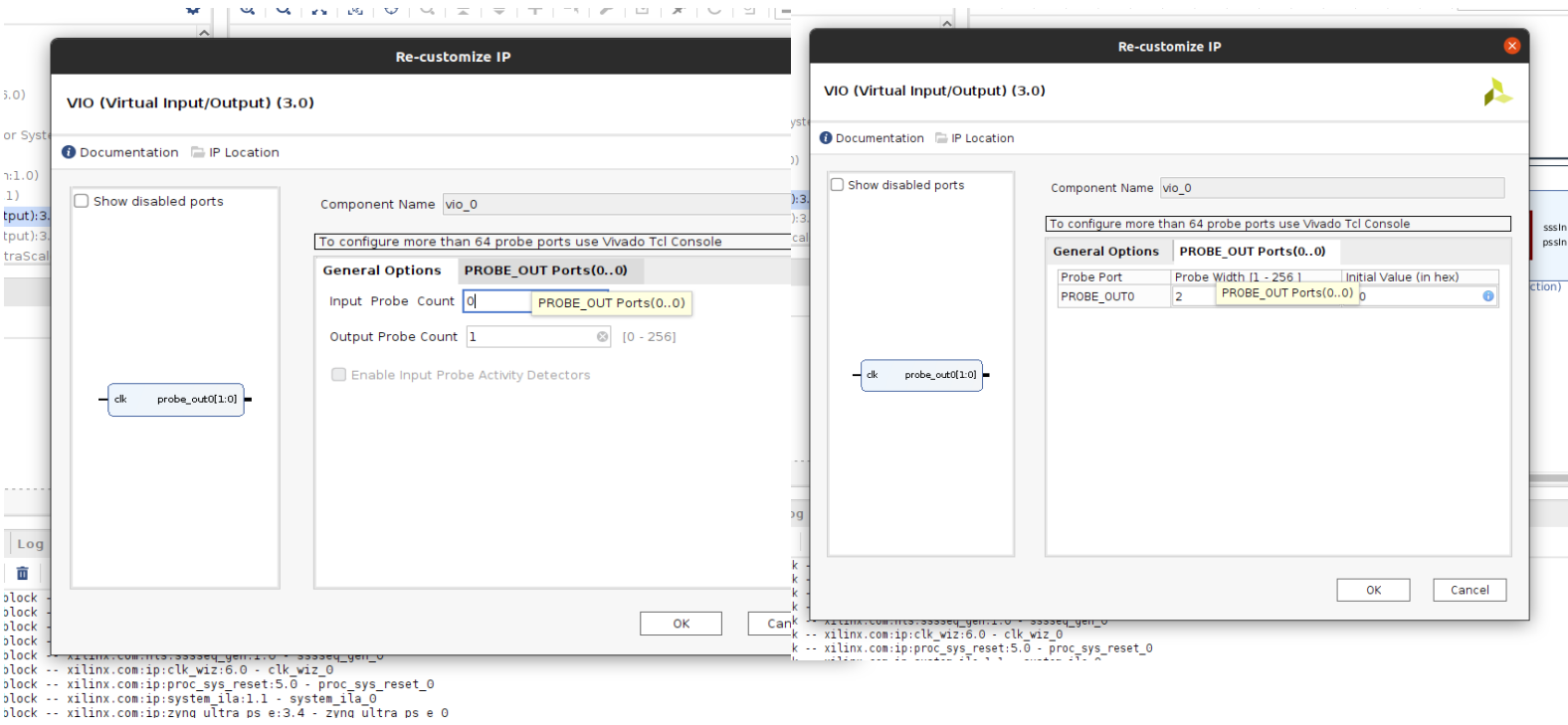
>connect the slowest_sync_clk port to vio’s clk and all athor IP’s ap_clk.

>connect the interconnect_aresetn to all IP’s ap_rst_n.

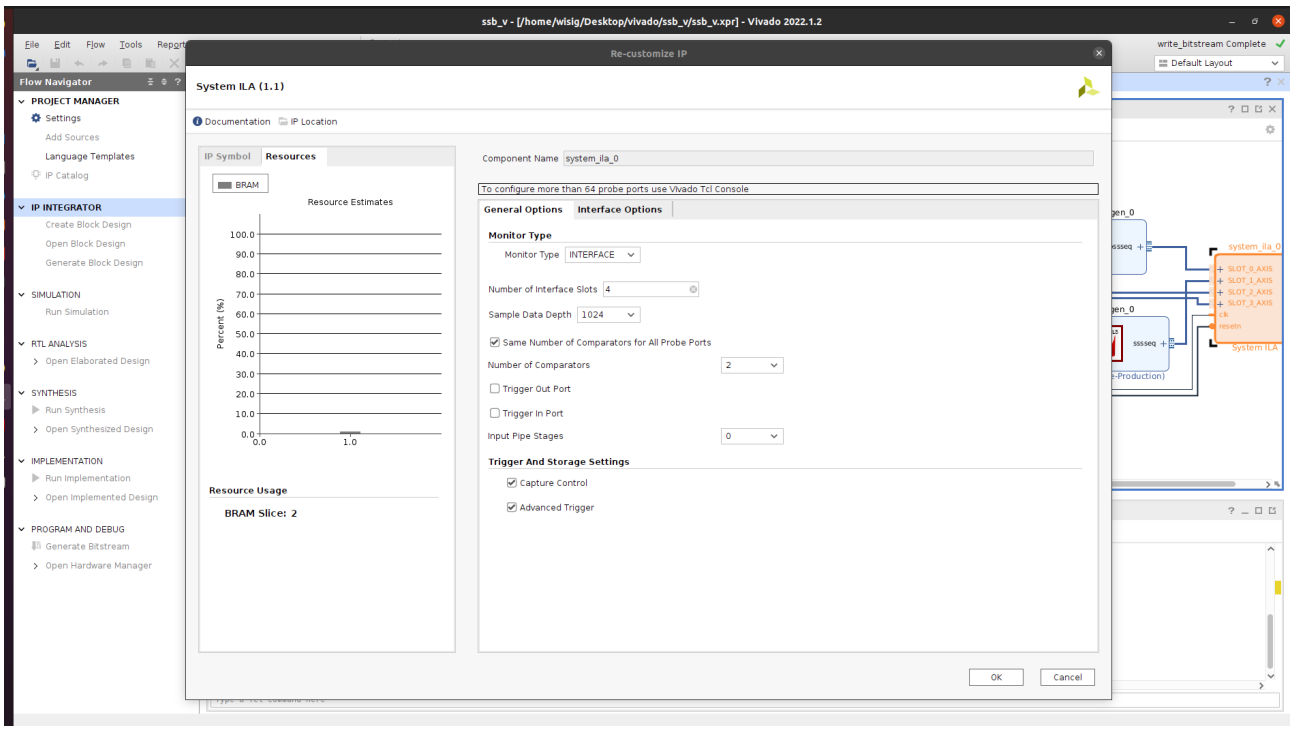
(iv) After that make connections as shown below figure.

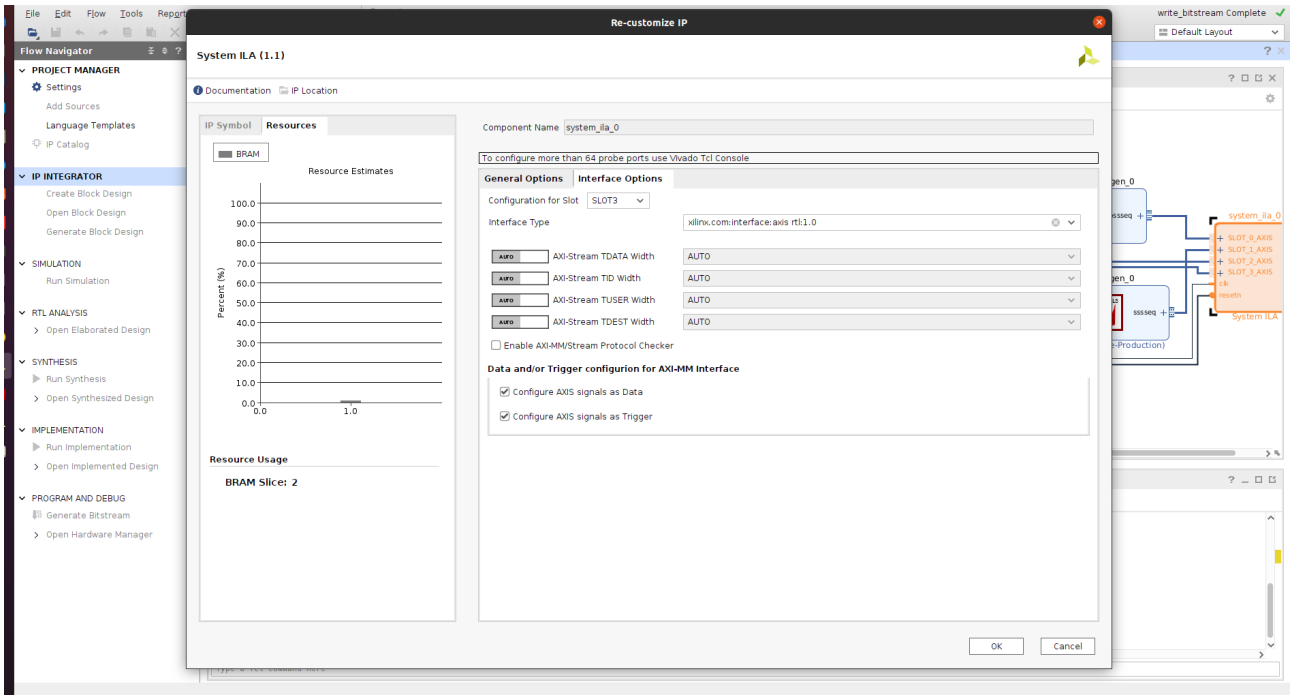


6. After double clicking on vio's IP below window will open.Based on your requirement recostmize the vio's IP as shown in below figures.

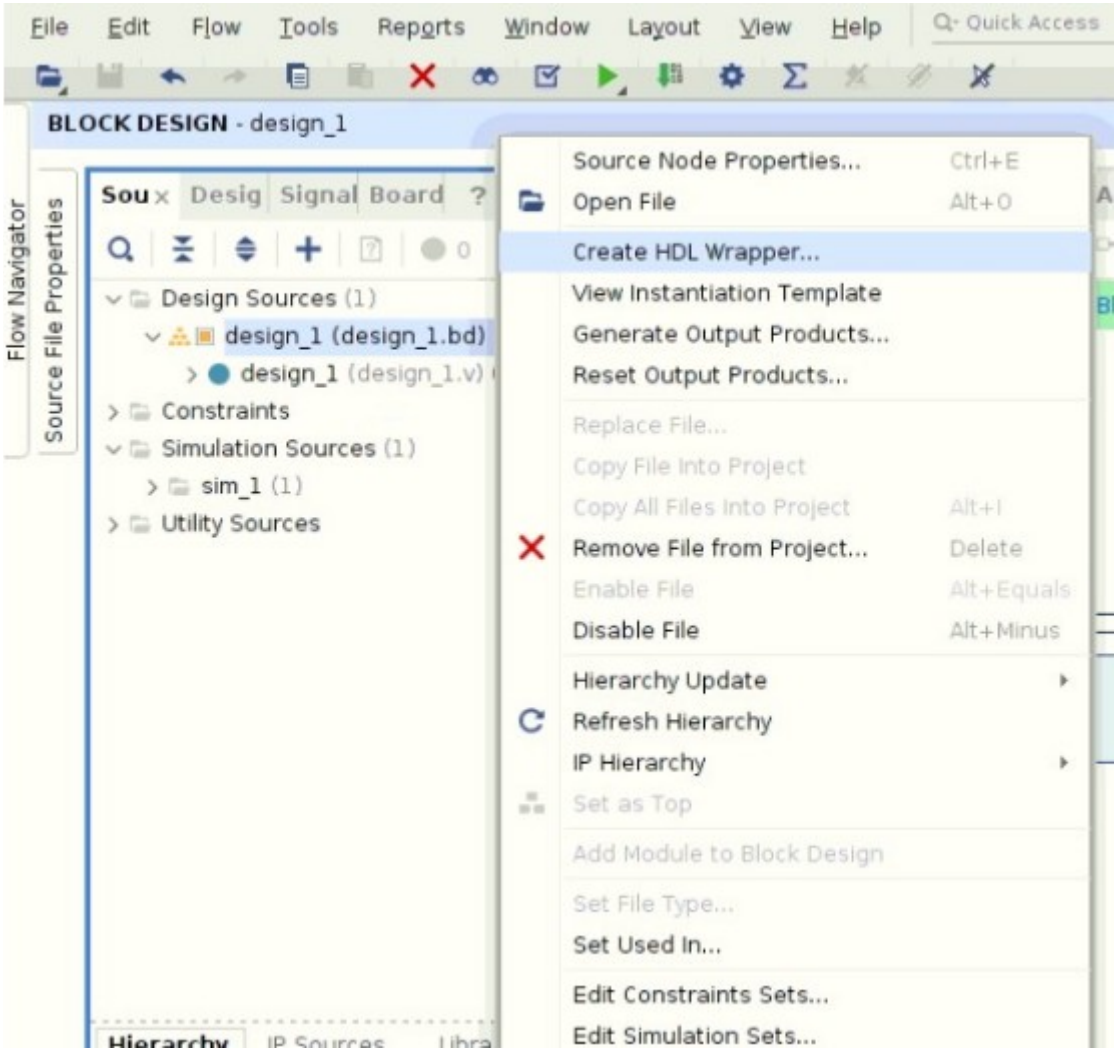


7. After double clicking on system ILA IP below window will open.Based on your requirement recostmize the system ILA IP as shown in below figures.





8.Save the design and then click on validate the design. If there are no errors in the design, we should generate a wrapper for the block design. To do so, look in the sources window of the screen. Expand the design sources and right click on the block diagram (file with .bd extension) and select “create HDL wrapper”. In the next window, select the option “let Vivado manage wrapper and auto-update” to let Vivado create a wrapper module for you and then press ok.



9.Now, can go ahead with generating the bit stream (.xsa file). To generate the bitstream first the design should undergo the processes of Synthesis and Implementation. These options are available on the right side of the screen in flow navigator window.Then click on Generate Bitstream to obtain .xsa file. The bitstream creation takes some time to run.After successfully completed Bitstream, then in the flow navigator window on the right, if we scroll down, we can see an option as “Open hardware manager”. Now click on “Open target” as shown in red box in figure Now select auto-connect option to connect to our board in Vivado.

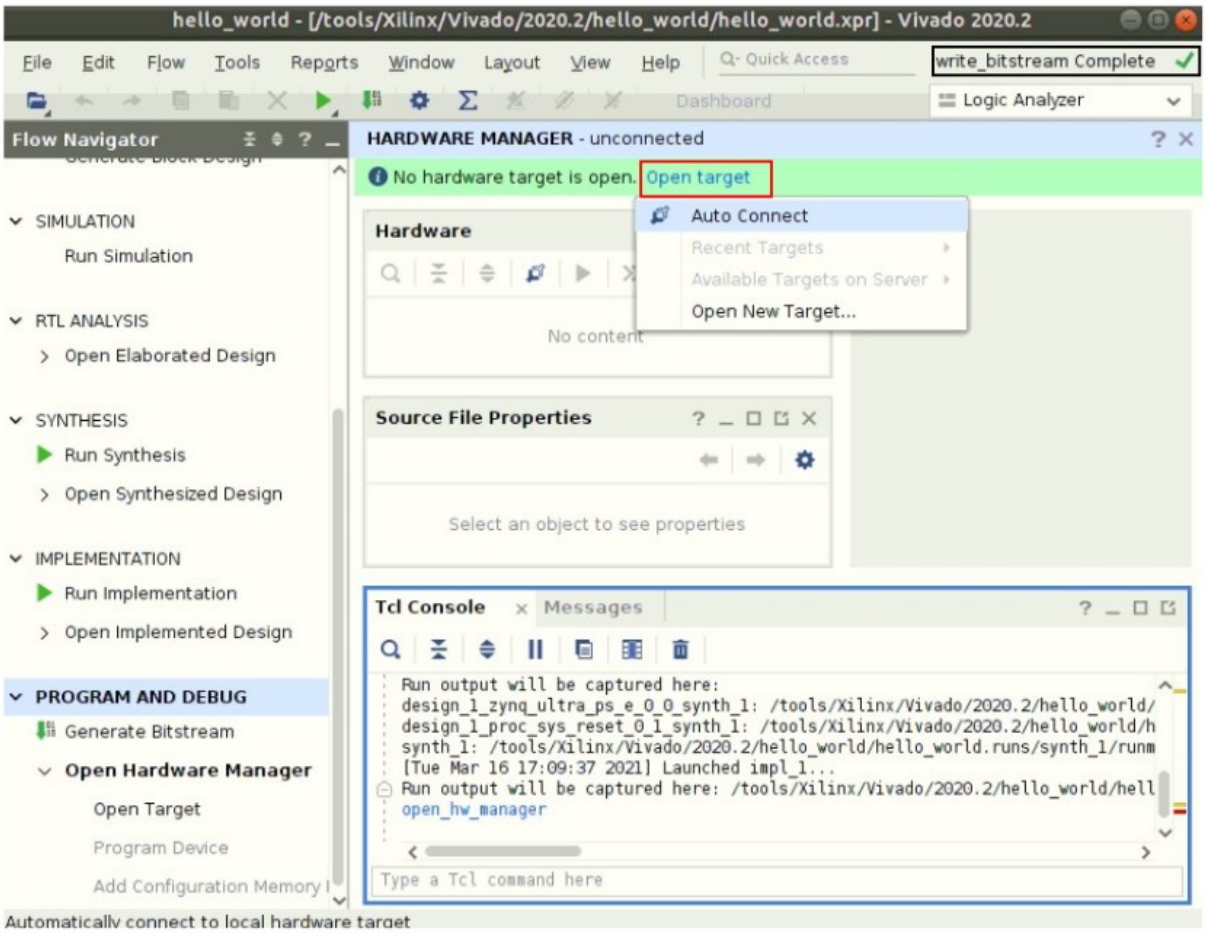
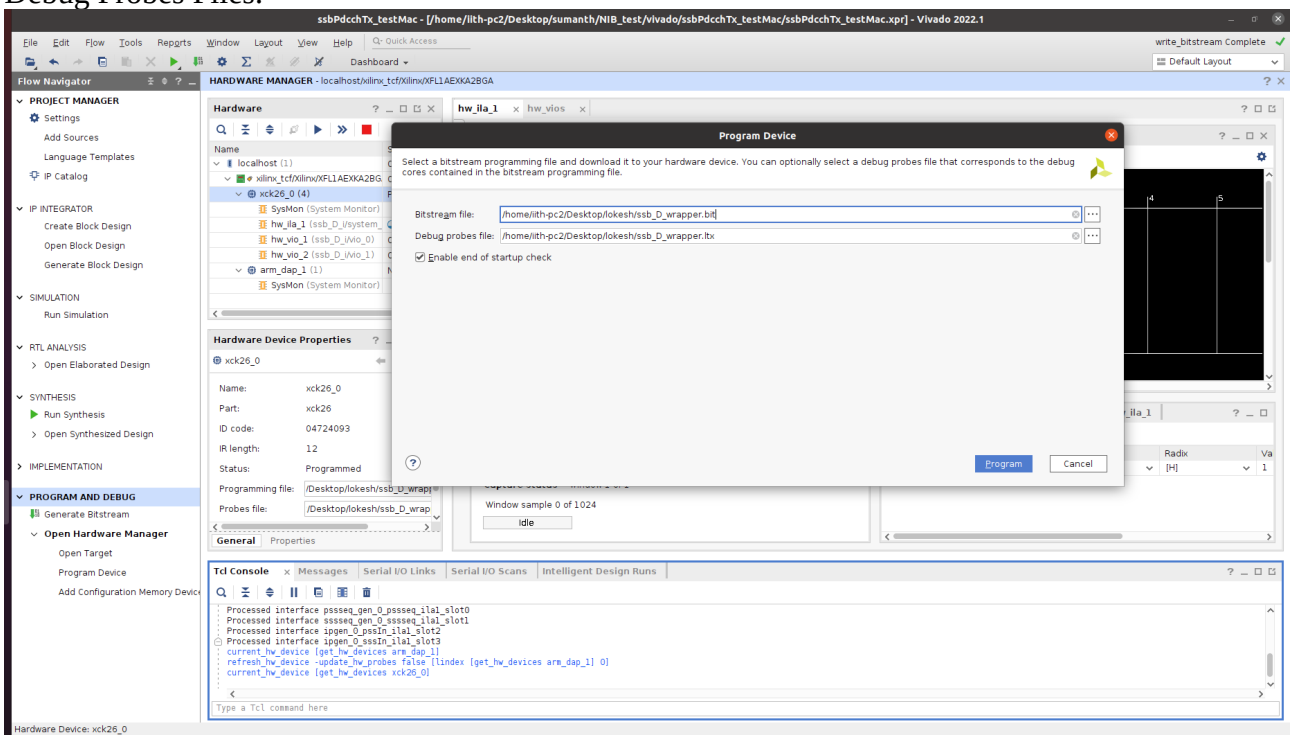
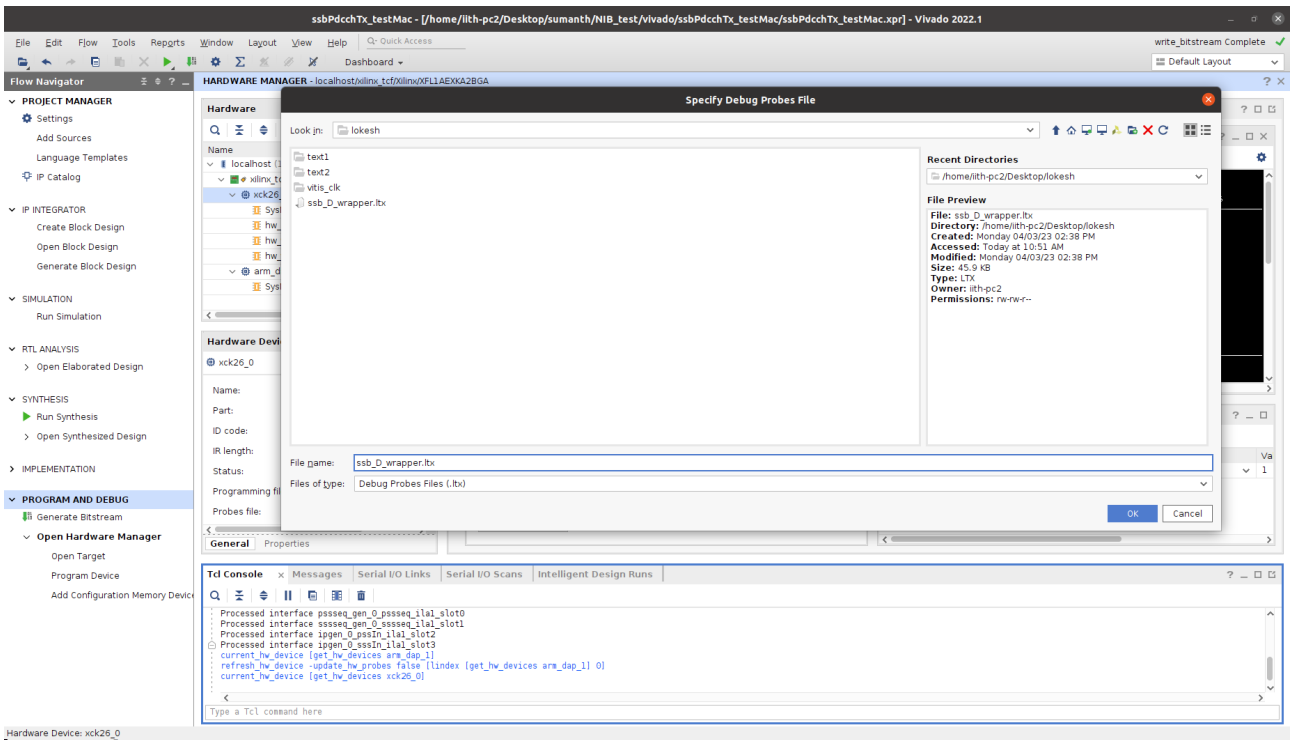


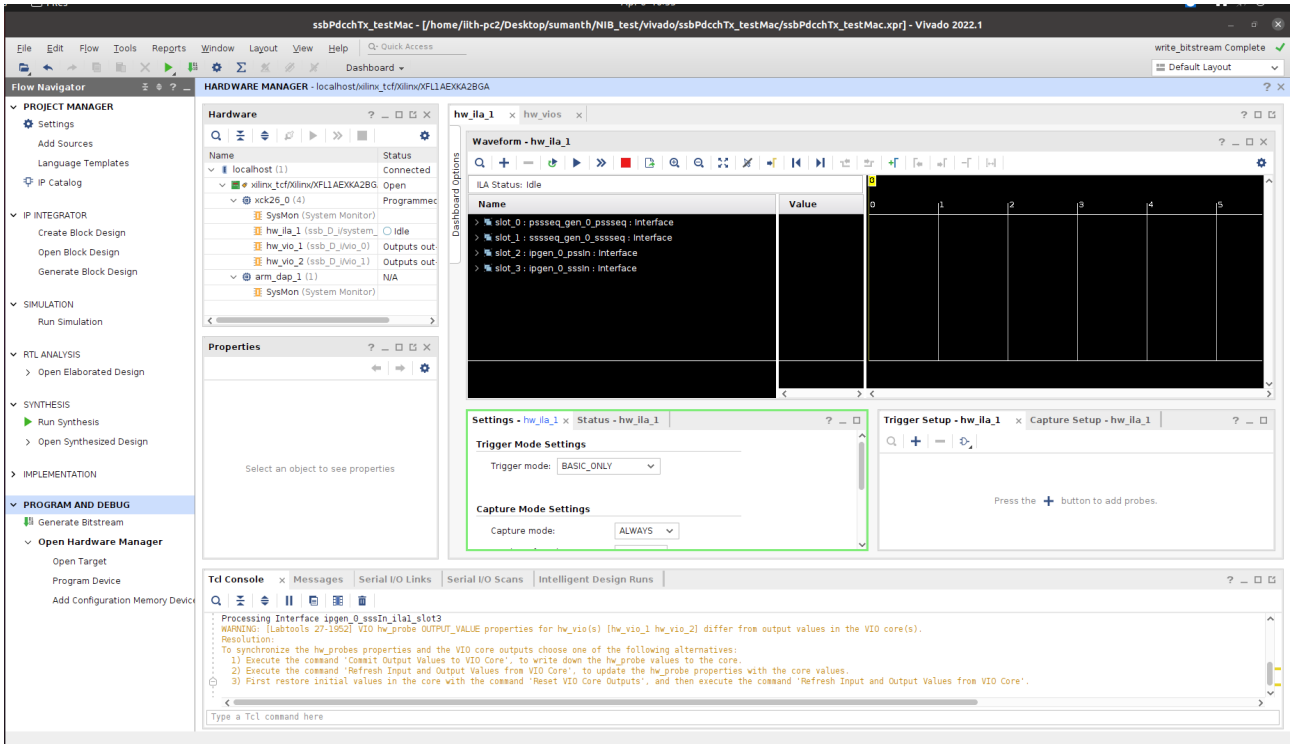
Figure 8 Opening Hardware manager

10. After that click on Program Device we get below window. Then specify Bitstream File and Debug Probes Files.

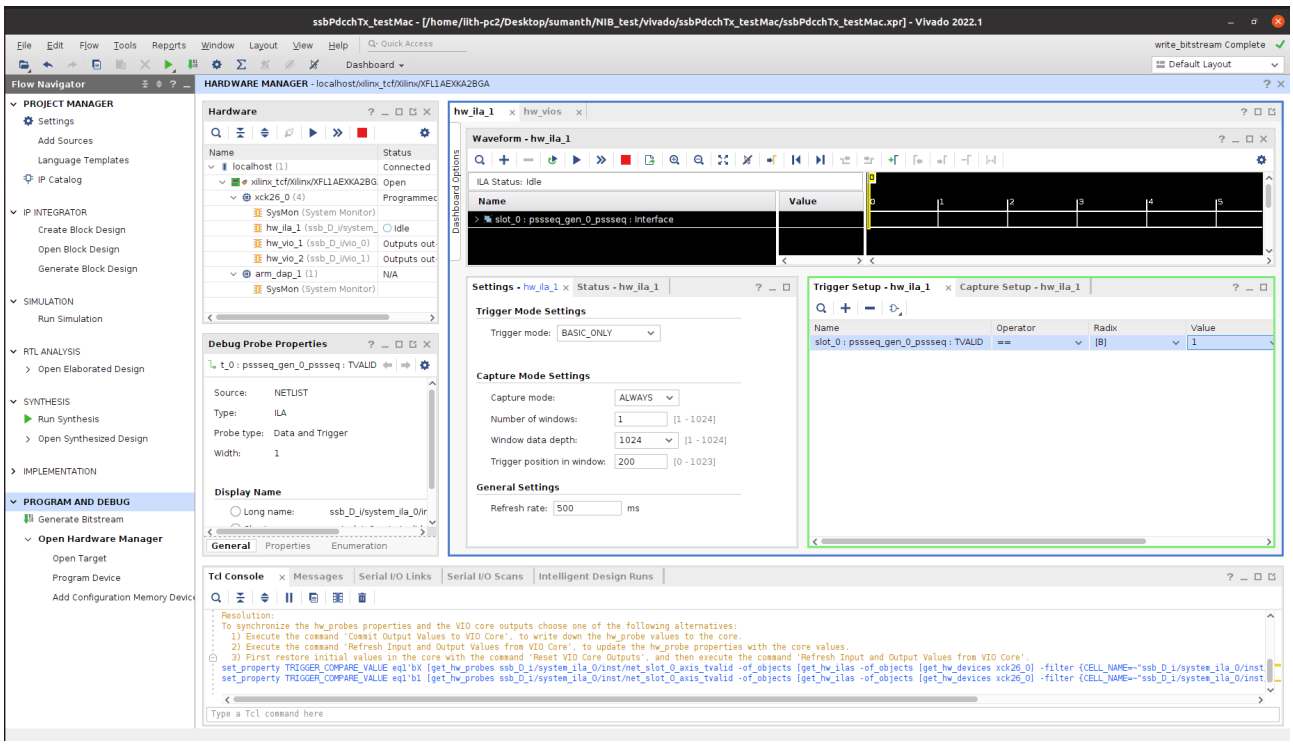


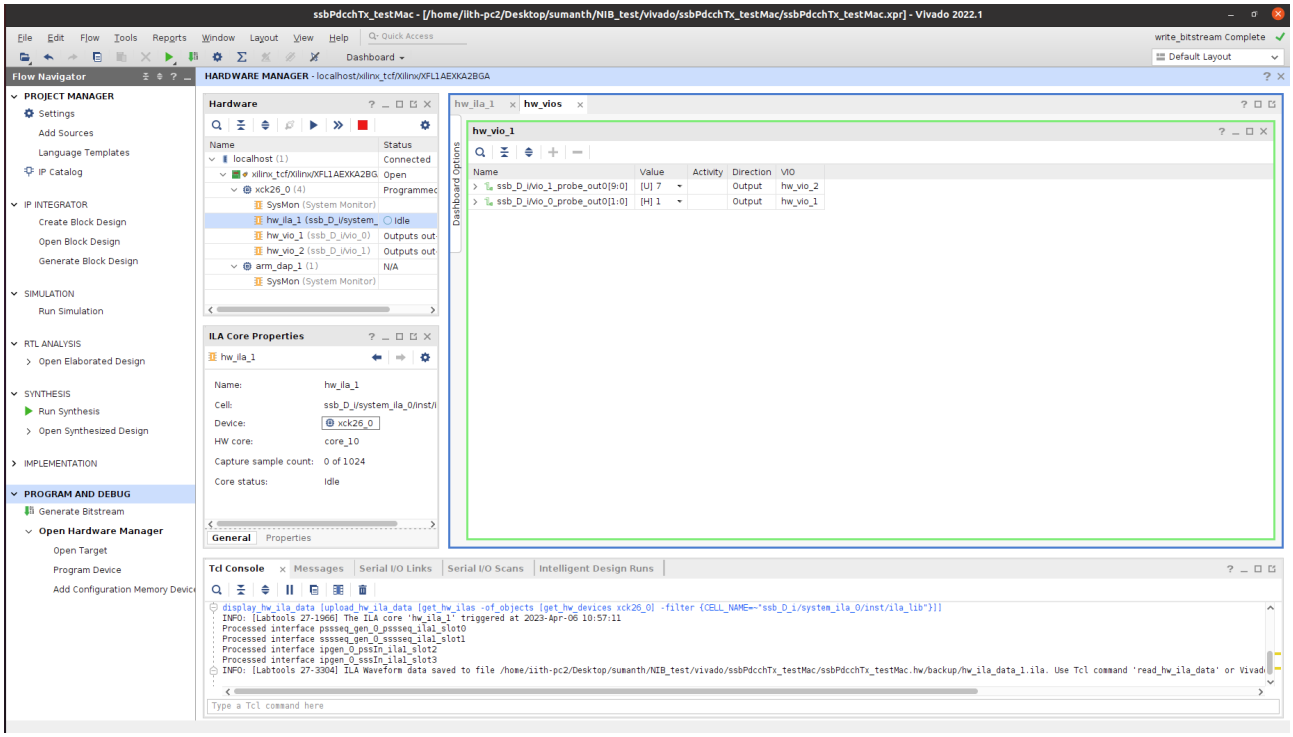


11.After that we get below window.

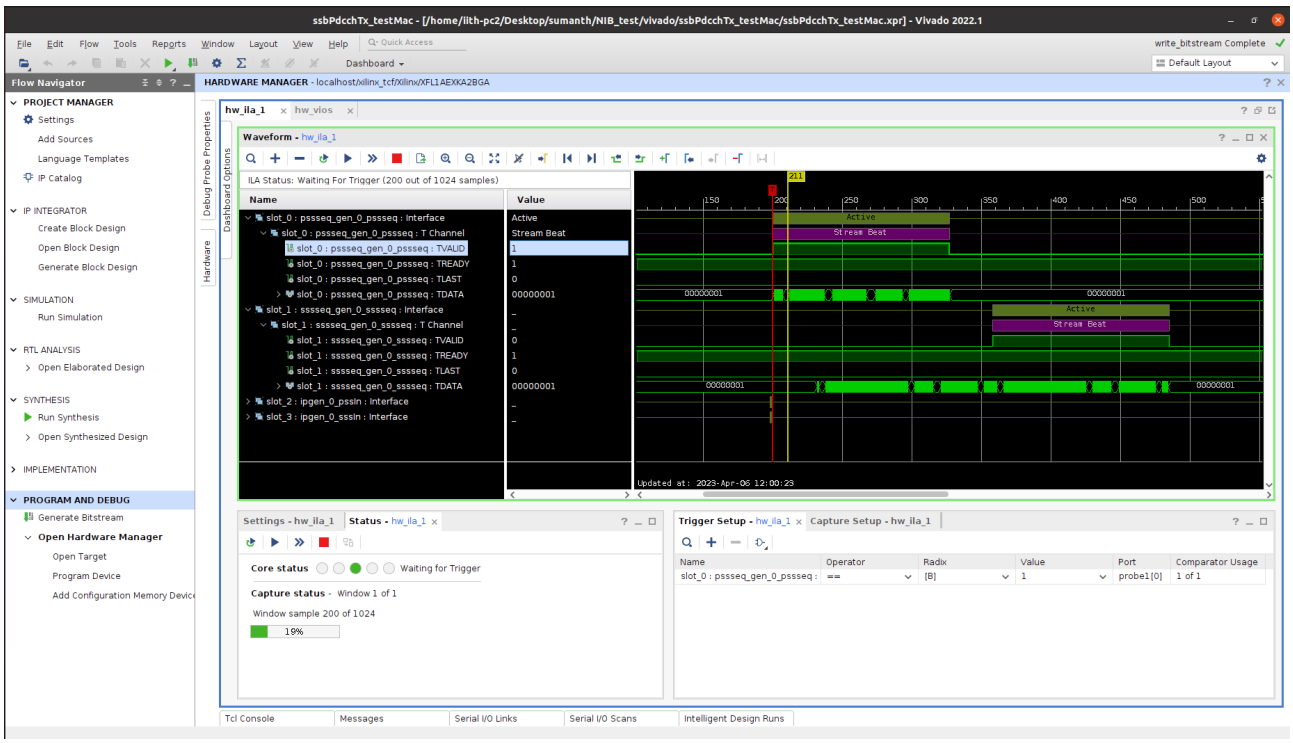


> Then give trigger condition and some Trigger Mode Settings and ila and vio's shown below.





12. Finally we get result and verified.



13.Next, by right-clicking on the Name dashboard, we can select the last option "Export ILA Data". Then, we need to specify the format as CSV and click on "OK".This will allow us to export the ILA data successfully.

14.To verify the pss sequence results, I have written a MATLAB code following these steps:

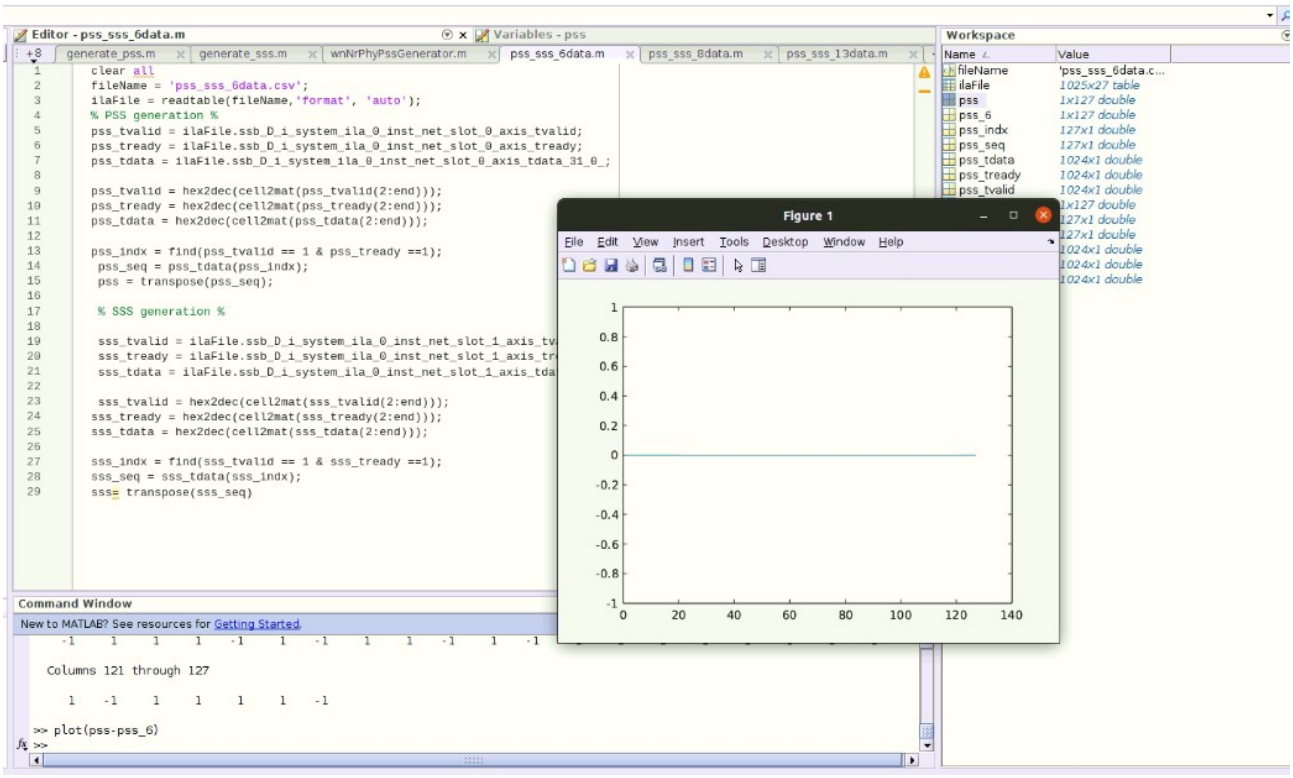
(i) Firstly, I convert the .CSV file into a table using the readtable function in MATLAB.

(ii) Then, I extract the columns pss_tvalid, pss_tready, and pss_tdata from the table using the dot operator (tablename.column_name).

(iii) The columns pss_tvalid, pss_tready, and pss_tdata contain hexadecimal values and are stored as cells. I convert these values to decimal and convert the cells into matrix form.

(iv) I retrieve the pss sequence values only when both pss_tvalid and pss_tready are high.

15. When verifying the ILAData results with the reference MATLAB code, I encountered an error resulting in a zero plot.



16.To verify the sss sequence results, I have written a MATLAB code following these steps:

(i) Firstly, I convert the .CSV file into a table using the readtable function in MATLAB.

(ii) Then, I extract the columns sss_tvalid, sss_tready, and sss_tdata from the table using the dot operator (tablename.column_name).

(iii) The columns sss_tvalid, sss_tready, and sss_tdata contain hexadecimal values and are stored as cells. I convert these values to decimal and convert the cells into matrix form.

(iv) I retrieve the sss sequence values only when both sss_tvalid and sss_tready are high.

17. When verifying the ILAData results with the reference MATLAB code, I encountered an error resulting in a zero plot.