Outline of Lecture

- **1. The Role of Computer Performance**
- **2. Measuring Performance**

The Role of Computer Performance

- Designing high performance computers is one of the major goals of any computer architect.
- As a result, <u>assessing</u> the performance of computer hardware is at the heart of computer design
 and greatly affect the demand and market value of the computer.
- However, measuring performance of a computer system is not a straight-forward task:
 - → Which application to use to measure performance?
 - → What component of computer to measure (e.g., processor, I/O, cache)?
 - How do other parameters affect performance (e.g., OS, compiler).

→ How do you define performance (e.g., faster, or most completed jobs during a certain period of time) - <u>execution time</u> vs. <u>throughput</u>.

Example

Do the following changes to a computer system increase throughput, decrease response time, or both?

- 1) Replacing the processor in a computer with a faster version
- Adding additional processors to a system that uses multiple processors for separate tasks - for example handling an airline reservation system.

<u>Answer</u>

- 1) Both response time and throughput are improved
- 2) Only throughput increases.

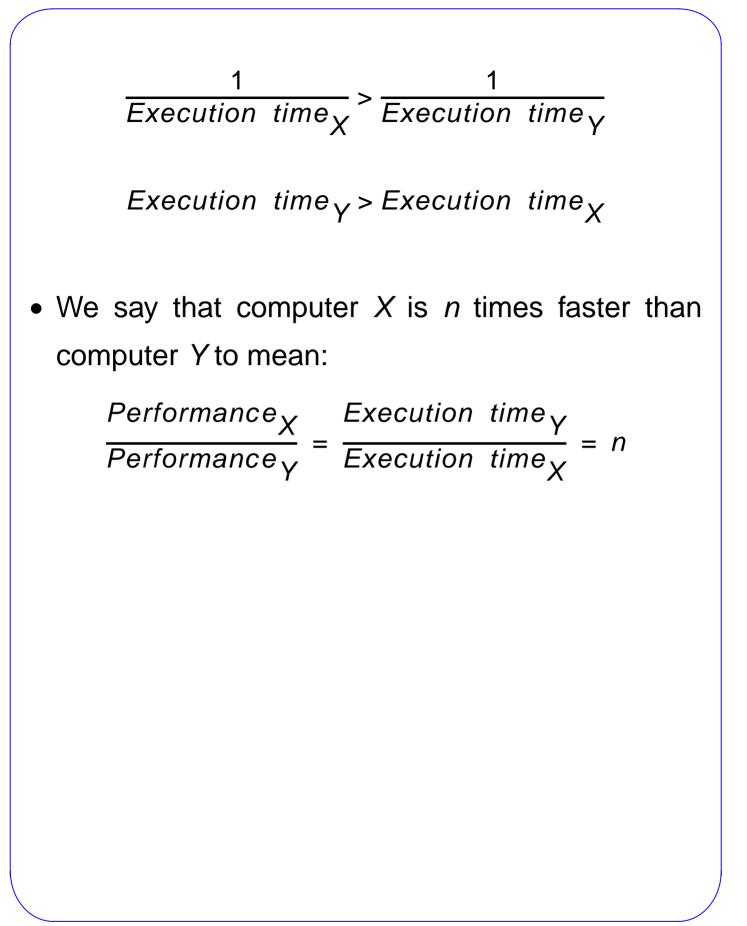
In this class, we will be primarily interested in <u>execution time</u> as a measure of performance.

 To maximize performance of an application, we need to minimize its execution time - the relationship between performance and execution time on a computer X is given by:

$$Performance_X = \frac{1}{Execution \ time_X}$$

• If the performance of computer X is better than the performance of computer Y, then:

Performance_X > Performance_Y



How to Measure Performance?

- In order to get an accurate measure of performance, we use <u>CPU time</u> instead of using response time.
- CPU time is the time the CPU spends computing a program and does not include time spent waiting for I/O or running other programs.
- CPU time can also be divided into <u>user CPU time</u> (program) and <u>system CPU time</u> (OS).
- In our performance measurements, we use user CPU time - because of its independence on the OS and other factors.

CPU Time Performance

- All computers are constructed using a <u>clock</u> to operate its circuits. It is typically measured by its period (e.g., 10 nsec) or by its <u>rate</u> (e.g., 100 MHz).
- The <u>CPU time</u> performance is probably the most accurate and fair measure of performance.
- The CPU time for a program is given by:

CPU time = CPU clock cyles for a program

×Clock cycle time

Alternatively the CPU time can be measured as:

 $CPU \ time = \frac{CPU \ clock \ cycles \ for \ a \ program}{Clock \ rate}$



A computer designer can improve the computer performance by either reducing the length of the clock cycle or the number of clock cycles required for a program.

In this class, we will understand how a computer designer achieves these goals, and what trade-offs the designer faces to achieve that.

Example

A given program runs in 10 sec on computer A, which has a 100 MHz clock. We are trying to help a computer designer build a computer B, that will run this program in 6 sec. 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?

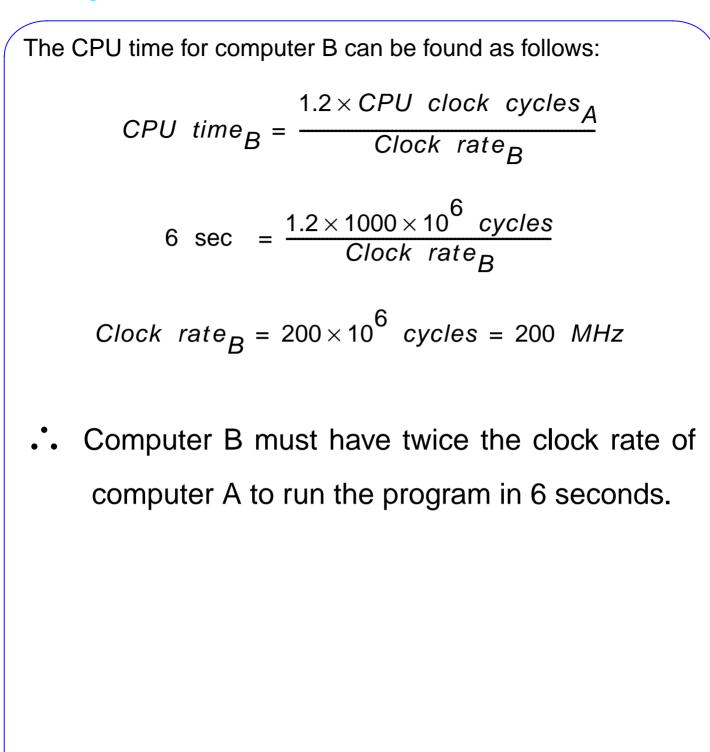
<u>Answer</u>

First, we find the number of clock cycles required for the program on computer A:

$$CPU \ time_{A} = \frac{CPU \ clock \ cycles_{A}}{Clock \ rate_{A}}$$

$$CPU \ clock \ cycles_{A} = 10 \ \sec \times 100 \times 10^{6} \ \frac{cycles}{\sec}$$

HKUST



• The CPU time for a program directly depends on the number of instructions in that program.

CPU clock cycles = Instructions for a program

× Average clock cycles per instruction

 The term <u>clock cycles per instruction</u> is often abbreviated as <u>CPI</u>.



The CPI of a program depends on the instruction set of the computer and on its compiler.

<u>Example</u>

Suppose we have 2 implementations of the same instruction set architecture. Computer A has a clock cycle time of 10 nsec and a CPI of 2.0 for some program, and computer B has a clock cycle time of 20 nsec and a CPI of 1.2 for the same program.

Which machine is faster for this program?

<u>Answer</u>

Assume the program requires / instructions to be executed:

CPU clock cycles_A = $I \times 2.0$ CPU clock cycles_B = $I \times 1.2$

CPU time_A = $I \times 2.0 \times 10$ nsec = $20 \times I$ nsec

CPU time_B = $I \times 1.2 \times 20$ nsec = $24 \times I$ nsec

• Computer A is faster than computer B.

• The CPU time for a program, which is our main measure of performance, can be written as:

CPU time = Instruction count × CPI × Clock cycle time

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

- The performance of the CPU is directly dependent on the clock speed, the number of cycles per instruction, and the number of instructions per program, known as <u>instruction count</u> (<u>IC</u>).
- . It is equally dependent on each one of them.

	IC	СРІ	Clock rate
Program	X		
Compiler	X	X	
Instr. Set	X	X	
Microarchitecture		X	X
Technology			X

 In order to take into account the frequency of instructions in a program, then the CPU performance can be expressed as:

$$CPU \ clock \ cycles = \sum_{i=1}^{n} CPI_i \times IC_i$$

where $\underline{IC_{i}}$ is the number of times instruction *i* is executed in a program and CPI_{i} represents the average number of clock cycles for instruction *i*.

$$CPU \ time = \left(\sum_{i=1}^{n} CPI_i \times IC_i\right) \times Clock \ cycle \ time$$

• The overall CPI can be expressed as:

$$CPI = \sum_{i=1}^{n} CPI_{i} \times \left(\frac{IC_{i}}{Instruction \ count}\right)$$