Factors affecting Performance

Programs are easier to write in high level languages
- examples are C, C++ and Java

Compilers must translate high level language programs
- target programs can be assembly language
- or machine code

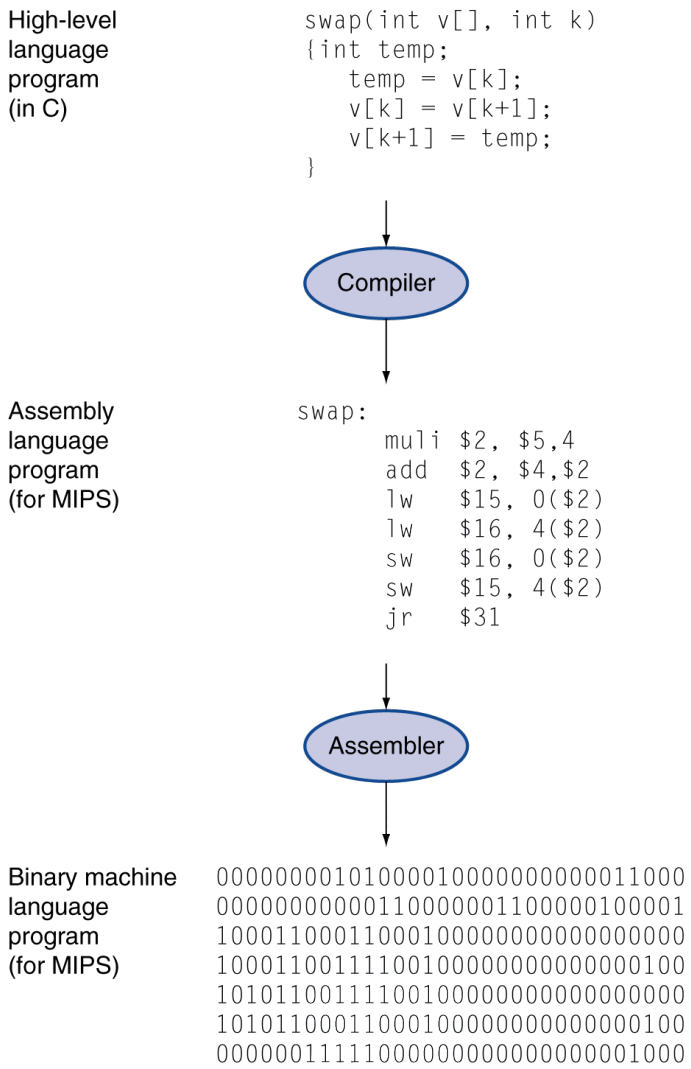
Assemblers generate machine code from assembly programs
- which are symbolic representations of machine code

Processors can only execute machine code

The *execution cycle* specifies how instructions are carried out

Factors affecting Performance

- High-level language
  - Level of abstraction closer to problem domain
  - Provides for productivity and portability
- Assembly language
  - Textual representation of instructions
- Hardware representation
  - Binary digits (bits)
  - Encoded instructions and data



Factors affecting Performance

The execution cycle consists of one or more steps

Each step requires a clock cycle

cycle time is the duration of a clock cycle (clock period)
- Instructions that take fewer cycles execute faster
- The shorter the cycle time the faster the execution

The clock rate is the number of cycles per second (Hz)
Clock rate = 1/cycle time

Simple RISC type instructions take fewer cycles

CISC type instructions tend to take more cycles

Factors affecting Performance

Programs that contain fewer instructions are faster

Programs that use mainly simple instructions are faster

Ways to improve performance include using:

- Efficient algorithms
- Efficient fast hardware
- Fast memory
- Parallel operations

Metrics are needed to access performance improvements

Measuring Performance

- Response time
  - How long it takes to do a task
- Throughput
  - Total work done per unit time
    - e.g., tasks/transactions/… per hour
- How are response time and throughput affected by
  - Replacing the processor with a faster version?
  - Adding more processors?
- We'll focus on response time for now…

Measuring Performance

- Define Performance = 1/Execution Time

$$\text{Performance}_X / \text{Performance}_Y = \text{Execution time}_Y / \text{Execution time}_X = n$$

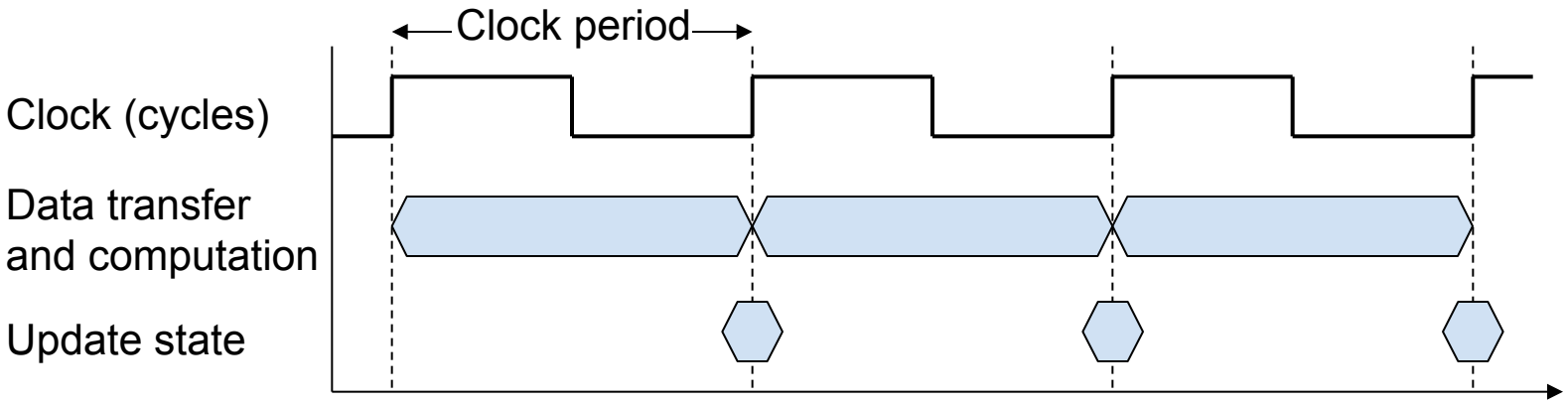
  - "X is n times as fast as than Y"
- Example: time taken to run a program
  - 10s on computer A, 15s on computer B
  - Execution Time$_B$ / Execution Time$_A$ = 15s / 10s = 1.5
  - So A is 1.5 times as fast as B

Measuring Performance

- Elapsed time
  - Total response time, including all aspects
    - Processing, I/O, OS overhead, idle time
  - Determines system performance
- CPU time
  - Time spent processing a given job
    - Discounts I/O time, other jobs' shares
  - includes user CPU time and system CPU time
  - Different programs are affected differently by CPU and system performance

Measuring Performance

- Digital hardware is driven by a constant-rate clock



  - Clock period: duration of a clock cycle
    - e.g., 250ps = 0.25ns = 250×10^{-12}s
  - Clock frequency (rate): cycles per second
    - e.g., 4.0GHz = 4000MHz = 4.0×10^{9}Hz