Computer Architecture

Ali Muhammad Asad
aa07190

Homework 1

Question 1 15 marks

Consider three different processors P1, P2, and P3 executing the same insstruction set. P1 has a 3 GHz clock rate and a CPI of 1.5. P2 has a 2.5 GHz clock rate and a CPI of 1.0. P3 has a 4.0 GHz clock rate and has a CPI of 2.2.

(a) Which processor has the highest performance expressed in instructions per second? (5 marks)

(b) If the processors each execute a program in 10 seconds, find the number of cycles and the number of instructions. (5 marks)

(c) We are trying to reduce the execution time by 30%, but this leads to an increase of 20% in the CPI. What clock rate should we have to get this time reduction? (5 marks)

Solution:

(a) $\text{IPS(instructions per second)} = \dfrac{\text{Instructions}}{\text{Seconds}} = \dfrac{\text{Instructions}}{\text{Clock Cycles}} \times \dfrac{\text{Clock Cycles}}{\text{Seconds}}$

$\text{CPI(cycles per instruction)} = \dfrac{\text{Clock Cycles}}{\text{Instruction}}$ and $\text{Clock Rate} = \dfrac{\text{Clock Cycles}}{\text{Seconds}}$

$\therefore \text{IPS} = \dfrac{1}{\text{CPI}} \times \text{Clock Rate} = \dfrac{\text{Clock Rate}}{\text{CPI}}$

$\text{P1} \implies \text{Clock Rate} = 3 \times 10^9, \text{CPI} = 1.5$
$\text{P2} \implies \text{Clock Rate} = 2.5 \times 10^9, \text{CPI} = 1.0$
$\text{P3} \implies \text{Clock Rate} = 4.0 \times 10^9, \text{CPI} = 2.2$

$\text{IPS}_{\text{P1}} = \dfrac{3 \times 10^9}{1.5} = 2 \times 10^9$
$\text{IPS}_{\text{P2}} = \dfrac{2.5 \times 10^9}{1.0} = 2.5 \times 10^9$
$\text{IPS}_{P3} = \dfrac{4.0 \times 10^9}{2.2} = 1.82 \times 10^9$

The more the instructions per second, the faster the CPU. Therefore **P2** has the best performance in terms of instructions per second: 2.5×10^9 IPS.

(b) CPU Time = 10 seconds

$\text{CPU Time} = \frac{\text{Clock Cycles}}{\text{Clock Rate}} = \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$

$\therefore$ Clock Cycles = CPU Time $\times$ Clock Rate

$\therefore$ Instruction Count $= \frac{\text{CPU Time}}{\text{CPI}}$

P1 $\implies$ Clock Cycles $= 10 \times 3.0{\times}10^9 = 3.0 \times 10^{10}$ cycles

Instruction Count $= \frac{3.0{\times}10^{10}}{1.5} = 2 \times 10^{10}$ instructions

P2 $\implies$ Clock Cycles $= 10 \times 2.5 \times 10^9 = 2.5 \times 10^{10}$ cycles

Instruction Count $= \frac{2.5^{10}}{1.0} = 2.5 \times 10^{10}$ instructions

P3 $\implies$ Clock Cycles $= 10 \times 4.0 \times 10^9 = 4.0 \times 10^{10}$ cycles

Instruction Count $= \frac{4.0{\times}10^{10}}{2.2} = 1.82 \times 10^{10}$ instructions

(c) Reduce execution time by 30% and increase in CPI of 20%.

Execution time = CPU Time = 10 seconds

$\therefore$ CPU Time = CPU Times $\times 0.7 = 10 \times 0.7 = 7$ seconds

$\therefore$ CPI = CPI $\times 1.2$

$$\text{CPU Time} = \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$$

$$\therefore \text{Clock Rate} = \frac{\text{Instruction Count} \times \text{CPI}}{\text{CPU Time}}$$

P1 $\implies$ Clock Rate $_{\text{P1}} = \frac{2 \times 10^{10} \times 1.5 \times 1.2}{7} = 5.14 \times 10^9$ Hz

P2 $\implies$ Clock Rate $_{\text{P2}} = \frac{2.5 \times 10^{10} \times 1.0 \times 1.2}{7} = 4.29 \times 10^9$ Hz

P3 $\implies$ Clock Rate $_{\text{P3}} = \frac{1.82 \times 10^{10} \times 2.2 \times 1.2}{7} = 6.86 \times 10^9$ Hz

Question 2 15 marks

Cosider 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). P1 with a clock rate of 2.5 GHz and CPIs of 1, 2, 3, and 3, and P2 with a clock rate of 3 GHz and CPIs of 2, 2, 2, and 2.

(a) Given a program with a dynamic instruction set count of 1.0E6 instructions divided into classes as follows: 10% class A, 20% class B, 50% class C, and 20% class D, which is faster, P1 or P2? (5 marks)

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

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

Solution:

Processor	Class A	Class B	Class C	Class D
P1	1	2	3	3
P2	2	2	2	2

P1 $\implies$ Clock Rate $= 2.5 \times 10^9$ Hz
P2 $\implies$ Clock Rate $= 3.0 \times 10^9$ Hz

(a) Instruction Count $= 1.0 \times 10^6$ instructions.

Average CPI = $\sum$(Instruction Count $\times$ CPI) / Total Instruction Count
The summation of instruction count will be the same as the total instruction count
$\therefore$ Average CPI = $\sum$(CPI)

P1 $\implies$ CPU Time $= \frac{1.0\times10^6\times(1(0.1)+2(0.2)+3(0.5)+3(0.2))}{2.5\times10^9} = 1.04 \times 10^{-3} = 1.04$ ms

P2 $\implies$ CPU Time $= \frac{1.06\times10^6\times(2(0.1)+2(0.2)+2(0.5)+2(0.2))}{3.0\times10^9} = 6.67 \times 10^{-4} = 0.667$ ms

The more the Execution Time, the slower the processor. P2 has a lesser Execution Time, therefore **P2** is faster.

(b) Global CPI = $\sum$(CPI) [as above]

P1 $\implies$ Global CPI $= 1(0.1) + 2(0.2) + 3(0.5) + 3(0.2) = 2.6$
P2 $\implies$ Global CPI $= 2(0.1) + 2(0.2) + 2(0.5) + 2(0.2) = 2$

(c) Clock Cycles = Instruction Count $\times$ Average CPI [average CPI = Global CPI]

P1 $\implies$ Clock Cycles $= 1.0 \times 10^6 \times 2.6 = 2.6 \times 10^6 = 2.6$ MHz
P2 $\implies$ Clock Cycles $= 1.0 \times 10^6 \times 2.0 = 2.0 \times 10^6 = 2.0$ MHz

Question 3 [15 marks]

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 1.0E9 and has an execution time of 1.1 s, while compiler B results in a dynamic instruction count of 1.2E9 and an execution time of 1.5 s.

(a) Find the average CPI for each program given that the processor has a clock cycle time of 1ns. (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 6.0E8 instructions and has an average CPI of 1.1. What is the speedup of using this new compiler versus using compiler A or B on the original processor? (5 marks)

Solution:

A $\implies$ Instruction count $= 1.0 \times 10^9$ instructions, Execution Time = CPU Time = 1.1 s
B $\implies$ Instruction count $= 1.2 \times 10^9$, Execution Time = CPU Time = 1.5 s

(a) Clock Cycle Time = 1ns $= 1 \times 10^{-9}$s

CPU Time = Instruction Count $\times$ CPI $\times$ Clock Cycle Time

$$\therefore \text{CPI} = \frac{\text{CPU Time}}{\text{Instruction Count} \times \text{Clock Cycle Time}}$$

$$\text{A} \implies \text{CPI} = \frac{1.1}{1.0 \times 10^9 \times 1 \times 10^{-9}} = 1.1$$

$$\text{B} \implies \text{CPI} = \frac{1.5}{1.2 \times 10^9 \times 1 \times 10^{-9}} = 1.25$$

(b) $$\text{CPU Time} = \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$$

$$\therefore \text{Clock Rate} = \frac{\text{Instruction Count} \times \text{CPI}}{\text{CPU Time}}$$

$$\implies \frac{\text{Clock Rate}_\text{A}}{\text{Clock Rate}_\text{B}} = \frac{(\text{Instruction Count} \times \text{CPI})_\text{A}}{(\text{Instruction Count} \times \text{CPI})_\text{B}}$$

CPU Time is same therfore is cancelled out and Clock Rate is made the subject

$$\therefore \frac{\text{Clock Rate}_\text{A}}{\text{Clock Rate}_\text{B}} = \frac{(1.0 \times 10^9 \times 1.1)}{1.2 \times 10^9 \times 1.25} = 0.733$$

Clock if A is 0.733 times faster than Clock of B, or Clock of B is 1.37 times faster than Clock of A.

(c) New Compiler C $\implies$ Instruction Count $= 6.0 \times 10^8$ instructions, Average CPI $= 1.1$

$$\frac{(\text{CPU Time})_{\text{C}}}{(\text{CPU Time})_{\text{A}}} = \frac{(\text{Instruction Count} \times \text{CPI})_{\text{A}}}{(\text{Instruction Count} \times \text{CPI})_{\text{C}}} = \frac{1.0 \times 10^9 \times 1.1}{6.0 \times 10^8 \times 1.1} = 1.67$$

$\therefore$ There will be a speedup of 1.67 times when using Compiler C versus Compiler A.

$$\frac{(\text{CPU Time})_{\text{C}}}{(\text{CPU Time})_{\text{B}}} = \frac{(\text{Instruction Count} \times \text{CPI})_{\text{B}}}{(\text{Instruction Count} \times \text{CPI})_{\text{C}}} = \frac{1.2 \times 10^9 \times 1.25}{6.0 \times 10^8 \times 1.1} = 2.27$$

$\therefore$ There will be a speedup of 2.27 times when using Compiler C versus Compiler B.

Question 4 15 marks

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 2.56E9 arithmetic instructions, 1.28E9 load/store instructions, and 256 million 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 × 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 relative to the single processor result. (5 marks)

(b) If the CPI of the arithmetic instructions was doubled, what would the impact be on the execution time of the program on 1, 2, 4, or 8 processors? (5 marks)

(c) To what should the CPI of load/store instructions be reduced in order for a single processor to match the performance of four processors using the original CPI values? (5 marks)

Solution:

CPIs: $\implies$ Arithmetic = 1
$\implies$ Load/Store = 12
$\implies$ Branch Instructions = 5

Instructions: $\implies$ Arithmetic = 2.56×10^9
$\implies$ Load/Store = 1.28×10^9
$\implies$ Branch Instructions = 256×10^6

Clock Frequency = 2 GHz = Clock Rate

(a) $$\text{CPU Time} = \frac{\sum(\text{Instruction Count}_i \times \text{CPI}_i)}{\text{Clock Rate}}$$

$$\text{Clock Cycles} = \sum(\text{Instruction Count}_i \times \text{CPI}_i)$$

$$\implies \text{Clock Cycles} = 1(2.56 \times 10^9) + 12(1.28 \times 10^9) + 5(256 \times 10^6) = 1.92 \times 10^{10}$$

$$\therefore \text{CPU Time} = \frac{1.92 \times 10^{10}}{2 \times 10^9} = 9.6\text{s}$$

Then execution time of **1 processor** is **9.6 seconds**.

For $p > 2$ where p represents the number of processors, arithmetic and load/store instructions are divided by $0.7 \times p$.

$$\implies \text{Clock Cycles}_p = \frac{2.56 \times 10^9}{0.7p} + \frac{12(1.28 \times 10^9)}{0.7p} + 5(256 \times 10^6) = \frac{2.56 \times 10^{10}}{p} + 1.28 \times 10^9$$

$$\therefore \text{CPU Time}_p = \frac{\frac{2.56 \times 10^{10}}{p} + 1.28 \times 10^9}{2 \times 10^9}$$

Then we can plug in the values of p for different execution times:

p = 2: CPU Time$_2$ = 7.04
Speedup = $\frac{9.6}{7.04} = 1.36$

p = 4: CPU Time$_4$ = 3.84
Speedup = $\frac{9.6}{3.84} = 2.5$

p = 8: CPU Time$_8$ = 2.24
Speedup = $\frac{9.6}{2.24} = 4.29$

(b) Arithmetic Instruction CPI is doubled.
$\implies$ Clock Cycles $=2(2.56 \times 10^9) + 12(1.28 \times 10^9) + 5(256 \times 10^6) = 2.176 \times 10^10$
$\implies$ CPU Time $= \frac{2.176 \times 10^10}{2 \times 10^9} = 10.88$s
Then if our CPI for arithmetic instructions was doubled, our execution time for 1 processor increasde by $\frac{10.88}{9.6} = 1.13$.
So for **1 processor**, time increased by a factor of 1.13.

Then for $p > 2$:
Clock Cycles$_p$ $= \frac{2(2.56 \times 10^9)}{0.7p} + \frac{12(1.28 \times 10^9)}{0.7p} + 5(256 \times 10^6) = \frac{2.93 \times 10^{10}}{p} + 1.28 \times 10^9$

$$\therefore \text{CPU Time}_p = \frac{\frac{2.93 \times 10^{10}}{p} + 1.28 \times 10^9}{2 \times 10^9}$$

Then we can plug in the values of p for different execution times:

p = 2: CPU Time$_2$ = 7.965
Increase Factor = $\frac{7.965}{7.04} = 1.13$

p = 4: CPU Time$_4$ = 4.3025
Increase Factor = $\frac{4.3025}{3.84} = 1.12$

p = 8: CPU Time$_8$ = 2.47125
Increase Factor = $\frac{2.47125}{2.24} = 1.10$

(c) CPU Time for 4 processors = 3.84s
Then execution time for a single processor should be equal to 3.84s.

$$\text{CPU Time} = \frac{\sum(\text{Instruction Count}_i \times \text{CPI}_i)}{\text{Clock Rate}}$$

$$\implies 3.84 = \frac{2.56 \times 10^9 + \text{CPI}_{\text{load/store}}(1.28 \times 10^9) + 5(256 \times 10^6)}{2 \times 10^9}$$

$$\implies \text{CPI}_{\text{load/store}} = 3$$

The CPI for Load/Store for a single processor should be reduced to 3 to match the performance of 4 processors with the original CPI values.

Question 5 20 marks

(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;

(10 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)

(10 marks)

Solution:

(a) The above C language in RISC-V can be written as:

```
ld x10, 64(x22)   #Value at A[8] is loaded in a temporary register x10, and 64 is the
                  # offset as each double word is of 8 bytes, so 8 x 8 = 64
add x19, x20, x10 #g = h + A[8]
sub x11, x19, x21 #g - j is stored in a register x11
sd x11, 80(x22)   #value of g - j is stored in A[10] (8 × 10 = 80)
```

(b) The above C statement in RISC-V can be written as:

```
addi x8, x7, -5   #h - 5 is stored in a temporary register x8
add x5, x6, x8    #f = g + (h - 5)
```

Question 6 20 marks

For the following RISC-V assembly code, assume that the variables f, g, h, i, and j are assigned to registers x5, x6, x7, x28, and x29, respectively. Assume that the base address of the arrays A and B are in registers x10 and x11, respectively.

```
---------------------------------
    addi x30, x10, 8
    addi x31, x10, 0
    sd x31, 0(x30)
    ld x30, 0(x30)
    add x5, x30, x31
---------------------------------
```

Answer the following questions:

(a) Write the equivalent C code for RISC-V assembly code. (5 marks)

(b) Translate the above assembly language instructions into RISC-V machine language instructions. (15 marks)

Solution:

(a) f = x5, g = x6, h = x7, i = x28, j = x29
A = x10, B = x11 [base addresses of the two arrays]
sd in the code shows us that the arrays are of doublewords

Line 1 $\implies$ x30 = &A[1]
Line 2 $\implies$ x31 = &A[0]
Line 3 $\implies$ &A[1] = &A[0]
Line 4 $\implies$ x30 = &A[0]
Line 5 $\implies$ f = &A[0] + &A[0] [f is a variable]

Then from the above code, we can conclude that we are adding A[0] and A[0] into a variable. Then the equivalent C code can be written as:

```
long long a* = &A[0];
long long b* = &A[1];
b* = a*;
f = b* + a*;
```

(b) Machine Code:

Line 1 $\implies$ 0x00850f13 $\implies$ 00000000100001010000111100010011
Line 2 $\implies$ 0x00050f93 $\implies$ 00000000000001010000111110010011
Line 3 $\implies$ 0x01ff0023 $\implies$ 00000001111111111001100000100011
Line 4 $\implies$ 0x000f0f03 $\implies$ 00000000000011110011111100000011
Line 5 $\implies$ 0x01ff02b3 $\implies$ 00000001111111110000001010110011

The above machine code in binary can be divided into few segments as follows(from right to left in machine code in binary):

- opcode(7-bits): Basic operation of the instruction
- rd(5-bits): The register destination operand
- funct3(3-bits): An additional opcode field
- rs1(5-bits): First register source operand
- rs2(5-bits): Second register source operand
- funct7(7-bits): An additional opcode field

Then the above machine code can be represneted as:

Line No.	**Instruction**	**format**	**funct7**	**rs2**	**rs1**	**funct3**	**rd**	**opcode**
Line 1	addi (add immediate)	I	0000000	01000	01010	000	11110	0010011
Line 2	addi (add immediate)	I	0000000	00000	01010	000	11111	0010011
Line 3	sd (store doubleword)	S	0000000	11111	11110	011	00000	0100011
Line 4	ld (load doubleword)	I	0000000	00000	11110	011	11110	0000011
Line 5	add (add)	R	0000000	11111	11110	000	00101	0110011