Lab 2: Simple iPod

Author: Ruyi Zhou 49581911

Lab Section: L2C

1. *My SOF file is located at:*

D:\Y3_2021W\CPEN311\Lab2\lab2_template_de1soc\ template_de1soc\rtl
Named: simple_ipod_Solution.sof

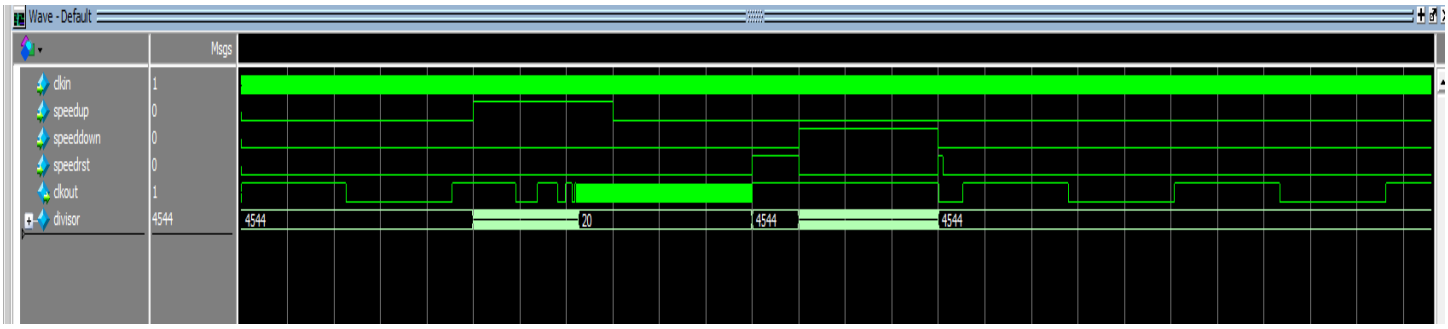
2. *The status of the lab:*

My design can work perfectly except the bonus part and achieves every requirement in the lab handouts. But *the simulation of the Flash_handle module is not working well*. The simulations of other FSMs are working well.

3. *Simulation screenshots:*

I wrote the testbenches for these FSMs: clk_divider_spd_controller, Read_Data, Flash_handle, keyboard. And I simulated them in ModelSim 10.3c.

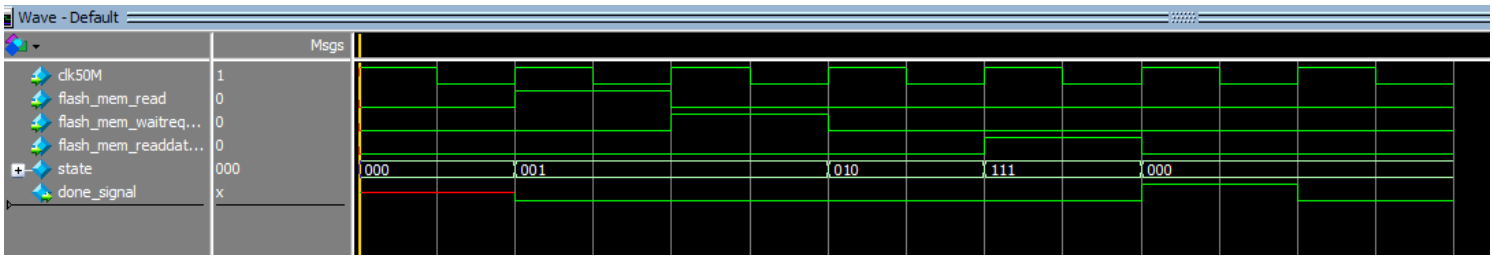
(a) clk_divider_spd_controller:



The smaller divisor, the higher speed of the song, and vice versa. The clkdout is the frequency that control the speed of the song. As shown above, when KEY[0] is pressed (speedup = 1), the divisor is decreasing until the lower limit (I set it to 20), and the corresponding clkout frequency is increasing. Then, we press KEY [2] (speedrst =1), the divisor is turned to the default divisor and the output frequency is the default frequency. Then we slow the speed of the song by press KEY [1] (speeddown), the

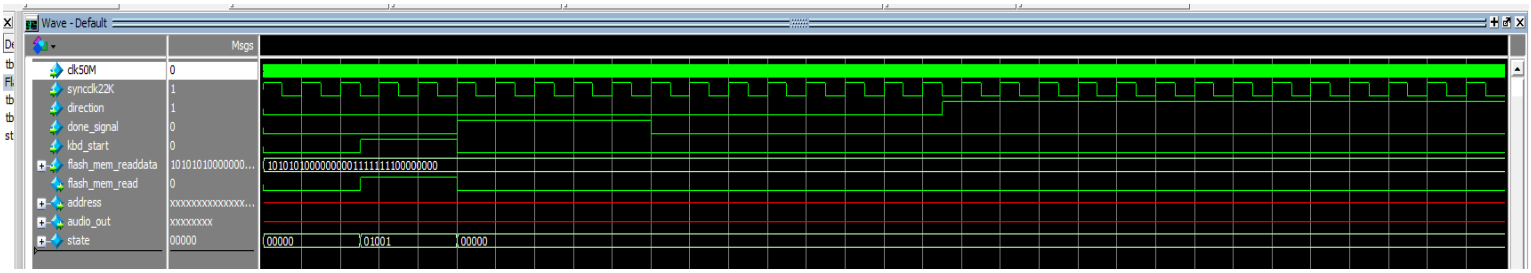
divisor is increased, and the corresponding output frequency is lower. Then we press the KEY [2] again. The divisor and the output frequency become the default value again.

(b) Read_Data:



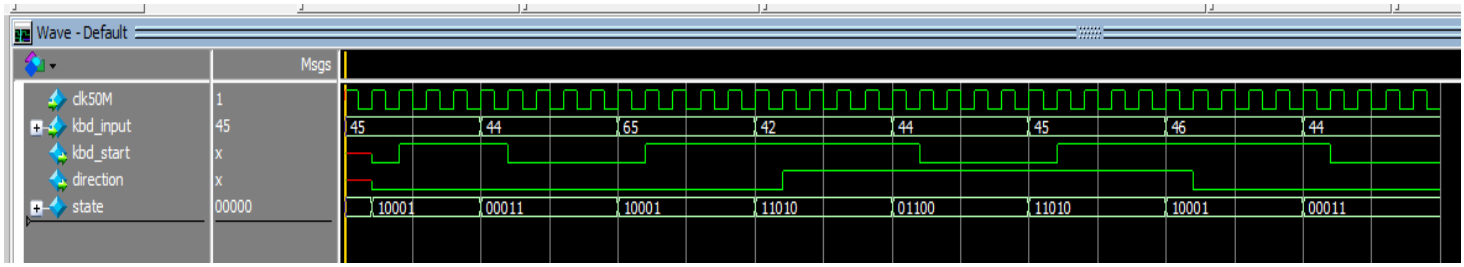
Initially, the state is ‘idle’. When the input ‘flash_mem_read’ commend is 1, the state goes to ‘reading’, and checking if the waitrequest if from 1 to 0. Once the waitrequest falls to 0, the state goes to ‘check_data’. If the data is valid (flash_mem_readdatavalid = 1), the state goes to ‘done’ and output 1 at ‘done’ state.

(c) Flash_handle:



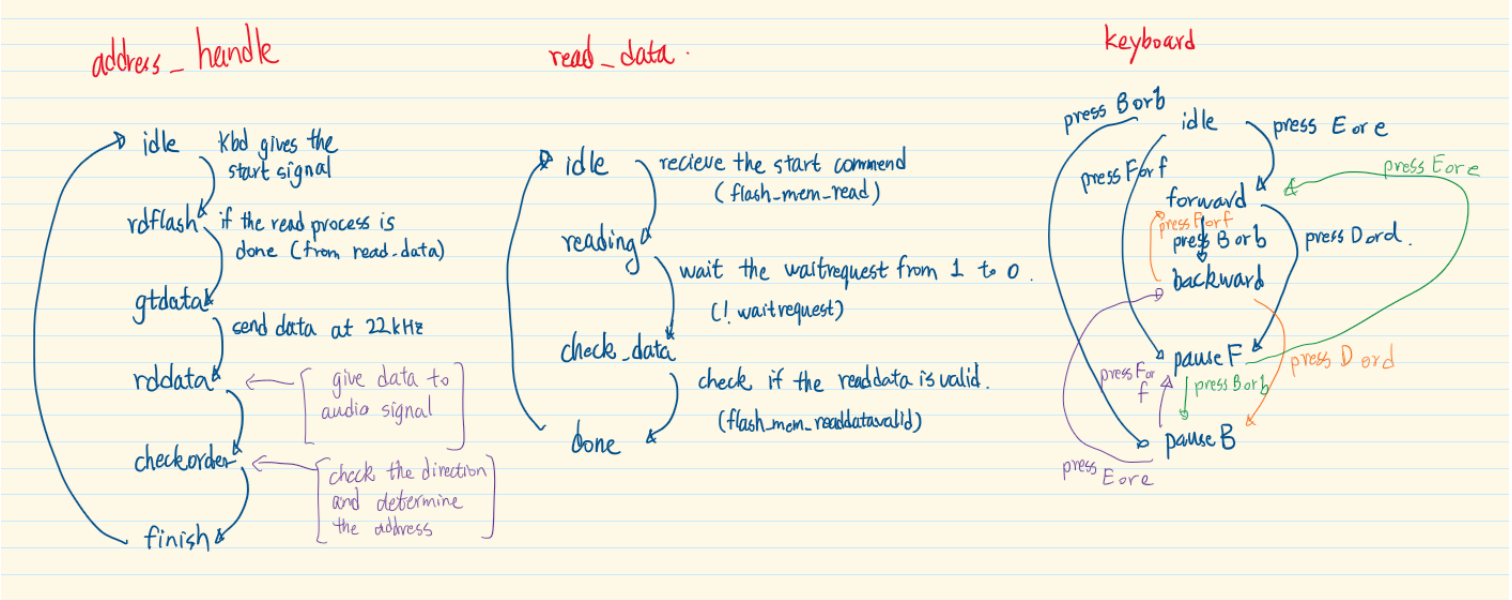
As shown here, the initial state is ‘idle’ and it goes to ‘rdflash’ when the keyboard provides the start commend. In the ‘rdflash’ state, when the done_signal (from read_data) is 1, the state should go to ‘gtdata’. However, it goes back to ‘idle’, which does not make sense. In addition, the output address and audio_out are always ‘x’ which also does not make sense because I already set the default values to them, and the Quartus report does not show any Latches. That’s how this simulation does not work.

(d) Keyboard

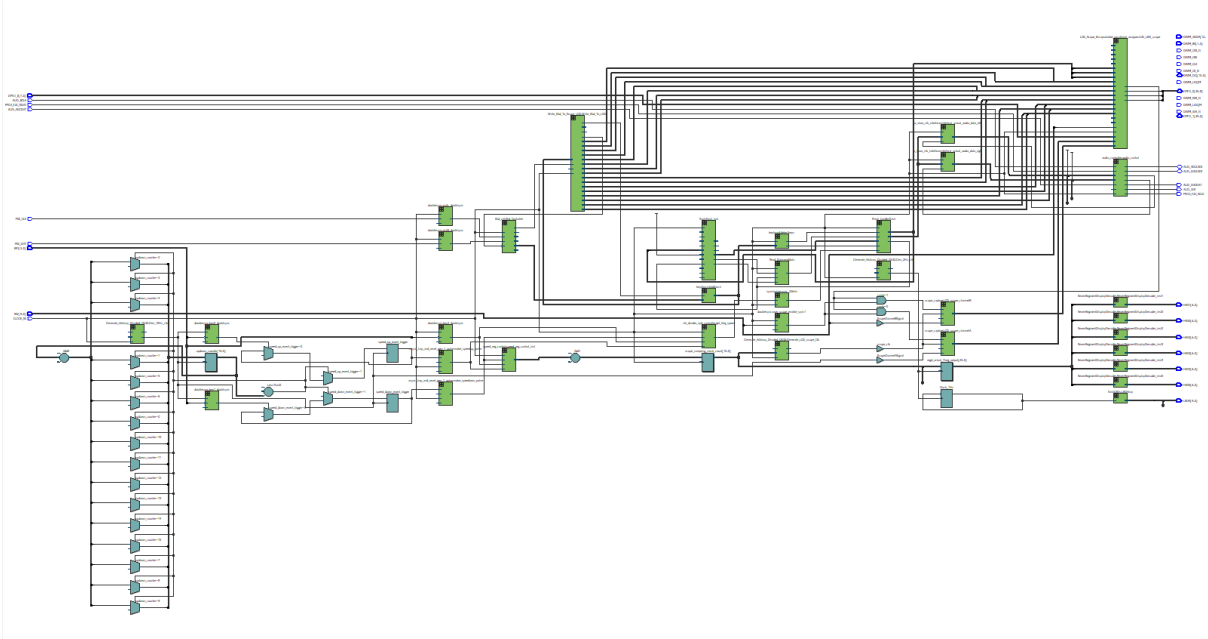


As shown above, the initial state is 'idle'. When 'E' (45) is pressed, the state goes to 'forward' and output the start commend =1. Then 'D'(44) is pressed, the state goes to 'pauseF' and stop outputs the start commend. Then the lower case 'e'(65) is pressed, the state goes back to 'forward' and continues to output the start commend. Then 'B'(42) is pressed, the state goes to 'backward' and output the start commend. Meanwhile, in 'backward' and 'pauseB' states, the direction is 1, which means the order is reversed. Then, after a while, 'F' (46) is pressed, the direction is 0 right now and the state is 'forward', and the module outputs start commend = 1 .

Flowchart of the FSMs:



RTL viewer of the design



4. How to run the simulations:

The testbenches for the clk_divider_spd_controller, Read_Data, Flash_handle and keyboard are called tb_clk_divider_spd_controller.sv, tb_Read_Data.sv, tb_Flash_handle.sv and tb_keyboard.sv respectively. These testbenches are located at:
D:\Y3_2021W\CPEN311\Lab2\lab2_template_de1soc\template_de1soc\rtl
I run the simulation in **ModelSim 10.3c** by clicking the 'simulation' button on the tool bar.
If you have my do. files, to simulate, click the 'start simulation' and choose the corresponding testbench. Then input the command 'do <filename>'. Then click 'run all' to get the waveforms.

4. Additional information:

-All the code files and simulation files (do. Files) are inside **rtl** because I simulate the design using **ModelSim**, which requires all the project files under the same dictionary.

-If you have any questions please let me know during my demo session. I am very willing to demo all the features I have in this lab. Finally, this lab is really hard.