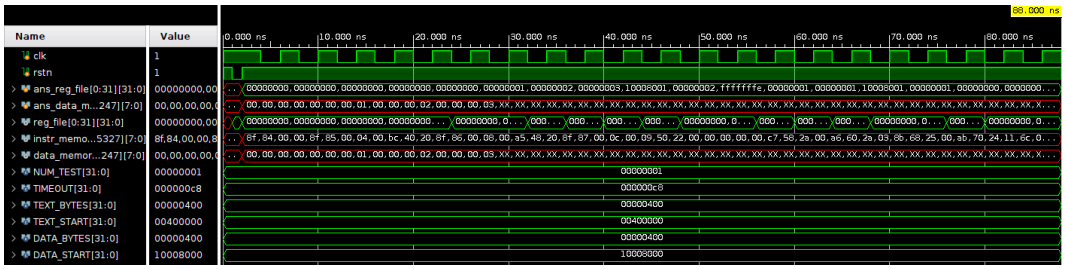
Report

1. Experimental Result

a. Show the waveform screen shot of the test we provided.



b. What other cases you've tested? Why you choose them?

8f 8e 00 04 // [0040002c] lw $14, 4($28)

11 6e 00 06 // [00400030] beq $11, $14, 24

20 11 00 01 // [00400034] addi $17, $0, 0x0001

20 12 00 02 // [00400038] addi $18, $0, 0x0002

20 13 00 03 // [0040003c] addi $19, $0, 0x0003

2 Stalls should happen when lw right before beq , and addi should forward.

2. Answer the following Questions

a. List out the equation to detect EX & MEM hazard in forwarding unit. Which part of the equation in textbook p.369 is wrong?

if EX/MEM_RS = (EX/MEM_reg_write && (EX/MEM_rd != 0)) && (EX/MEM_rd == ID/EX_rs); forward_A = 10

if EX/MEM_RT = (EX/MEM_reg_write && (EX/MEM_rd != 0)) && (EX/MEM_rd == ID/EX_rt); forward_B = 10

The parenthesis are unbalanced

if (MEM/WB.RegWrite and (MEM/WB.RegisterRd $\neq$ 0) and

not(EX/MEM.RegWrite and (EX/MEM.RegisterRd $\neq$ 0) and

(EX/MEM.RegisterRd = ID/EX.RegisterRs)) and (MEM/WB.RegisterRd = ID/EX.RegisterRs)) Forward = 01

if (MEM/WB.RegWrite and (MEM/WB.RegisterRd ≠ 0) and not(EX/MEM.RegWrite and (EX/MEM.RegisterRd ≠ 0) and (EX/MEM.RegisterRd = ID/EX.RegisterRt)) and (MEM/WB.RegisterRd = ID/EX.RegisterRt)) Forward = 01

b. In forwarding for **beq**, is forwarding from MEM/WB to ID needed? Why?

Forwarding from MEM/WB to ID is not typically necessary for the beq instruction because the register file is updated in time for the ID stage to read the correct values.

c. Briefly explain how you insert 2 stalls when **beq** reads registers right after lw writes it.

Make two parameters, first one is for lw when it is two cycles after beq instruction, the other is when lw is one cycle after beq instruction. The instruction suppose to be done in ID stage but add two stalls to make it in MEM stage.

```
stall_2 = (EX_MEM_mem_read && control_branch && ((EX_MEM_write_reg == IF_ID_rs) || (EX_MEM_write_reg == IF_ID_rt))) ? 1'b1:1'b0;
 stall_3 = (ID_EX_mem_read && control_branch && ((ID_EX_write_reg == IF_ID_rs) || (ID_EX_write_reg == IF_ID_rt))) ? 1'b1:1'b0;
```

d. **sw** right after **lw** is quite common since copy and paste a data from one address to another is used frequently. In textbook, a stall is followed by the **lw** in this case. Is it possible to remove this stall? How?

By adding the forwarding path and adjusting the control logic, you can eliminate the stall that would normally occur between a lw followed by a sw.

Forwarding Path: Create a forwarding path from the memory read data in the MEM stage directly to the EX stage where the sw instruction will use it.

Control Logic: Adjust the control logic to detect this specific lw followed by sw scenario and enable the forwarding path.