Performance

Time is the measure of computer performance: the computer that performs the same amount of work in the least time is the fastest. Program *response time or execution time* is measured in seconds per program. Computers are often shared, however, and a processor may work on several programs simultaneously. In such cases, the system may try to optimize throughput rather than attempt to minimize the response time for one program. Hence, we often want to distinguish between the response time and the time that the processor is working on our behalf.

Response time

- The total time required for the computer to complete a task, including disk accesses, memory accesses, I/O activities, operating system overhead, CPU execution time, and so on.

Throughput

- Another measure of performance, it is the number of tasks completed per unit time.

Relative Performance

To maximize performance, we want to minimize response time or execution time for some task. Thus, we can relate performance and execution time for a computer X:

$$\text{Performance}_X = \frac{1}{\text{Execution Time}_X}$$

This means that for two computers X and Y, if the performance of X is greater than the performance of Y, we have

$$\text{Performance}_X > \text{Performance}_Y$$

$$\frac{1}{\text{Execution Time}_X} > \frac{1}{\text{Execution Time}_Y}$$

$$\text{Execution Time}_Y > \text{Execution Time}_X$$

That is, the execution time on Y is longer than that on X, if X is faster than Y. In discussing a computer design, we often want to relate the performance of two different computers quantitatively. We will use the phrase "X is *n* times faster than Y" or equivalently "X is *n* times as fast as Y". So we can write

$$\frac{\text{Performance}_X}{\text{Performance}_Y} = n$$

If X is *n* times faster than Y, then the execution time on Y is *n* times longer than it is on X:

$$\frac{\text{Performance}_X}{\text{Performance}_Y} = \frac{\text{Execution Time}_Y}{\text{Execution Time}_X} = n$$

Prob. 1 If computer A runs a program in 10 seconds and computer B runs the same program in 15 seconds, how much faster is A than B?

We know that A is *n* times faster than B if

$$\frac{\text{Performance}_A}{\text{Performance}_B} = \frac{\text{Execution Time}_B}{\text{Execution Time}_A} = n$$

Thus the performance ratio is

$$\frac{\text{Performance}_A}{\text{Performance}_B} = \frac{15}{10} = 1.5$$

and A is therefore 1.5 times faster than B.

Measuring Performance

CPU Execution Time or **CPU Time**

This is the actual time spend by the CPU for computing a task and does not include time spent waiting for I/O or running other programs. CPU time can be further divided into two parts:

User CPU Time

- The CPU time spent in a program itself.

System CPU Time

- The CPU time spent in the operating system performing tasks on behalf of the program.

The term ***system performance*** is used to refer ***Response Time*** and ***CPU performance*** to refer ***User CPU Time*** for unloaded system.

Clock Cycle

Almost all computers are constructed using a clock that determines when events take place in the hardware. These discrete time intervals are called **clock cycles** (or **ticks**, **clock ticks**, **clock periods**, **clocks**, **cycles**).

Clock Period

- Clock Period is the length of each clock cycle.

Designers refer to the length of a **clock period** as

- The time for a complete *clock cycle* (e.g., 250 picoseconds, or 250 ps).

- The *clock rate* (e.g., 4 gigahertz, or 4 GHz), which is the inverse of the clock period.

CPU Performance and Its Factors

A simple formula relates the clock cycles and clock cycle time to CPU time for a program:

$$\text{CPU execution time} = \text{CPU clock cycles } \times \text{Clock cycle time}$$

Alternatively, because clock rate and clock cycle time are inverses,

$$\text{CPU execution time} = \frac{\text{CPU clock cycles}}{\text{Clock Rate}}$$

This formula makes it clear that the hardware designer can improve performance by reducing the number of clock cycles required for a program or the length of the clock cycle.

Prob. 2	A program runs in 10 seconds on computer A, which has a 2 GHz clock. We are trying to help a computer designer build a computer, B, which will run this program in 6 seconds. The designer has determined that a substantial increase in the clock rate is possible, but this increase will affect the rest of the CPU design, causing computer B to require 1.2 times as many clock cycles as computer A for this program. What clock rate should we tell the designer to target?

Let's first find the number of clock cycles required for the program on A:

$$\text{CPU Time}_A = \frac{\text{CPU Clock Cycles}_A}{\mathit{Clock\ Rate}_A}$$

$$10\ \mathit{Sec.} = \frac{\text{CPU Clock Cycles}_A}{2 \times 10^9\ \mathit{Cycles/Sec.}}$$

$$\text{CPU Clock Cycles}_A = 10 \times 2 \times 10^9 = 20 \times 10^9\ \mathit{cycles}$$

CPU time for B can be found using this equation:

$$\text{CPU Time}_B = \frac{1.2 \times \text{CPU Clock Cycles}_A}{\mathit{Clock\ Rate}_B}$$

$$6\ \mathit{sec} = \frac{1.2 \times 20 \times 10^9 \mathit{cycles}}{\mathit{Clock\ Rate}_B}$$

$$\mathit{Clock\ Rate}_B = \frac{1.2 \times 20 \times 10^9 \mathit{cycles}}{6\ \mathit{sec}} = \frac{0.2 \times 20 \times 10^9 \mathit{cycles}}{\mathit{sec}}$$

$$\mathit{Clock\ Rate}_B = \frac{4 \times 10^9 \mathit{cycles}}{\mathit{sec}} = 4\ \mathit{GHz}$$

To run the program in 6 seconds, B must have twice the clock rate of A.

Instruction Performance

The performance equations till know, not include any reference to the number of instructions needed for the program. However, since the compiler clearly generated instructions to execute, and the computer had to execute the instructions to run the program, the execution time must depend on the number of instructions in a program. In other words response time or execution time is that it equals the number of instructions executed multiplied by the average time per instruction. Therefore, the number of clock cycles required for a program can be written as

$$CPU\ clock\ cycles = Instructions\ for\ a\ program \times Average\ clock\ cycles\ per\ instruction$$

- The term ***clock cycles per instruction***, which is the average number of clock cycles each instruction takes to execute, is often abbreviated as **CPI**.
- Since different instructions may take different amounts of time depending on what they do, CPI is an average of all the instructions executed in the program.
- CPI provides one way of comparing two different implementations of the same instruction set architecture, since the number of instructions executed for a program will, of course, be the same.

Prob. 3

If we have two implementations of the same instruction set architecture. Computer A has a clock cycle time of 250 ps and a CPI of 2.0 for some program, and computer B has a clock cycle time of 500 ps and a CPI of 1.2 for the same program. Which computer is faster for this program and by how much?

We know that each computer executes the same number of instructions for the program; let's call this number *I*. First, find the number of processor clock cycles for each computer:

$$CPU\ clock\ cycles_A = I \times 2.0$$

$$CPU\ clock\ cycles_B = I \times 1.2$$

Now we can compute the CPU time for each computer:

$$CPU\ time\ _A = CPU\ clock\ cycles_A \times Clock\ cycle\ time$$

$$CPU\ time\ _A = I \times 2 \times 250ps = 500 \times I\ ps$$

Likewise, for B:

$$CPU\ time\ _B = CPU\ clock\ cycles_B \times Clock\ cycle\ time$$

$$CPU\ time\ _B = I \times 1.2 \times 500\ ps = 600 \times I\ ps$$

Clearly, computer A is faster. The amount faster is given by the ratio of the execution times:

$$\frac{\text{CPU Performance}_A}{\text{CPU Performance}_B} = \frac{\text{Execution Time}_B}{\text{Execution Time}_A} = \frac{600 \times I\ ps}{500 \times I\ ps} = 1.2$$

We can conclude that computer A is 1.2 times as fast as computer B for this program.

The Classic CPU Performance Equation

We can write the basic performance equation in terms of **instruction count** (the number of instructions executed by the program), CPI, and clock cycle time and known as *The Classic CPU Performance Equation*:

$$CPU\ time = Instructions\ count \times CPI \times clock\ cycle\ time$$

or, since the clock rate is the inverse of clock cycle time:

$$CPU\ time = \frac{Instructions\ count \times CPI}{Clock\ rate}$$

These formulas are particularly useful because they separate the three key factors that affect performance. We can use these formulas to compare two different implementations or to evaluate a design alternative if we know its impact on these three parameters.

We can see how these factors are combined to yield execution time measured in seconds per program:

$$Time = \frac{Seconds}{Program} = \frac{Instructions}{Program} \times \frac{Clock\ Cycles}{Instruction} \times \frac{Seconds}{Clock\ cycle}$$

Prob. 3 A compiler designer is trying to decide between two code sequences for a particular computer. The hardware designers have supplied the following facts:

	CPI for each instruction class		
CPI	A	B	C
	1	2	3

For a particular high-level language statement, the compiler writer is considering two code sequences that require the following instruction counts:

	Instruction counts for each instruction class		
Code sequence	A	B	C
1	2	1	2
2	4	1	1

Which code sequence executes the most instructions? Which will be faster? What is the CPI for each sequence?

Sequence 1 executes 2 + 1 + 2 = 5 instructions. Sequence 2 executes 4 + 1 + 1 = 6 instructions. Therefore, sequence 1 executes fewer instructions.

We can use the equation for CPU clock cycles based on instruction count and CPI to find the total number of clock cycles for each sequence:

$$CPU\ clock\ cycle = \sum_{i=1}^{n} (CPI_i \times C_i)$$

This yields

$$CPU\ clock\ cycles_1 = (2 \times 1) + (1 \times 2) + (2 \times 3) = 2 + 2 + 6 = 10\ cycles$$

$$CPU\ clock\ cycles_2 = (4 \times 1) + (1 \times 2) + (1 \times 3) = 4 + 2 + 3 = 9\ cycles$$

So code sequence 2 is faster, even though it executes one extra instruction. Since code sequence 2 takes fewer overall clock cycles but has more instructions, it must have a lower CPI. The CPI values can be computed by

$$CPI = \frac{CPU\ clock\ cycles}{Instruction\ count}$$

$$CPI_1 = \frac{CPU\ clock\ cycles_1}{Instruction\ count_1} = \frac{10}{5} = 2.0$$

$$CPI_2 = \frac{CPU\ clock\ cycles_2}{Instruction\ count_2} = \frac{9}{6} = 1.5$$

Prob. 3

Suppose a program (or a program task) takes 1 billion instructions to execute on a processor running at 2 GHz. Suppose also that 50% of the instructions execute in 3 clock cycles, 30% execute in 4 clock cycles, and 20% execute in 5 clock cycles. What is the execution time for the program or task?

* Do by yourself