Homework 1
Computer Organization and Architecture
CSCI 361, Fall 2014
Due: September 19, 2014 in class

Instructions: Complete the problems enumerated below. Answers must typed and neatly formatted. You will submit a printed hardcopy of your work at the beginning of class on the specified due date. Be sure to include your name and email address on your submission.

Kaleb J. Himes
kaleb@wolfssl.com
kaleb.himes@msu.montana.edu

Processor	Clock Rate	CPI
P1	3.0 GHz	1.5
P2	2.5 GHz	1.0
P3	4.0 GHz	2.2

Problem 1.5a [5 points]: Which processor has the highest performance expressed in instructions per second?

IC * CPI / GHz
Assume IC = 100 for calculation purposes.
P1: (100 * 1.5) / (3.0 * 10^9) = .00000005 = .5^-7
P2: (100 * 1.0) / (2.5 * 10^9) = .00000004 = .4^-7
P3: (100 * 2.2) / (4.0 * 10^9) = .000000055 = .55^-7

Conclussion: P3 has the highest performance expressed in instructions per second.

Problem 1.5b [5 points]: If the processors each execute a program in 10 seconds, find the number of cycles and the number of instructions.

Formula: CPU Clock Cycles = CPU execution time * Clock Rate

P1: 10s * (3.0 * 10^9) = 30 * 10^9 Clock Cycles
P2: 10s * (2.5 * 10^9) = 25 * 10^9 Clock Cycles
P3: 10s * (4.0 * 10^9) = 40 * 10^9 Clock Cycles

Formula: Instruction Count = (CPU Time * Clock Rate) / CPI = CPU Clock Cycles / CPI

P1: (30 * 10^9) / 1.5 = 20 * 10^9 Instructions/sec
P2: (25 * 10^9) / 1.0 = 25 * 10^9 Instructions/sec
P3: (40 * 10^9) / 2.2 = 18.181818... * 10^9 Instructions/sec

Problem 1.5c [10 points]: 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.

Given: .3 execution time and 1.2 CPI find clock rate.

NOTE: CPI_o = the old Cycles per instruction
Result: CPI_o + CPI_o * .20
CPU Time = 10 - (10 * .30)   = 10 – 3 = 7 sec
Clock Rate = IC * CPI / CPU Time

P1: CPI   = 1.5 + (1.5 * .20) = 1.5 + 0.3 = 1.8
Clock Rate = (20 * 10^9 * 1.8)/ 7 = (36 / 7) = 5.14 GHz

P2: CPI   = 1.0 + (1.0 * .20) = 1.2
Clock Rate = (25 * 10^9 * 1.2) / 7 = (30 / 7) = 4.29 GHz

P3: CPI   = 2.2 + (2.2 * .2) = 2.2 + 0.44 = 2.64
Clock Rate = (18.18 * 10^9 * 2.64) / 7 = (47.9952 / 7) = 6.8564571429 GHz = 6.86 GHz

---

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).

	Instr. Class CPI				Clock Rate
	A	B	C	D	
P1	1	2	3	3	2.5 GHz
P2	2	2	2	2	3.0 GHz

You are given a program with a dynamic instruction count of 1.0E6 instructions divided into class as follows:

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

Problem 1.6a [10 points] What is the global CPI for each implementation?

Formula: CPI = (#cycles_a * %overall_cycles) + (#cycles_b * %overall_cycles) + … + (#cycles_n * %overall_cycles)

P1: CPI = (1*.10) + (2*.20) + (3*.50) + (3*.20) = 2.6 Cycles per Instruction
P2: CPI = (2*.10) + (2*.20) + (2*.50) + (2*.20) = 2 Cycles per Instruction

Problem 1.6b [10 points] Find the clock cycles required in both cases.

Formula: Total Clock Cycles = 1.0E6 * CPI

P1: Total Clock Cycles = 1.0E6*2.6 = 2.6E6 = 2,600,000 Cycles
P2: Total Clock Cycles = 1.0E6*2.0 = 2.0E6 = 2,000,000 Cycles

---

The Pentium 4 Prescott processor, released in 2004, had a clock rate of 3.6 GHz and voltage of 1.25 V. 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, has 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.

Problem 1.8.1 [5 points] For each processor find the average capacitive loads.

Formula: Capacitive Load = Power / Voltage^2 * Clock Rate

Prescott 4: Capacitive Load = (10 + 90) / (1.25V^2 * 3.6GHz) = 100 / 5.625 = 17.7778
Core i5: Capacitive Load = (30 + 40) / (0.9^2 * 3.4) = 70 / 2.754 = 25.4176

Problem 1.8.2 [5 points] Find the percentage of the total dissipated power comprised by static power and the ratio of static power to dynamic power for each technology.

Formula: total power dissipated (tpd) = (static power / total power) * 100

Presott 4: 10W + 90W = 100W → 10W/100W * 100 = 10% Power Dissipated
Core i5: 30W + 40W = 70W → 30W/70W * 100 = 42.86% Power Dissipated

Problem 1.8.3 [10 points] 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.)

Formula: Static Power = Voltage * Leakage Current

Prescott 4: sp = 1.25 * lc → 10W / 1.25 = lc = <u>8 leakage current</u>
... To reduce the power dissipated by 10% we could therefore calculate: (10 - (.10*10) / 8 = V → 9/8 = V = 1.125 → 1.125 / 1.25 = 0.9 = 90% of the former voltage. Therefore to reduce total power dissipated by 10% we would also have to reduce the voltage by 10%.

Core i5: sp = 0.9 * lc → 30W / 0.9 = lc = <u>33.333 leakage current</u>

The results of the SPEC CPU2006 bzip2 benchmark running on an AMD Barcelona has an instruction count of 2.389E12, an execution time of 750 s, and a reference time of 9650 s.

Problem 1.11.1 [5 points] Find the CPI if the clock cycle time is 0.333 ns.

Formula: CPI = Clock Rate * Cpu Time / Instruction Count
Clock Rate = 1 / Cycle Time = 3GHz
CPI = (3E9 * (750 + 9650)) / 2.389E12 = 13.05986

Problem 1.11.2 [5 points] Find the SPEC ratio.

Formula: Reference Time / Execution Time

SPEC = 9650 s / 750 s = 12.86667 s

Problem 1.11.4 [5 points] Find the increase in CPU time if the number of instructions of the benchmark is increased by 10% and the CPI is increased by 5%.

Formula: CPU time = # of instructions * (CPI / Clock Rate)
CPU time increase = newCPU time / oldCPUtime

oldCPU time = 2.389E12 * (13.05986 / 3E9) = 10,400.0018
newCPU time = (1.1*2.389E12) * ((1.05*13.05986) / 3E9) = 12,012.0021
CPU time increase = 12012.0021 / 10400.0018 = 1.155 – 1 * 100 = 15.5% increase in CPU time

Problem 1.11.6 [10 points] Suppose that we are developing a new version of the AMD Barcelona processor with a 4 GHz clock rate. We have added some additional instructions to the instruction set in such a way that the number of instructions has been reduced by 15%. The execution time is reduced to 700 s and the new SPECratio is 13.7. Find the new CPI.

Given:

4E9 clock rate
0.85 * #of instructions = 0.85 * 2.389E12 = <u>2.03065E12</u>
700 s execution time
13.7 SPEC (= Reference Time / Execution Time (= 700 s)) → 13.7 * 700 = RT = <u>9590 s</u>
CPU time = #of instructions * (CPI / Clock Rate)
CPI = Clock Rate * CPU Time / #of instructions
Clock Rate = 1 / Cycle Time = 4GHz

CPI = (4E9 * (9590 + 700)) / 2.03065E12 = <u>20.2694</u>

NEXT PAGE

Another 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.

Problem 1.13.1 [5 points] By how much is the total time reduced if the time for FP operations is reduced by 20%?

Given:

0.80 * FP operations = 0.80 * 70 = 56

NOTE: 250s total NE to 70(fp) + 85(l/s) + 40(branch) so I'm adding in T_ghost for whatever the ghost operations are that cause us to get to 250s: 70+85+40 = 195. 250 – 195 = 55s dedicated to ghost operations.

Formula: Time_total (T_t) = T_fp + T_l/s + T_branch + T_ghost

T_t = 56 + 85 + 40 + 55 = 236s → 250 – 236 = 14 → 14 / 250 = 0.056 = <u>5.6% reduction in T_t</u>

Problem 1.13.2 [5 points] By how much is the time for INT operations reduced if the total time is reduced by 20%?

NOTE: I am assuming that the INT operations are what I called the Ghost Operations in Problem 1.13.1 so the missing 55s was dedicated to INT operations.

T_t = 250 * 0.8 = 200

T_fp + T_l/s + T_branch = 85 + 70 + 40 = 195 so all that's left is 5s for T_int.

5 / 55 = 9.09% → 100% - 9.09% = 90.91% reduction in T_int

Problem 1.13.3 [5 points] Can the total time can be reduced by 20% by reducing only the time for branch instructions?

T_t = 250 * 0.8 = 200

T_fp + T_int + T_l/s = 70 + 55 + 85 = 210 → <u>210 > 200</u>

Therefore even if we completely eliminated branch instructions (IE 0s) We would still have a run time greater than that of a 20% reduction. We would need to also reduce the run times of one of the other operations in order to get our program down to a 200 s runtime. So the answer is <u>NO, we could not</u>.
