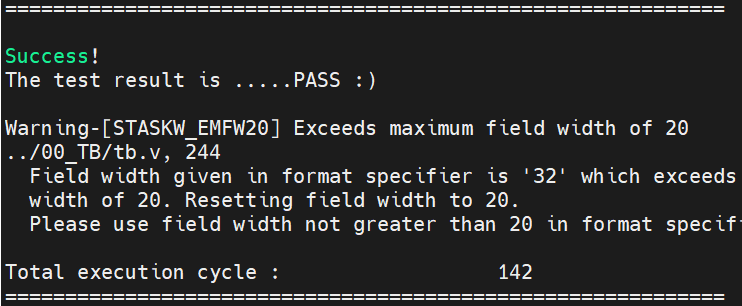
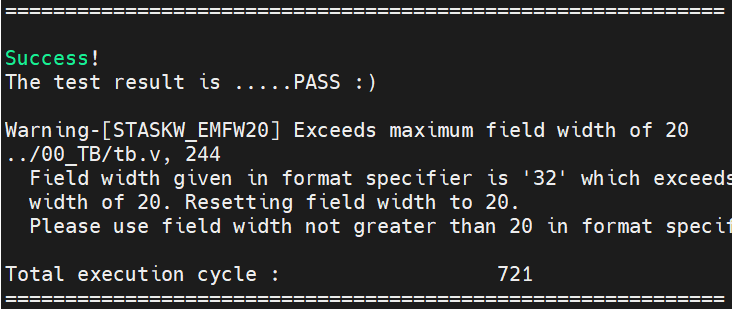
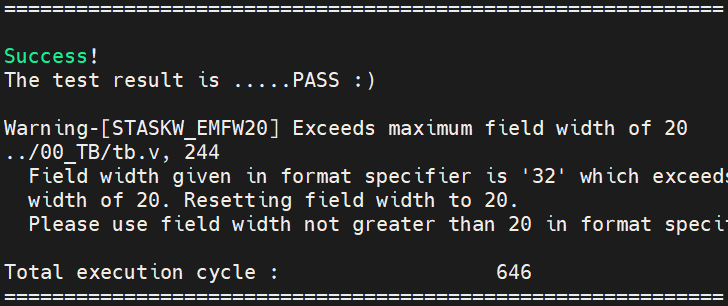
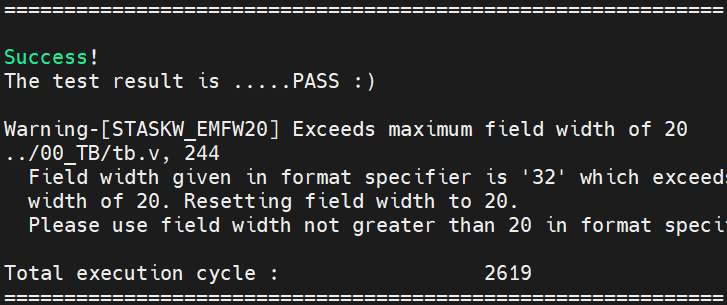
Computer Architecture Final Project Report

B10901166 許蘊琰 B10901006 鄒昀樺

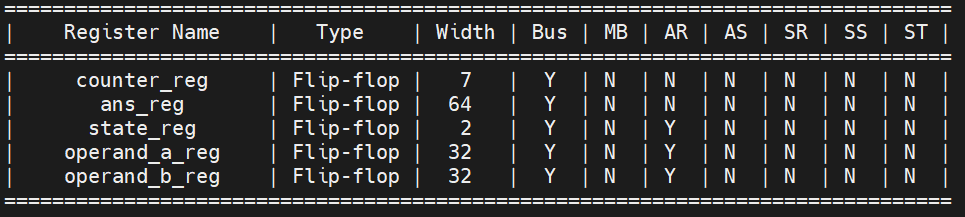
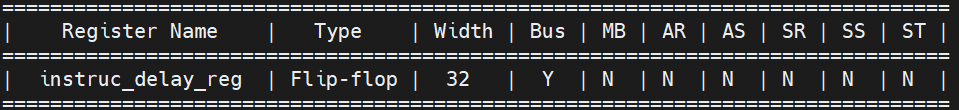
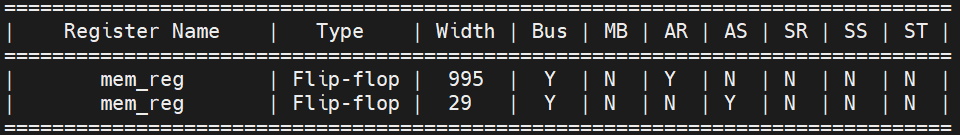
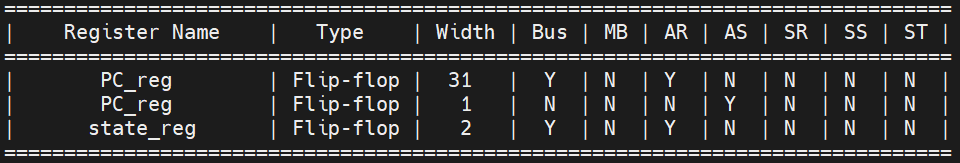
**1. Execution Cycle Number**

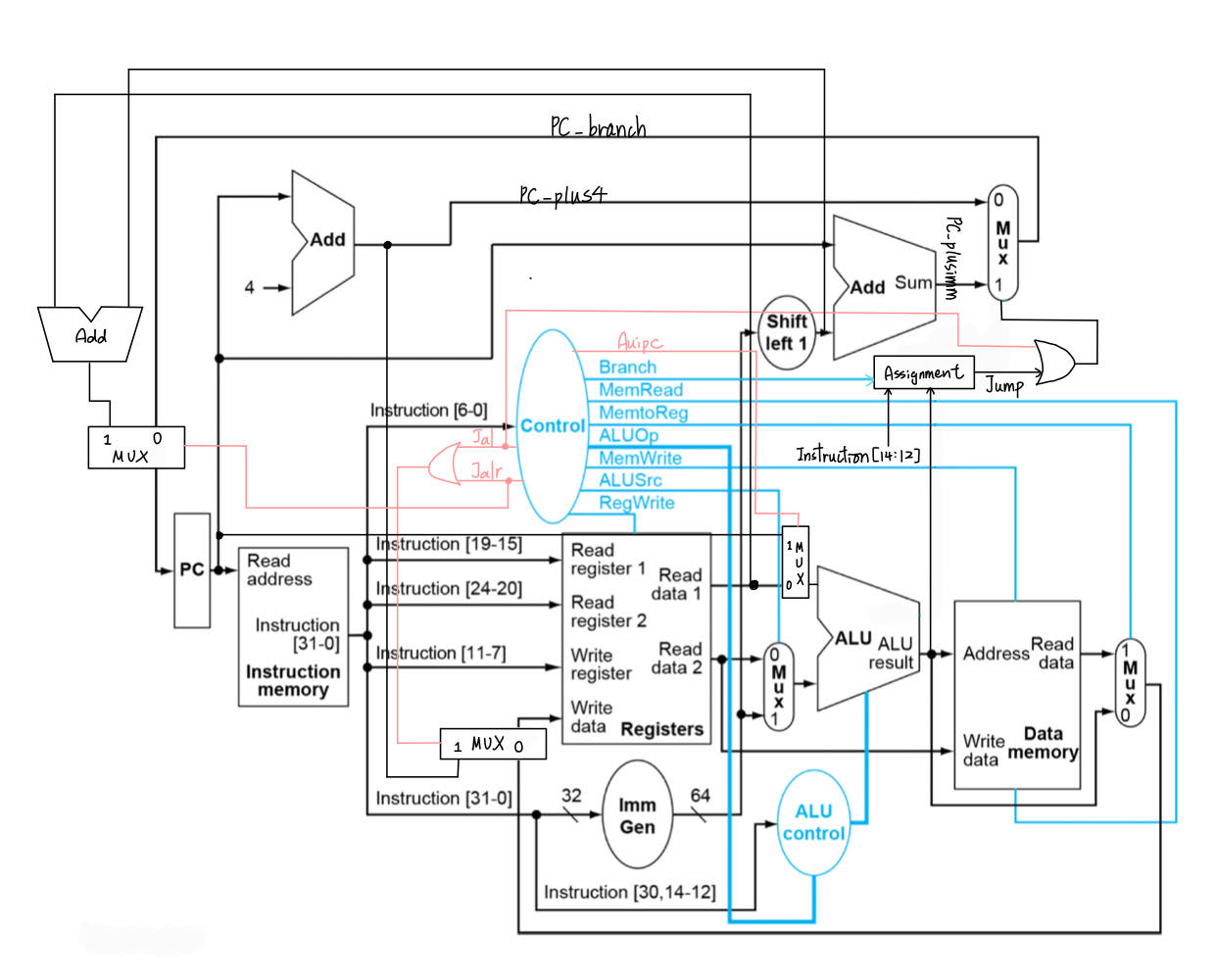
 

|  |  |
| --- | --- |
| Instruction Set | Execution Cycle |
| I0 | 142 |
| I1 | 721 |
| I2 | 646 |
| I3 | 2619 |

**2. Synthesizable Check: all flip-flops**



**3. Design Description**



**CPU FSM**

**IF & Single Cycle Instruction**

**MUL**

**Access Dmem**

alu\_done = 1

(counter == 32)

**Wait for Dmem**

* Instruction fetch and handle single cycle instructions
* Access data memory
* Wait for data memory access finished
* MUL calculations

**CPU Submodules**

* ImmGen: Generate different immediate values for various instructions.
* Controller: Generate control signals (e.g. ALUSrc, Branch…) and ALUOp according to different instructions.
* ALUControl: Assign ALU control input depending on ALUOp and funct3, 7.
* ALU: Implement arithmetic operations according to ALU control input
* MULDIV\_unit: Handle multiplications
* Reg\_file

**Special Instructions**

* Jal
  + Decode opcode and use ImmGen to generate imm
  + PC\_branch = PC + imm
  + Write PC + 4 into reg
* Jalr
  + Decode opcode and use ImmGen to generate imm
  + Jalr = 1, PC\_nxt = rd1 + imm
  + Write PC + 4 into reg
* Auipc
  + Decode opcode and use ImmGen to generate imm
  + Use Auipc (pulled to high) signal to determine MUX value
  + MUX = 1, ALUresult = PC + imm
  + MUX = 0, ALUresult = rd1 + imm

**Handling Multi-cycle Instructions**

Once the CPU receives **mul** instruction, it will jump to multi-cycle state. During the calculations, use a counter to count how many cycles has passed. As soon as the counter reaches 32, pull up alu\_done (which means the calculations is done) and the FSM will change to the next state.

4**. Work distribution Table**

|  |  |  |
| --- | --- | --- |
| Name  Work | 鄒昀樺 | 許蘊琰 |
| ImmGen | 70% | 30% |
| Control | 60% | 40% |
| ALU related | 30% | 70% |
| MULDIV | 40% | 60% |
| Report | 50% | 50% |