**Computer Organization**

**STUDENT NAME: 李欣洳 黃壽欽**

**STUDENT ID: 0416111 0540112**

**Architecture diagram:**

****

**Detailed description of the implementation:**

In this program, we have made a simple single CPU. The operation includes ADD, ADDI, SUB, AND, OR, SLT, SLTIU, BEQ, SRA, SRAV, LUI, ORI and BNE. In the instruction memory, the instruction is initialized and new instruction is loaded. In the register file, the register file is distributed into RS, RT and RD. In the sign-extend, the 16bits data is extend to 32bits. And for the Decoder, we send the option field of the instruction to it and the decoder generates the control signal and also the ALU\_op which ALU\_control needs. In the ALU-control, the ALU\_op from the Decoder and function field in the register are sent into it determine which function is needed to do and the order is sent to ALU. In the ALU, we do the functions according to the order from the ALU\_control given. The result of ALU will sent back to register file and the register is set as the answer.

**Problems encountered and solutions:**

When we implement the instruction SRA we do suffer from a problem, we found that we cannot send the data shamt in the SRA instruction. At first I find that the data from the singed\_extendtion unit contains the part of shamt and we decide to use it. Unfortunately, it didn’t work because of the input port src2 of ALU is busy. And then we found that the part RS of the SRA is ignored so that we decide to add a MUX unit and a control signal, which is generated from ALU\_control unit. That’s, we sent the data shamt of SRA via the src1 input port of the ALU unit. Finally, it did work.

**Lesson learnt (if any):**

In this project we do learn the knowledge about the CPU architecture. And it can help us understand how the data stream go through the inside CPU. And also we learn something about some architecture of the MIPS instruction.