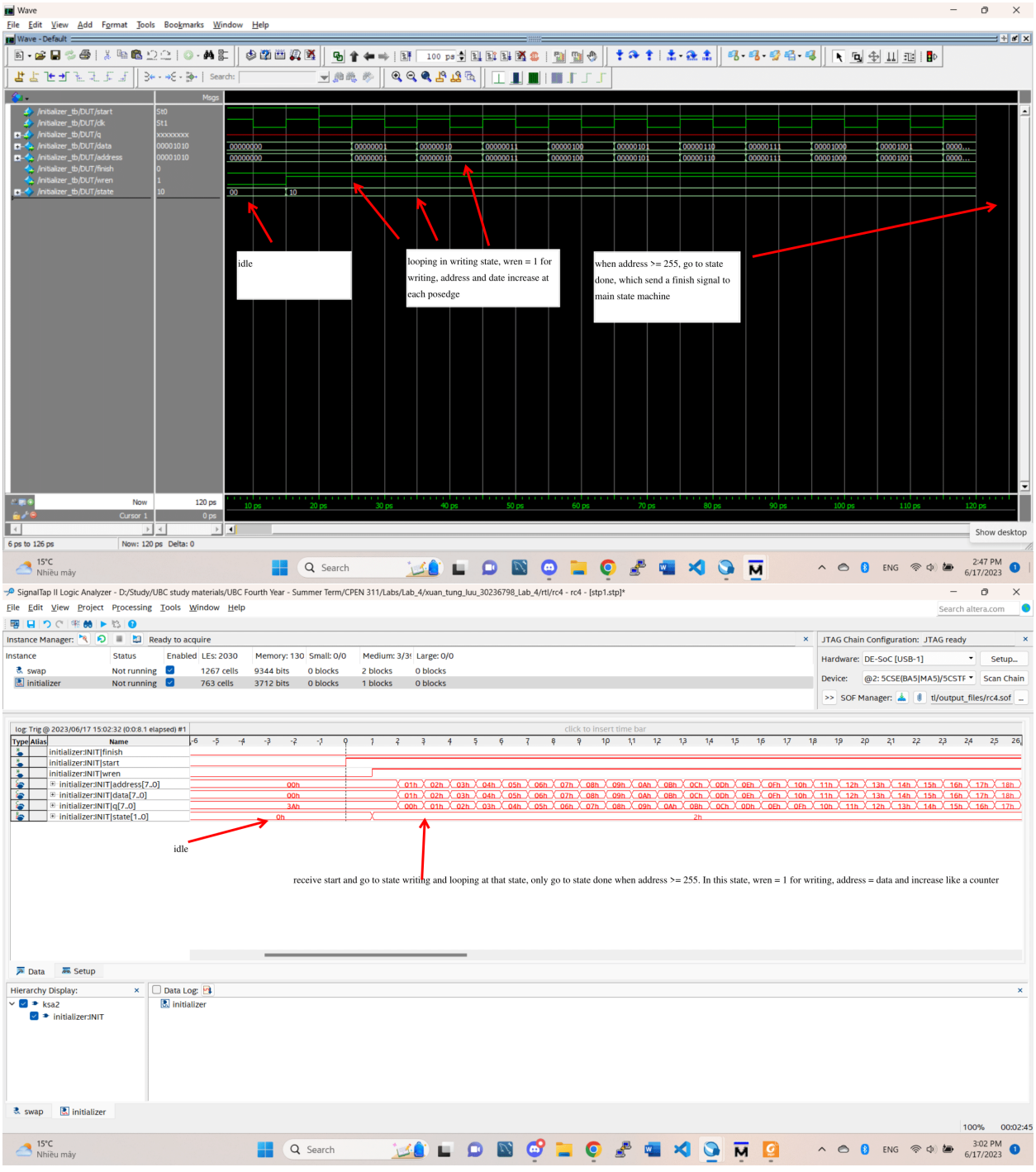
Tim Yang - 53414207 - Xuan Tung Luu - 30236798 - README

1. My SOF file is located at tim_yang_53414207_xuan_tung_luu_30236798_Lab_4/rtl/rc4.sof

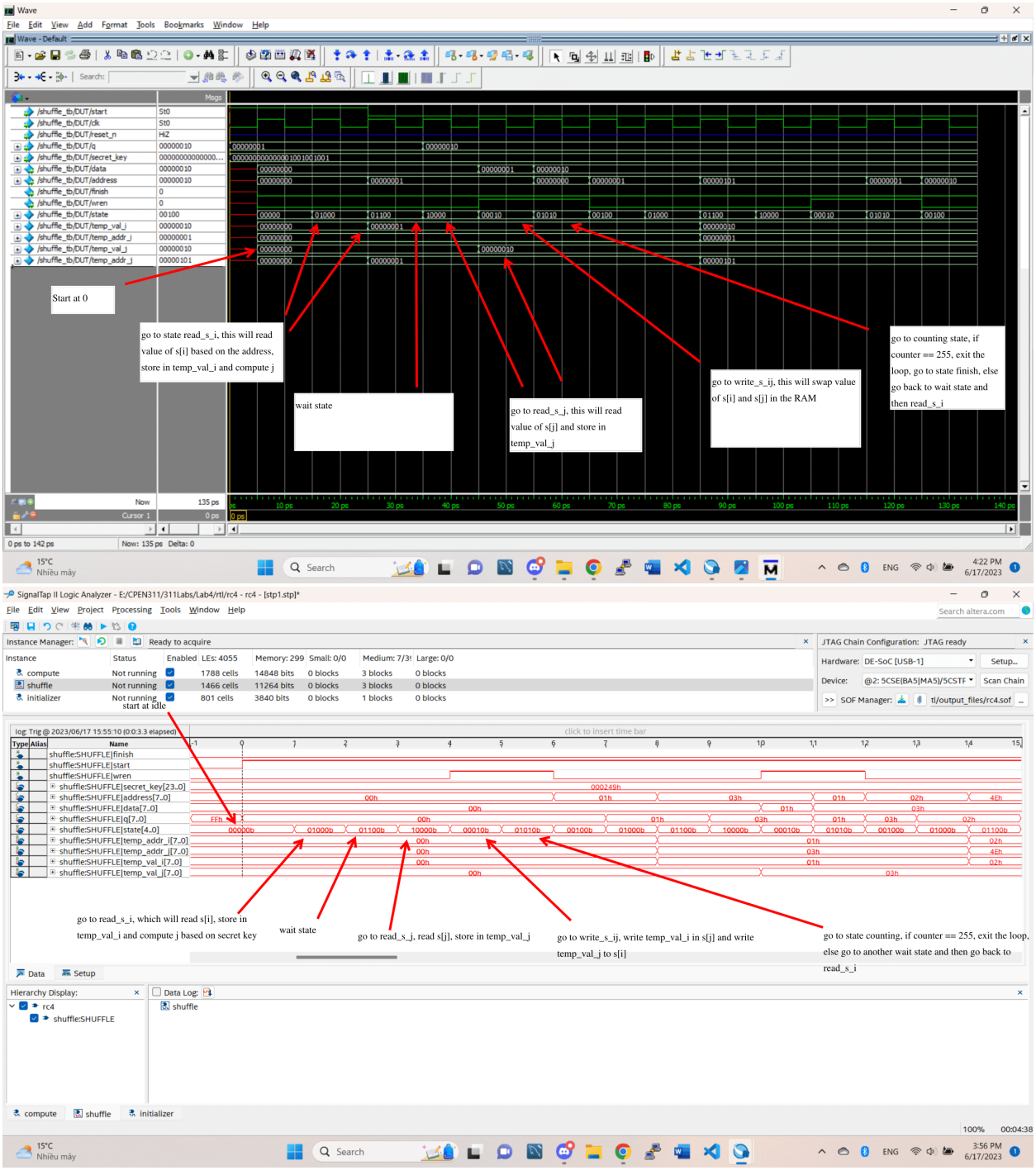
2. Everything works.

3&4. We have 4 FSM. The first one is named initializer, which will initialize each address to store the same value of the address.



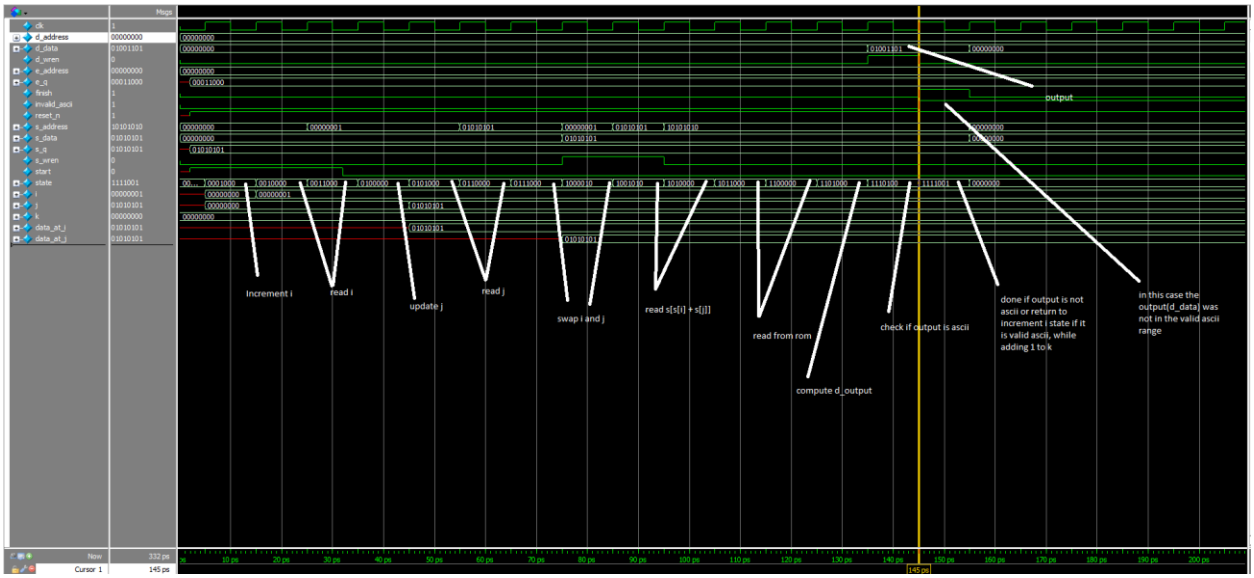
Initializer

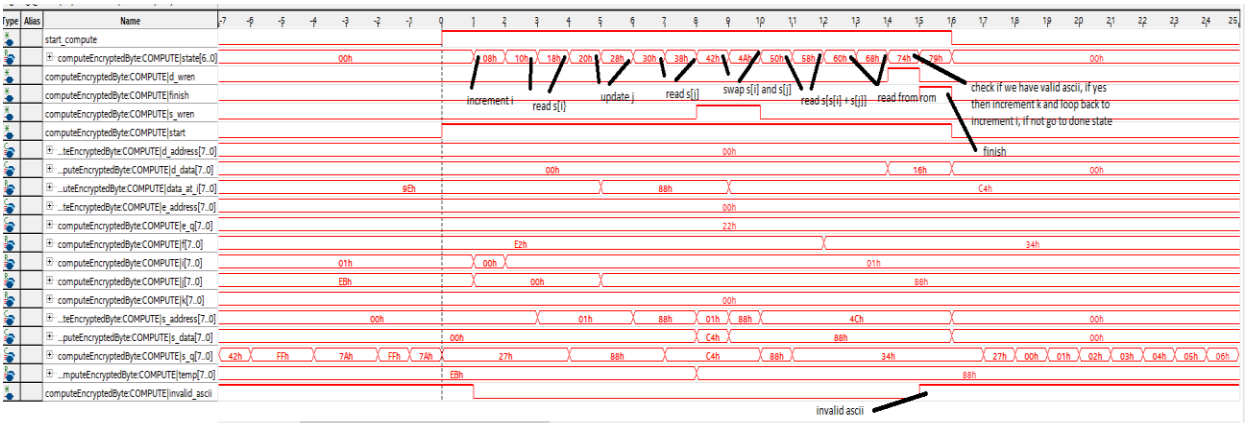
Next FSM is called shuffle, which will calculate the value j in each loop as the code in task 2a and swap the value at address i and j.



Shuffle

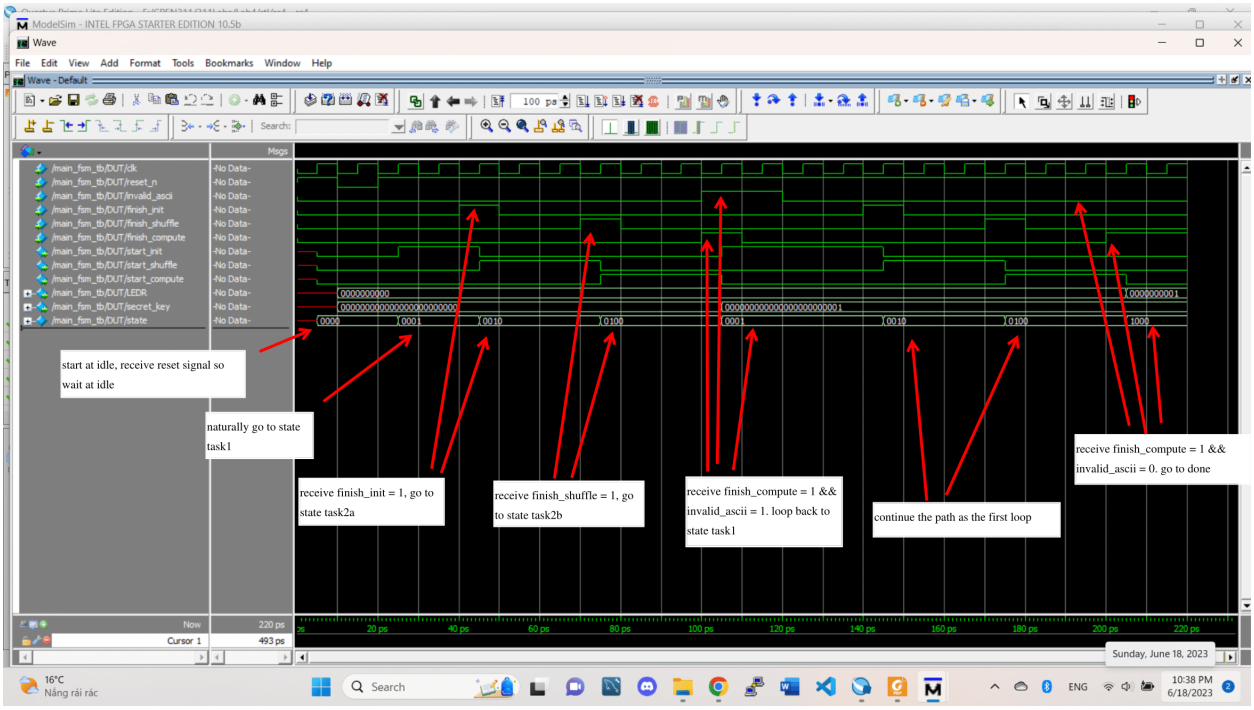
Third FSM is called computeEncyptedByte which will take the s array and use it to compute a decrypted output based on the encrypted rom.

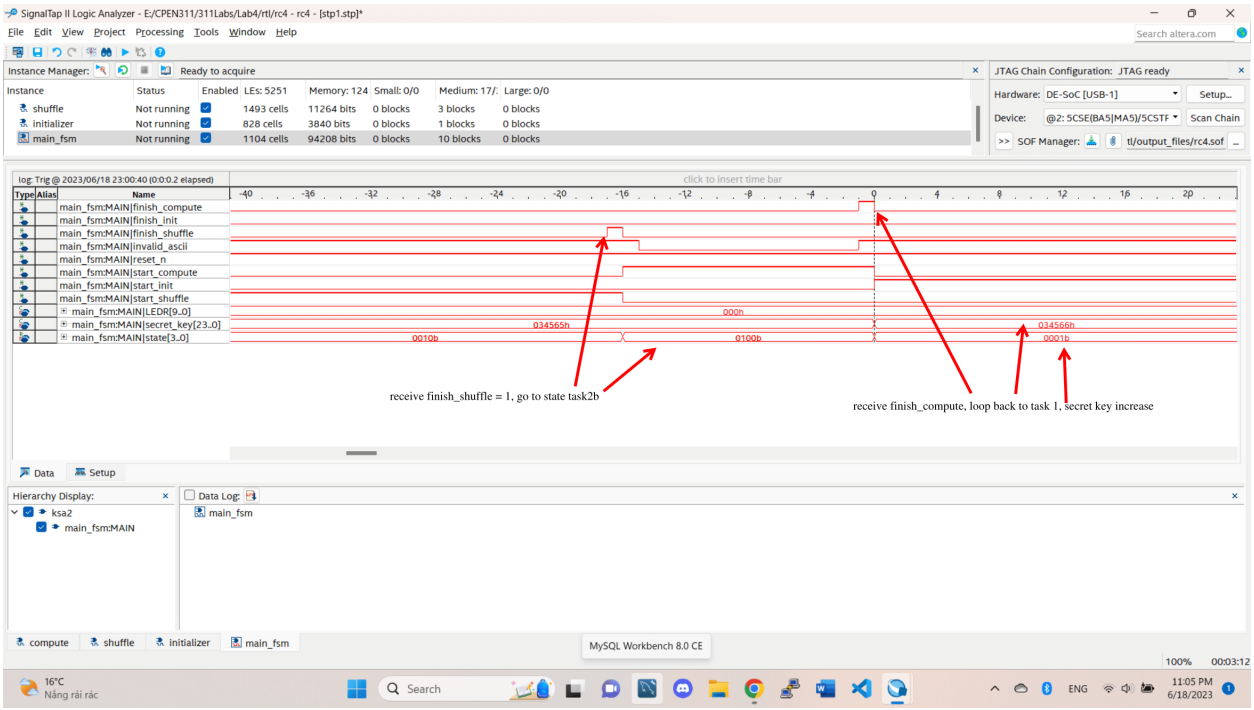


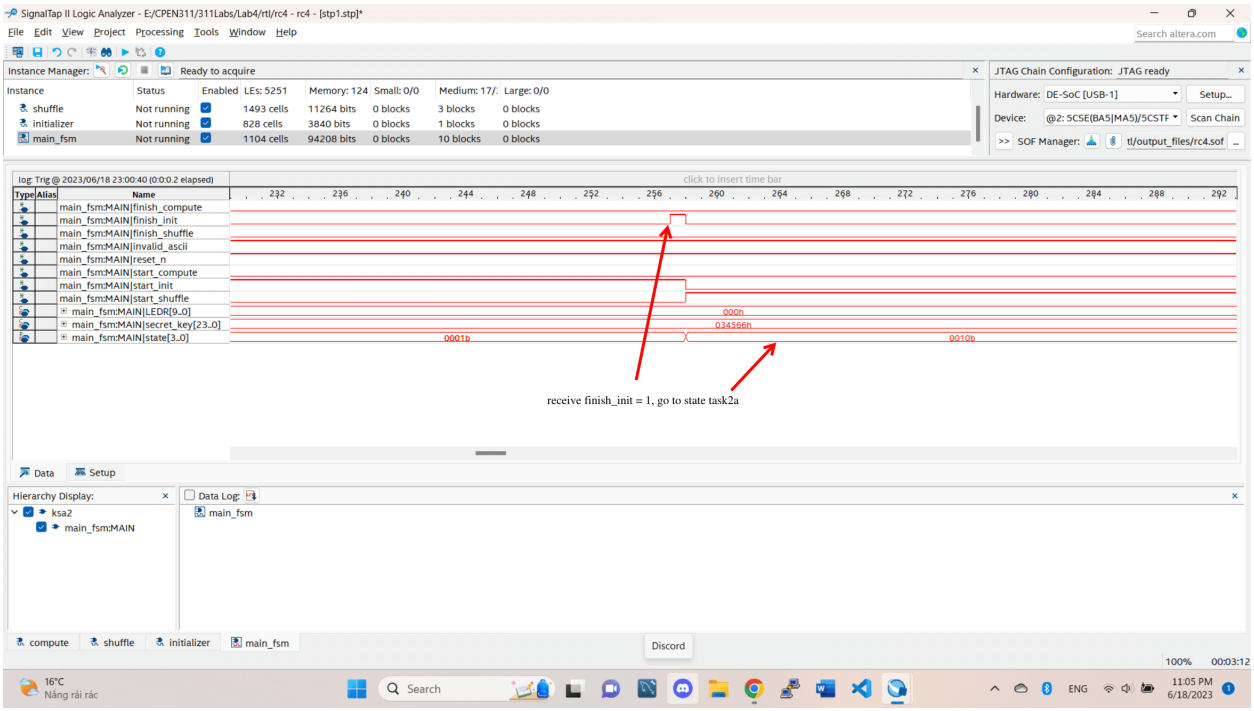


ComputeEncryptedByte

Lastly, we have the FSM main_fsm, which control when other state machine will start and loop around to find the secret key.







Main_fsm

5. We use modelsim to run the simulation. The main files for each state machine are

tim_yang_53414207_xuan_tung_luu_30236798_Lab_4/rtl/initializer.sv
tim_yang_53414207_xuan_tung_luu_30236798_Lab_4/rtl/shuffle.sv
tim_yang_53414207_xuan_tung_luu_30236798_Lab_4/rtl/computeEncryptedByte.sv
tim_yang_53414207_xuan_tung_luu_30236798_Lab_4/rtl/main_fsm.sv

The testbench files are
tim_yang_53414207_xuan_tung_luu_30236798_Lab_4/rtl/initializer_tb.sv
tim_yang_53414207_xuan_tung_luu_30236798_Lab_4/rtl/shuffle_tb.sv
tim_yang_53414207_xuan_tung_luu_30236798_Lab_4/rtl/computeEncryptedByte_tb.sv
tim_yang_53414207_xuan_tung_luu_30236798_Lab_4/rtl/main_fsm_tb.sv

To simulate, just create the project with each couple of files (e.g initializer.sv with initializer_tb.sv) and add the waves similar to the pictures (or any wave that you want).