Habib University

Dhanani School of Science and Engineering

Computer Architecture
EE 371 / CS 330 / CE 321 (Spring 2024)

Homework 1 Solution

Release Date: January 25, 2024	**Due by:** February 1, 2024 11:59 PM

Total marks: 100	**Marks obtained:**

Student Name:	**Student ID:**	**Section:**

Purpose:

The purpose of this assignment is to help you in understanding the role of ISA in modern processors. In addition, this homework will aid you in practicing assembly language programming, and instruction encoding of RISC-V microprocessor.

Instructions:

1. This assignment should be done individually.
2. All questions should be answered in **black ink only**.
3. Scan your answer sheet and upload it on HU LMS before the due date.

Grading Criteria:

1. Your assignments will be checked by instructor.
2. You can also be asked to give a viva where you will be judged whether you understood the question yourself or not. If you are unable to answer correctly to the question you have attempted right, you may lose your marks.
3. Zero will be given if the assignment is found to be plagiarized.
4. Untidy work will result in reduction of your points.

Submission policy:

1. No submission will be accepted after the instructor releases the solution on HU LMS.

CLO Assessment:

This assignment assesses students for the following course learning outcomes.

Course Learning Outcomes		CLO Assessed
CLO 1	*Explain* the role of ISA in modern processors and instruction encodings and assembly language programming	✓
CLO 2	*Explain* the architecture and working of a single cycle processor	
CLO 3	*Design* the architecture to mitigate issues of a pipelined processor	
CLO 4	*Analyze the* performance of cache operations	

Question 1

Consider the following performance measurements for a program:

Measure	Computer A	Computer B
Instruction Count	20 million	16 million
Clock Rate	4 GHz	4 GHz
CPI	1.0	1.1

Answer the following questions:

a. Which computer has the higher MIPS rating? **(5 marks)**

b. Which computer is actually faster in terms of CPU execution time? **(5 marks)**

We know that,

$$MIPS = \frac{Clock\ Rate}{CPI \times 10^6}$$

a. Computer A:

$$MIPS_A = \frac{4 \times 10^9}{1 \times 10^6} = 4 \times 10^3$$

Computer B:

$$MIPS_B = \frac{4 \times 10^9}{1.1 \times 10^6} = 3.636 \times 10^3$$

We know that,

$$Execution\ Time = \frac{Instruction\ Count \times CPI}{Clock\ rate}$$

b. Computer A:

$$Execution\ Time_A = \frac{20 \times 10^6 \times 1}{4 \times 10^9} = 5ms$$

Computer B:

$$Execution\ Time_B = \frac{16 \times 10^6 \times 1.1}{4 \times 10^9} = 4.4ms$$

Computer B is Faster.

Question 2

a. Suppose a program runs in 200 seconds on a computer, with multiply operations responsible for 160 seconds of this time. Suppose a new multiply unit is introduced into the computer which will allow multiplication operations to run 4 times faster.

What will be the execution time after introducing this improvement? **(5 marks)**

b. Suppose a program runs in 100 seconds on a computer, with multiply operations responsible for 80 seconds of this time.

How much do I have to improve the speed of multiplication if I want my program to run 5 times faster? **(5 marks)**

Amdahl's Law is given by:

$$Ex.Time\ After\ Improvement = \frac{Ex.Time\ Affected\ by\ Improvement}{n} + Ex.Time\ Unaffected$$

a. $Ex.Time\ After\ Improvement = \frac{160}{4} + (200 - 160) = 80$

b. $\frac{100}{5} = \frac{16}{x} + (100 - 80)$

$20 = \frac{16}{x} + 20$

$\frac{16}{x} = 0$

There is no x for which the above equation holds. This means that we have to improve the multiply operation by infinite times which is practically impossible.

Question 3

A compiler designer is trying to decide between two code sequences for a computer. The hardware designers have supplied the following facts:

	CPI for each instruction class		
	Add	Multiply	Divide
CPI	1	2	4

For a particular high-level language statement, the compiler writer is considering two code sequences that require the following instruction counts:

	Instruction counts for each instruction class		
Code Sequence	Add	Multiply	Divide
A	3	2	3
B	5	2	2

Answer the following questions:

a. Which code sequence executes the most instructions? **(5 marks)**

b. Which code sequence will be faster in terms of number of clock cycles taken? **(5 marks)**

c. What is the CPI for each code sequence? **(5 marks)**

a. Sequence A executes $3 + 2 + 3 = 8$ instructions. Sequence B executes $5 + 2 + 2 = 9$ instructions. Therefore, Sequence B executes most instructions.

b. $Clock\ Cycles = \sum_{i=i}^{n} (CPI_i \times C_i)$

$Clock\ Cycles_A = 3 \times 1 + 2 \times 2 + 3 \times 4 = 19$
$Clock\ Cycles_B = 5 \times 1 + 2 \times 2 + 2 \times 4 = 17$

Sequence B is faster since it takes lesser clock cycles.

c. $CPI_A = \frac{Clock\ Cycles_A}{Instruction\ Count_A} = \frac{19}{8} = 2.375$

$$CPI_B = \frac{Clock\ Cycles_B}{Instruction\ Count_B} = \frac{17}{9} = 1.889$$

Question 4

Consider two different implementations of the same instruction set architecture. The instructions can be divided into four classes according to their CPI (classes A, B, C, and D). P_1 with a clock rate of 2.5 GHz and CPIs of 1, 2, 3, and 3, and P_2 with a clock rate of 3 GHz and CPIs of 2, 2, 2, and 2.

a. Given a program with a dynamic instruction count of 10^6 instructions divided into classes as follows: 20% class A, 20% class B, 30% class C, and 30% class D, which is faster: P_1 or P_2? **(5 marks)**

b. What is the global CPI for each implementation? **(5 marks)**

c. Find the clock cycles required in both cases. **(5 marks)**

a.

Class A = $1.0 \times 10^6 \times 0.2 = 2 \times 10^5$

Class B = $1.0 \times 10^6 \times 0.2 = 2 \times 10^5$

Class C = $1.0 \times 10^6 \times 0.3 = 3 \times 10^5$

Class D = $1.0 \times 10^6 \times 0.3 = 3 \times 10^5$

CPU Clock Cycles$_1$ = $(2 \times 10^5 \times 1) + (2 \times 10^5 \times 2) + (3 \times 10^5 \times 3) + (3 \times 10^5 \times 3) = 24 \times 10^5$

Execution Time$_1$ = $24 \times 10^5 / 2.5 \times 10^9 = 9.6 \times 10^{-4}$ sec

CPU Clock Cycles$_2$ = $(2 \times 10^5 \times 2) + (2 \times 10^5 \times 2) + (3 \times 10^5 \times 2) + (3 \times 10^5 \times 2) = 20 \times 10^5$

Execution Time$_2$ = $20 \times 10^5 / 3 \times 10^9 = 6.67 \times 10^{-4}$ sec

P_2 is faster.

b.

CPI$_1$ = CPU Clock Cycles$_1$ / Instruction Count = $24 \times 10^5 / 10^6 = 2.4$

CPI$_2$ = CPU Clock Cycles$_2$ / Instruction Count = $20 \times 10^5 / 10^6 = 2.0$

c.

CPU Clock Cycles$_1$ = $(2 \times 10^5 \times 1) + (2 \times 10^5 \times 2) + (3 \times 10^5 \times 3) + (2 \times 10^5 \times 3) = 24 \times 10^5$

CPU Clock Cycles$_2$ = $(2 \times 10^5 \times 2) + (2 \times 10^5 \times 2) + (3 \times 10^5 \times 2) + (3 \times 10^5 \times 2) = 20 \times 10^5$

Question 5

Compilers can have a profound impact on the performance of an application. Assume that for a program, compiler A results in a dynamic instruction count of 10^9 and has an execution time of 1.8 s, while compiler B results in a dynamic instruction count of 1.5×10^9 and an execution time of 1.3 s.

a. Find the average CPI for each program given that the processor has a clock cycle time of 1 ns. **(5 marks)**

b. Assume 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? **(5 marks)**

c. A new compiler is developed that uses only 9×10^8 instructions and has an average CPI of 1.7. What is the speedup of using this new compiler versus using compiler A or B on the original processor? **(5 marks)**

a. CPI_A = Execution Time$_A$ / (Clock cycle time × Instruction count$_A$) = 1.8 / ($1 \times 10^{-9} \times 1 \times 10^9$) = 1.8

CPI_B = Execution Time$_B$ / (Clock cycle time × Instruction count$_B$) = 1.3 / ($1 \times 10^{-9} \times 1.5 \times 10^9$) = 0.867

b. After dividing the common execution time, the following expression can be used:

Clock rate$_A$ / Clock rate$_B$ = (Instruction count$_A$ × CPI_A) / (Instruction count$_B$ × CPI_B) = ($1 \times 10^9 \times 1.8$) / ($1.5 \times 10^9 \times 0.86$) = 1.39

c. Execution Time$_A$ / Execution Time$_{new}$ = (Instruction count$_A$ × CPI_A × Clock cycle time) / (Instruction count$_{new}$ × CPI_{new} × Clock cycle time) = ($1 \times 10^9 \times 1.8 \times 1 \times 10^{-9}$) / ($9 \times 10^8 \times 1.7 \times 1 \times 10^{-9}$) = 1.176

Execution Time$_B$ / Execution Time$_{new}$ = (Instruction count$_B$ × CPI_B × Clock cycle time) / (Instruction count$_{new}$ × CPI_{new} × Clock cycle time) = ($1.5 \times 10^9 \times 0.867 \times 1 \times 10^{-9}$) / ($9 \times 10^8 \times 1.7 \times 1 \times 10^{-9}$) = 0.85

Question 6

Assume for arithmetic, load/store, and branch instructions, a processor has CPIs of 1, 12, and 5, respectively. Also assume that on a single processor a program requires the execution of 3×10^9 arithmetic instructions, 1.5×10^9 load/store instructions, and 2.5×10^8 branch instructions. Assume that each processor has a 2 GHz clock frequency.

Assume that, as the program is parallelized to run over multiple cores, the number of arithmetic and load/store instructions per processor is divided by $0.7 \times p$ (where p is the number of processors) but the number of branch instructions per processor remains the same.

a. Find the total execution time for this program on 1, 2, 4, and 8 processors, and show the relative speedup of the 2, 4, and 8 processors result as compared to the single processor result. **(10 marks)**

b. It is said that the GPUs (Graphics Processing Units) perform way better than the CPUs. Although there are many other fundamental considerations such as the independence of instructions that are necessary for a GPU to give better performance, let us ignore those assumptions and put this fact to test. Consider a certain GPU, say Nvidia T4 which consists of 2560 cores. Find the total execution time of the same program as in (a) on this GPU and compute the relative speedup that this GPU gives as compared to the single processor CPU. Comment on the performance of this GPU compared to the single processor CPU. **(5 marks)**

a.

P	Arithmetic	Load/Store	Branch	Clock Cycles	Execution Time	Speedup
1	3×10^9	1.5×10^9	0.25×10^9	$(3 \times 10^9 \times 1) + (1.5 \times 10^9 \times 12) + (0.25 \times 10^9 \times 5) = 22.25 \times 10^9$	$22.25 \times 10^9 / 2 \times 10^9 = 11.125$ sec	1
2	$3 \times 10^9 / (0.7 \times 2) = 2.143 \times 10^9$	$1.5 \times 10^9 / (0.7 \times 2) = 1.07 \times 10^9$	0.25×10^9	$(2.143 \times 10^9 \times 1) + (1.07 \times 10^9 \times 12) + (0.25 \times 10^9 \times 5) = 16.233 \times 10^9$	$16.233 \times 10^9 / 2 \times 10^9 = 8.1165$ sec	1.37
4	$3 \times 10^9 / (0.7 \times 4) = 1.07 \times 10^9$	$1.5 \times 10^9 / (0.7 \times 4) = 0.536 \times 10^9$	0.25×10^9	$(1.07 \times 10^9 \times 1) + (0.536 \times 10^9 \times 12) + (0.25 \times 10^9 \times 5) = 8.752 \times 10^9$	$8.752 \times 10^9 / 2 \times 10^9 = 4.376$ sec	2.54
8	$3 \times 10^9 / (0.7 \times 8) = 0.536 \times 10^9$	$1.5 \times 10^9 / (0.7 \times 8) = 0.269 \times 10^9$	0.25×10^9	$(0.536 \times 10^9 \times 1) + (0.269 \times 10^9 \times 12) + (0.25 \times 10^9 \times 5) = 4.889 \times 10^9$	$4.889 \times 10^9 / 2 \times 10^9 = 2.444$ sec	4.55

b.

P	Arithmetic	Load/Store	Branch	Clock Cycles	Execution Time	Speedup
2560	$3 \times 10^9 / (0.7 \times 2560) = 1.67 \times 10^6$	$1.5 \times 10^9 / (0.7 \times 2560) = 0.837 \times 10^6$	0.25×10^9	$(1.67 \times 10^6 \times 1) + (0.837 \times 10^6 \times 12) + (0.25 \times 10^9 \times 5) = 1.26 \times 10^9$	$1.26 \times 10^9 / 2 \times 10^9 = 0.6308$ sec	17.63

The GPU provides a very great speedup of about 17.63. Furthermore, this speed up would've been even higher if there were no branch instructions.

Question 7

a. Assume that *A* is an array of 100 doublewords and that the compiler has associated the variables g, h, and j with the registers x19, x20, and x21 respectively. Let's also assume that the starting address, or base address, of the array A is in x22. Compile these C statements into RISC-V assembly language:

```
g = h + A[8];
A[10] = g – j;
```

(5 marks)

b. For the following C statement, write the corresponding RISC-V assembly code. Assume that the C variables f, g, and h have already been placed in registers x5, x6, and x7, respectively. Use a minimal number of RISC-V assembly instructions.

```
f = g + (h – 5);
```

(5 marks)

a.

```
ld x5, 64(x22)      // temporary register x5 gets A[8]
add x19, x20, x5
sub x6, x19, x21    // temporary register x6 gets g – j
sd x6, 80(x22)
```

b.

```
addi x5, x7, -5
add x5, x5, x6
```

[Note: addi f,h,-5 (note, no subi) followed by add f,f,g]

Question 8

Translate the below assembly language instructions into RISC-V machine language instructions. Clearly indicate which instruction format is being used in each instruction. You can refer to RISC-V Green Card for the encoding of instructions.

---

addi x11, x10, 8
ld x31, 64(x30)
add x12, x11, x31
sd x12, 96(x30)

(10 marks)

addi x11, x10,8 (I-type instruction)

Immediate	rs1	funct3	rd	opcode
000000001000	01010	000	01011	0010011

ld x31, 64(x30) (I-type instruction)

Immediate	rs1	funct3	rd	opcode
000001000000	11110	011	11111	0000011

add x12, x11, x31 (R-type instruction)

funct7	rs2	rs1	funct3	rd	opcode
0000000	11111	01011	000	01100	0110011

sd x12, 96(x30) (S-type instruction)

Immediate [11:5]	rs2	rs1	funct3	Immediate [4:0]	opcode
0000011	01100	11110	011	00000	0100011