112-1 NTU-CA Lab1 Report

`b10902138 陳德維`

---

1. Modules Explanation

- **`Adder.v`**
  - Adder module takes two 32-bit data `data1_i`, `data2_i` as input and output the bit-sum of the 2 datas (i.e. `data_o = data1_i + data2_i`). In this data path, the adder is used to increment `PC` by 4.
- **`ALU_Control.v`**
  - ALU Control module takes an instruction segments ([31:25]+[14-12]) and a 2-bit select signal `ALUOp` from Control module as input and output a 3-bit data for ALU as select signal to determine what operation should ALU do.
- **`ALU.v`**
  - ALU module takes a select signal input from ALU_Control module, two 32-bit input data from either register of sign-extended immediate and output a zero signal for branching and a ALU_result for the result after ALU finished the selected operation.
- **`Control.v`**
  - Control module takes a segment of the instruction([6:0]) and output 3 signal `RegWrite`, `ALUOp`, `ALUSrc` based on the type of the instruction for other modules to use.
- **`MUX32.v`**
  - MUX32 module implement a 32-bit input 2x1 multiplexer having one 1-bit select signal and two 32-bit data, and output the data which select signal indicated.
- **`Sign_Extend.v`**
  - Sign_Extend module takes a 12-bit immediate and extend its most significant bit to a 32-bit immediate.
- **`CPU.v`**
  - CPU module instantiate all the module above and uses wire to connect each module in order to construct the data path given in `Figure1`.

2. Development Environment

- OS: `Ubuntu 22.04` given by the `Dockerfile`
- Compiler: `iverilog 11.0-1.1`

- Text Editor: `VScode` + `Wavetrace`