計算機組織 Lab3

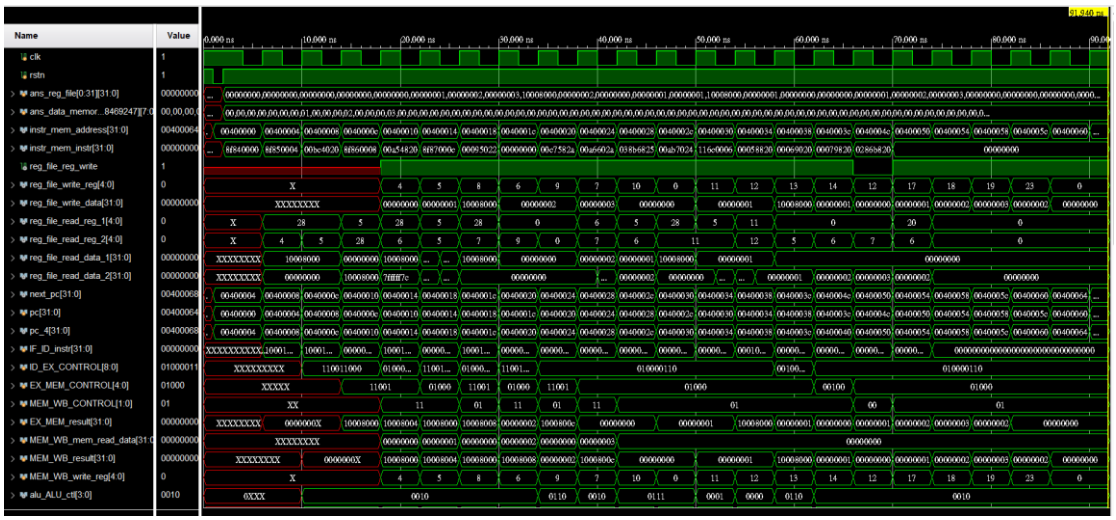
學號：111613025

系級：材料 15

姓名：黃綵誼

I. Experimental Result

1. Show the waveform screen shot of the test we provided.



II. Answer the following Questions

1. For each code sequence below, state whether it must stall, can avoid stalls using only forwarding, or can execute without stalling or forwarding.

Sequence 1	Sequence 2	Sequence 3

lw $t0,0($t0)
add $t1,$t0,$t0
	add $t1,$t0,$t0
	addi $t2,$t0,#5
	addi $t4,$t1,#5
		addi $t1,$t0,#1
		addi $t2,$t0,#2
		addi $t3,$t0,#2
		addi $t3,$t0,#4
		addi $t5,$t0,#5

(1) Sequence 1 must stall.
(2) Sequence 2 can avoid stalls using only forwarding.
(3) Sequence 3 can execute without stalling.

2. Explain the difference between throughput and latency.

Throughput refers to the amount of data that can be transferred from one place to another within a given period of time.

Latency, on the other hand, refers to the time it takes for a data packet to travel from the source to the destination. It is the delay between the initiation of a request and the beginning of a response.

3. A group of students were debating the efficiency of the five-stage pipeline when one student pointed out that not all instructions are active in every stage of the pipeline. After deciding to ignore the effects of hazards, they made the following five statements. Which ones are correct? Explain why or why not.

(1) Allowing jumps, branches, and ALU instructions to take fewer stages than the five required by the load instruction will increase pipeline performance under all circumstances.

This statement is not entirely correct. While reducing the number of stages for certain types of instructions like jumps, branches, and ALU instructions may improve performance in some cases, it's not guaranteed to do so under all circumstances. Different instructions have different dependencies and resource requirements, so optimizing the pipeline for one type of instruction may negatively impact the performance of others. It's a trade-off between optimizing for certain instructions and maintaining overall balance in pipeline performance.

(2) Trying to allow some instructions to take fewer cycles does not help, since the throughput is determined by the clock cycle; the number of pipe stages per instruction affects latency, not throughput.

This statement is not correct. Throughput is indeed determined by the clock cycle, but reducing the number of pipeline stages for certain instructions can still indirectly affect throughput. If some instructions take fewer cycles to execute, it means they can progress through the pipeline more quickly, potentially allowing for more instructions to be executed per unit of time, thus improving throughput. However, this improvement in throughput might be limited by other factors such as resource availability or dependencies between instructions.

(3) You cannot make ALU instructions take fewer cycles because of the writeback of the result, but branches and jumps can take fewer cycles, so there is some opportunity for improvement.

This statement is generally correct. ALU instructions typically involve several stages, including fetching operands, executing the operation, and writing back the result. The writeback stage can't be eliminated for ALU instructions because it's necessary to update the

register file with the result. However, branches and jumps may be optimized to take fewer cycles by reducing the number of stages involved in their execution, as they don't require a writeback stage in the same way as ALU instructions.

(4) Instead of trying to make instructions take fewer cycles, we should explore making the pipeline longer, so that instructions take more cycles, but the cycles are shorter. This could improve performance.

This statement is not correct. Increasing the length of the pipeline can indeed improve performance by allowing for more efficient instruction processing and better parallelism. However, simply making instructions take more cycles without considering the overall balance and efficiency of the pipeline can lead to increased latency and potential performance degradation. Lengthening the pipeline should be done cautiously, considering the trade-offs between latency, throughput, and the potential for pipeline hazards.

III. (optional) Problems Encountered & Solution

List some important problem you've met during this lab and the solution.

(1) reg_file_write_reg, reg_file_reg_write, reg_file_write_data 有正確訊號但 register 寫入失敗


```
$6 expect 0x00000002 found 0x7fffff7c
$8 expect 0x10008000 found 0x00000000
$11 expect 0x00000001 found 0x00000000
$12 expect 0x00000001 found 0x00000000
$14 expect 0x00000001 found 0x00000000
$18 expect 0x00000002 found 0x00000000
$19 expect 0x00000003 found 0x00000000
$23 expect 0x00000002 found 0x7fffff7c
```

```
always @(posedge clk)
    if (rstn) begin  // make sure to check reset!
        if (reg_write & write_reg) begin
            registers[write_reg] <= write_data;
        end
    end
```

Sol: reg_file.v 中用 ”===“ 來判斷條件

```
always @(posedge clk)
    if (rstn) begin  // make sure to check reset!
        if (reg_write === 1 & write_reg !== 'b0 & write_reg !== 'bx) begin
            registers[write_reg] <= write_data;
        end
    end
```