Eric Tashji

Homework Assignment 1

Problem 1 (15 points)

Consider three different processors P1, P2, and P3 executing the same instruction set. P1 has a 3.2 GHz clock rate and a CPI of 1.5. P2 has a 2.0 GHz clock rate and a CPI of 1.0. P3 has a 4.0 GHz clock rate and has a CPI of 2.3.

Solution:

a. Calculate the performance of each processor expressed in instructions per second.

- **i.** **P1**: 3.2E9/1.5 = 2,133,333,333 instructions per second
- **ii.** **P2**: 2.0E9/1.0 = 2,000,000,000 instructions per second
- **iii.** **P3**: 4.0E9/2.3 = 1,739,130,434 instructions per second

Explanation: I calculated the number of instructions per second by dividing the clock rate in Hz by the number of cycles per instruction.

b. If each of these processors executes a program in 10 seconds, calculate the number of cycles and the number of instructions.

- **i.** If each processor executes a program in 10 seconds, the total number of instructions as the program executes is as follows:
  1. **P1**: 2,133,333,333 * 10 = 21,333,333,330 instructions
  2. **P2**: 2,000,000,000 * 10 = 20,000,000,000 instructions
  3. **P3**: 1,739,130,434 * 10 = 17,391,304,340 instructions

Explanation: I calculated the total number of instructions used with a program executed in 10 seconds by multiplying the number of instructions per second by 10.

- **ii.** If each processor executes a program in 10 seconds, the total number of cycles as the program executes is as follows:
  1. **P1**: 3.2E9 * 10 = 3.2E10 cycles
  2. **P2**: 2.0E9 * 10 = 2.0E10 cycles
  3. **P3**: 4.0E9 * 10 = 4.0E10 cycles

Explanation: I calculated the total number of cycles used with a program executed in 10 seconds by multiplying each processor's clock rate (cycles per second) by 10.

c. We are trying to reduce the execution time of P2 by 30%, but this leads to an increase of 20% in the CPI. What should the clock rate be to obtain this reduction in execution time?

- **i.** Target instructions per second: (2,000,000,000 * 10) / 7 = 2,857,142,857
- **ii.** Clock rate: (2,857,142,857 * (1.0 * 1.2)) = **3.43GHz**

Explanation: The clock rate to obtain this reduction in execution time would have to be approximately **3.43GHz**. I got this result first by finding the number of instructions per second that would result in a 30% decrease in execution time. I did this by multiplying the number of target instructions per second by 10 to get the number of instructions in a 10 second program. I then divided that by 7 so that we could get the target number of instructions per second to achieve the same amount of instructions in 7 seconds that would occur in 10 seconds. This way, we know it is a 30% decrease in execution time because the number of seconds is decreased by 30%, but the number of instructions per second changed so that the same number of instructions occur.

I then multiplied my result by the new CPI rate (as it has increased by 20%) to get the resulting clock rate. Finally, I divided the clock rate by 1E9 to get the clock rate in GHz, resulting in an approximate overall clock rate of **3.43GHz**.

Problem 2 (15 points)

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). P1 with a clock rate of 2.5 GHz and CPIs of 1, 2, 3, and 3 for the corresponding classes, 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: 30% class A, 20% class B, 30% class C, and 20% class D, which is faster: P1 or P2?

Solution:

a. What is the total CPI for each implementation?
   i. **P1**: (1 * .3) + (2 * .2) + (3 * .3) + (3 * .2) = 2.2 cycles per instruction
   ii. **P2**: (2 * .3) + (2 * .2) + (2 * .3) + (2 * .2) = 2.0 cycles per instruction

Explanation: I found the total CPI for each implementation by multiplying the CPI of each class for each implementation by the percentage of instructions divided into that class, as represented by a decimal value.

b. Calculate the clock cycles required in both cases.
   i. **P1:** 2.2 * 1E6 = 2.2E6 clock cycles
   ii. **P2:** 2.0 * 1E6 = 2E6 clock cycles

Explanation: I found the total number of clock cycles in both cases by multiplying the total cycles per instruction by the number of instructions for each implementation.

(Continue on next page)

c. Which is faster: P1 or P2?

   i. 2.2E6/2.5E9 = .00088 seconds

   ii. 2.0E6/3.0E9 = .00067 seconds

   iii. .00067 < .00088, so **P2** is faster than **P1**

Explanation: We know that **P2** is faster than **P1** because **P2** can complete the cycles required for completing 1,000,000 instructions in .00067 seconds, where as **P1** completes the cycles required for 1,000,000 instructions in .00088 seconds. This was found by dividing the number of cycles required for 1E6 instructions by the total cycles each iteration is capable of completing in a second. Because .00067 < .00088, **P2** is faster than **P1**.

Problem 3 (20 points)

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 1.2 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.

   i. 1.2 * 1E9 / 1E9 = 1.2 cycles per instruction

   ii. 1.5 * 1E9 / 1.2E9 = 1.25 cycles per instruction

Explanation: I found the CPI for each program by dividing the total number of cycles by the total number of instructions. I found the total number of cycles by multiplying the total execution time of each program by 1E9, as the clock cycle time for the processor is 1ns.

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 As code versus the clock of the processor running compiler Bs code?

   i. **Processor A:** (1.2 * 1.0E9) / 1E9 = 1.2 GHz

   ii. **Processor B:** (1.25 * 1.2E9) / 1E9 = 1.5 GHz

   iii. Processor B's clock is .1.25 times faster than Processor A's clock

Explanation: I first found the total number of cycles that occurred when executing the program by multiplying the number of cycles per instruction for each program by the number of total instructions. This gives us the total number of cycles for each program. From there, I determined that Processor B could complete 1.25 timest the number of cycles that processor A could complete in the same amount of time. Therefore, the clock of the processor running compiler B is 1.25 times faster than the clock of the processor running compiler A.

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?
   i. New Compiler: 6.0E8 * 1.1 * 1ns = 6.6E8ns
   ii. Compiler A: 1.0E9 * 1.2 * 1ns = 1.2E9ns
   iii. Compiler B: 1.2E9 * 1.25 * 1ns = 1.5E9ns
   iv. The new compiler is 1.82 times faster than compiler A and is 2.27 times faster than Compiler B.

Explanation: I first found the total number of nano seconds it takes for each compiler to compile the program by multiplying the number of instructions by the cycles per instruction for each by 1 nanosecond, which is the clock time of the original processor. I then compared the new compiler with compilers A and B to find how much faster the new compiler is than compilers A and B.

Problem 4 (15 points)

Assume a 15 cm diameter wafer costs $12, contains 84 dies, and has 0.022 defects/cm2 . Assume a 20 cm diameter wafer costs $15, contains 100 dies, and has 0.031 defects/cm2 .

a. Calculate the yield for both wafers.
   i. **Wafer 1 Yield:** 1 / (1 + (.022 * (15/84)/2))^2 = 1 / 1.0039 = **99.6%**
   ii. **Wafer 2 Yield:** 1 / (1 + (.031 * (20/100)/2))^2 = 1 / 1.0062 = **99.4%**

Explanation: I found the yield for each wafer by multiplying the defects per area by the die area (which is found by dividing the wafer area by the number of dies) divided by two. I then added 1 to it, squared that result, and then divided 1 by the result of the previous calculations. That gives us the yield.

b. Calculate the cost per die for both wafers.
   i. **Wafer 1 Cost per Die:** $12 / (84 * .996) = $12 / 83.664 = **$0.14**
   ii. **Wafer 2 Cost per Die:** $15 / (100 * .993) = $15 / 99.3 = **$0.15**

Explanation: I found the cost per die of each wafer by dividing the price of each wafer by the product of the number of dies and the yield of each wafer.

c. If the number of dies per wafer is increased by 10% and the defects per area unit increase by 15%, find the die area and yield.

i. **Die Area:**
   1. **Wafer 1:** 15 / (84 * 1.1) = **0.16cm**
   2. **Wafer 2:** 20 / (100 * 1.1) = **0.18cm**

Explanation: I found the die area by dividing the wafer area by the number of dies multiplied by 1.1 to account for the 10% die increase.

ii. **Yield:**
   1. **Wafer 1:** 1 / (1 + ((.022 * 1.15) * .16/2))^2 = 1 / 1.004 = **99.6%**
   2. **Wafer 2:** 1 / (1 + ((.031 * 1.15) * .18/2))^2 = 1 / 1.006 = **99.4%**

Explanation: I found the yield by multiplying the defects per area by 1.15 to indicate a 15% increase in defects per area. Then, I multiplied it by the die area divided by 2. I then added 1 to that number and squared the result. Finally, I divided 1 by the result to get the percentage.

Problem 5 (15 points)

A version of Pentium has a clock rate of 3.2 GHz and voltage of 1.2V. It consumes 12W of static power and 48 W of dynamic power.

a. What is the average capacitive load of this processor?
   i. **Capacitive Load:** ((2 * 48) / 1.2^2) / 3.2e^9 = **2.0e^-8**

   Explanation: I found the capacitive load by doubling the dynamic power, then dividing it by the voltage squared, and then dividing the result by the clock rate.

b. If the total dissipated power is to be reduced by 10% (from both dynamic and static power), how much should the voltage be reduced to maintain the same leakage current?
   i. Dynamic Current: 48 / 1.2 = 40 amps
   ii. Static current: 12 / 1.2 = 10 amps
   iii. Voltage (Dynamic): (48 * 0.9) / 40 = 43.2 / 40 = 1.08V
   iv. Voltage (Static): (12 * 0.9) / 10 = 10.8 / 10 = 1.08V
   v. Final voltage: 1.2V - 1.08V = **0.12V**
   vi. The voltage must be reduced by **0.12V** to maintain the same leakage current

   Explanation: I determine the voltage reduction by first finding the current of both the dynamic and static power. I did this by dividing the power by the voltage for both dynamic and static power. Then, I used the result of those to find the new voltage by first multiplying the power by 0.9 (to adjust for a 10% decrease) and then dividing that by the current for both static and dynamic power. I then subtracted the new voltage from the old voltage and concluded that the voltage would have to be decreased by **0.12V** to maintain the same leakage current?

Problem 6 (20 points)

Assume a program requires the execution of 50E06 floating point (FP) instructions, 100E06 integer (INT) instructions, 80E06 load/store (L/S) instructions, and 16E06 branch (B) 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.

a. How much faster can we execute the program if the CPI of INT and FP instructions is reduced by 40% and the CPI of L/S and Branch is reduced by 30%?

  i. Original Time: (50E06 * 1 + 100E06 * 1 + 80E06 * 4 + 16E06 * 2) / 2E09 = 0.251 seconds

  ii. New Time: (50E06 * 1 (.6) + 100E06 * 1 (.6) + 80E06 * 4 (.7) + 16E06 * 2 (.7)) / 2E09 = (30E06 + 60E06 + 224E06 + 22.4E06) / 2E09 = 0.1682 seconds

  iii. Difference: 0.251/.1682 = **1.5 times faster**

Explanation: The program can be executed **1.5** times faster. I reached this result by first finding the total cycles given the original CPI. I did this by multiplying the CPI by the number of instructions, and then dividing that by the total number of cycles the processor can complete within 1 second to get the total number of seconds the original time took. I then repeated this but multiplied the CPIs of the various instructions by percentages according to how the question describes, and used that to calculate the new time. I then compared the original and new times to determine how much faster the new program execution is compared to the older one.

b. By how much must we improve the CPI of FP instructions if we want the program to run 20% faster? (That is to take 80% of the original time.)

  i. Total Cycles: (50E06 * 1 + 100E06 * 1 + 80E06 * 4 + 16E06 * 2) = 502E06 cycles

  ii. Percentage of instructions: 50E06 / 502E06 = 9.9% of instructions

  iii. Because the FP instructions make up less than 20% of the total instructions, there is no CPI that would allow the program to execute 20% faster.

Explanation: What the question is asking is not possible because in order to make the program run 20% faster, we would need to adjust the CPI to reduce the overall cycles by 20%. However, the FP instructions make up at most 9.9% of the overall cycles. This means that even if we were to set the CPI of the FP instructions to 0, it would only make the program 9.9% faster, and could not reach 20%. Therefore, there is no CPI for the FP instructions that would make the program run 20% faster.

c. By how much must we improve the CPI of L/S instructions if we want the program to run 30% faster?

   i. Total Cycles: 50E06 * 1 + 100E06 * 1 + 80E06 * 4 + 16E06 * 2 = 502E06 cycles

   ii. New Cycles: 502E06 * .7 = 351,400,000 cycles

   iii. Cycles to Subtract: 502,000,000 - 351,400,000 = 150,600,000 cycles

   iv. New number of L/S Cycles 80E06 * 4 - 150,600,000 cycles = 169,400,000 cycles

   v. New CPI: 169,400,000 / 80,000,000 = 2.1175 cycles per instruction

   vi. Difference: 4 - 2.1175 = **1.8825 CPI**

Explanation: The CPI must be reduced by **1.8825 cycles per second**. We know this is true because in order to improve the overall time of the program, you would need to reduce the CPI of the L/S instructions so as to decrease the overall cycles by the corresponding percentage (30%). I found the correct CPI to accomplish this by first finding the total cycles of the original program. Then, I calculated the number of cycles equal to 70% of the total cycles (indicating a 30% decrease in overall cycles, and thus a 30% improvement in time). Then, I figured out the number of cycles to be reduced by subtracting the new number of cycles from the original number of cycles. Then, I found the new number of L/S cycles by subtracting the difference of the previous calculation from the number of L/S cycles multiplied by its corresponding CPI. Finally, I found the new CPI by dividing the result of the previous calculation by the number of L/S cycles. I then found the difference between the old CPI and the new CPI to determine how much the old CPI would need to be improved.