Computer Architecture Fall 2015　Project 1

Team 11 : B02902125 吳哲宇 B03902082 江懿友 B02705021 謝志邦

Team Work :

吳哲宇　B02902125

Debug and testing
Report

江懿友　B03902082

Data_Memory
Control signals for instructions `beq`, `j`, `lw`, `sw`
Multiplexers for branching and jumping

謝志邦　B02705021

Fowarding Unit
Hazard Detection Unit
Latches

How to implement the pipelined CPU :

Use latches to store the outputs of each stage, and propagate the output values to the next stage at rising edge.

Tools used:

Github, Sublime Text, Notepad++, ModelSim, cygwin, msys, WINDOWS!!!!!!!!

Module Implementation :

1. testbench.v
- Put in my own signal to count stall and flush
2. Adder.v:
- Output the sum of the two 32 bits inputs
3. ALU.v
- Do the calculation for the two 32 bits inputs, and output the result and set the zero signal
4. ALU_Control.v
- Input the last 6 bits from instruction and ALUOp code, then output the 3-bit ALU control code
5. Control.v
- Set the signal lines according to the first 6 bits of the instruction.
6. CPU.v
- Connect all components together.
7. Data_Memory.v
- A module to simulate memory read / write.
8. Forwarding_Unit.v

- Sends signals to multiplexers to decide if data should be forwarded or not.

9. Hazard_Detection.v
- Sends signals to indicate stall condition.

10. IF/ID flush
- If branch taken or jump instruction then flush the pipeline by sending an all-zero instruction through IF/ID latch.

11. Instruction_Memory.v
- Read the instruction from the memory at the input address.

12. MUX5.v
- Select one 5-bit input from two 5-bit inputs

13. MUX32.v
- Select one 32-bit input from two 32-bit inputs

14. PC.v
- Program counter

15. Registers.v
- Write the input to the register at rising edge, output the value of register at falling edge.

16. Sign_Extend.v
- Extend a 16-bit input into a 32-bit output

17. IF_ID.v
- The latch between Instruction Fetch stage and Instruction Decode stage

18. ID_EX.v
- The latch between Instruction Decode stage and Execute stage

19. EX_MEM.v
- The latch between Execute stage and Load/Store Memory stage.

20. MEM_WB.v
- The latch between Load/Store Memory stage and Write Back stage

Problems and Solutions

1. Our output is different from the TA’s output
- Try to change some update timings to fix the output values.

2. Misunderstand the posedge, negedge, non-blocking assignment, blocking assignment.
- Search tutorials and do some little experiments to understand the behaviors.