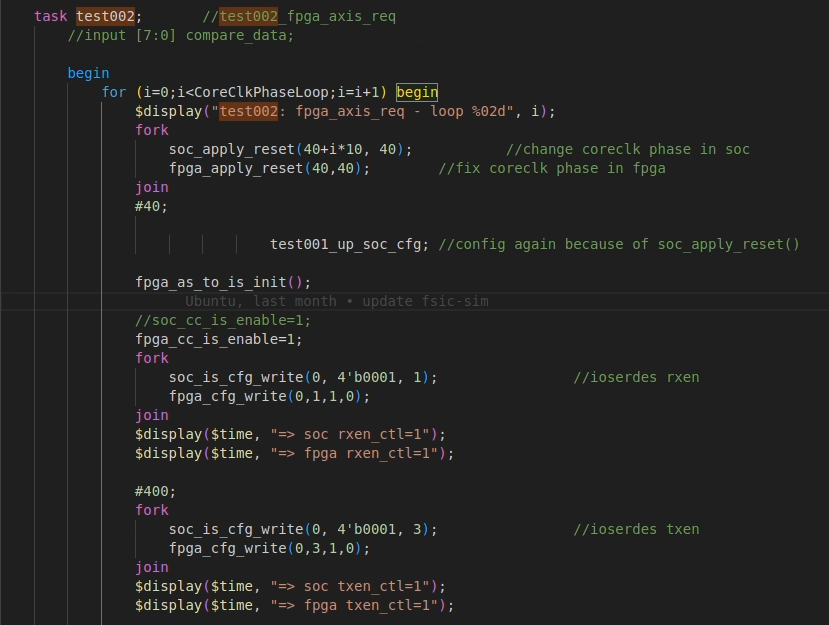
Advanced SoC Lab-fsic\_sim

111061545 陳揚哲

1. Show the code that you use to program configuration address [‘h3000\_5000].

There are multiple steps in every task: First, it will reset the system. Second, the system is initialized. Third, it enables the RX and TX side in system.



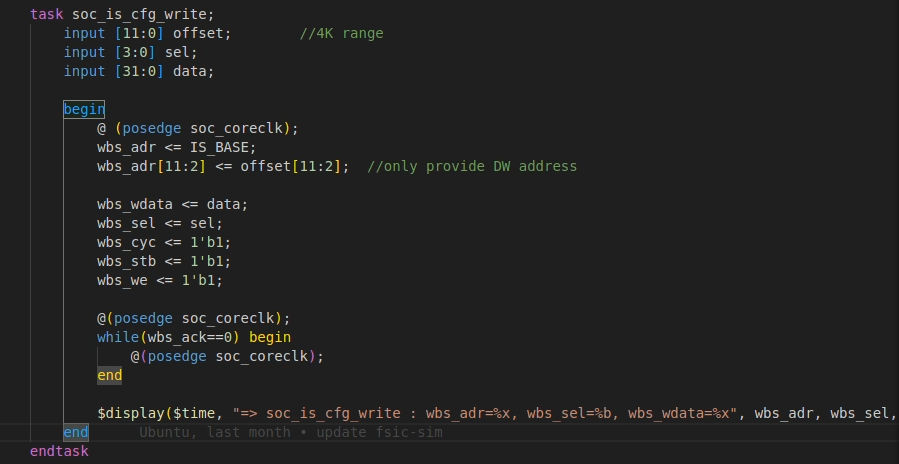
**RX**

**TX**

**initialize**

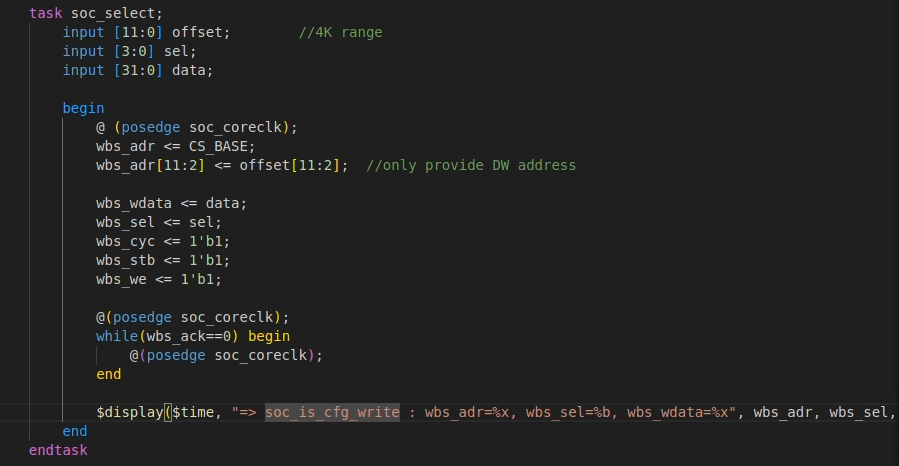
**reset**

From the step which is enable the RX and TX side in system, we can find out the how the system programs the specific address.



Therefore, I add new task changing the target address to [‘h3000\_5000].

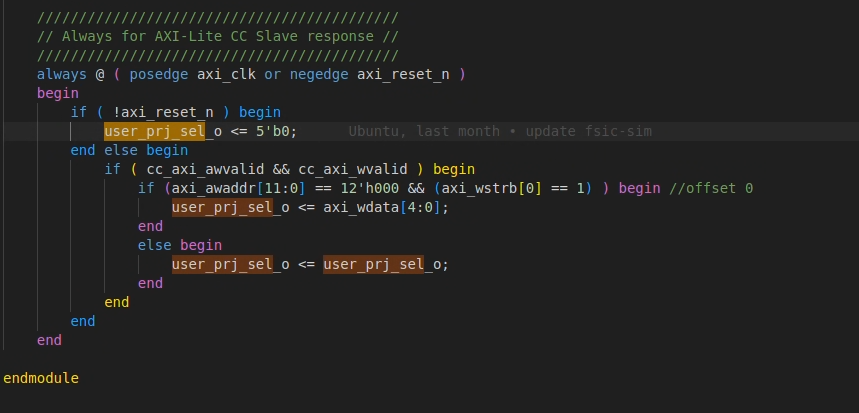
https://cdn.discordapp.com/attachments/896787580941639684/1221198917576818869/image.png?ex=6611b51e&is=65ff401e&hm=6961eaf02c5a33e2bf5fe55281c195d02825781b7b53a99c280b1ffd7c88bff1&=



1. Explain why “By programming configuration address [‘h3000\_5000], signal user\_prj\_sel[4:0] will change accordingly”? (Hint: trace code in config\_ctrl.v)

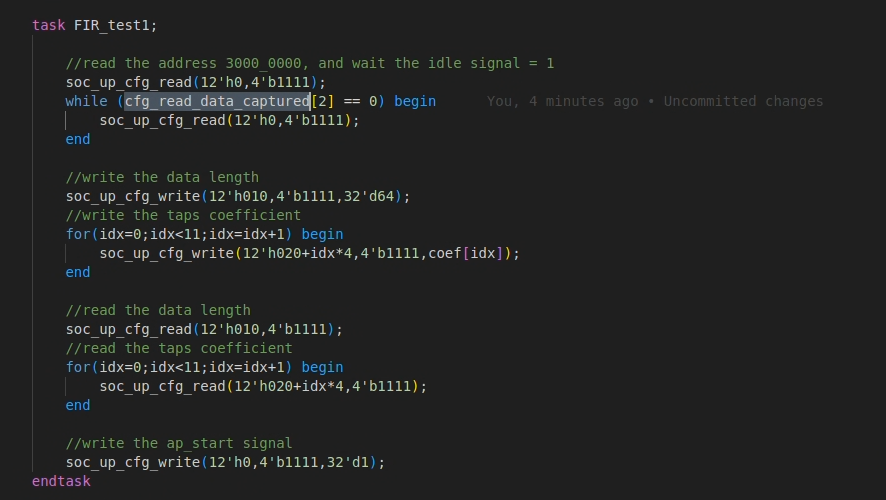
From these code, we change the user\_prj\_sel[4:0] signal if

(address[11:0] == 12’h000 && axi\_wstrb[0] ==1). In the case, the program data would be the user\_prj\_sel signal.



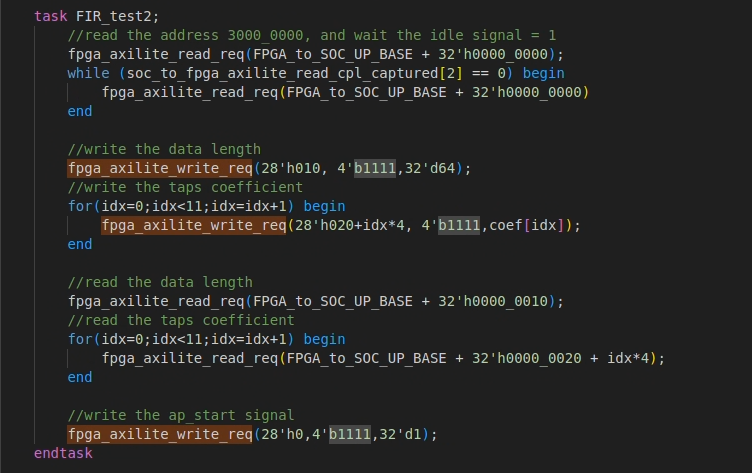
1. Briefly describe how you do FIR initialization (tap parameter, length) from SOC side (Test#1).

I use the soc\_up\_cfg\_read and soc\_up\_cfg\_write task to perform FIR initialization, since these tasks are programming the address base of 32’h3000\_0000. First, I will check idle signal, then program the data length and taps coefficient and read back.



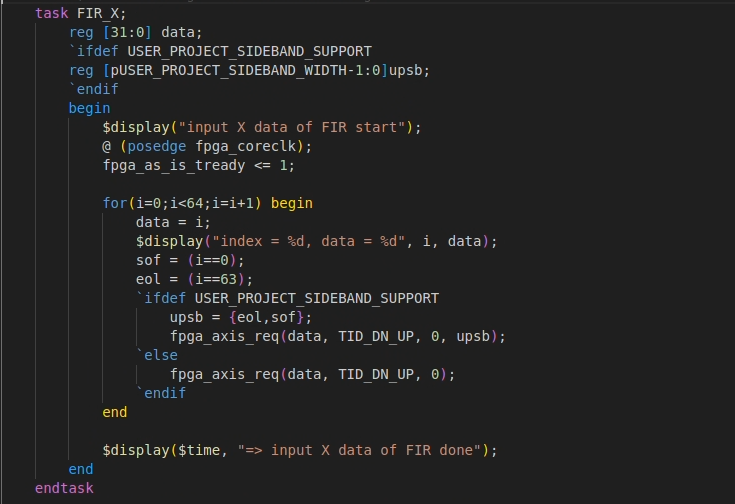
1. Briefly describe how you do FIR initialization (tap parameter, length) from FPGA side (Test#2).

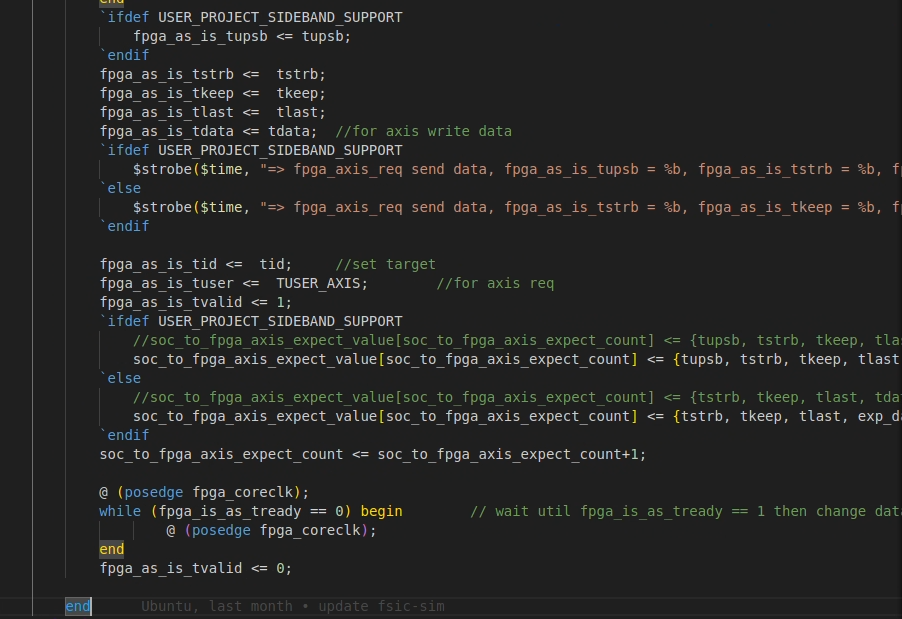
Similar to test#1, I use the fpga\_axilite\_read\_req and fpga\_axilite\_write\_req task to perform FIR initialization, since these tasks are programming the address base of 28’h0000\_0000. First, I will check idle signal, then program the data length and taps coefficient and read back.



1. Briefly describe how you feed in X data from FPGA side.

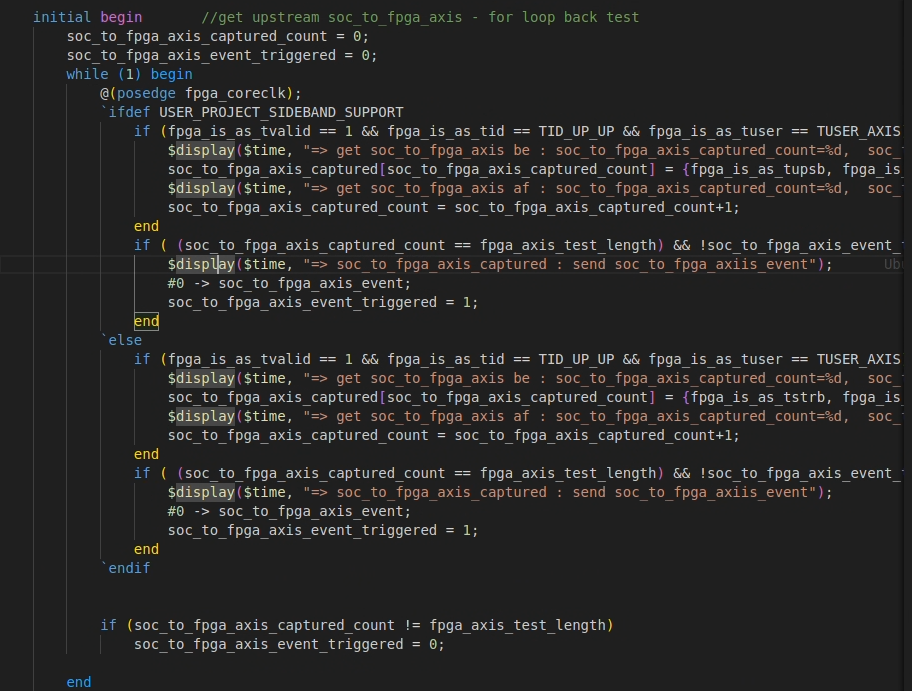
In order to feed in the X data, I modify the code of test002\_fpga\_axis\_req, which is the test of axis. I only need to change the test002\_fpga\_axis\_req, rather than the task it used. The task fpga\_axis\_req will config fpga\_as signal (AXIS-Switch).





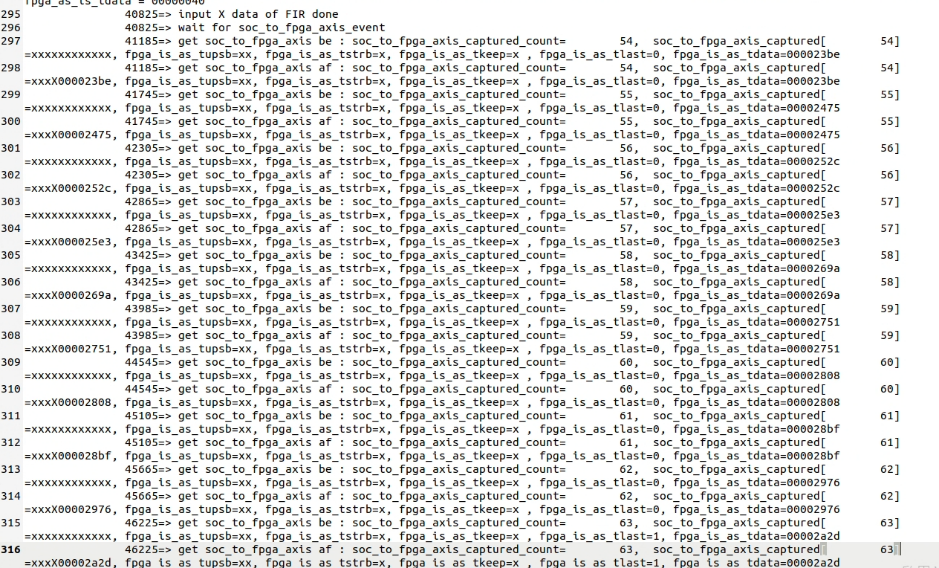
1. Briefly describe how you get output Y data in testbench, and how to do comparison with golden values.

I read back the output data Y by following code, which is the original code I didn’t do any change. These code will read out and save the output in soc\_to\_fpga\_axis\_captured[soc\_to\_fpga\_axis\_captured\_count][31:0].

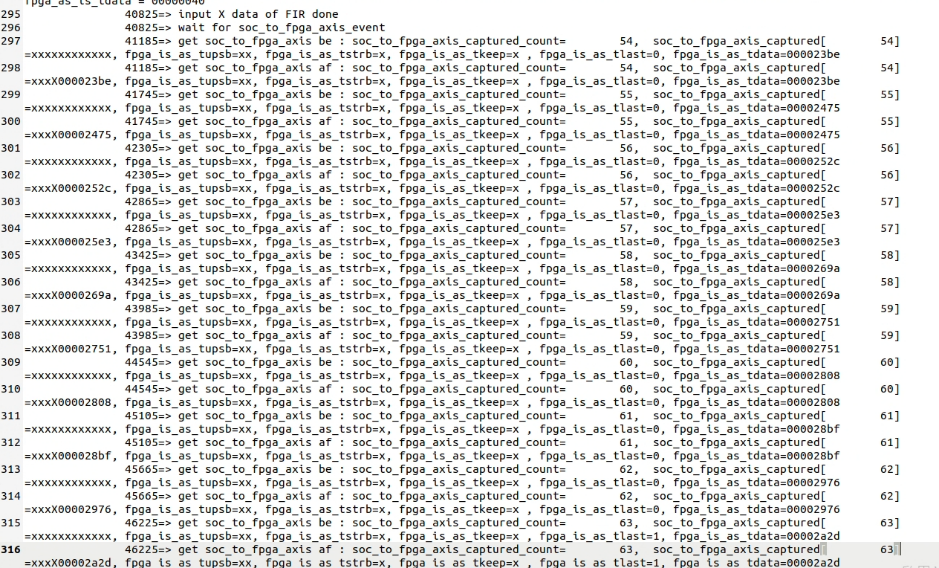


1. Screenshot simulation results printed on screen, to show that your Test#1 Test#2 complete successfully

Test#1:

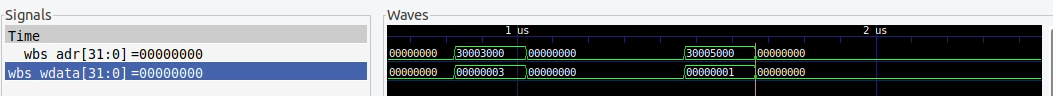


Test#2



1. Screenshot simulation waveform:

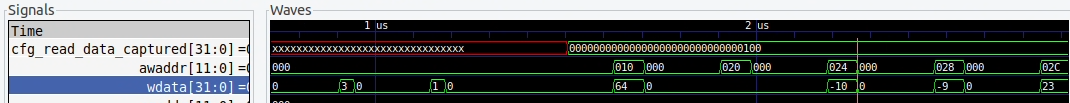
* Configuration cycle (when we program [‘h3000\_5000] = 32’h01, signal user\_prj\_sel changes accordingly)

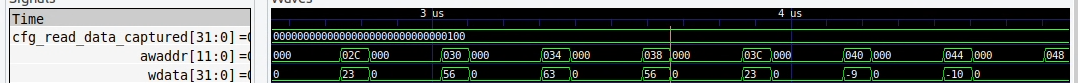


* AXI-Lite transaction cycles (feed in tap parameters, data\_length)

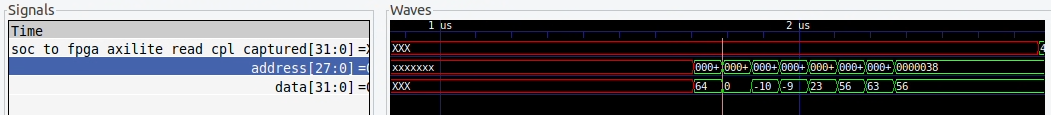
Write the data length and tap parameters

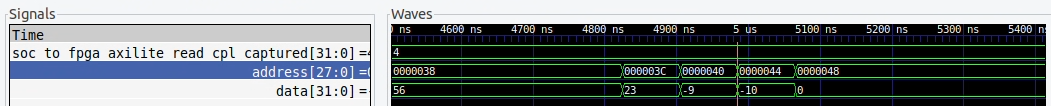
From SOC side:





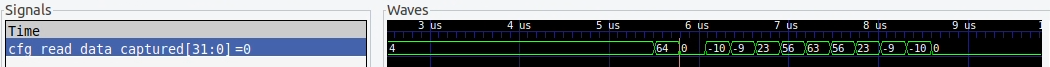
From FPGA side:



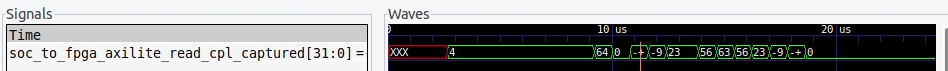


Read back the data length and tap parameters

From SOC side:

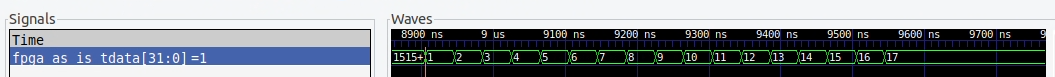


From FPGA side:



* Stream-in, Stream-out

There are several threads like the waveform below, until stream-in data reach 64.



The stream-out waveform shows the FIR will need several cycle to load new data and reset the previous output data.

