NTU Computer Architecture 2022 HW3 Report

B09901027 賀崇恩

1. Modules Explanation

Registers.v

The Registers module takes clock, RS1 and RS2 (Register source) address, RD address and data (Register destination), and Register Write signal and input. The output is the RS1 and RS2 data bus. This module is used to simulate the Register file in CPU, which is a register array. The register is updated at the positive edge of the clock, if RegWrite is enabled. The output of RS1 and RS2 is combinational.

PC.v

The Program counter module takes clock, start signal, reset signal, and next cycle's pc as input. The output is the PC of current cycle. The module updates the internal register pc_o at the positive edge of clock, which is 0 if reset is set, otherwise set to next cycle's pc if start signal is set.

Instruction_Memory.v

This module takes address as input and the instruction as output. The module simulates memory via a register array with 255 words (1 word = 32 bits). The module is purely combinational.

Adder.v

This is a simple module which adds two 32-bit integer together. The module is purely combinational.

ALU_Control.v

This module takes func code (funct7 + funct3) and ALUOp code (from controller) as input and ALUCtrl as output, which is sent to ALU to indicate the operation. The ALUOp code is used to classify the R-type or I-type command. The module is purely combinational.

ALU.v

This module takes two data and ALUCtrl as input, which indicate the operation for ALU and is provided by ALU_Control. The output of the module includes the arithmetic operation result and “Zero” used for branch operation (not used in this hw). The module is purely combinational.

Control.v

This module is used for parsing the instruction and generates the control signal for ALUSrc for MUX, ALUOp for ALU_Control, and RegWrite for Registers. The input is the opcode in the instruction, which is either R-type and I-type. The module is purely combinational.

MUX32.v

This module takes two 32-bit data and one selection bit as input and assign one of the data to output determined by the selection bit. The module is purely combinational.

Sign_Extend.v

This module takes the input data, which contains 11 bits, and output the sign extension result with 32 bits via copying the MSB of input data 20 times. The module is purely combinational.

CPU.v

This module is the top module in this design. It includes all submodules listed above and connect them with data bus with appropriate size. There is no internal state in this module.

2. Development Environment

I used Windows subsystem for linux (WSL) for the compile command (“iverilog -o CPU.out *.v” and “vvp CPU.out”) and gtkwave in Windows command line via “gtkwave.exe CPU.vcd” for debugging.