ECE4300 HW 1 Kenneth Bach

Sunday, September 22, 2024 7:08 PM

1.3 [2] <§1.3> Describe the steps that transform a program written in a high-level language such as C into a representation that is directly executed by a computer processor.

The high-level language is fed into a compiler to be translated into assembly language, which would become processed by the assembler afterwards. This will result in machine language (binary representation) that can be interpreted and executed by the processor.

1.4 [2] <§1.4> Assume a color display using 8 bits for each of the primary colors (red, green, blue) per pixel and a frame size of 1280 × 1024.

a. What is the minimum size in bytes of the frame buffer to store a frame?

b. How long would it take, at a minimum, for the frame to be sent over a 100 Mbit/s network?

8 x 8 x 8 = 24 bits = 3 bytes

a. 1280 x 1024 x 3 bytes = **3,932,160 bytes minimum**

b. 3,932,160 bytes x 8 $\frac{bits}{byte}$ = 31,457,280 bits

31,457,280 bits x 100,000,000 $\frac{bits}{sec}$ = **0.3145 sec**

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?

P1 = $\frac{Clock\ rate}{CPI} = \frac{3.0*10^9}{1.5} = 2 * 10^9$ ips

P2 = $\frac{2.5*10^9}{1} = 2.5 * 10^9$ ips

P3 = $\frac{4.0*10^9}{2.2} = 1.81 * 10^9$ ips

a. P2 has the highest performance with $2.5 * 10^9$ ips

b. P1: $3.0 * 10^9 * 10 = \mathbf{30 * 10^9}$ **cycles**

$\frac{30*10^9}{1.5} = \mathbf{20 * 10\text{^}9}$ **instructions**

P2: $2.5 * 10^9 * 10 = \mathbf{25 * 10^9}$ **cycles**

$\frac{25*10^9}{1} = \mathbf{25 * 10\text{^}9}$ **instructions**

P3: $4.0 * 10^9 * 10 = \mathbf{40 * 10^9}$ **cycles**

$\frac{40*10^9}{2.2} = \mathbf{18.18 * 10\text{^}9}$ **instructions**

c. Execution time: $\frac{\#\ of\ instructions * CPI}{Clock\ Rate}$

New clock rate: $\frac{Old\ CR * Old\ CPI * 1.20}{0.70} \Rightarrow \frac{1.20}{0.70} = \mathbf{1.714}$ *** Old Clock Rate**

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.

Global CPI = $(\%Class\ A) * CPI_A + (\%Class\ B) * CPI_B + (\%Class\ C) * CPI_C + (\%Class\ D) * CPI_D$

Clock Cycles = # of instructions x Global CPI
Instruction count: 1.0E6 = 1,000,000 instructions

a. P1: $(0.10 * 1) + (0.20 * 2) + (0.50 * 3) + (0.20 * 3) = \mathbf{2.60\ CPI}$
P2: $(0.10 * 2) + (0.20 * 2) + (0.50 * 2) + (0.20 * 2) = \mathbf{2.00\ CPI}$

b. P1: $\left(1 * 10^6\right) * 2.60 = \mathbf{2.60} * \mathbf{10^6}$ **clock cycles**
P2: $\left(1 * 10^6\right) * 2.00 = \mathbf{2.00} * \mathbf{10^6}$ **clock cycles**

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?

1.11 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.

1.11.1 [5] <§§1.6, 1.9> Find the CPI if the clock cycle time is 0.333 ns.

1.11.2 [5] <§1.9> Find the SPECratio.

1.11.3 [5] <§§1.6, 1.9> Find the increase in CPU time if the number of instructions of the benchmark is increased by 10% without affecting the CPI.

1.11.4 [5] <§§1.6, 1.9> 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%.

1.11.5 [5] <§§1.6, 1.9> Find the change in the SPECratio for this change.

1.11.6 [10] <§1.6> 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.

1.11.7 [10] <§1.6> This CPI value is larger than obtained in 1.11.1 as the clock rate was increased from 3 GHz to 4 GHz. Determine whether the increase in the CPI is similar to that of the clock rate. If they are dissimilar, why?

1.11.8 [5] <§1.6> By how much has the CPU time been reduced?

1.11.9 [10] <§1.6> For a second benchmark, libquantum, assume an execution time of 960 ns, CPI of 1.61, and clock rate of 3 GHz. If the execution time is reduced by an additional 10% without affecting to the CPI and with a clock rate of 4 GHz, determine the number of instructions.

1.11.10 [10] <§1.6> Determine the clock rate required to give a further 10% reduction in CPU time while maintaining the number of instructions and with the CPI unchanged.

1.11.11 [10] <§1.6> Determine the clock rate if the CPI is reduced by 15% and the CPU time by 20% while the number of instructions is unchanged.

1.11.9 [10] <§1.6> For a second benchmark, libquantum, assume an execution time of 960 ns, CPI of 1.61, and clock rate of 3 GHz. If the execution time is reduced by an additional 10% without affecting to the CPI and with a clock rate of 4 GHz, determine the number of instructions.

1.11.10 [10] <§1.6> Determine the clock rate required to give a further 10% reduction in CPU time while maintaining the number of instructions and with the CPI unchanged.

1.11.11 [10] <§1.6> Determine the clock rate if the CPI is reduced by 15% and the CPU time by 20% while the number of instructions is unchanged.

1.12 Section 1.10 cites as a pitfall the utilization of a subset of the performance equation as a performance metric. To illustrate this, consider the following two processors. P1 has a clock rate of 4 GHz, average CPI of 0.9, and requires the execution of 5.0E9 instructions. P2 has a clock rate of 3 GHz, an average CPI of 0.75, and requires the execution of 1.0E9 instructions.

1.12.1 [5] <§§1.6, 1.10> One usual fallacy is to consider the computer with the largest clock rate as having the largest performance. Check if this is true for P1 and P2.

1.12.2 [10] <§§1.6, 1.10> Another fallacy is to consider that the processor executing the largest number of instructions will need a larger CPU time. Considering that processor P1 is executing a sequence of 1.0E9 instructions and that the CPI of processors P1 and P2 do not change, determine the number of instructions that P2 can execute in the same time that P1 needs to execute 1.0E9 instructions.

1.12.3 [10] <§§1.6, 1.10> A common fallacy is to use MIPS (millions of instructions per second) to compare the performance of two different processors, and consider that the processor with the largest MIPS has the largest performance. Check if this is true for P1 and P2.

1.12.4 [10] <§1.10> Another common performance figure is MFLOPS (millions of floating-point operations per second), defined as

MFLOPS = No. FP operations / (execution time × 1E6)

but this figure has the same problems as MIPS. Assume that 40% of the instructions executed on both P1 and P2 are floating-point instructions. Find the MFLOPS figures for the programs.