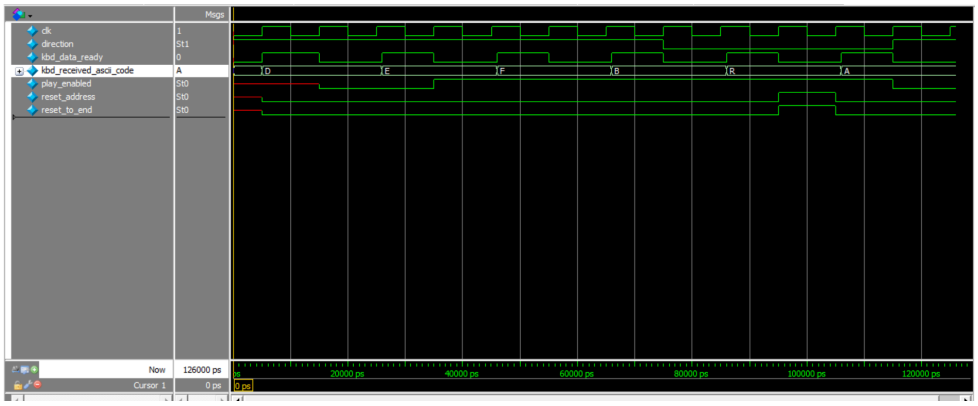
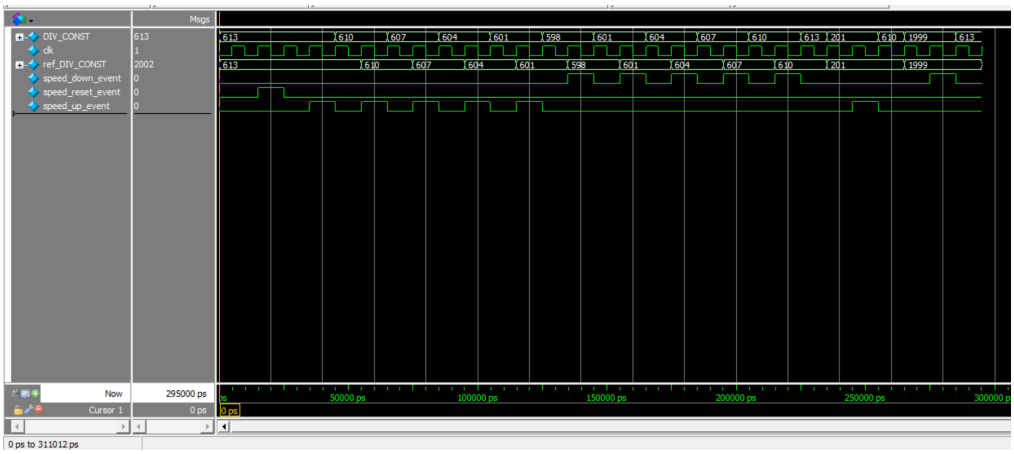
README

1. SOF file is located in the rtl directory, all code is there, simulation is located in the SIM directory with the screen shots as requested.
2. Everything that was for marks works including both bonuses.
3. Simulations
    a. Keyboard_controller
        i. 

            1. Based upon the keyboard keys pressed we can see the correct events being displayed for example when the E key is pressed we can see the play enabled is activated when R is pressed the reset signals are also activated. Direction is also changed when B is pressed. This passed the self checking test bench that was made.
        ii.
```
VSIM 15> run -all
# PASS: D
# PASS: E
# PASS: F
# PASS: B
# PASS: R
# PASS: Default
# ** Note: $finish    : C:/Users/a_abe/Desktop/Ali_Abedian_49665789_Lab_2/sim/Keyboard_controller_testbench/Ketboard_controller_
#    Time: 126 ns  Iteration: 0  Instance: /Keyboard_controller_tb
```
    b. Speed_controller
        i. 

            1. As we can see based upon the inputs from the speed event the change is reflected in the division constant used for the trigger of the FSM which is generated via clock divider as a result it controls the speed as given.

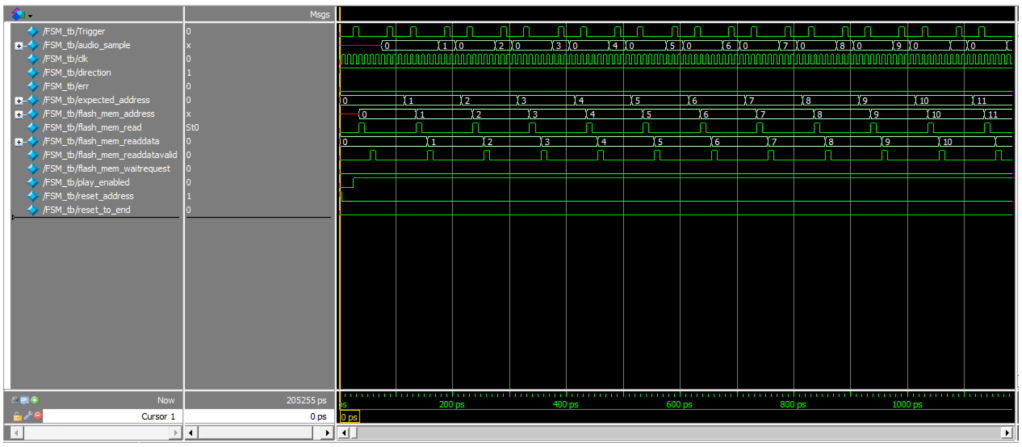
2. Passed self checking test bench.

```
VSIM 8> run -all
# PASS: Reset -- DIV_CONST: 613
# PASS: Speed Up -- DIV_CONST: 610
# PASS: Speed Up -- DIV_CONST: 607
# PASS: Speed Up -- DIV_CONST: 604
# PASS: Speed Up -- DIV_CONST: 601
# PASS: Speed Up -- DIV_CONST: 598
# PASS: Speed Down -- DIV_CONST: 601
# PASS: Speed Down -- DIV_CONST: 604
# PASS: Speed Down -- DIV_CONST: 607
# PASS: Speed Down -- DIV_CONST: 610
# PASS: Speed Down -- DIV_CONST: 613
# All tests completed.
```

ii.

c. FSM_controller

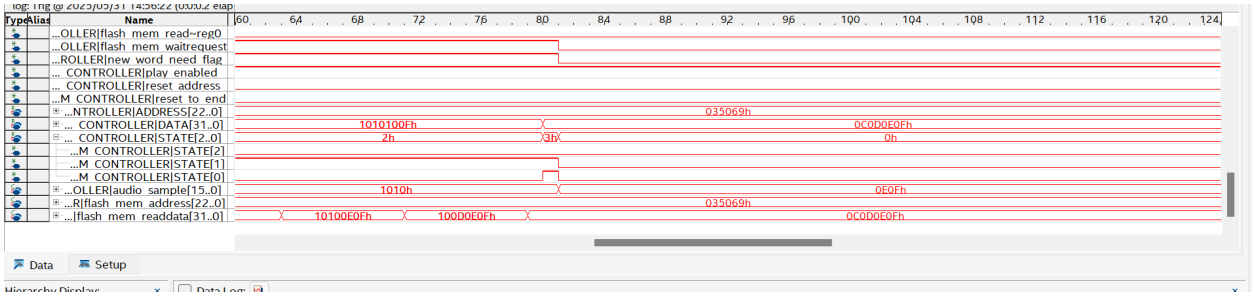
i.



1. The FSM starts to emit an address and when the trigger is activated. As we can see in the above simulation it sends address to the flash driver and waits in the given states (read and wait) until the ok signals (waitreq and readvalid) are sent audio sample is generated by the simulation and we can see a direct correlation with read data.
2. The FSM passed the self checking test bench that I made.

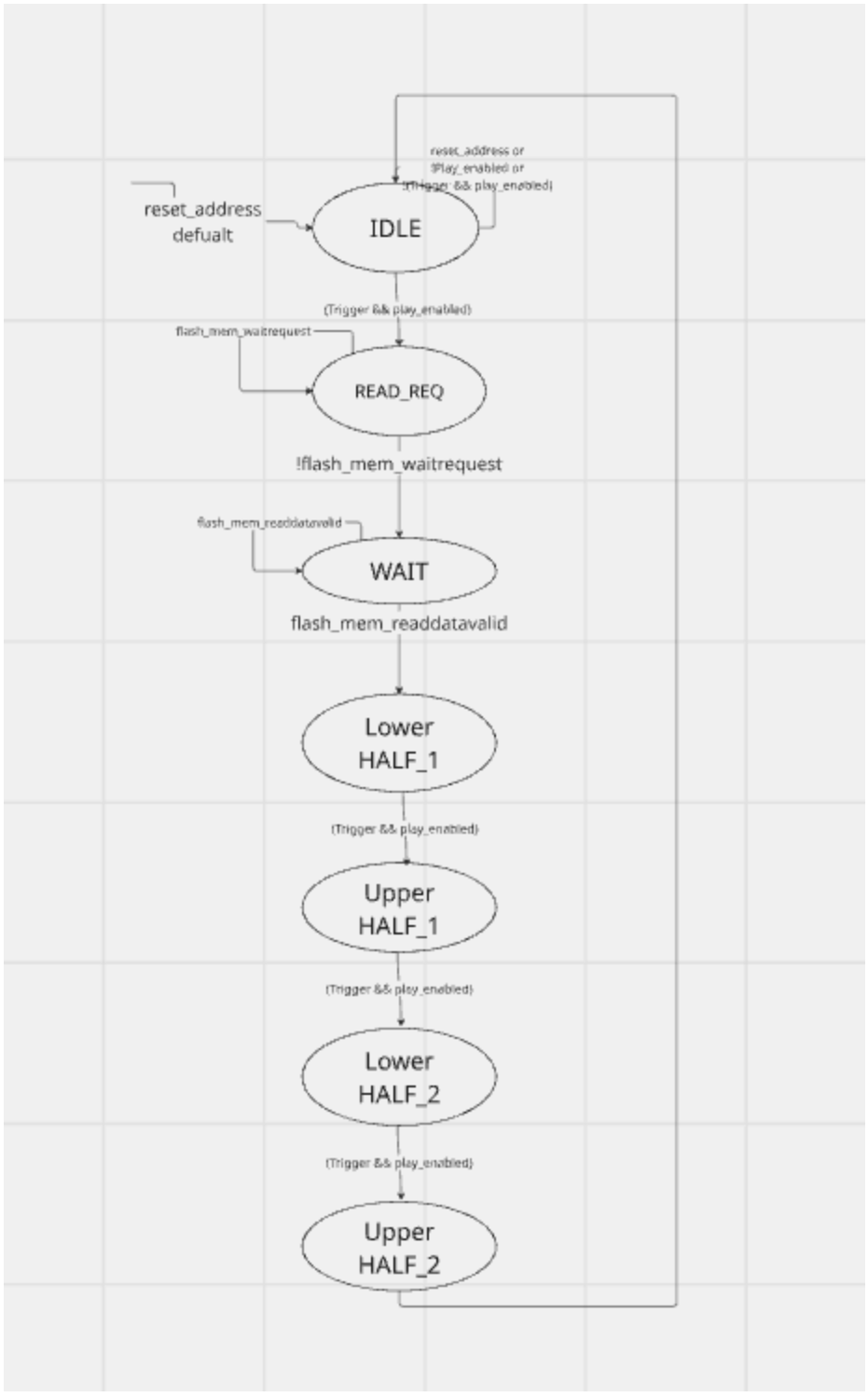
ii.

```
VSIM 26> run -all
# All tests passed.
# ** Note: $stop    : C:/Users/a_abe/Desktop/Ali_Abedian_49665789_Lab_2/sim/FSM_testbench/FSM_tb.sv(203)
#    Time: 205255 ps  Iteration: 1  Instance: /FSM_tb
# Break in Module FSM_tb at C:/Users/a_abe/Desktop/Ali_Abedian_49665789_Lab_2/sim/FSM_testbench/FSM_tb.sv line 203
```

4. 

    a. The above shows the signal tap of the state machine as we can see it transitions states successfully it also changes the audio sample after 3h which is indicative of the flash data being received which would be correct as required. We can also see the other signals including the address and the read data propagating to that saved DATA in the state machine.
5. Launch each of the modelsim projects for each simulation and compile them if needed then click on simulate and run the test bench.
6. DONE!

7.


reset_address or
!Play_enabled or
!(Trigger && play_enabled)
reset_address
defualt
IDLE
(Trigger && play_enabled)
flash_mem_waitrequest
READ_REQ
!flash_mem_waitrequest
flash_mem_readdatavalid
WAIT
flash_mem_readdatavalid
Lower
HALF_1
(Trigger && play_enabled)
Upper
HALF_1
(Trigger && play_enabled)
Lower
HALF_2
(Trigger && play_enabled)
Upper
HALF_2