**Computer Organization**

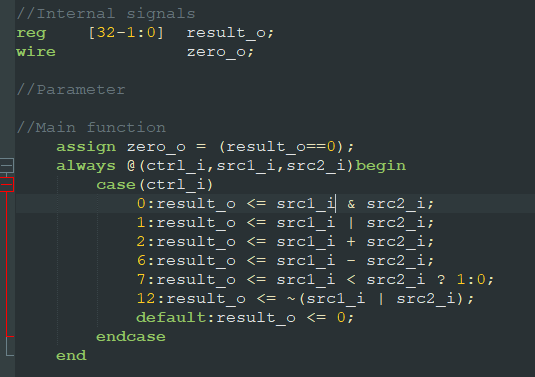
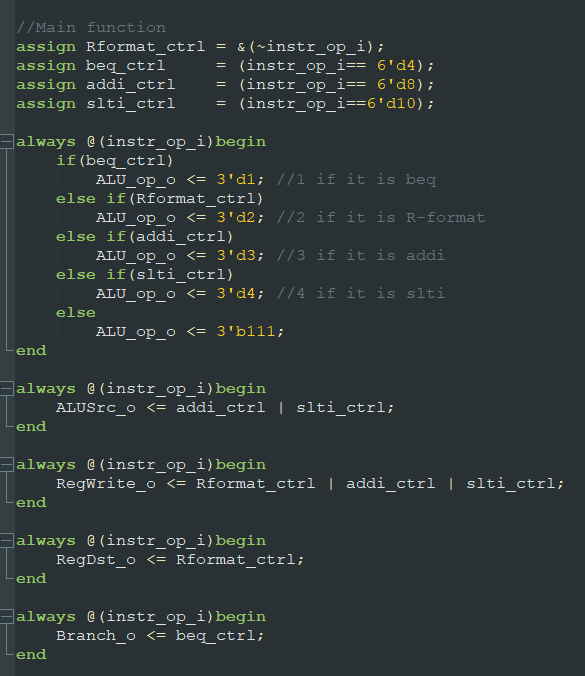
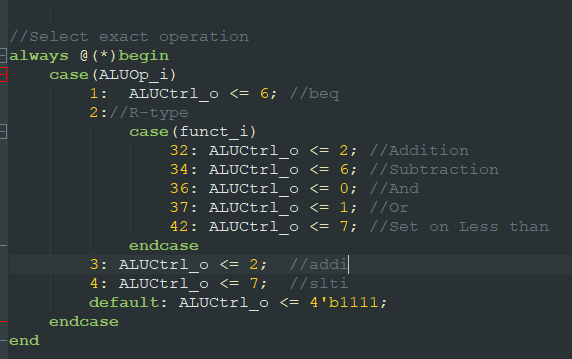
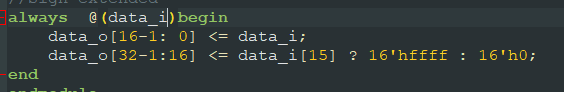
**Architecture diagrams:**

****

**Hardware module analysis:**

* Top module: Simple\_Single\_CPU
  + Program Counter: “count” for the next instruction that needs to do in the following clock cycle.
  + Adder: there is 2 adder in this hardware-
    - first: add 4 to the current value of Program Counter, which called the next instruction in the next cycle
    - second: add the value of Program Counter for the next instruction with the address stated by instruction[15:0], when is the instruction that specified by branch operation.
  + Instruction memory: fetch the instructions from the test file according to the Program Counter.
  + Decoder: decode the opcode of the instruction and output to other module for use.
  + ALU Control: further decode the function code of the instruction and output to ALU for use.
  + Register File:performs the functions of getting the value from register according to the instructions and storing the result of ALU into the register specified by current instruction.
  + Sign Extend: extend the 16-bit of address stored in instruction into 32-bit.
  + ALU: perform some operations such as
    - and
    - or
    - addition
    - subtraction
    - set on less than
    - nor
  + multiplexor: for selecting the input according to the selector.

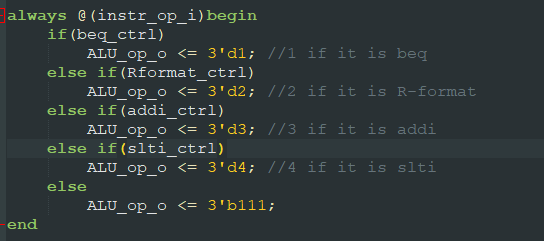
**Finished part:**

1. Adder.v
   1. 
2. ALU.v
   1. 
3. Decoder.v
   1. 
4. ALU\_Ctrl.v
   1. 
5. MUX\_2to1.v
   1. 
6. Shift\_Left\_Two\_32.v
   1. 
7. Sign\_Extend.v
   1. 
8. Simple\_Single\_CPU.v (module connection and signal declaration)

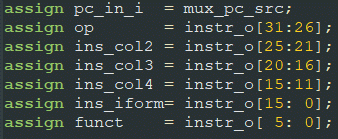
**Problems you met and solutions:**

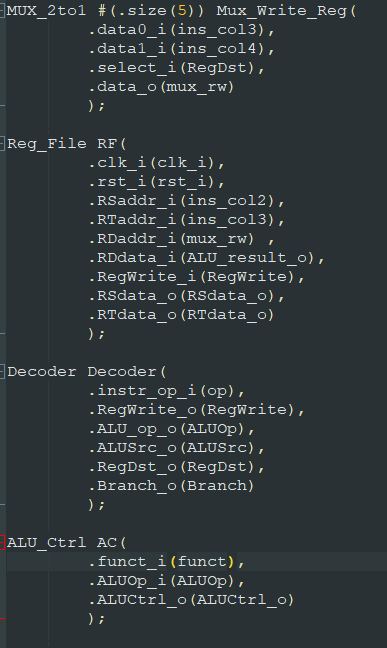
1. I am really confused with the connection of Decoder and ALU\_Ctrl for decoding the instruction, especially the 3-bit ALUOp signal that did not appear in textbook and lecture.

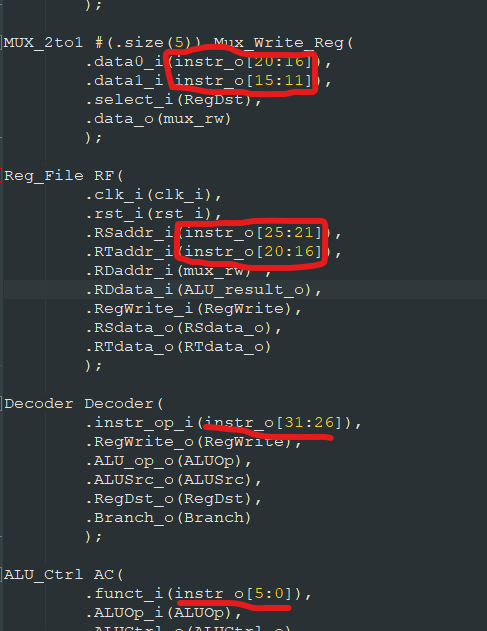
* solution: confused by the simplification of ALUOp and finally realize the meaning of ALUOp is only to encode the opcode into simpler form:



1. When I try to connect some wire than differentiate the instruction, the signals uncaught by the next connected module.





* solution: connect the signal directly.
* 

**Summary:**

**This Lab2 guide us to do the basic MIPS Simple Single CPU that operate *add, addi, sub, sub, and, or, slt, slti* and *beq.***