Zhixuan Mo

CSE664

Lab4 part A report

1. Problem Statement:

In this lab, we need to use two different tools to deal with software and hardware problems. We use keil Uvision to edit and compile software code in assembly language and use modelsim to modify the structure of the machine and simulate the design.

2. Problem 1: Simulation of original design:

First we need to create a new project, in option for target -> user, we need to generate executable file in hex format with the following command:

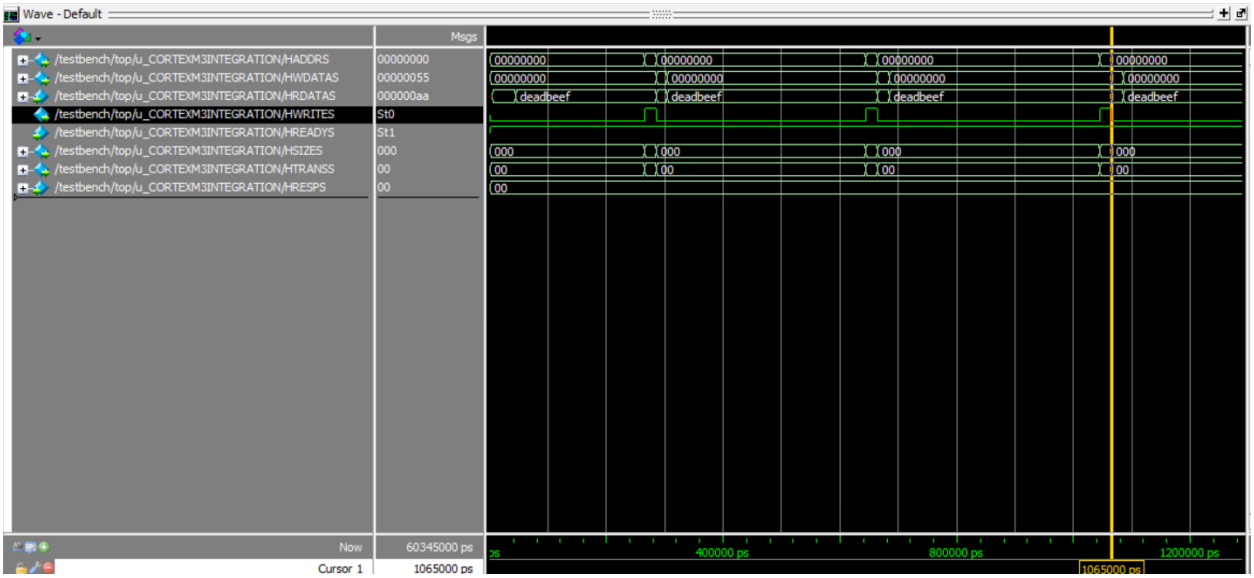
```
fromelf -cvf code.axf --vhx --32x1 -o lab1.hex
```

Then put the hex file into a new modelsim library, compile modules defined in software folder into this new library and run simulation on the testbench, then type do run.do command to initialize the clock. The result waveform is below:

First LED write transaction at 265000 ps, LED changed from 00000000 to 01010101

Second LED write transaction at 665000 ps, LED changed from 01010101 to 10101010

HADDRS = address , HWDATAS = write data , HWRITES = write permission



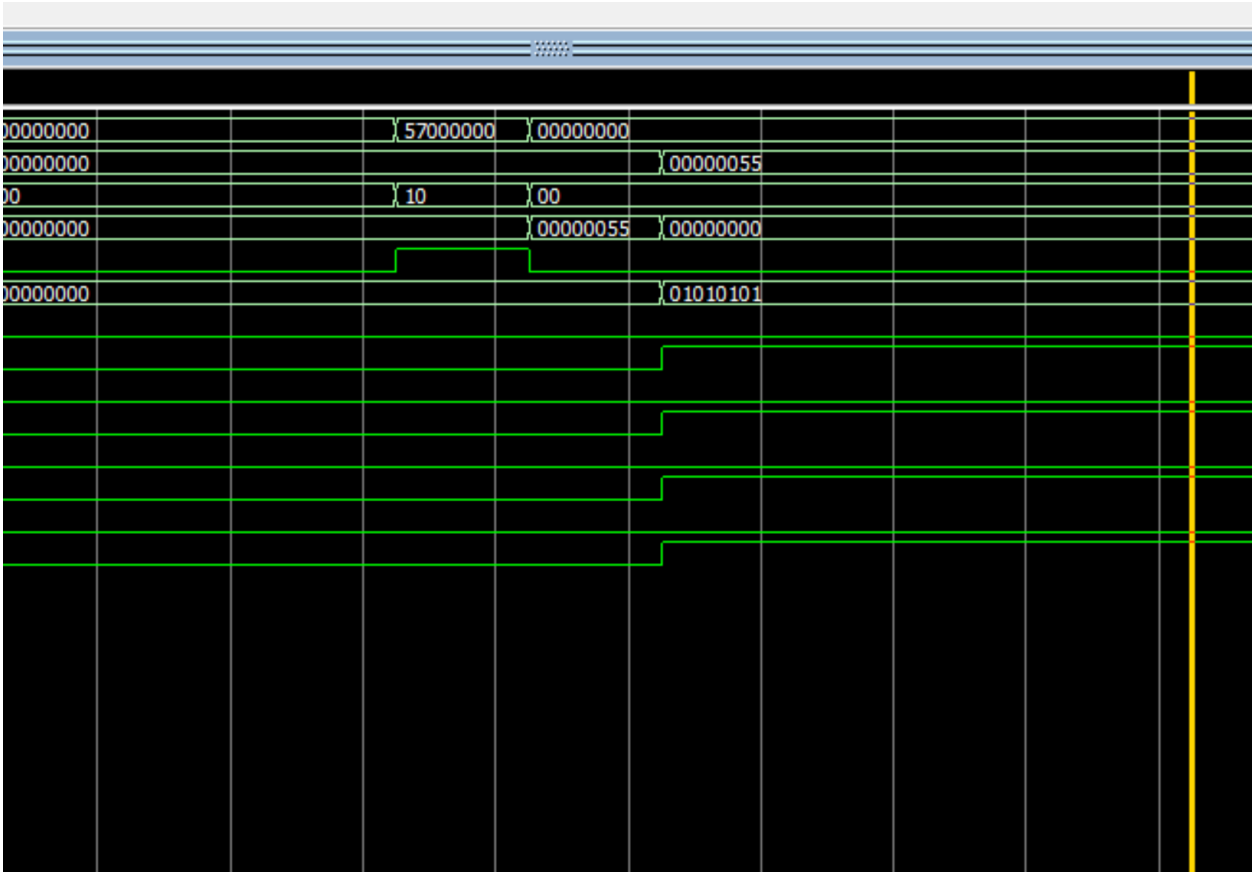
3. ## Problem 2: Change the write address

   You can't do this. Error occurs when you do this on Keil Uvision, it says you don't have write permission for address 0x5000_0010.

4. ## Problem 3: Map LED to different Address

   You can do this by changing the wire hsel_led in topmodule. Originally, the signal comes from HSEL_S1 output in address decoder, meaning that when you are accessing address 0x5000_0000 to 0x50FF_FFFF, the LED will notice and react. We need to make the wire connect to HSEL_S8 instead, which means LED will notice and react when the address 0x5700_0000 to 0x57FF_FFFF. The following snapshot proves my method:

   I also changed to address in assembly code so you can see, when I write 0x55 to 57000000, the LED changed.



5. Problem 4: masking

   In order to do this, I added a 8 bit register in LED module that can only be modified by writing data to address 57000004.

   Because led only deals with address that is 57xxxxxx , so we only need to check last 3 digits.

```
else if (rHADDR[2:0] == 3'b100)
        begin
                mask <= HWDATA[7:0];
        end
```

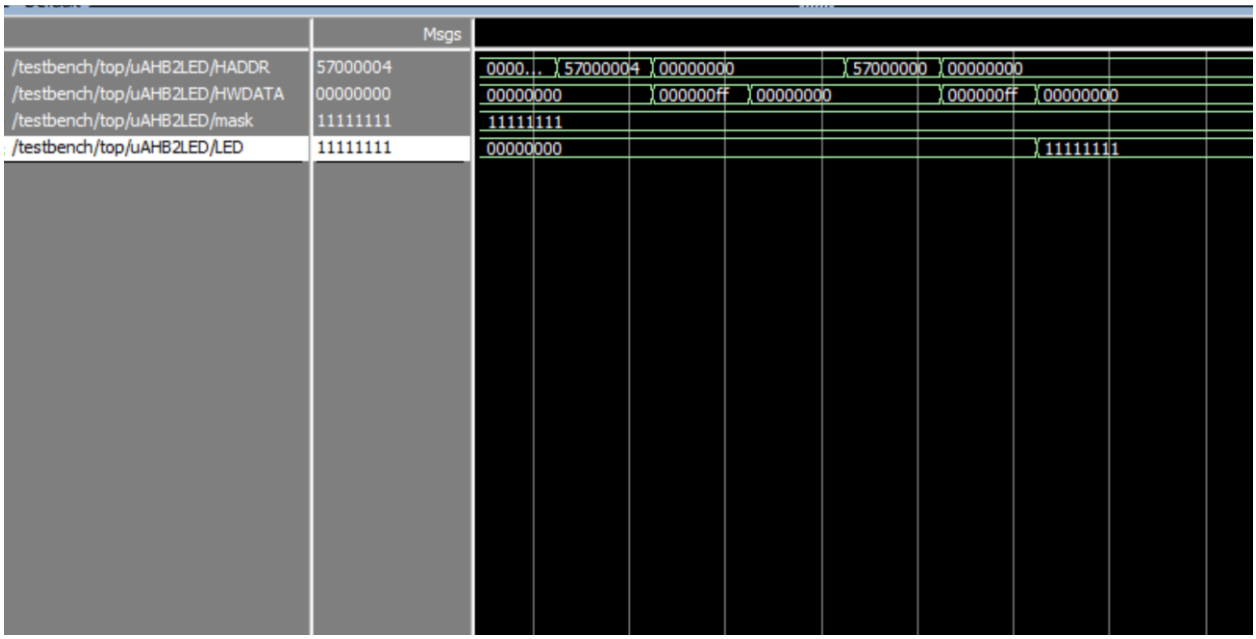
   And then I use a for loop to update LED values:

```
else if(rHSEL & rHWRITE & rHTRANS[1])
        begin
                for(i = 0 ; i < 8 ; i = i + 1) begin
                        if(mask[i] == 1)
                                rLED[i] = HWDATA[i];
                        else
                                rLED[i] = LED[i];
                end
```

   In this way LED digits can only be modified when Mask[i] is 1.

The waveform is shown below:

You can see, I tried to change the mask to 0xff, however it is 0xff already, with mask = 0xff, we can change the led values as before. When we write to 57000000 with data 0xff , the LED values changed to 0xff.



On the second graph, at first the LED = 11111111 , the mask is 00001111, which means the first half of LED values can not be modified. Then we try to write 0xaa to LED. The result is correct, we get 11111010 = 0xfa.