Tutorial 3
Update MicroBlaze Platform with a Custom Coprocessor (3-register adder) connected via AXI-Lite – HW project (Vivado)

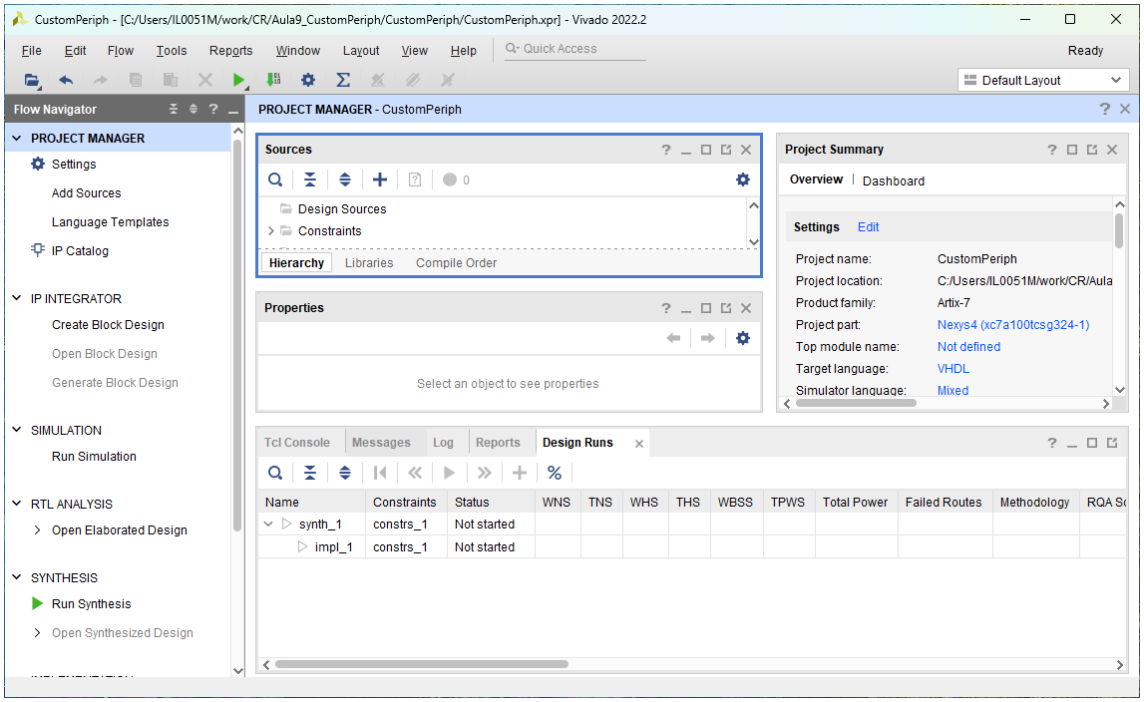
Vivado version: 2022.2

NOTE: this guide requires a MicroBlaze platform implementation (a *source project*) as the one described in **Tutorial_MicroBlaze_Polling_Interrupts.pdf**. It is important that the file path to the source project is short and has no spaces or other special characters. If that is not the case, two options are available.

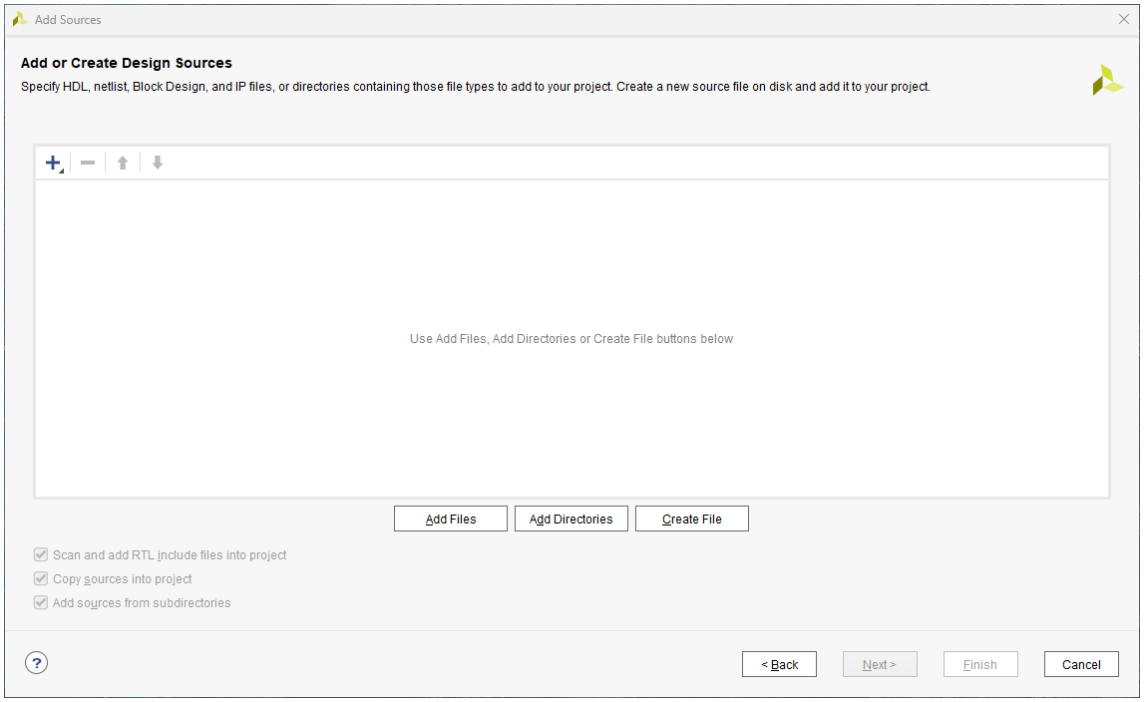
- Open the project and save in a new location – NOT TESTED.
- Implement the MicroBlaze platform from scratch, following tutorials Tutorial_MicroBlaze_platform.pdf and Tutorial_MicroBlaze_Polling_Interrupts – takes about 10-20 minutes.

1. Start **Vivado**. Create a new project. Add no sources. Select the **Nexys** board.

2. Click on the **'+'** button (**Add sources**). Click **Next**.



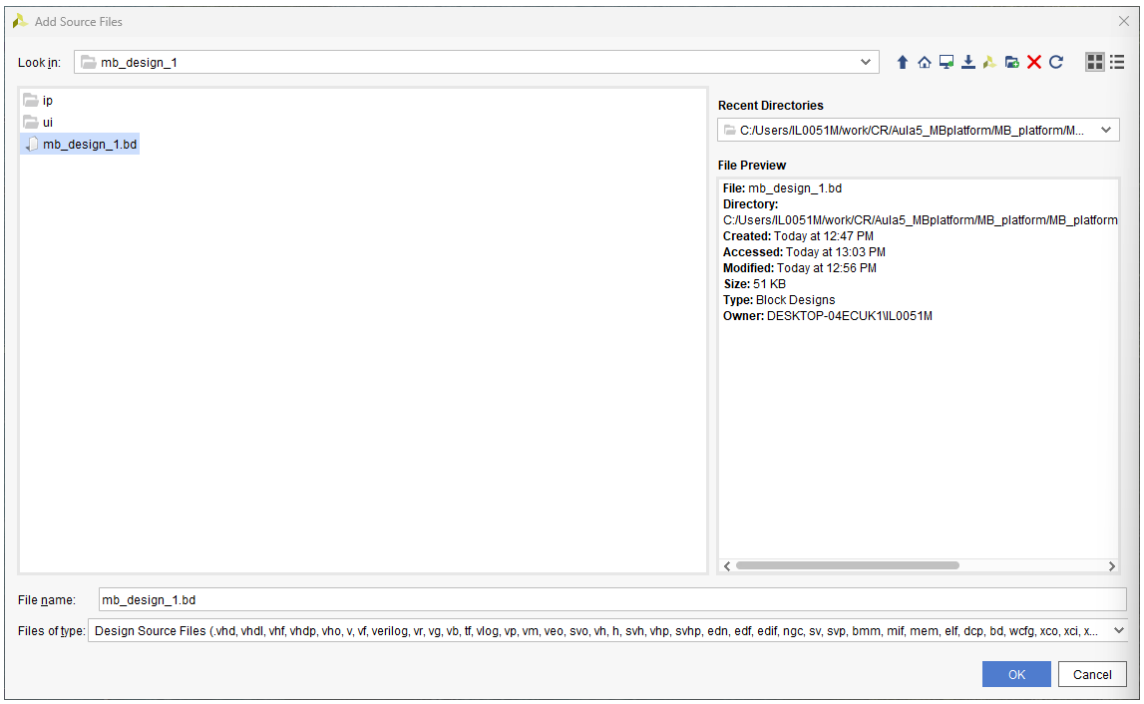
3. Select **Add Files**.



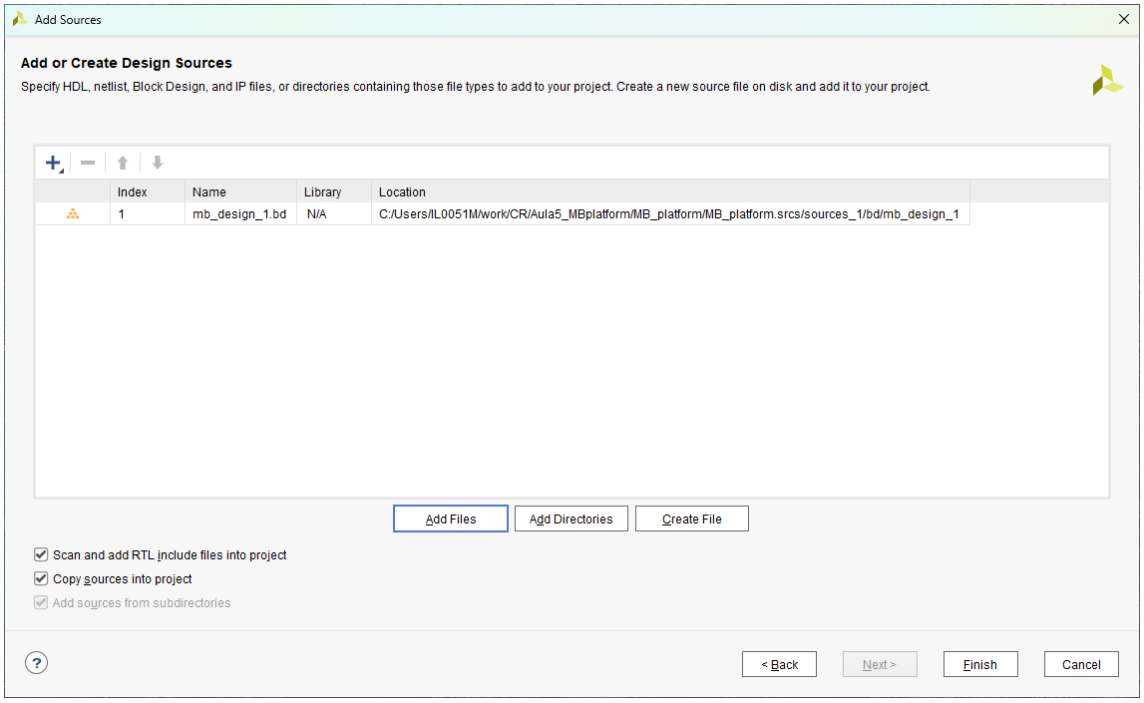
4. Navigate to find the **.bd** file of the source project. Click **OK.**

   The file can be found in:
   `<project folder>/MB_platform.srcs/sources_1/bd/mb_design_1/mb_design_1.bd`

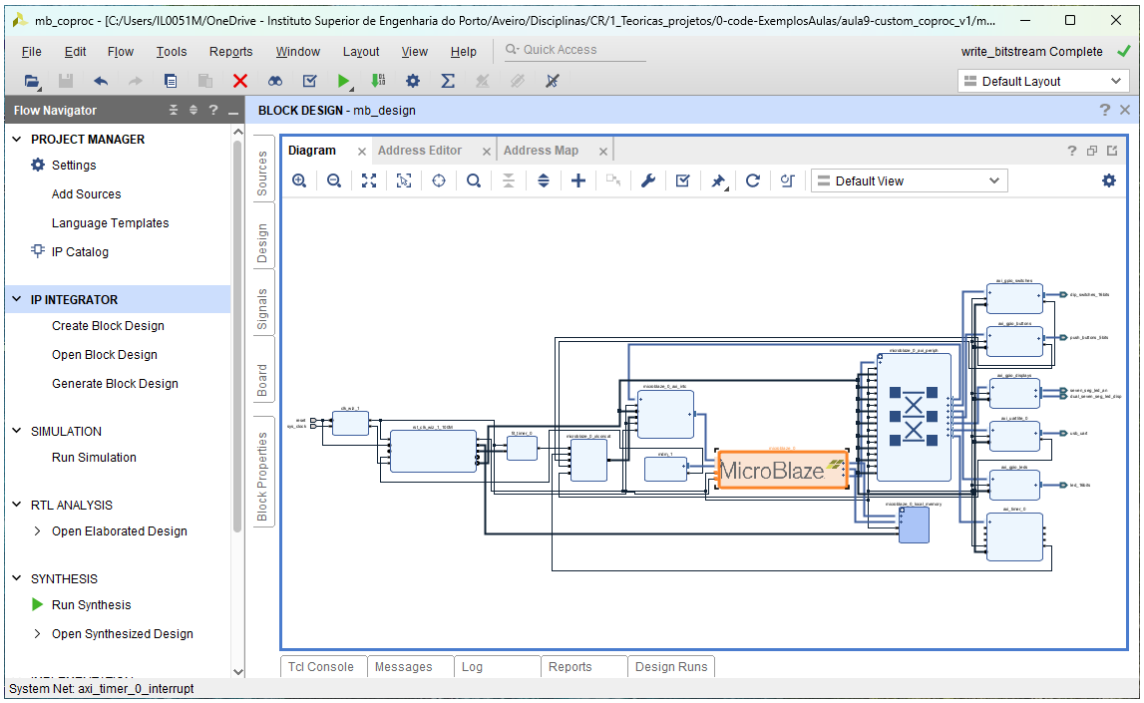
   For example:
   `C:\Users\user\work\CR\Aula5_MBplatform\MB_platform\MB_platform.srcs\sources_1\bd\mb_design_1\mb_design_1.bd`



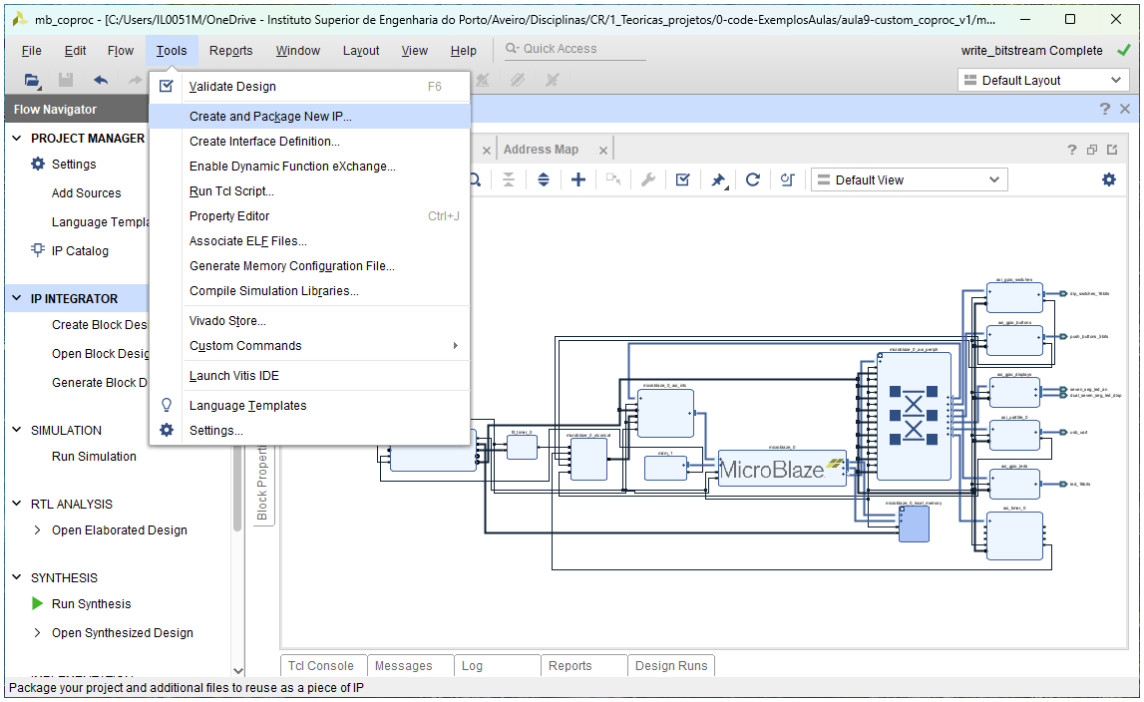
5. Make sure that the "**Copy sources into project**" option is selected. Click **Finish**.



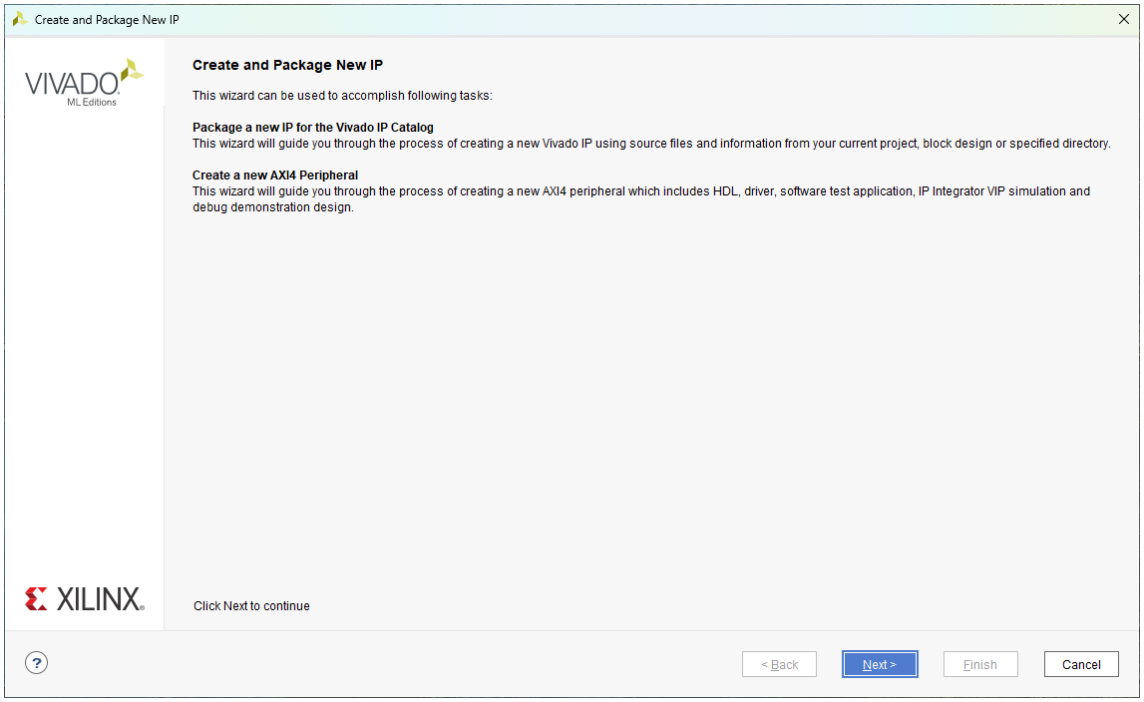
6. Once loaded, open the design. The block diagram of the platform should be presented as shown.



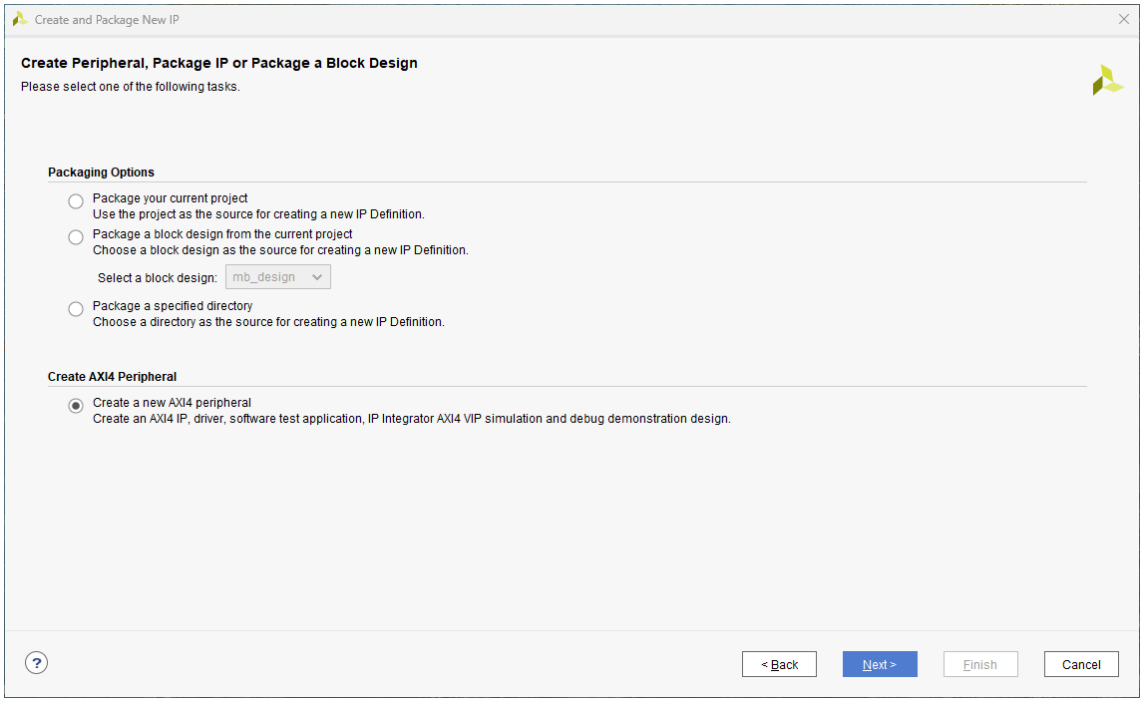
7. Click **Tools** > **Create and Package IP**.



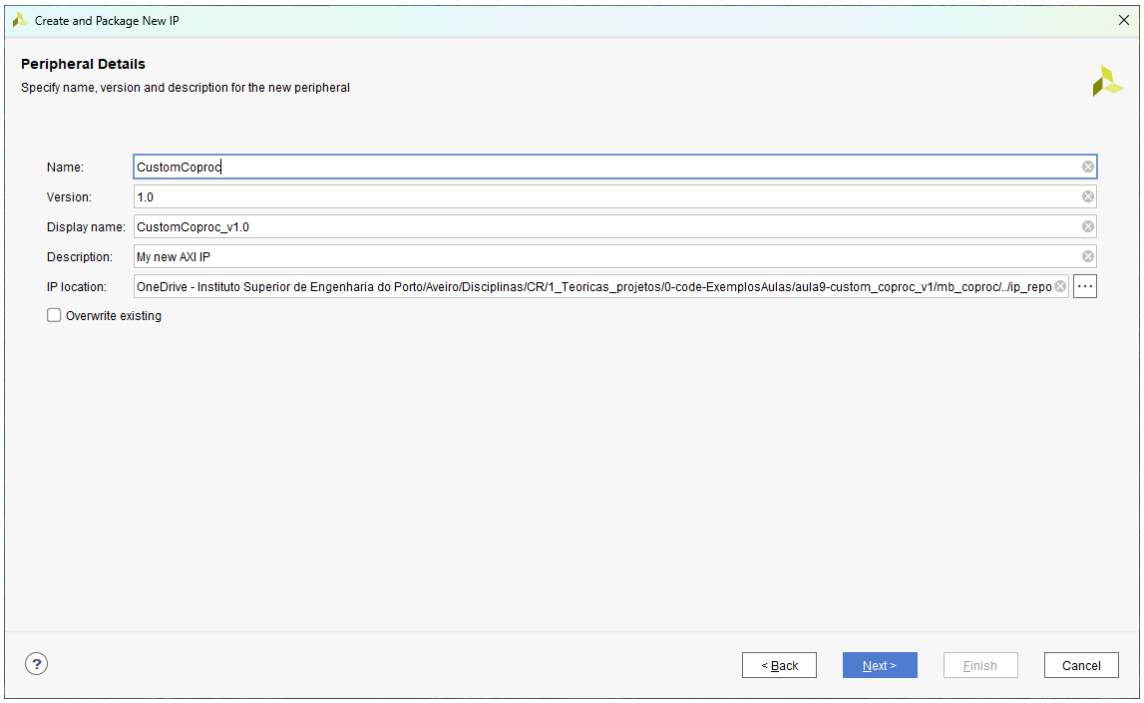
8. Click **Next**.



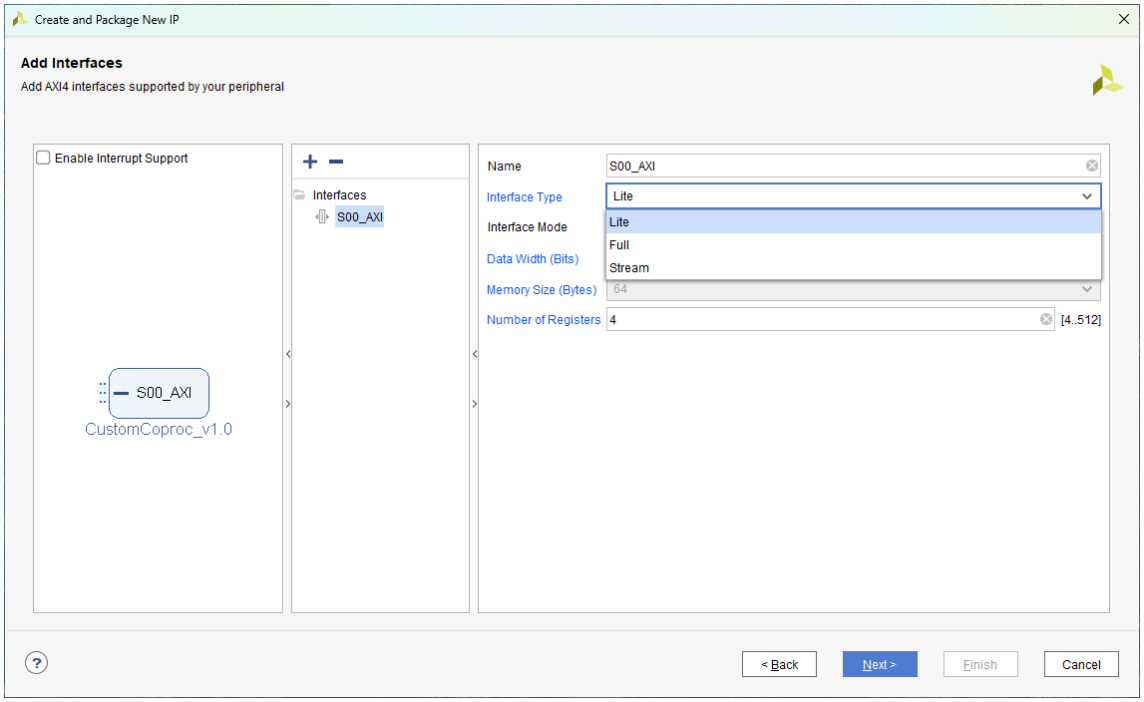
9. Select **Create AXI4 Peripheral**.



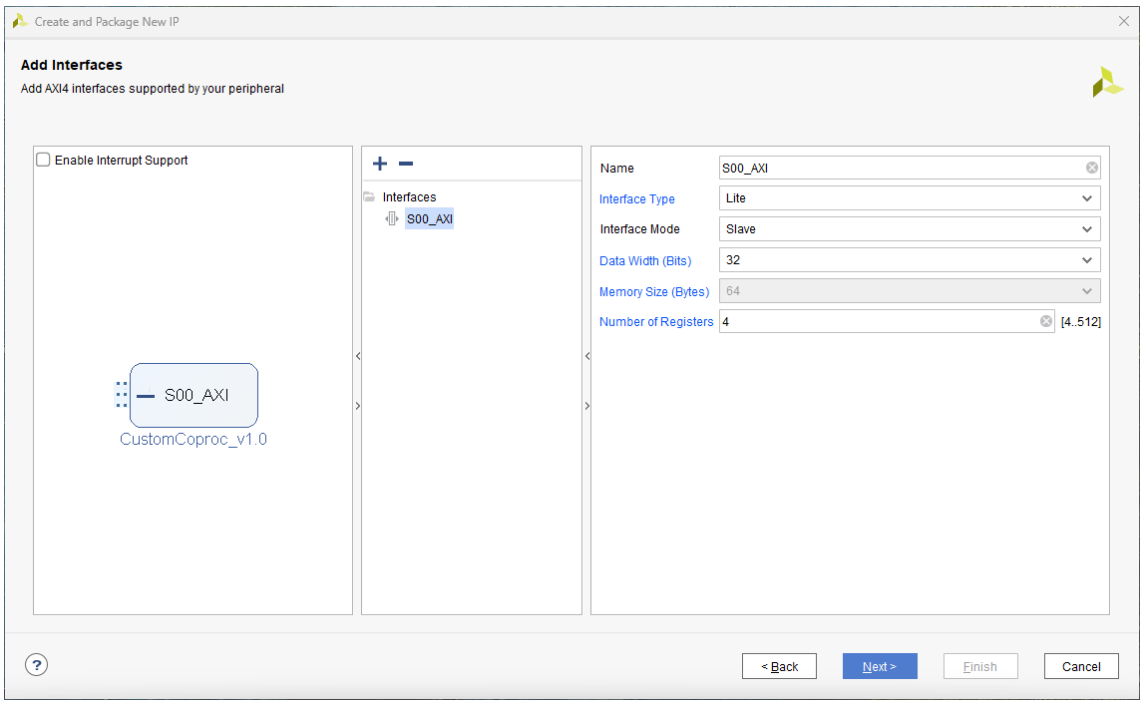
10. Assign the name **CustomCopr** to the new IP block at your discretion. Alternate names are possible but will require small edits on code later on.



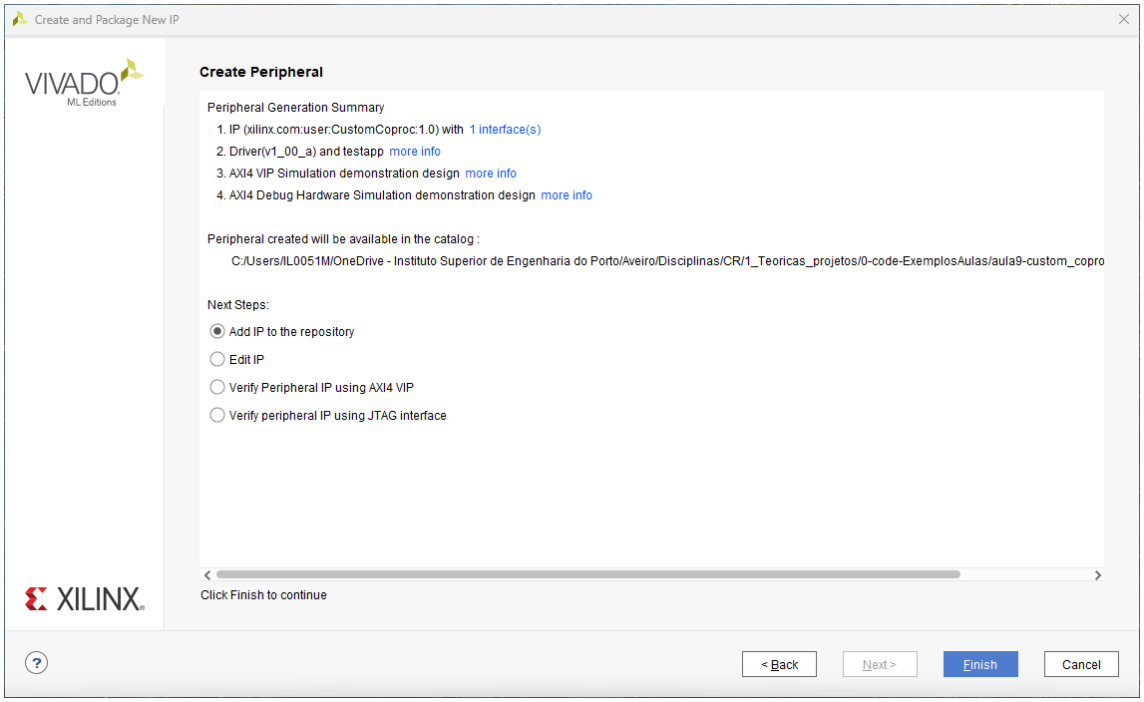
11. Make sure that the **Lite** interface is selected; this is the bus type we will be using in this work. Click **Next**.



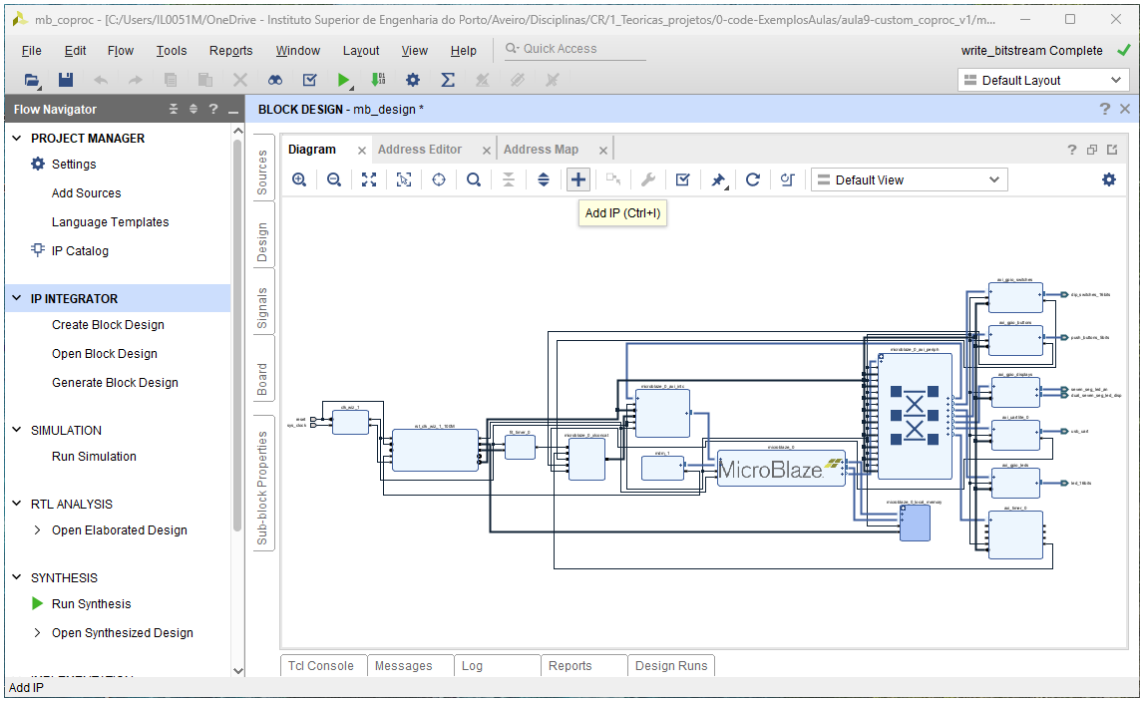
12. Additionally, make sure the **Slave** option is selected.
    Finally, depending on the module you are designing, you may also need to change the **Number of Registers**. For the application in mind (3-register adder), we need **4**. Click **Next**.



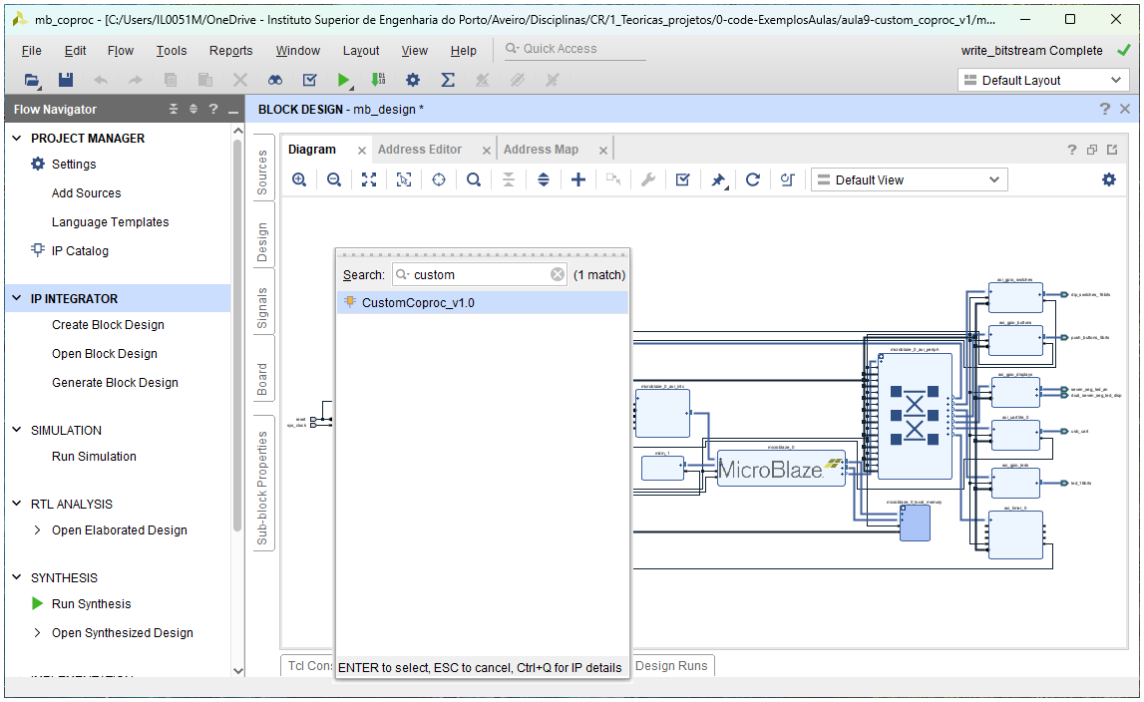
13. Add the IP core to **Add IP to the repository**. Click **Finish.**



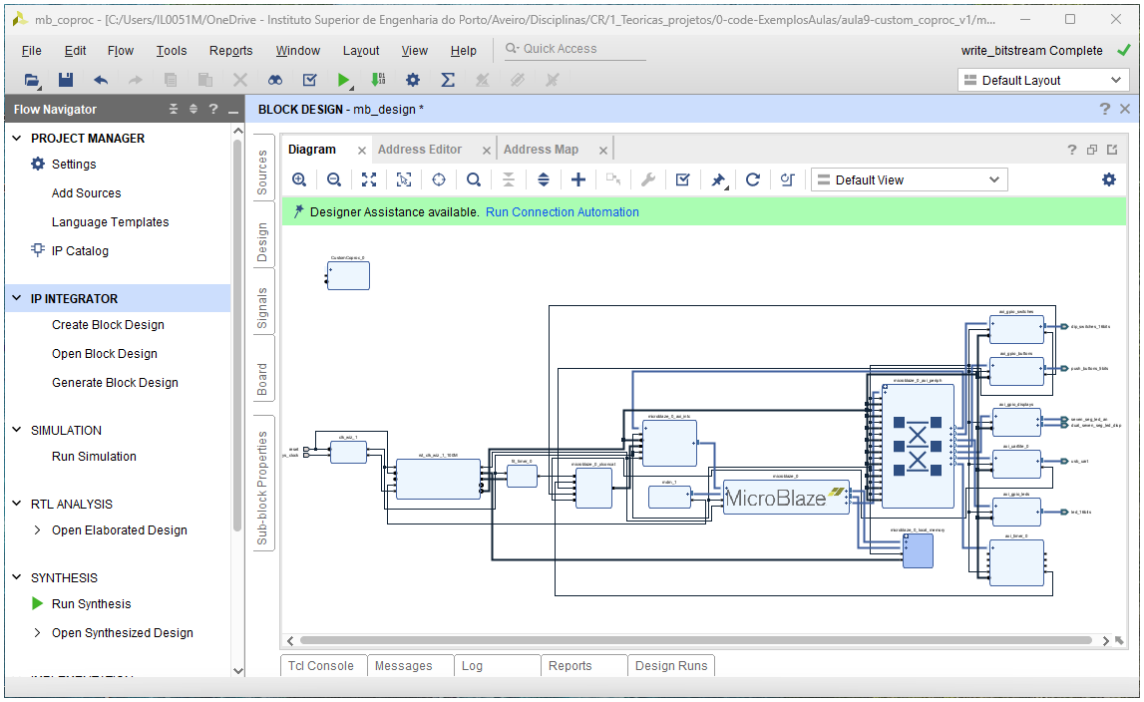
14. Click button **Add IP**.



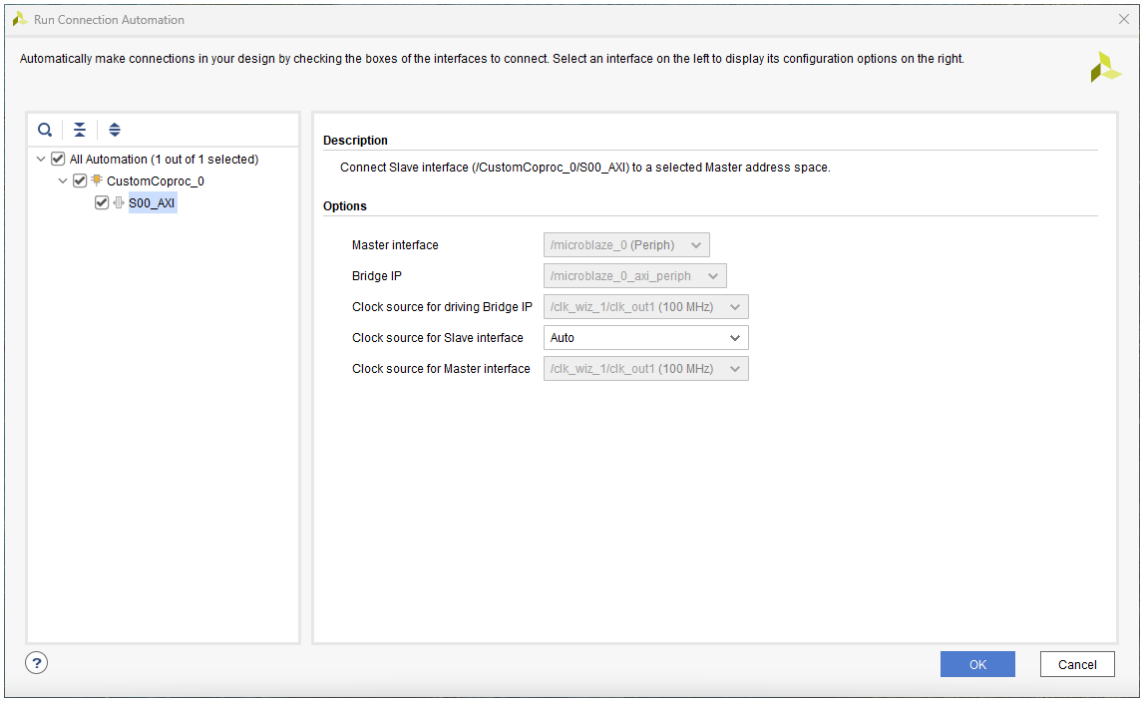
15. Search for the name of your module.



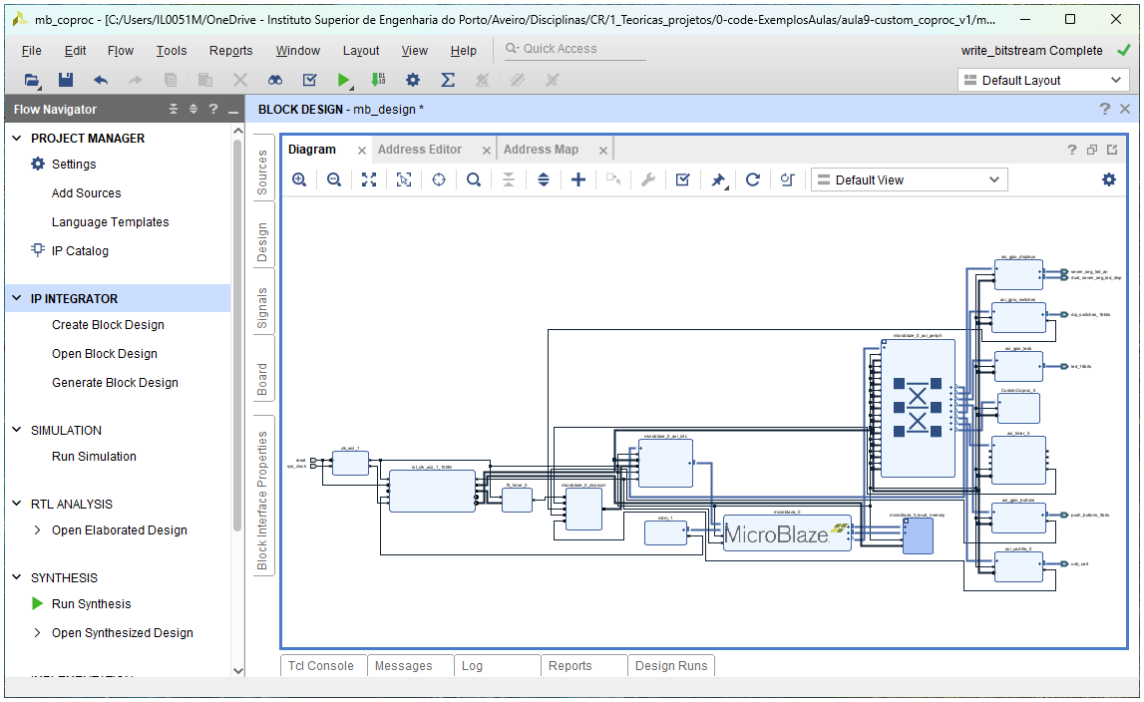
16. The new module shows up. Click on **Run Connection Automation**.



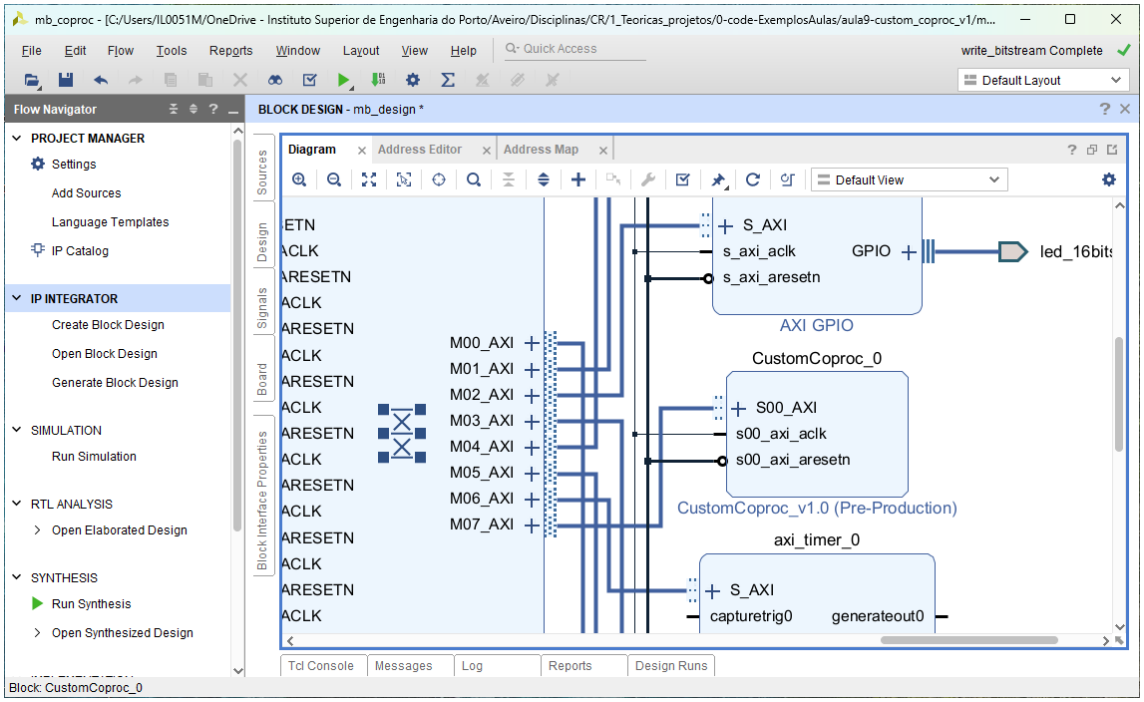
17. Select the appropriate tick box. Click **OK**.



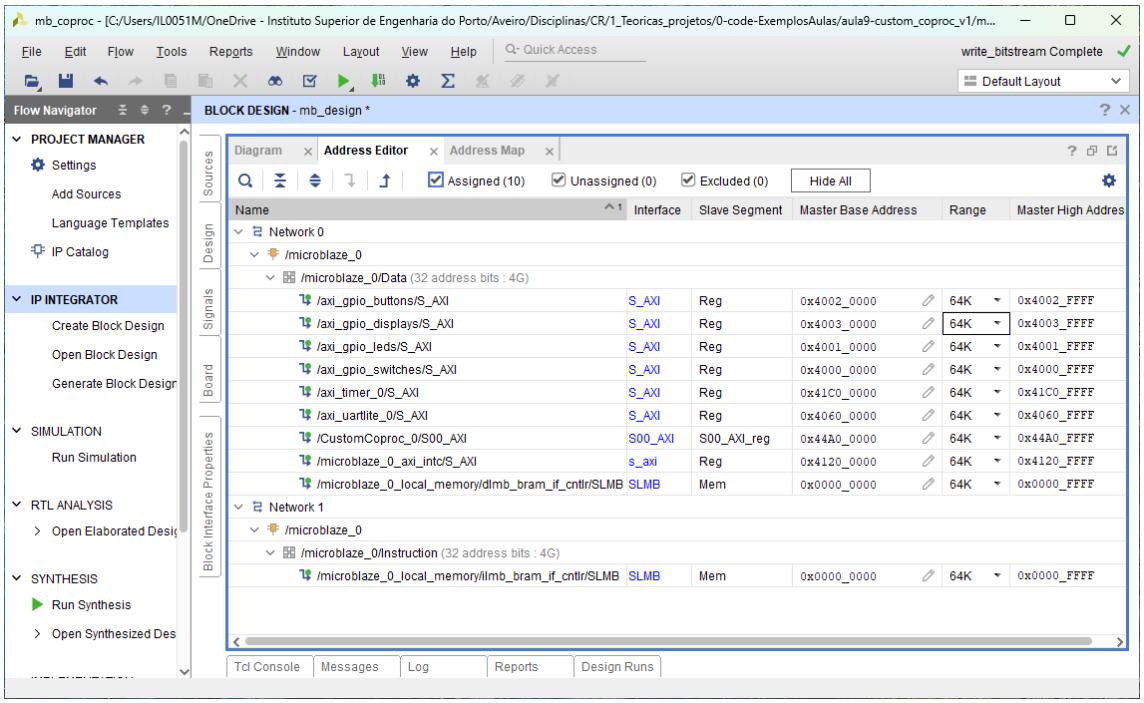
18. Click on **Regenerate Layout** (optional).



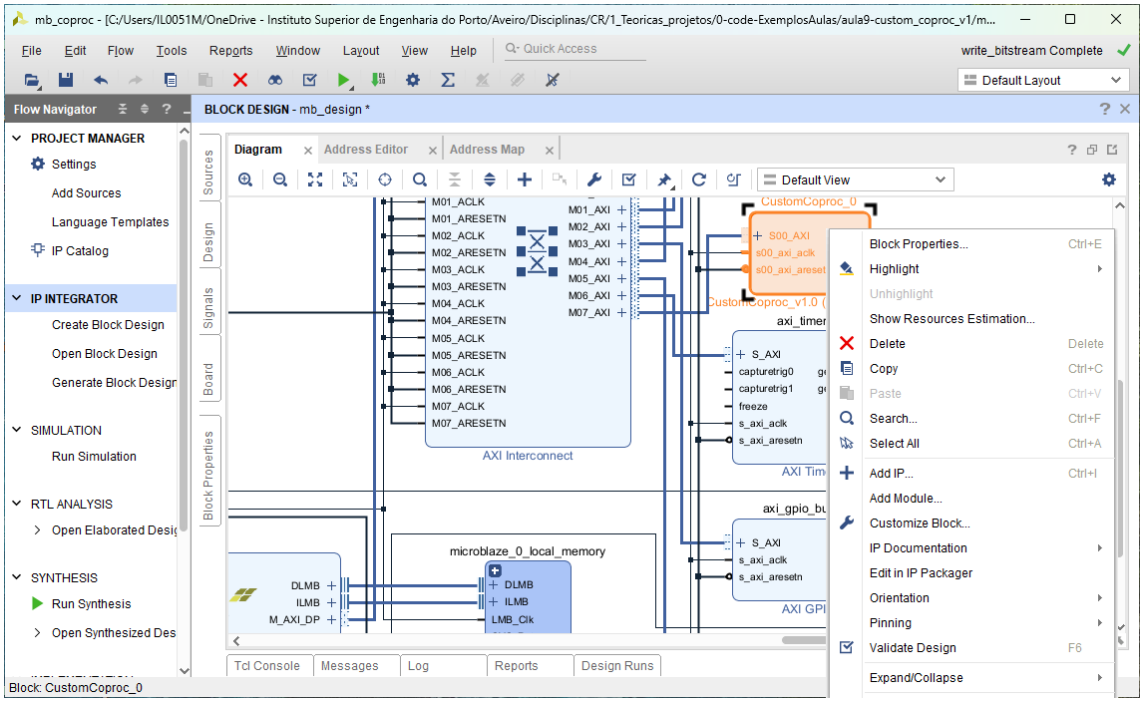
19. Click in the **Address Editor** tab.



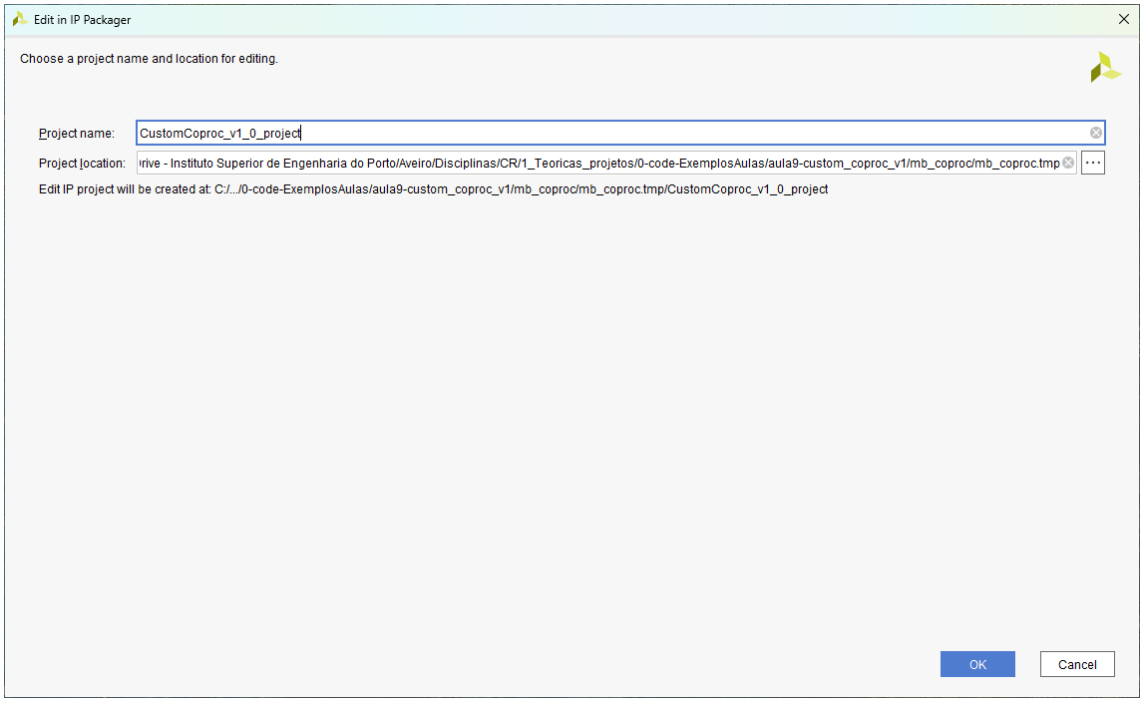
20. Observe the **Master Base Address** and **Range** assigned to your IP core.
    - Notice that your module has been assigned 64Kbytes, but in practice it only requires 4 registers times 4 bytes = 16 bytes. Although highly inefficient, we will leave 64K for convenience.
    - Register addresses are consecutive.
      Register 0: 0x44A0_0000; Register 1: 0x44A0_0004; Register 2: 0x44A0_0008; Register 3: 0x44A0_00012



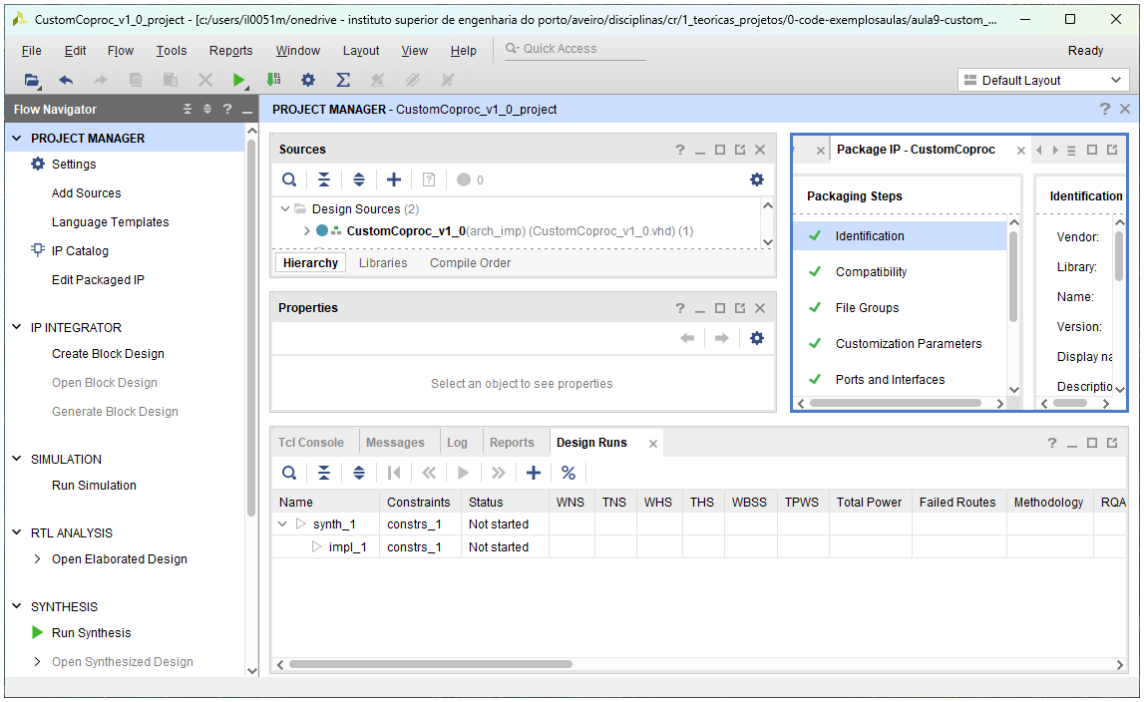
21. Return to the **Diagram** tab. Select the **Edit in IP Packager**.



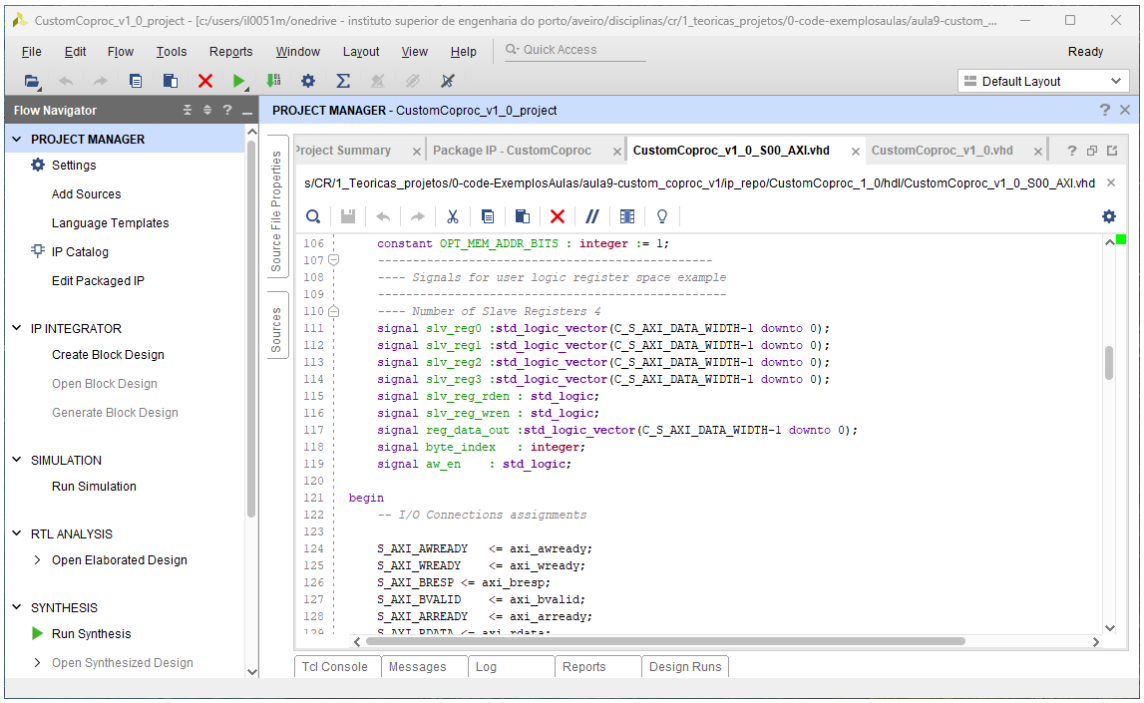
22. Click **OK**.



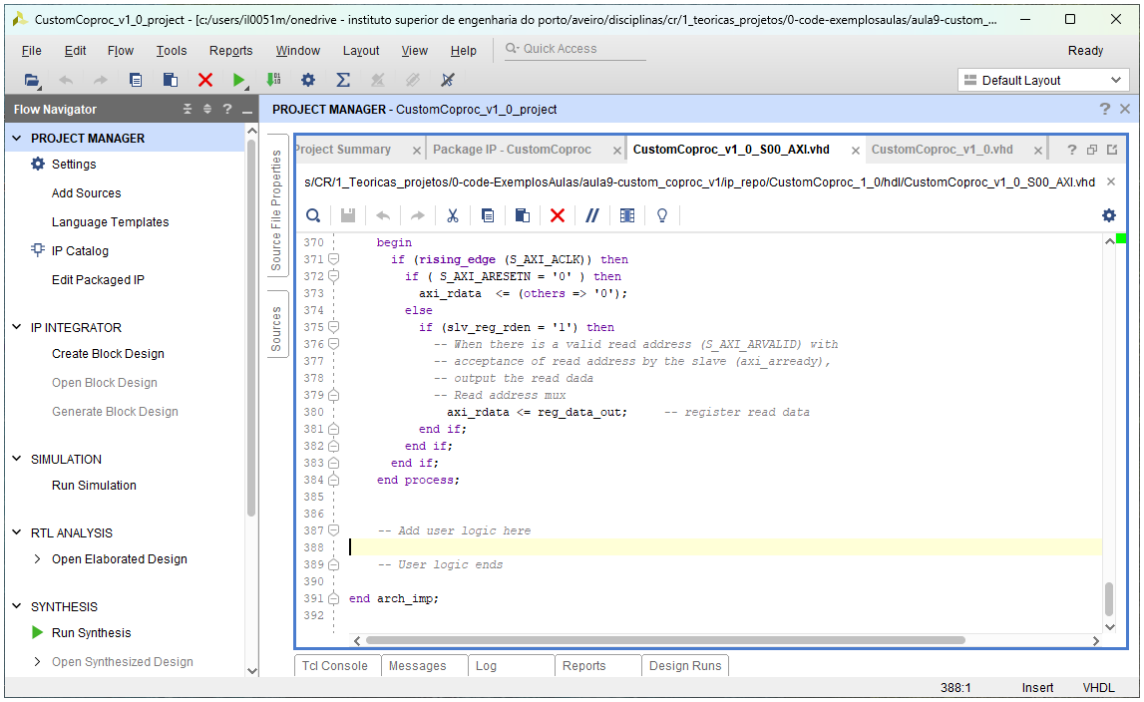
23. A new Vivado window will open.



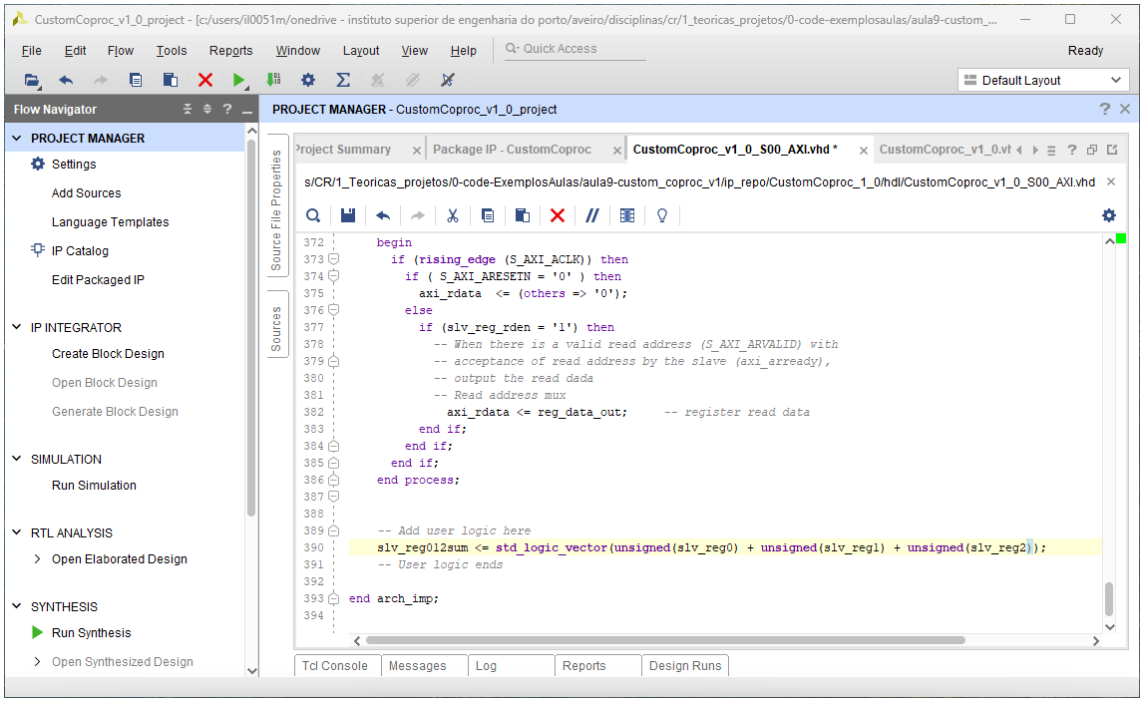
24. Open the VHDL files, **CustomCoproc_v1_0_S00_AXI.vhd** and **CustomCoproc_v1_0.vhd**. Focus on the first.



25. Navigate to the end of the file. The application logic goes here.

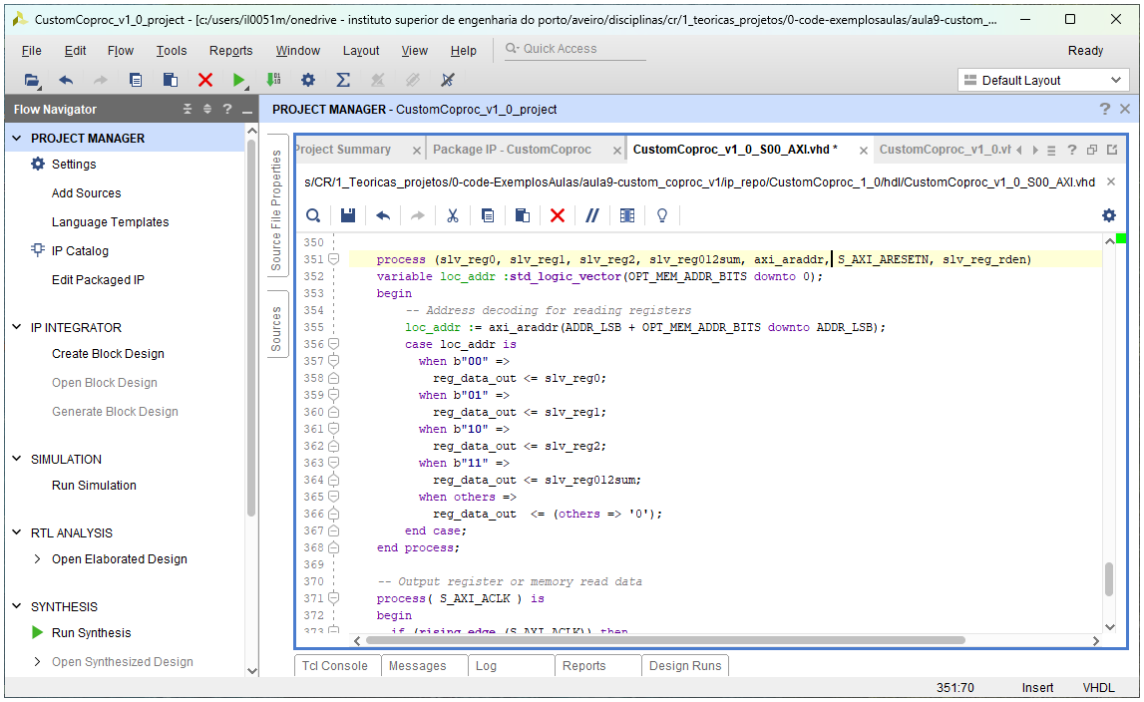


26. Implement the application logic.



A new signal **slv_reg012sum** stores the sum of registers **slv_reg0, slv_reg1** and **slv_reg2.**

27. Replace **slv_reg3** by **slv_reg012sum** in the case **when b"11"** and in the sensitivity list.

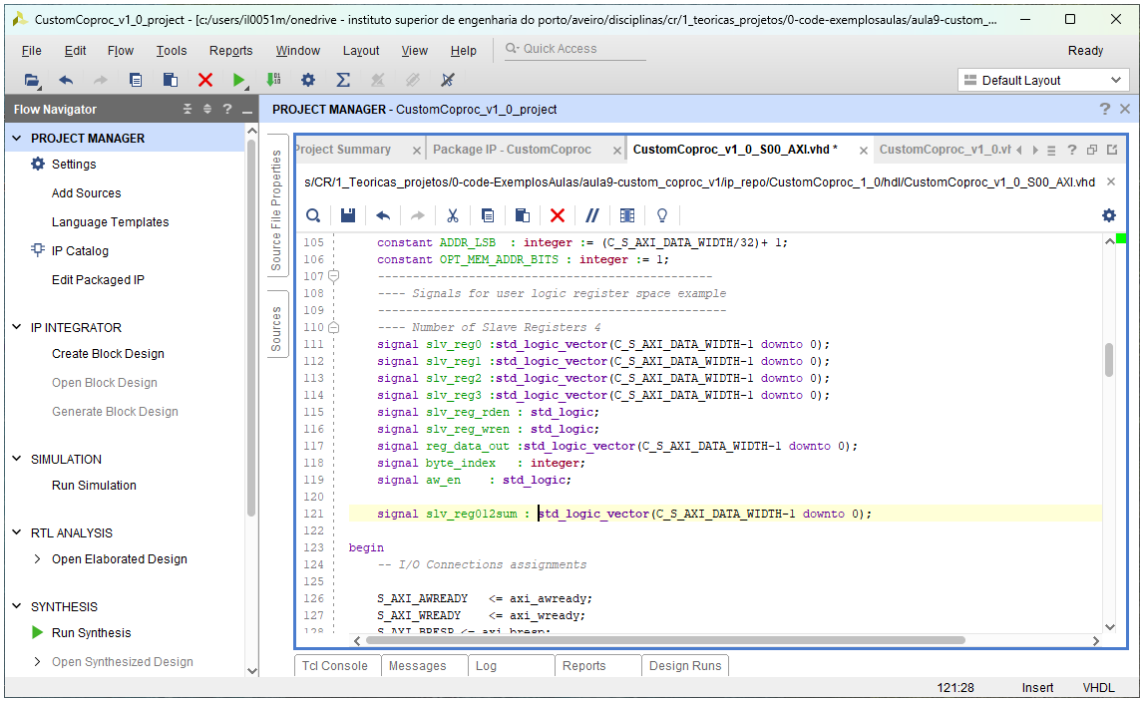


Summary of alterations:

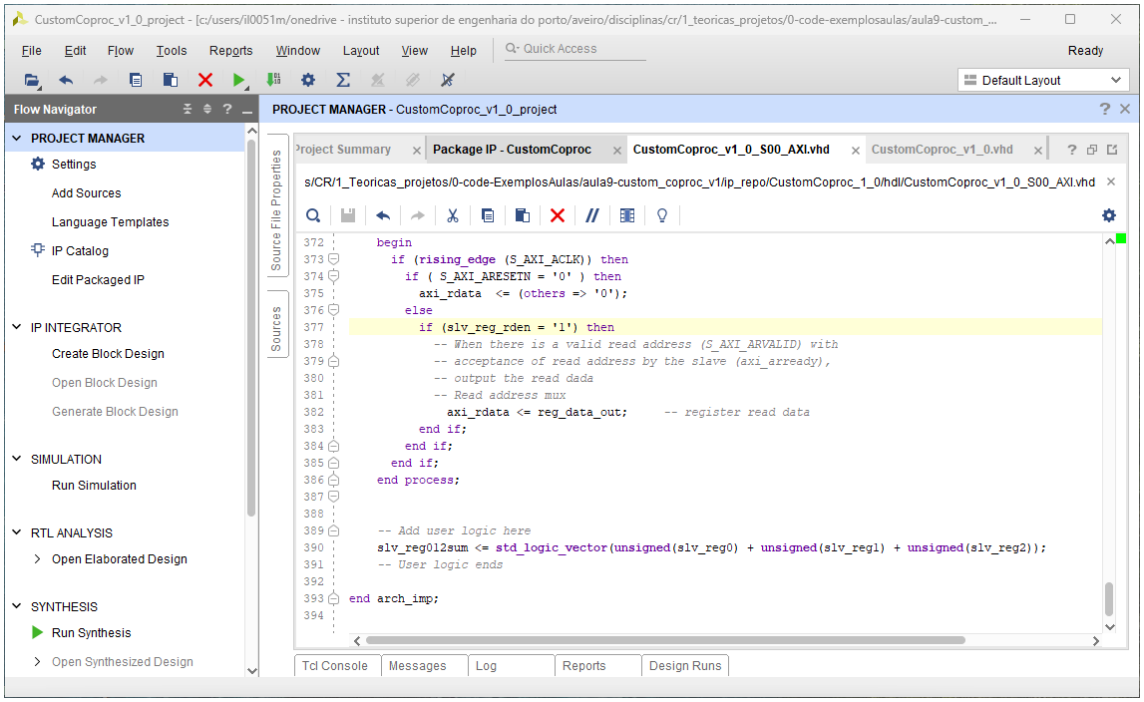
c:/Iouliia/Aulas/2020-2021/CR/T/Aula9_CustomHW/ip_repo/CustomCopr_1.0/hdl/CustomCopr_v1_0_S00_AXI.vhd

```
355         case loc_addr is
356           when b"00" =>
357             reg_data_out <= slv_reg0;
358           when b"01" =>
359             reg_data_out <= slv_reg1;
360           when b"10" =>
361             reg_data_out <= slv_reg2;
362           when b"11" =>
363             reg_data_out <= slv_reg012Sum; --slv_reg3;
364           when others =>
365             reg_data_out  <= (others => '0');
366         end case;
367       end process;
368
369       -- Output register or memory read data
370       process( S_AXI_ACLK ) is
371       begin
372         if (rising_edge (S_AXI_ACLK)) then
373           if ( S_AXI_ARESETN = '0' ) then
374             axi_rdata  <= (others => '0');
375           else
376             if (slv_reg_rden = '1') then
377               -- When there is a valid read address (S_AXI_ARVALID) with
378               -- acceptance of read address by the slave (axi_arready),
379               -- output the read dada
380               -- Read address mux
381                 axi_rdata <= reg_data_out;     -- register read data
382             end if;
383           end if;
384         end if;
385       end process;
386
387
388       -- Add user logic here
389       slv_reg012Sum <= std_logic_vector(unsigned(slv_reg0) + unsigned(slv_reg1) + unsigned(slv_reg2));
390       -- User logic ends
```

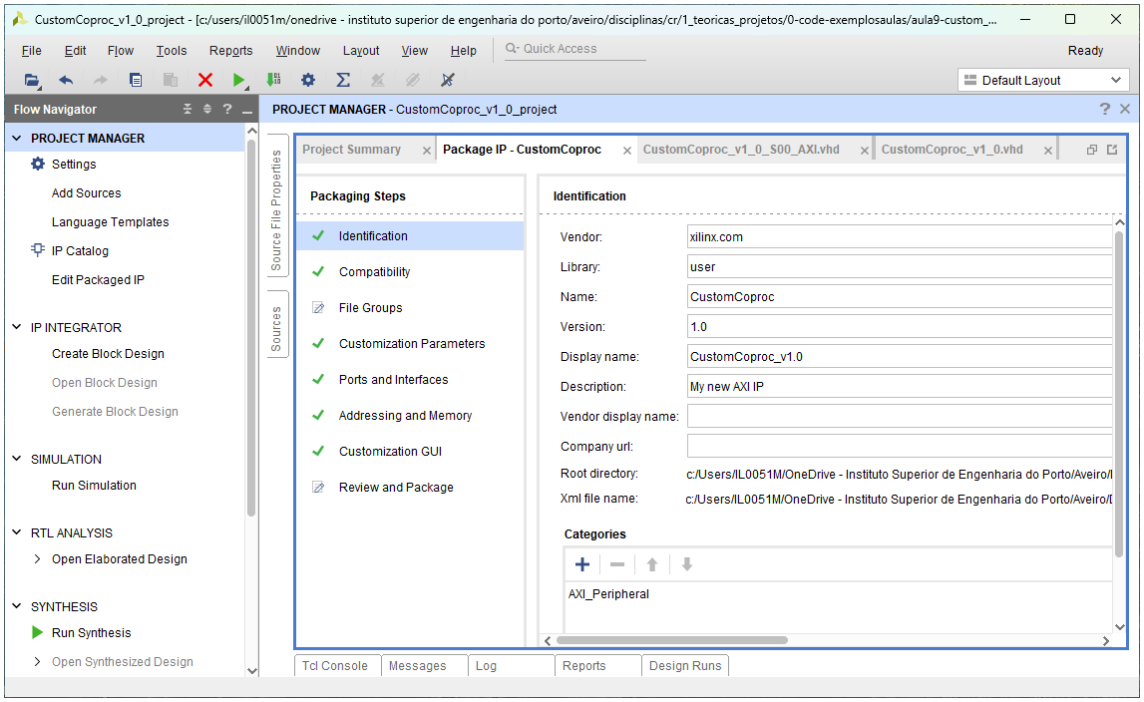
28. Add also the declaration for signal **slv_reg012sum**. Save the file.



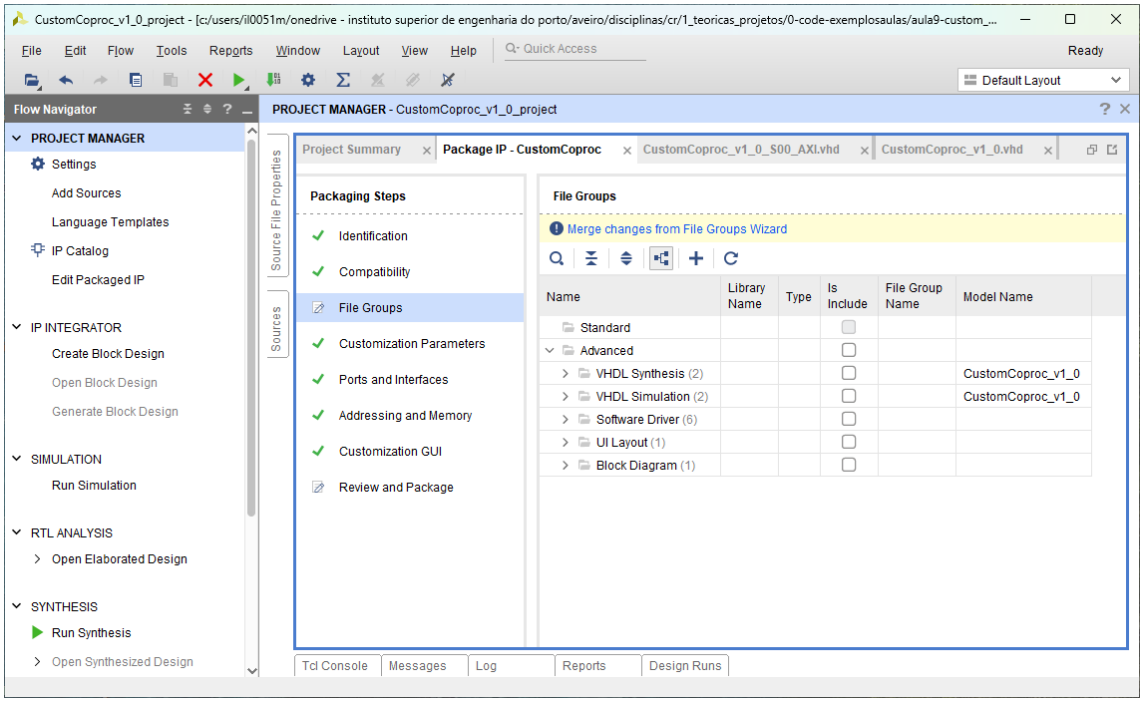
29. Click the **Package IP – CustomCoproc** tab.



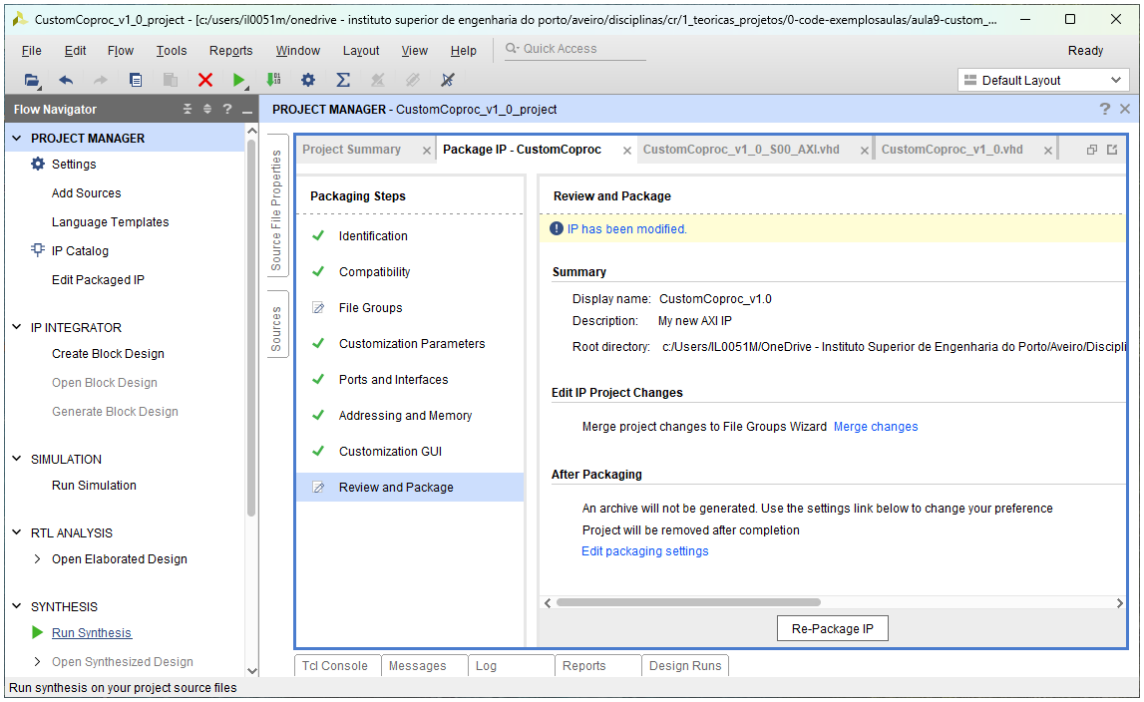
30. A tab with project details shows up. A menu is available on the left side.



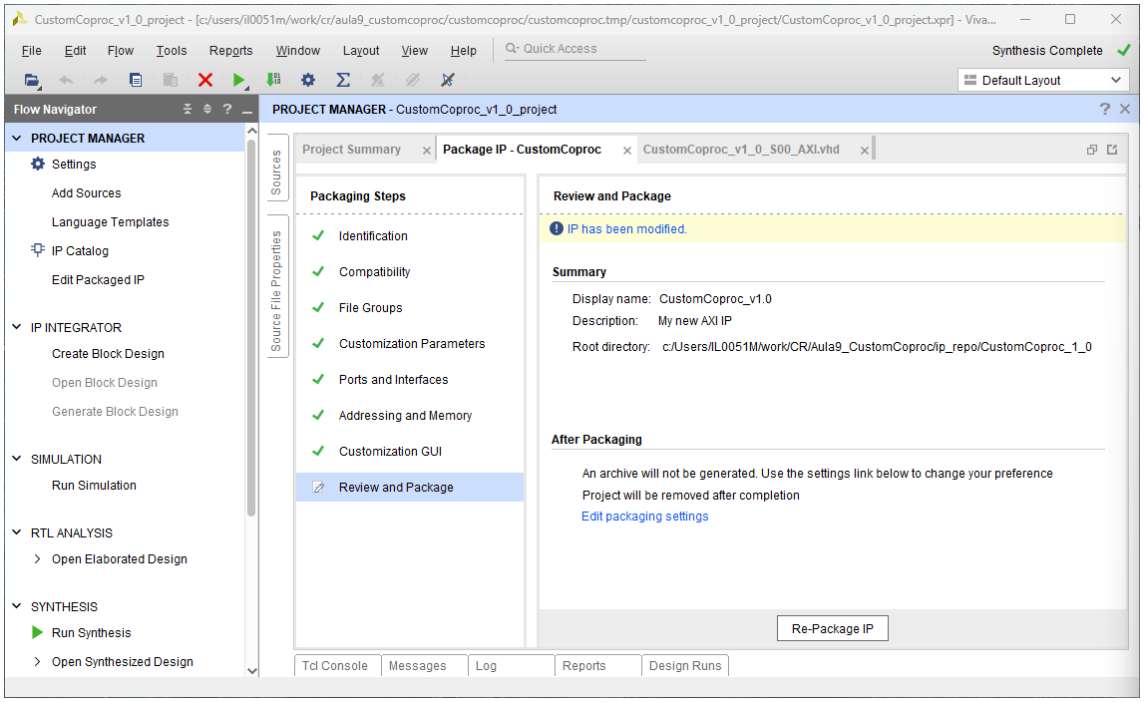
31. Select tab **File Groups**. Click on **Merge Changes from File Groups Wizard**.



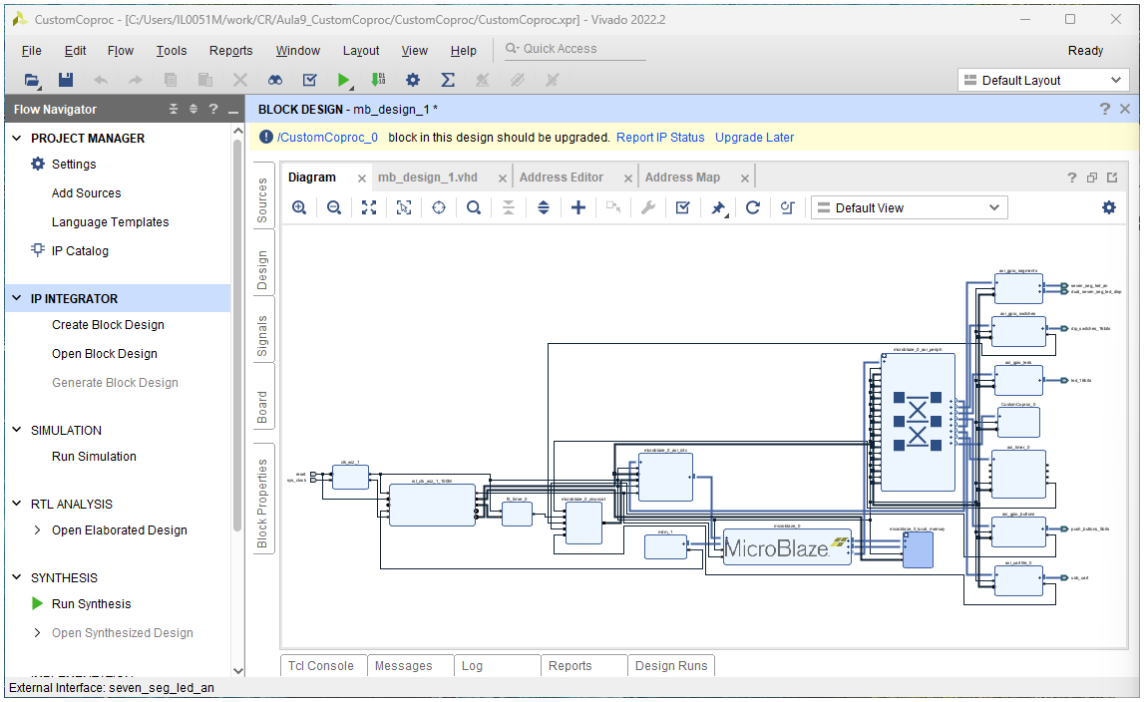
32. Run **Synthesis** (to prevent errors that may). If the module has errors, it is more convenient to detect and address them now.



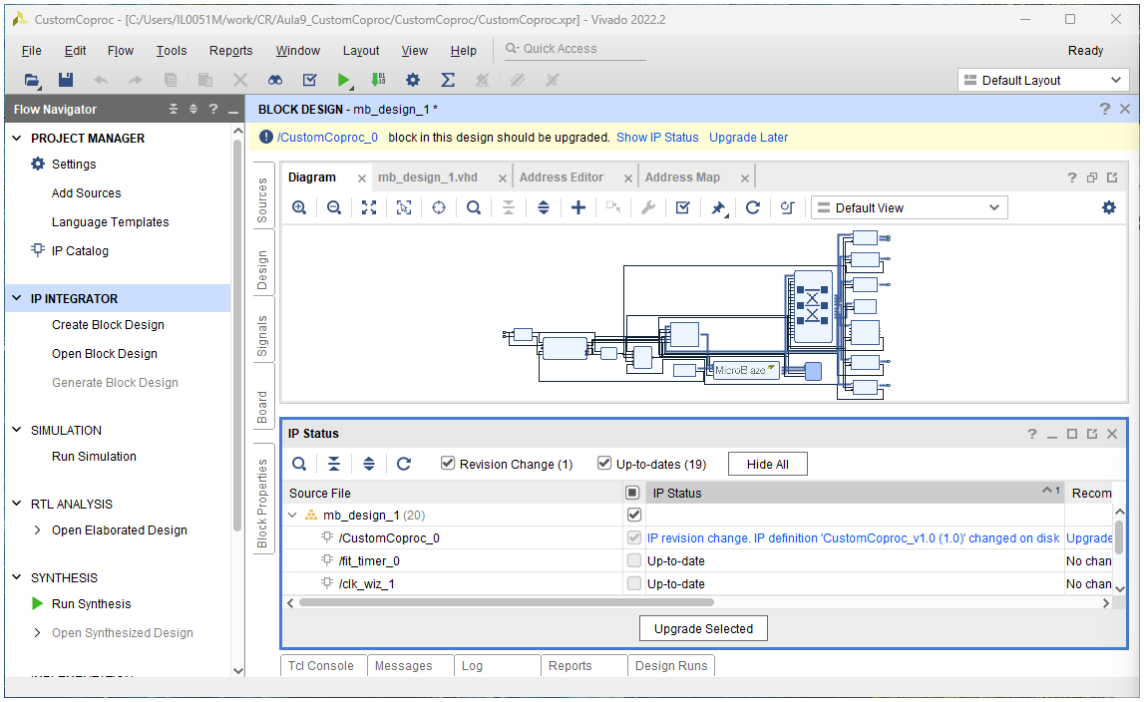
33. Navigate to the **Review and Package** tab. Click in **Re-Package IP**. Click **Yes** in the pop-up window. The second **Vivado** window will close, and you will return to the initial one.



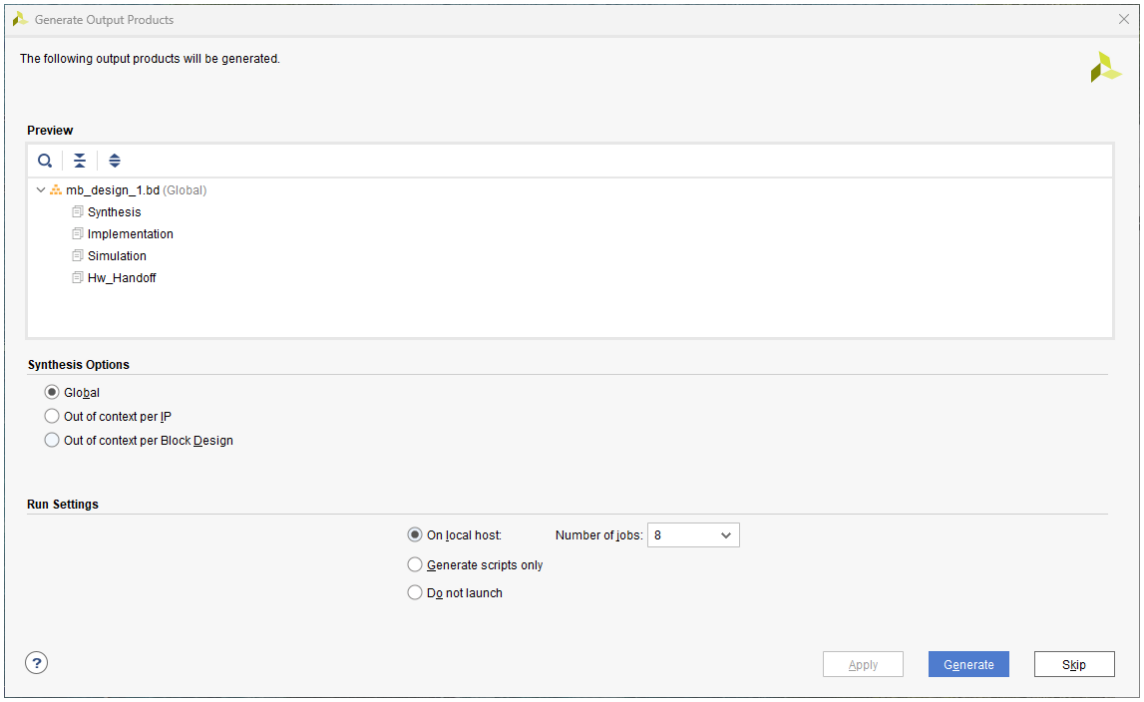
34. Back in the initial **Vivado** window, a new interface bar informs you that the **CustomCoproc** block has been updated. Click in **Report IP Status**.



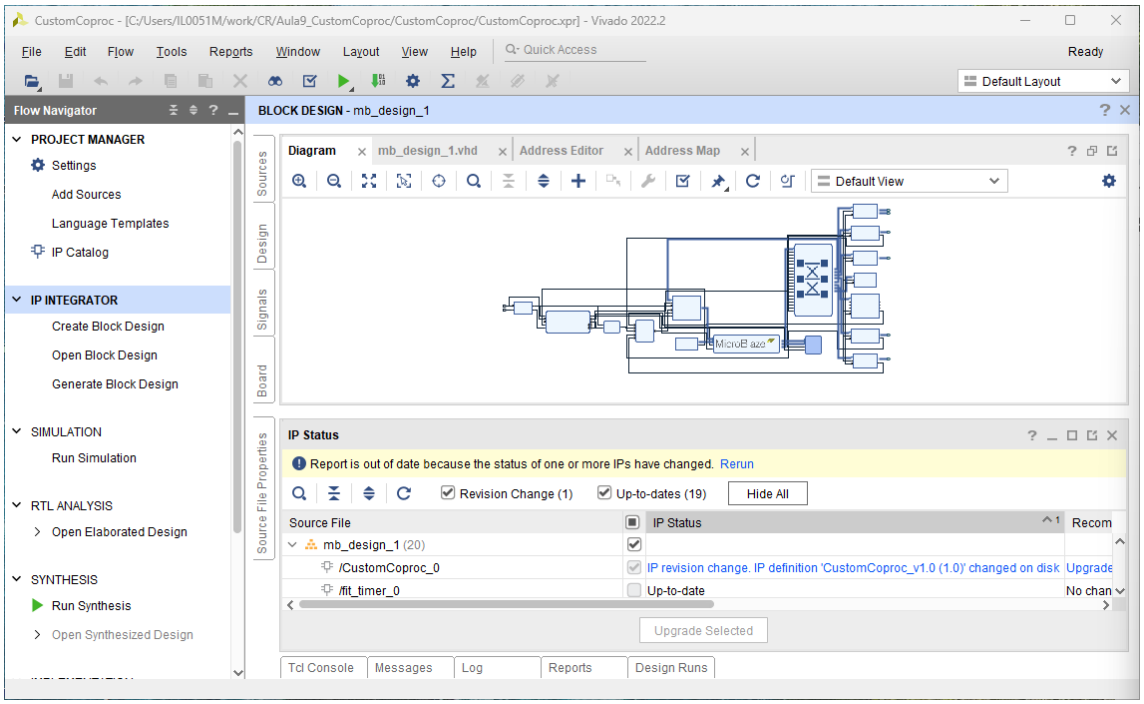
35. A panel names **IP Status** show up, listing the design blocks and their status. Notice that **CustomCoproc** block is reported revised. Click in **Upgrade Selected**. A pop-up window shows up once the process is over.



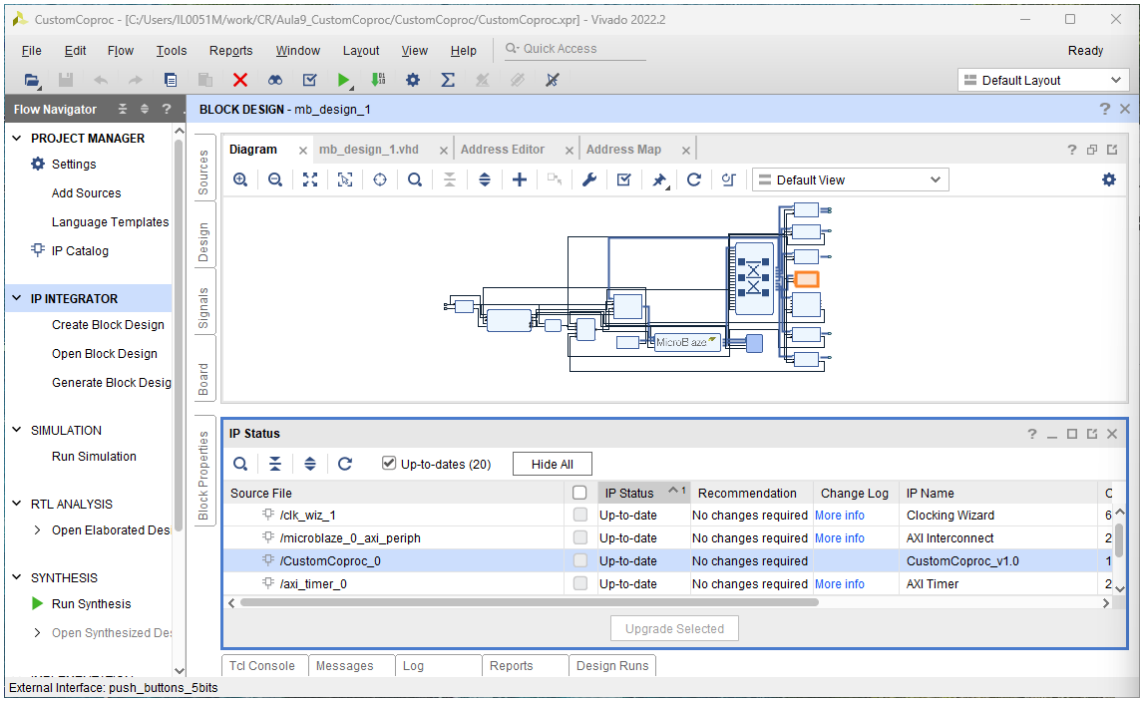
36. A window regarding generation of output products shows up. **DO NOT CLICK *GENERATE*!** Due to a bug in Vivado, close the window instead in the top right **X**.



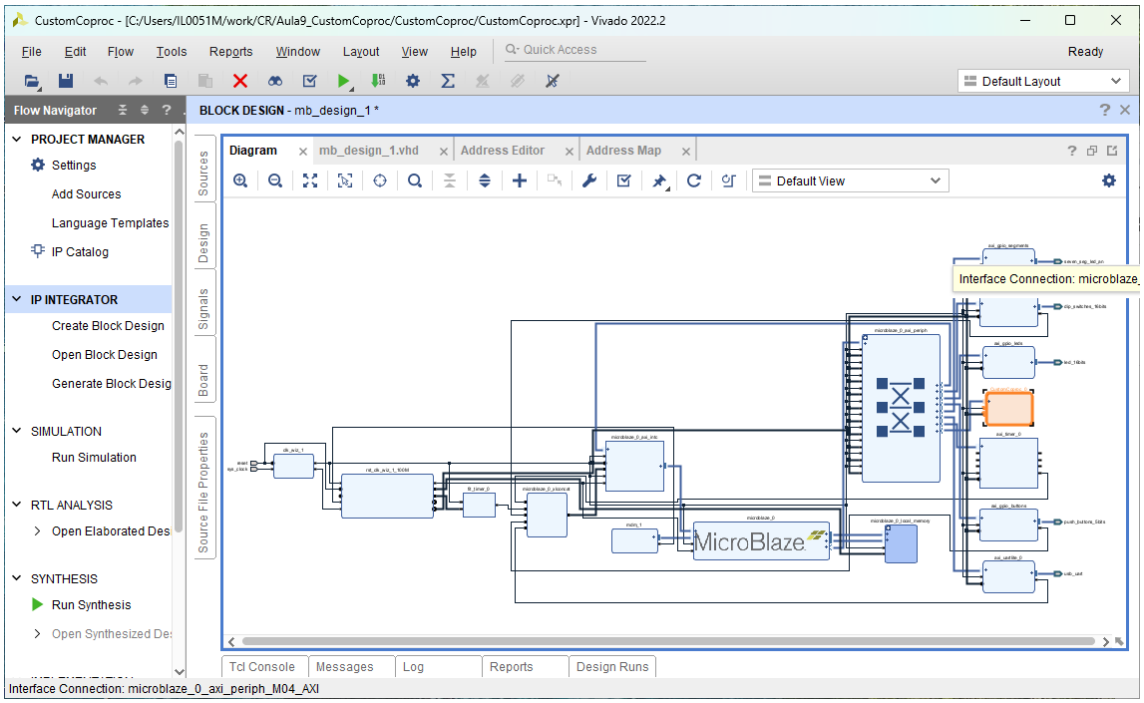
37. In the **IP status** panel, notice that the status for the **CustomCoproc** block has not been updated, as intended. Also, an interface bar has shown up. Click **Re-run**.



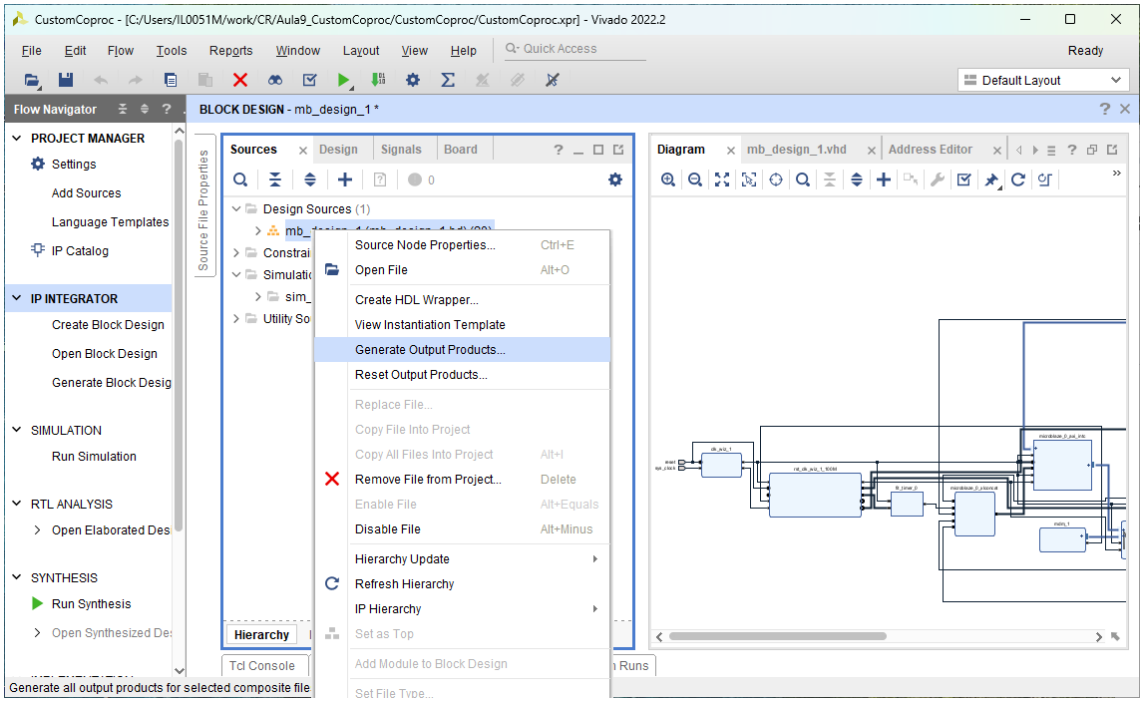
39. The status for the **CustomCoproc** block is now properly updated. Check this; the block's report may have moved to a different position in the list.



40. Click on **Validate Design**.



41. Click on **Generate Output Products**, and after that, on **Create HDL Wrapper**.



42. Click on **Generate Bitstream.**

43. Click on **File** > **Export** > **Export Hardware.**

Congratulations!

Sources: AMD Xilinx documentation.