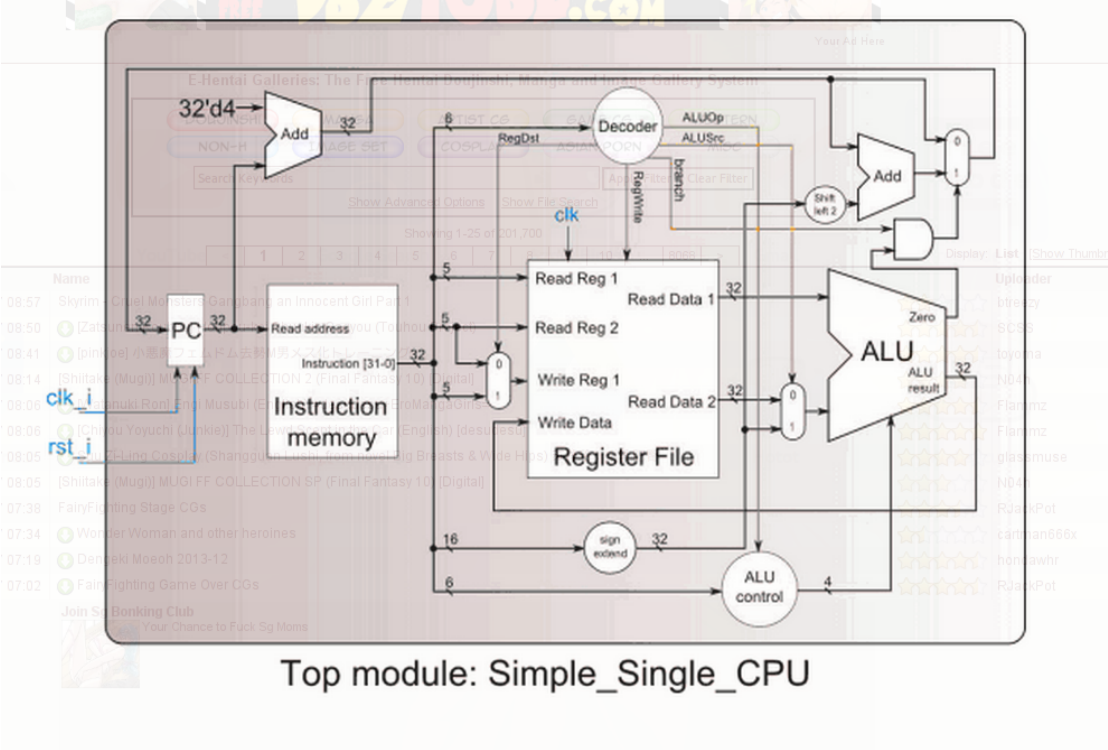
# Computer Organization

## Architecture diagrams:




Basically, my architecture is the same as in the pdf. In order to handle 'ori' cases, I add a wire from decoder to sign_extention module. (see the module analysis below).

## Hardware module analysis:

ALU.v / alu_top.v: the ALU component from lab1.

ALU_Ctrl.v: using 2 muxes to make a mapping from ALUOP_i to real ALUCtrl_o signals. Input: which type the instruction is. Output: what operation should the ALU do.

Adder.v: add up two 32-bit wire signals.

Decoder.v: decode the type of instruction and output some control signals for other muxes like RegDst_o or ALUSrc_o. Since the 'ori' instruction need a zero extension not signed extension, so I add a 'SinExt_o' wire here to tell the signed

extension module to do signed extension or not.

Instr_Memory.v: the instruction Memory of the CPU (from TA).

Reg_File.v: the register file, we have r0 to r14. We have RegWrite_i to select if we want to write over the registers or not. (from TA)

MUX_2to1.v: implement a 2x1 mux, just use the ?: syntax.

ProgramCounter.v: the PC of our CPU, actually it just pass the 32-bits bus from pc_in_i to pc_out_i.

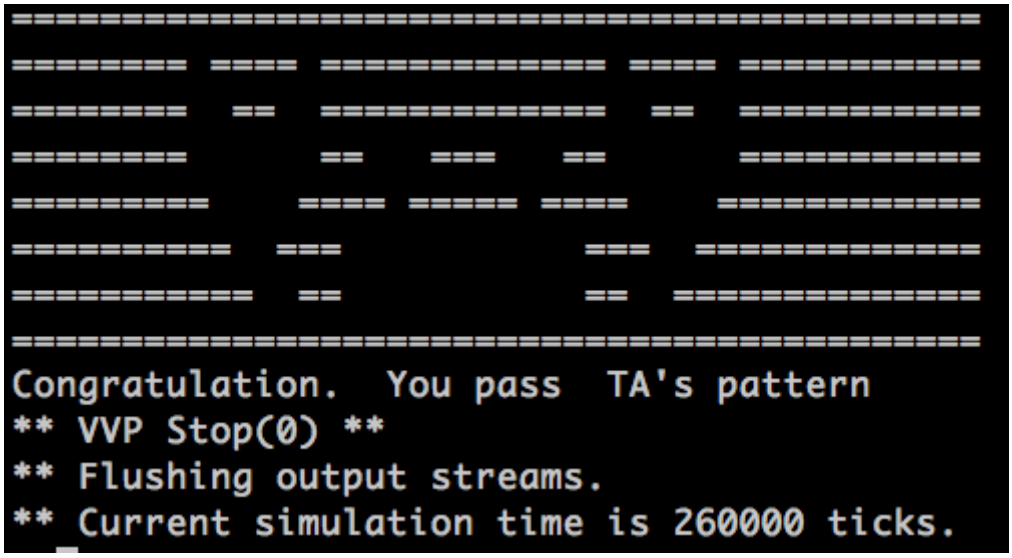
Shift_Left_Two_32.v: I use the bits concatenation operation to append two zeros.

Sign_Extend.v: Just repeat the MSB for 16 times and insert that in front of data_i. In 'ori' case, we insert 16 zeros.

Simple_Single_CPU.v: the top module, interconnect all the sub-modules.

## Experiment result:

I finished the basic part and got a crab from TAs, ya!




## Problems you met and solutions:

The modelSim in my XP virtual machine crashed again and again. So in this homework, I use iverilog to compile and test all my codes. It's really a good tool for me from the open source world.

There's lots of annoying trailing spaces in the TA's files. Also, it's not a good idea to mix tab and spaces during indenting our codes. I use the emacs editor to help me remove these stupid coding style issues.

## Summary:

I implemented a very simple single cycle CPU myself. The homework is not as hard as I thought before. Just implemented each module and then connect them together according to the homework pdf.