Homework #1

Due 11:59pm Wednesday, 7 September, 2016
Multiple submissions accepted.
Late homeworks are dinged 25% per full or partial day.

Problems

1.5 [4] <§1.6> Consider three different processors P1, P2, and P3 executing the same instruction 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?
b. If the processors each execute a program in 10 seconds, find the number of cycles and the number of instructions.
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?

1.6 [20] <§1.6> Consider two different implementations of the same instruction set architecture. The instructions can be divided into four classes according to their CPI (class 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. Given a program with a dynamic instruction count of 1.0E6 instructions divided into classes as follows:

10% class A,
20% class B,
50% class C, and
20% class D,

which implementation is faster?

a. What is the global CPI for each implementation?
b. Find the clock cycles required in both cases.

1.7 [15] <§1.6> 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 1 ns.
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?

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?

1.8 The Pentium 4 Prescott processor, released in 2004, had a clock rate of 3.6GHz and voltage of 1.25V. Assume that, on average, it consumed 10 W of static power and 90 W of dynamic power.

The Core i5 Ivy Bridge, released in 2012, had a clock rate of 3.4 GHz and voltage of 0.9 V. Assume that, on average, it consumed 30 W of static power and 40 W of dynamic power.

1.8.1 [5] <§1.7> For each processor find the average capacitive loads.
1.8.2 [5] <§1.7> Find the percentage of the total dissipated power comprised by static power and the ratio of static power to dynamic power for each technology.
1.8.3 [15] <§1.7> If the total dissipated power is to be reduced by 10%, how much should the voltage be reduced to maintain the same leakage current? Note: power is defined as the product of voltage and current.

1.13 A pitfall cited in Section 1.10 is expecting to improve the overall performance of a computer by improving only one aspect of the computer. Consider a computer running a program that requires 250 s, with 70 s spent executing FP instructions, 85 s executed L/S instructions, and 40 s spent executing branch instructions.

1.13.1 [5] <§1.10> By how much is the total time reduced if the time for FP operations is reduced by 20%?
1.13.2 [5] <§1.10> By how much is the time for INT operations reduced if the total time is reduced by 20%?
1.13.3 [5] <§1.10> Can the total time can be reduced by 20% by reducing only the time for branch instructions?

1.14 Assume a program requires the execution of:
50×10^6 FP instructions,
110×10^6 INT instructions,
80×10^6 L/S instructions, and
16×10^6 branch instructions.
The CPI for each type of instruction is 1, 1, 4, and 2, respectively. Assume that the processor has a 2 GHz clock rate.

1.14.1 [10] <§1.10> By how much must we improve the CPI of FP instructions if we want the program to run two times faster?
Hint: 2 GHz is 2×10^9. T(cpu) = clock cycles / clock rate = clock cycles /(2 x

10^9)
Add up the # of instructions * the CPI for each of the four classes of instructions and that give you clock cycles. Then calculate your starting point time T(cpu).
To cut in half the # of clock cycles by improving the CPI of FP instructions, the CPI of the improved FP gets substituted in where the old CPI was and the clock cycles has to drop by 50%. Solve for the new clock cycles/2

1.14.2 [10] <§1.10> By how much must we improve the CPI of L/S instructions if we want the program to run two times faster?
Hint: Same exact idea here, just solve for clock cycles/2 by adjusting the value of CPI(l/s).

1.14.3 [5] <§1.10> By how much is the execution time of the program improved if the CPI of INT and FP instructions is reduced by 40% and the CPI of L/S and Branch is reduced by 30%?
Hint: T(cpu) = clock cycles / clock rate = clock cycles / $2*10^9$
Remember that the CPI for each type of instruction is 1, 1, 4, and 2, respectively, and reducing by 40% the CPI of int and fp means they're 60% of what they were before. So plug in the old and the new #s and calculate the old and new total CPU time in seconds for each scheme. Those two times are your answer.

1.15 [5] <§1.8> When a program is adapted to run on multiple processors in a multiprocessor system, the execution time on each processor is comprised of computing time and the overhead time required for locked critical sections and/or to send data from one processor to another.

Assume a program requires $\underline{t = 100\ s}$ of execution time on one processor.

When run on *p* processors, each processor requires $\underline{t/p\ sec}$, as well as an additional 4 s of overhead, irrespective of the number of processors.

Compute the per-processor execution time for 2, 4, 8, 16, 32, 64, and 128 processors. For each case, list the corresponding speedup relative to a single processor and the ratio between actual speedup versus ideal speedup (speedup if there was no overhead).

Hint: We will change the value of t/p for each of the 8 scenarios (the first is with just 1 processor), but always add 4 seconds for overhead. So that 4 seconds is going to become a larger and larger factor in the execution time. Built a table like the one below. The first calculated column is just t/p. The second one is t/p + 4 seconds (but only if there is more than one processor!). The speedup is the ratio of the time it took on 1 processor and the time it takes on one of the new count. Finally, the actual speedup divides that speedup ratio by the # of processors. After all, ideally, if you

had 2 processors, the program would need x/2 time. Four processors? x/4 time.

# of Processors	Execution Time per Processor	Time/proc w/ Overhead	Speedup ratio	Actual speedup / ideal speedup
1				
2				
4				
8				
16				
32				
64				
128				