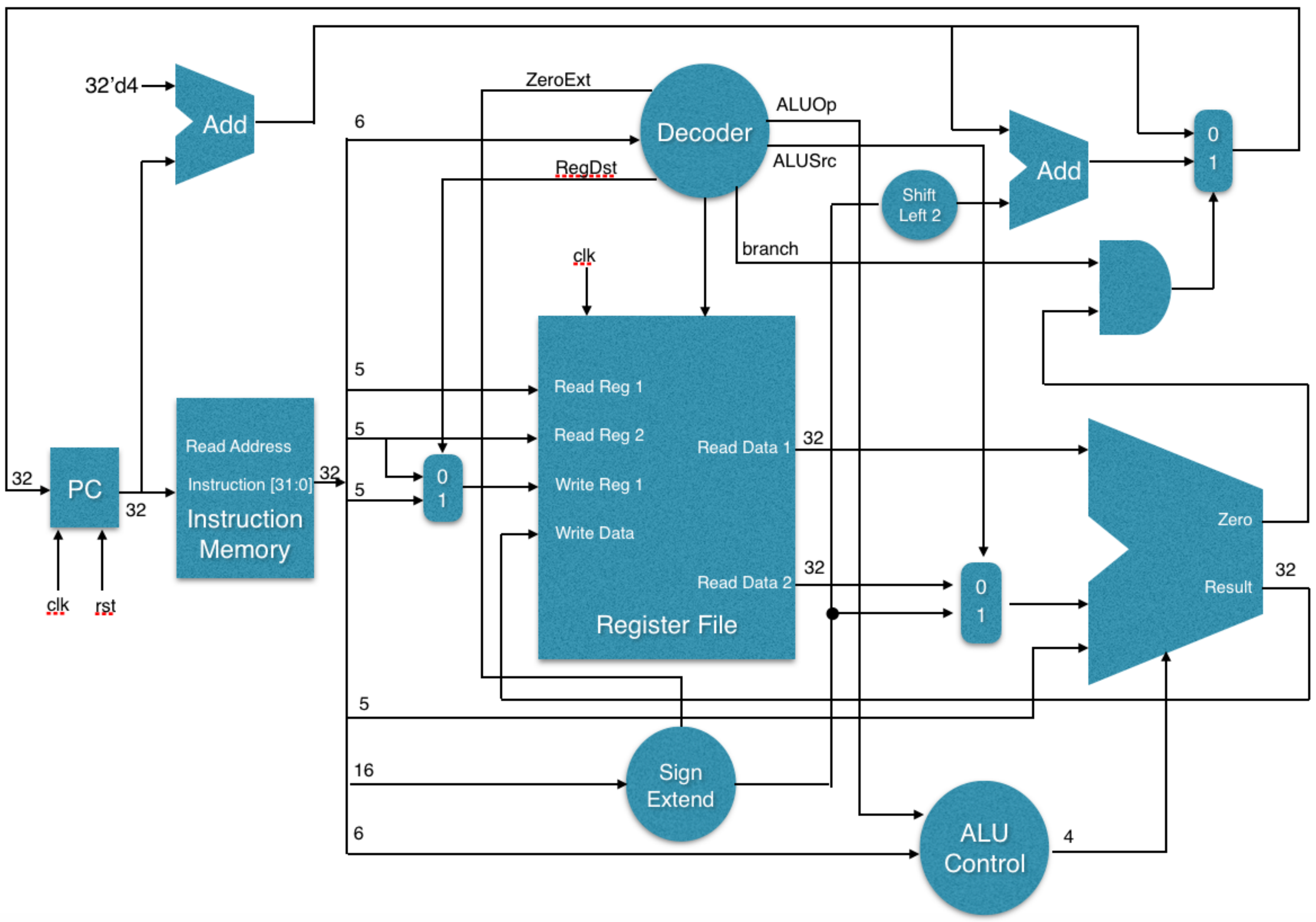
Computer Organization

Architecture diagram:



Detailed description of the implementation:

We have a program counter to continuously count current instruction count. Then we get the address to fetch instruction. For program counter, the next counter input will be either a branch address or just have addition of 4 to get to next instruction.

For instruction which is 32-bits, we divided into different parts. The first 6 bits are op codes to let decoder know what operation and how to control signals. On the other hand, bits[25:21], bits[20:16], and bits[15:11] are the rs, rt, and rd value for register file to process. As to the bits[15:0] are the Immediate value, it will extend as an input source of ALU if needed. The last bits[5:0] are the function code, which are used when instruction is a R-Type instruction.

ALU gets source from register file and do arithmetic selected by alu_ctrl. The result goes back to register file module for either save to register or something else.

Problems encountered and solutions:

Decoder signal effect the instruction, so be careful to set the output signal.
2'bit ALUOp is not enough for all the instruction. We extend to 4 bits
SRA, SRAV need to transform into signed signal to perform correct result.

Lesson learnt (if any):

We learned how to implement a simple CPU to perform a MIPS instructions.