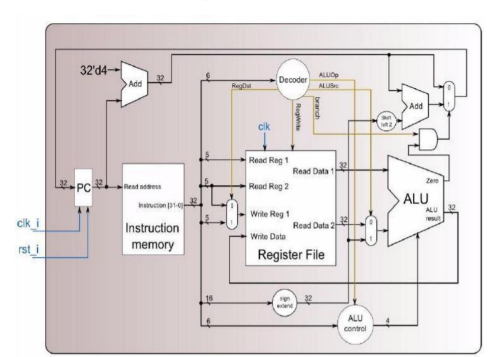
**Computer Organization**

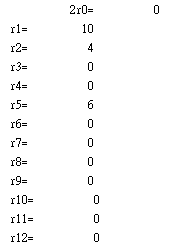
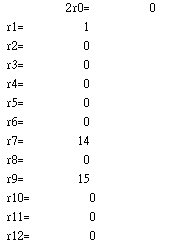
**Architecture diagrams:**

**Hardware module analysis:**

1. **Adder: this module add two inputs**
2. **ALU\_Ctrl: it decide the control of the alu depends on funct\_i and ALUOp\_i**
3. **ALU : this is the arithmetic logic unit**
4. **Decode: it decide the ALUOp\_i and the control signal to two of multiplexer ,register file , ALU control ,etc**
5. **Instr\_memory : this get the instruction depends on program counter**
6. **MUX\_2to1: this is the normal multiplexer**
7. **Program counter: this produce the program counter**
8. **Reg\_file: this module do the jobs of read and write operation on registers which are used in instructions**
9. **Shift\_Left\_Two: this shift the input left by two bits**
10. **Sign\_Extend: this extend the 16 bits of input repeat its sign bit until the input becomes 32 bits**
11. **Simple\_Single\_CPU: this is the top module to operate as a cpu**

**Finished part:**

**All the modules are completed and the results are below the first one is data1 and the following is data2**

****

**Problems you met and solutions:**

**Initially I don’t use the techniques that the professor taught during the class, I use the case statement to deal with the decision that the decoder and ALU\_Ctrl do. As a result, the original module that I design on my own idea can’t be done in one cycle clk. I found this by adding some display statement in my module to check the value of each signal. Finally I decide to turn to use the techniques that the professor taught**

**Which use two level the decide the outputs of the decoder and the ALU\_Ctrl**

**And it solve the problem perfectly.**

**Summary:**

**I think this lab is not so complicated but it need much patience to deal with the inputs and outputs of the signals of each modules.**