

CSSE3001/7000 2007 – Tutorial 1

4.19 [5] <§4.3> Suppose we enhance a computer to make all floating-point instructions run five times faster. Let's look at how speedup behaves when we incorporate the faster floating-point hardware. If the execution time of some benchmark before the floating-point enhancement is 10 seconds, what will the speedup be if half of the 10 seconds is spent executing floating-point instructions?

4.20 [10] <§4.3> We are looking for a benchmark to show off the new floating-point unit described in Exercise 4.19, and we want the overall benchmark to show a speedup of 3. One benchmark we are considering runs for 100 seconds with the old floating-point hardware. How much of the initial execution time would floating-point instructions have to account for to show an overall speedup of 3 on this benchmark?

4.22 [5] <§4.3> You are going to enhance a computer, and there are two possible improvements: either make multiply instructions run four times faster than before, or make memory access instructions run two times faster than before. You repeatedly run a program that takes 100 seconds to execute. Of this time, 20% is used for multiplication, 50% for memory access instructions, and 30% for other tasks. What will the speedup be if you improve only multiplication? What will the speedup be if you improve only memory access? What will the speedup be if both improvements are made?

4.23 [5] <§4.3> You are going to change the program described in Exercise 4.22 so that the percentages are not 20%, 50%, and 30% anymore. Assuming that none of the new percentages is 0, what sort of program would result in a tie (with regard to speedup) between the two individual improvements? Provide both a formula and some examples.

4.35 [10] <§§4.2, 4.3> In the embedded market, where cost is crucial, processors sometimes implement floating point only in software. We are interested in two implementations of a computer, one with and one without special floating-point hardware.

Consider a program, P, with the following mix of operations:

Floating-point multiply	10%
Floating-point add	15%
Floating-point divide	5%
Integer instructions	70%

Computer MFP (computer with floating point) has floating-point hardware and can therefore implement the floating-point operations directly. It requires the following number of clock cycles for each instruction class:

Floating-point multiply	6
Floating-point add	4
Floating-point divide	20
Integer instructions	2

Computer MNFP (computer with no floating point) has no floating-point hardware and so must emulate the floating-point operations using integer instructions. The

integer instructions all take 2 clock cycles. The number of integer instructions needed to implement each of the floating-point operations is as follows:

Floating-point multiply	30
Floating-point add	20
Floating-point divide	50

Both computers have a clock rate of 1000 MHz. Find the native MIPS ratings for both computers.

4.36 [10] <§4.2> If the computer MFP in Exercise 4.35 needs 300 million instructions for this program, how many integer instructions does the computer MNFP require for the same program?

4.37 [5] <§§4.2, 4.3> Assuming the instruction counts from Exercise 4.35, what is the execution time (in seconds) for the program in Exercise 4.36 run on MFP and MNFP?

4.38 [10] <§§4.2, 4.3> You are the lead designer of a new processor. The processor design and compiler are complete, and now you must decide whether to produce the current design as it stands or spend additional time to improve it. You discuss this problem with your hardware engineering team and arrive at the following options:

a) Leave the design as it stands. Call this base computer: Mbase. It has a clock rate of 500 MHz, and the following measurements have been made using a simulator:

Instruction Class	CPI	Frequency
A	2	40%
B	3	25%
C	3	25%
D	5	10%

b) Optimize the hardware. The hardware team claims that it can improve the processor design to give it a clock rate of 600 MHz. Call this computer Mopt. The following measurements were made using a simulator for Mopt:

Instruction Class	CPI	Frequency
A	2	40%
B	2	25%
C	3	25%
D	4	10%

What is the CPI for each computer?

4.39 [5] <§§4.2, 4.3> What are the native MIPS ratings for Mbase and Mopt in Exercise 4.38?

4.40 [10] <§§4.2, 4.3> How much faster is Mopt than Mbase in Exercise 4.38 (In term of CPI only)?

4.41 [5] <§§4.2, 4.3> The compiler team has heard about the discussion to enhance the computer discussed in Exercises 4.38 through 4.40. The compiler team proposes to improve the compiler for the computer to further enhance performance. Call this combination of the improved compiler and the base computer Mcomp. The instruction improvements from this enhanced compiler have been estimated as follows:

Instruction Class	Percentage of instructions executed versus base computer
A	90%
B	90%
C	85%
D	95%

For example, if the base computer executed 500 class A instructions, Mcomp would execute $0.9 \times 500 = 450$ class A instructions for the same program. What is the CPI for Mcomp?

4.42 [5] <§§4.2, 4.3> Using the data of Exercise 4.41, how much faster is Mcomp than Mbase (In term of CPI only)?

4.43 [10] <§§4.2, 4.3> The compiler group points out that it is possible to implement both the hardware improvements of Exercise 4.38 and the compiler enhancements described in Exercise 4.41. If both the hardware and compiler improvements are implemented, yielding computer Mboth, how much faster is Mboth than Mbase?

4.44 [10] <§§4.2, 4.3> You must decide whether to incorporate the hardware enhancements suggested in Exercise 4.38 or the compiler enhancements of Exercise 4.41 (or both) to the base computer described in Exercise 4.38. You estimate that the following time would be required to implement the optimisations:

Optimisation	Time to Implement	Computer name
Hardware	6 months	Mopt
Compiler	6 months	Mcomp
Both	8 months	Mboth

Assume that CPU performance improves by approximately 50% per year, or about 3.4% per month. Assuming that the base computer has performance equal to that of its competitors, which optimizations (if any) would you choose to implement?