CA Lab1 Report

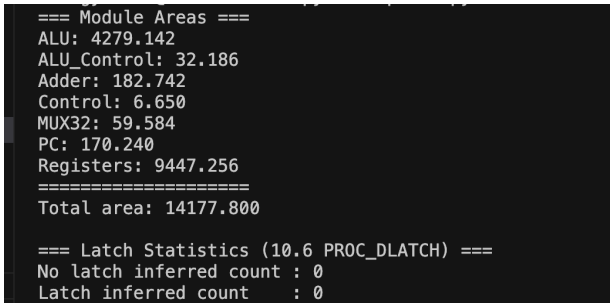
B12902125 張晏華

October 29, 2025

1. Explain modules in plain language

   - Adder
     Adder module takes two 32-bit inputs, data1_i and data2_i, and outputs their 32-bit sum as data_o. This module performs combinational addition without using any clock or reset signals. It is mainly used to calculate the next program counter value (pc_next = pc_cur + 4) in this homework. Since it is combinational, the output updates immediately when either input changes.
   - ALU
     ALU module takes two 32-bit input operands, data1_i and data2_i, and performs arithmetic or logical operations according to the 3-bit control signal ALUCtrl_i. It supports operations such as AND, XOR, SLL, ADD, SUB, MUL, ADDI, and SRAI. This module is combinational so the output data_o updates immediately when inputs or control signals change. For SLL and SRAI, only the lower 5 bits of data2_i are used as the shift amount. Signed arithmetic is applied in SRAI operation.
   - ALU_Control
     ALU_Control module determines the specific ALU operation based on the instruction's function fields and the high-level ALUOp signal from the Control unit. The 10-bit input func_i combines funct7 and funct3 fields of the instruction, while ALUOp_i specifies whether the instruction is R-type or I-type. According to these signals, the module outputs a 3-bit control signal ALUCtrl_o to select the correct ALU operation such as AND, XOR, SLL, ADD, SUB, MUL, ADDI, or SRAI. This module is combinational and updates its output immediately when any input changes.
   - Control
     Control module generates the main control signals for the CPU based on the 7-bit instruction opcode. It determines whether the instruction is R-type or I-type and produces three outputs: ALUOp_o, ALUSrc_o, and RegWrite_o. For R-type instructions, it sets ALUOp_o to 01, selects the second ALU operand from a register, and enables register write. For I-type instructions, it sets ALUOp_o to 00, selects an immediate value as the second ALU operand, and enables register write. For unsupported opcodes, all control signals are disabled. This module is combinational and updates its outputs immediately when the opcode input changes.
   - CPU
     CPU module integrates the single-cycle datapath and control. It receives clk_i and rst_i, and connects the provided PC, Instruction_Memory, and Registers with the designed Control, ALU_Control, Sign_Extend, MUX32, Adder, and ALU.
   - MUX32
     MUX32 module selects one of two 32-bit inputs to pass to the output based on the 1-bit control signal select_i. In this work, MUX32 is used to choose the second operand for the ALU. When alu_src is 0, the output data_o is taken from the register value rs2_data; when it is 1, the output comes from the sign-extended immediate value imm_ext. This module is purely combinational, so the output updates immediately whenever any input changes.
   - Sign_Extend
     Sign_Extend module extends a 12-bit immediate input data_i to a 32-bit signed value data_o. The most significant bit (data_i[11]) is used as the sign bit and replicated 20 times to fill the upper bits of the output. In this work, the Sign_Extend module is used to convert the immediate field of I-type instructions (instr[31:20]) into a 32-bit signed value before sending it to the ALU. This module is combinational, so the output updates immediately whenever the input changes.

2. Synthesizability / latch check screenshot.



3. Development environment

Category	**Tool / Version**
OS	Ubuntu 22.04 (Docker container)
Verilog Compiler	Icarus Verilog 11.0–1.1
Synthesizer	Yosys
Environment	Docker + Docker Compose
IDE	Visual Studio Code