Computer Architecture 2022 Fall Midterm Exam

Student ID: B09902105 **Name:** 劉元瑞

Unless specified otherwise, assume that:

- RISC-V assembly format and calling convention are listed in Appendix.

---

1. [5pts] We would like to parallelize a program on a two-core processor. The workload of the target program is evenly partitioned between these two cores. Let's assume when running the parallelized version, each core runs at 70% of the voltage and frequency of the settings of a serial version. Due to the synchronization overhead between two threads, the execution time of each thread increases 10% compared to the serial version. What is the energy saving we can get from parallelizing the target program on a two-core processor?

2. [5pts] Suppose a program runs in 150 seconds on a computer, with multiply operations responsible for 100 seconds of the total execution time. Is it possible to execute the program 4 times faster by improving the speed of multiplication? Please explain your answer.

3. [5pts] You are asked to design a single-cycle processor to accelerate a workload which spends 40% of time in function `f()`. Each function `f()` executes 20 dynamic instructions. One possible design is to add a new instruction that could perform function `f()` in one cycle. But adding this new instruction would double the cycle time. Is this a good solution? Please explain your answer.

4. [15pts] Compilers can have a profound impact on the performance of an application on a given processor. This problem will explore the impact compilers have on execution time. (Please round all answer to the 2nd decimal place)

Compiler A		Compiler B	
instruction count	Execution Time	instruction count	Execution Time
1.70E+9	3.4s	1.70E+9	4.08s

(1) [5pts] For the same program, two different compilers are used. The table above shows the execution time of the two different compiled programs. Find the average CPI for each program given that the processor has a clock cycle time of 2ns.

(2) [5pts] Assume the average CPIs found in (1), but that the compiled programs run on two different processors. If the execution times on the two processors are the same, how much faster is the clock of the processor running compiler A's code versus the clock of the processor running compiler B's code?

(3) [5pts] A new compiler is developed that uses only 1.40E+9 instructions and has an average CPI of 1.4. What is the speed-up of using this new compiler versus using Compiler A or B on the original processor of (1)?

5. *[8pts] Registers are usually used to hold temporaries. Ideally, the more registers, the more objects could be allocated to registers, and memory references to those objects could be saved. Assume that we would like to expand the RISC-V (32-bit instruction/32 registers) register file to 64 registers. Please answer the following questions.*

(1) [4pts] What is the total number of bits needed for a R-type format instruction?

(2) [4pts] How would the proposed change affect the size of a RISC-V assembly program?

6. [6pts] The pseudo instruction is an instruction that will be mapped to a sequence of real instructions by the assembler. For instance, `la rd, symbol` (load the address of `symbol` to `rd`) will be mapped to two instructions:

```
auipc rd, delta[31:12] + delta[11]
addi rd, rd, delta[11:0]
```

where `delta = symbol - pc`

For example, assuming that the address of the `la` instruction is 0x2000 and the address of `symbol` is 0x3000, the assembler calculates 0x3000 - 0x2000 = 0x1000 as `delta` which is then used to construct the `auipc` and `addi` instructions.

`auipc` (add upper immediate to pc) is used to build pc-relative addresses and uses the U-type format. `auipc` forms a 32-bit offset from the 20-bit U-immediate, filling in the lowest 12 bits with zeros, adds this offset to the address of the `auipc` instruction, then places the result in register `rd`.

Please answer the following questions.

(1) [2pts] What is the advantage of having pseudo instructions?

(2) [4pts] In the above example, the `la` instruction is translated into two instructions. Why does the first instruction need to add `delta[11]` to the upper twenty bits of delta? Please explain what would happen if we replace `auipc rd, delta[31:12] + delta[11]` with `auipc rd, delta[31:12]`?

7. [8pts] Translate the following C code to 64-bit RISC-V assembly code. Use a minimum number of instructions. Assume that the C-level integer variable `i` and `a` are held in x5 and x6, respectively. And x17, x18, and x19 hold the base address of the **long** type array `A`, `B`, and `C`, respectively. (Assume the size of a long type variable in C codes is 8 bytes.)

```
for (i = 0; i < a; i++) {
    C[i] = A[i] + B[4] + i;
}
```

8. [8pts] Most ISAs have separate register files: general purpose registers and floating point registers. Why do most modern ISAs provide different sets of registers? Would it be better to use just one set of general purpose registers? For example, we could have instructions such as:

```
add R1, R2, R3 /* Integer Add */
fadd R1, R2, R3 /* Floating point add */
```

Although these are different types of instructions, they could use the same general purpose registers.

Assume that we have two different architectures (1) unified register file architecture that has 2N registers and (2) separate register file architecture that has N general purpose registers and N floating-point registers. Please answer the following questions:

(1) [4pts] What changes are required in the instruction set architecture to support the unified register file? Are these changes better or worse than the separate register file architecture?

(2) [4pts] How would the unified register file affect the performance? Please explain the advantage and the disadvantage of the proposed change.

9. [10pts] Consider the single-cycle RISC-V CPU shown in Figure 1. Please answer true or false for each of the following statements. Note that you must point out why the statement is wrong if you answer false.

(1) [2pts] If the path from the 'Read data 2' port in the register file to the 'Write data' port in data memory has been cut, the instructions **`add`**, **`slt`**, and **`sw`** still can run correctly.

(2) [2pts] If the path from 'ImmGen' to 'Shift left 1' has been cut, the instructions **`lw`**, **`sw`** and **`beq`** may fail.

(3) [2pts] If the control signal ALUsrc is stuck on 0, the instructions **`lw`**, **`sw`**, and **`beq`** may fail to run correctly.

(4) [2pts] If the control signal RegWrite is stuck on 0, the instructions **`lw`** and **`sw`** still can run correctly.

(5) [2pts] If the control signal MemToReg is stuck on 1, the instructions **`add`**, **`sub`**, **`lw`**, and **`slt`** may fail to run correctly.

10. [15pts] Consider the single-cycle CPU architecture in Figure 1. The latency of each block is given in the following. Assume that the hardware module has zero delays if not specified.

I-Mem	Add	Mux	ALU	Regs	D-Mem	ImmGen	Shift-left-1
200ps	50ps	20ps	120ps	80ps	250ps	20ps	10ps

```
1020   sub $s4, $s3, $t3
1024   beq $t2, $s4, Else
1028   addi $t2, $t2, -4
1032   add $t0, $t1, $t2
1036   sw $s2, 0($t0)
       ...
1088   Else: addi $t2, $t2, 4
```

(1) [8pts] Suppose now the processor only needs to support instructions shown above. Please calculate the latency of the critical path for each instruction.

(2) [2pts] What would the cycle time be?

(3) [5pts] Before this code segment, the register contents in decimal are given as follows. After execution, for some clock cycles, if X=1028, what are the values of signal Y (after the MUX controlled by ALUSrc) and signal Z (input of PC)?

$t0	$t1	$t2	$t3	$s2	$s3	$s4
48	8	12	288	75	300	102

12

11. [15pts] A new S-type format instruction **swu** has been added to the RISC-V instruction set. Its format is **swu rs2, 1(rs1)**. It takes arguments register **rs2**, register **rs1**, and immediate 1, and it stores the contents of **R[rs2]** at the memory address (**R[rs1]+1**).

(1) [7pts] Given the single-cycle datapath in Figure 1, fill in the control signals in the table below for this new instruction. Each control signal must be specified as 0, 1, or X (don't care) ('add', 'sub', 'mul', … for ALUOp). Writing a 0 or 1 when an X is more accurate is incorrect.

RegWrite	ALUSrc	ALUOp	MemWrite	MemRead	MemtoReg	PCSrc

(2) [8pts] Now one more new I-type format instruction **jin imm12(rs1)** is developed for the single-cycle processor, which is a Jump Indirect instruction and will cause the processor to jump to the address stored in the word at memory location **imm12 + R[rs1]** (the same address computed by **lw**). Draw the necessary modifications to *implement the* **jin** instruction directly on Figure 1 in your question sheet.

Student ID: B0990215 **Name:** 鄭天聖

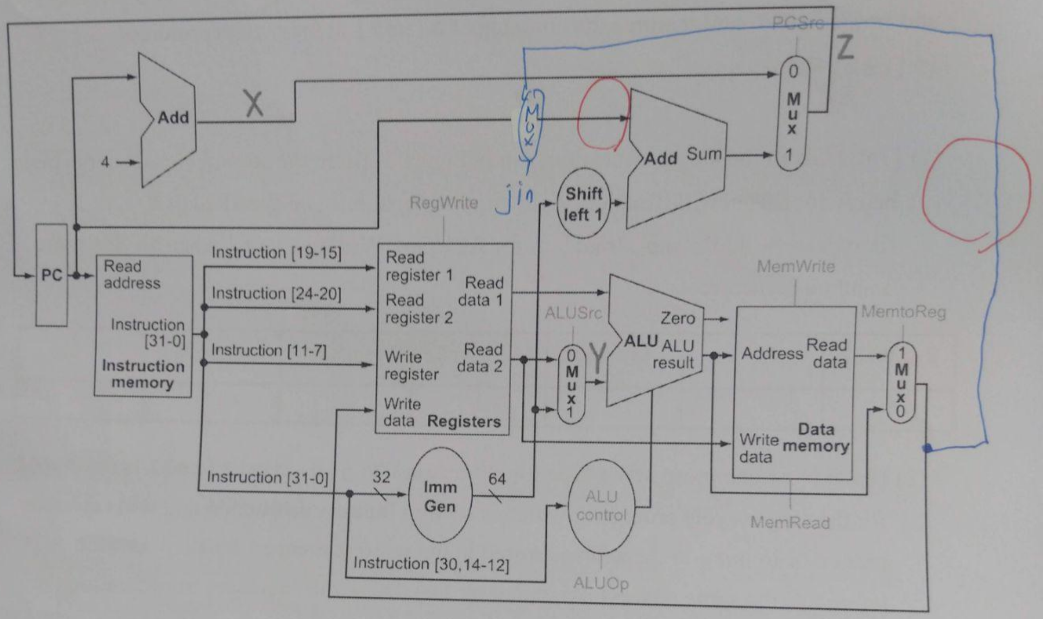

Figure 1

+3

Student ID: B09902105 **Name:** 鄭[illegible]

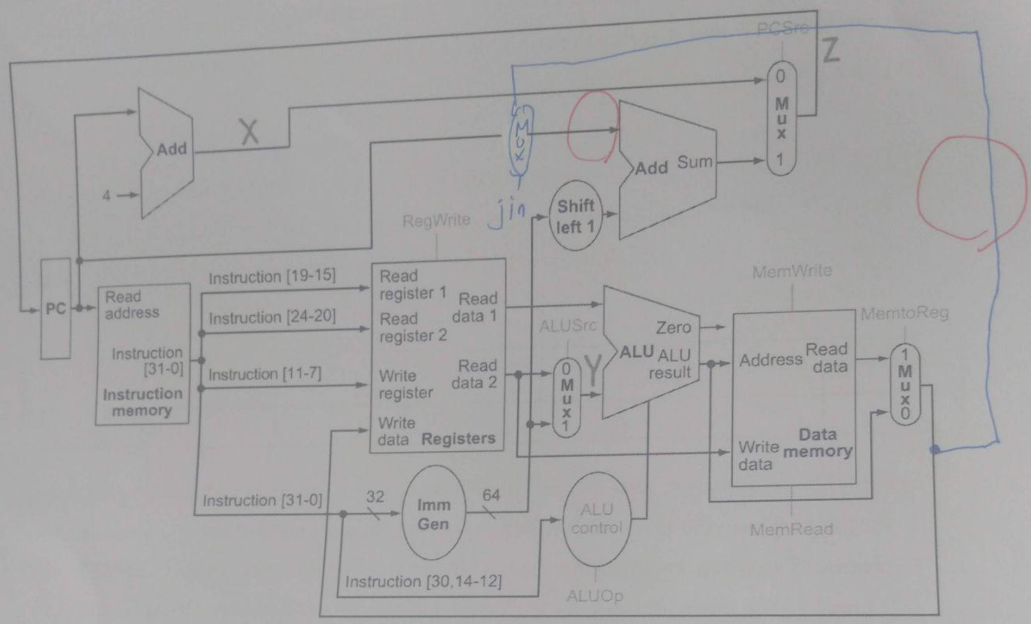

Figure 1

+3