Response time: How long it takes to do a task

Throughput(吞吐量): Total work done per unit time

Relative Performance

- Define Performance = 1/Execution Time
- "X is n time faster than Y"

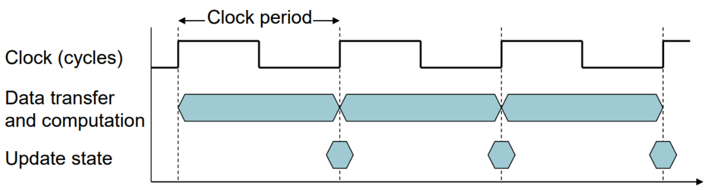
$$\begin{aligned} & \text{Performance}_X / \text{Performance}_Y \\ = & \text{Execution time}_Y / \text{Execution time}_X \\ = & n \end{aligned}$$

Execution Time

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

CPU Clocking

- Operation of digital hardware governed 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 = 4.0×10^9Hz

CPU Time

$$\text{CPU Time} = \text{No. of Clock Cycles} \times \text{Clock Period} = \frac{\text{No. of Clock Cycles}}{\text{Clock Rate}}$$

- Performance improved by
  - Reducing number of clock cycles (cycle count)
  - Increasing clock rate
  - Hardware designer must often trade off clock rate against cycle count

Instruction Count and CPI

$$\text{Clock Cycles} = \text{Instruction Count} \times \text{Cycles per Instruction}$$

$$\text{CPU Time} = \text{Instruction Count} \times \text{CPI} \times \text{Clock Cycle Time}$$

$$= \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$$

- Instruction Count for a program
  - Determined by program, ISA and compiler
- Average cycles per instruction
  - Determined by CPU hardware
  - If different instructions have different CPI
    - Average CPI affected by instruction mix

- If different instruction classes take different numbers of cycles

$$\text{Clock Cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{Instruction Count}_i)$$

- Weighted average CPI

$$\text{CPI} = \frac{\text{Clock Cycles}}{\text{Instruction Count}} = \sum_{i=1}^{n} \underbrace{\left(\text{CPI}_i \times \frac{\text{Instruction Count}_i}{\text{Instruction Count}} \right)}_{\text{Relative frequency}}$$

Summary

$$\text{CPU Time} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock cycle}}$$
$$= \text{IC} \times \text{CPI} \times \text{Tc}$$

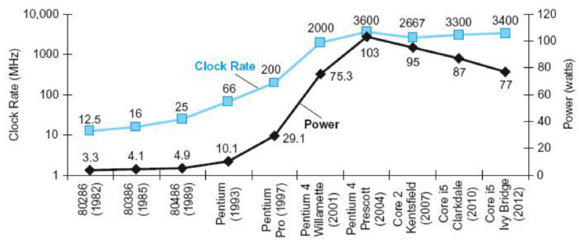
- Performance depends on
  - Algorithm: affects IC, possibly CPI
  - Programming language: affects IC, CPI
  - Compiler: affects IC, CPI
  - Instruction set architecture: affects IC, CPI, T_c

Energy Consumption

- Energy consumption = dynamic energy + static energy
  - Dynamic energy (energy spent when transistors switch from 0→1 1→0) is primary
  - Static energy is the energy cost when no transistor switches
- Energy for 0→1→0: $\textit{Energy} \propto \textit{Capacitive load} \times \textit{Voltage}^2$
- Energy for 0→1 or 1→0: $\textit{Energy} \propto 1/2 \times \textit{Capacitive load} \times \textit{Voltage}^2$
- Energy per second (power):

$$\textit{Power} \propto 1/2 \times \textit{Capacitive load} \times \textit{Voltage}^2 \times \textit{Frequency switched}$$

Power Trends



- In CMOS IC technology

$$\underbrace{\text{Power}}_{\times 23} \propto \frac{1}{2}\text{Capacitive load} \times \underbrace{\text{Voltage}^2}_{5\text{V} \to 1.5\text{V}} \times \underbrace{\text{Frequency}}_{\times 270}$$

Multiprocessors

- Multicore microprocessors
  - More than one processor per chip
- Requires explicitly parallel programming
  - Compare with instruction level parallelism
    - Hardware executes multiple instructions at once
    - Hidden from the programmer
  - Hard to do
    - Programming for performance
    - Load balancing
    - Optimizing communication and synchronization

Benchmark Suites

- Each vendor announces a SPEC rating for their system
  - a measure of execution time for a fixed collection of programs
  - is a function of a specific CPU, memory system, IO system, operating system, compiler
  - enables easy comparison of different systems
- The key is coming up with a collection of relevant programs

SPEC CPU Benchmark

- Programs used to measure performance
  - Supposedly typical of actual workload
- Standard Performance Evaluation Cooperative (SPEC)
  - Develops benchmarks for CPU, I/O, Web, …
- SPEC CPU2006
  - Elapsed time to execute a selection of programs
    - Negligible I/O, so focuses on CPU performance
  - Normalized relative to reference machine
  - Summarize as geometric mean of performance ratios
    - CINT2006 (integer) and CFP2006 (floating-point)

$$\sqrt[n]{\prod_{i=1}^{n} \text{Execution time ratio}_i}$$

SPEC Power Benchmark

- Power consumption of the server at different workload levels
  - Performance: ssj_ops (server side Java operations per second)
  - Power: Watts (Joules/sec)

$$\text{Overall ssj\_ops per Watt} = \left(\sum_{i=0}^{10} \text{ssj\_ops}_i\right) \Bigg/ \left(\sum_{i=0}^{10} \text{power}_i\right)$$

Amdahl's Law

- Architecture design is very bottleneck-driven – make the common case fast, do not waste resources on a component that has little impact on overall performance/power
- Amdahl's Law: performance improvements through an enhancement is limited by the fraction of time the enhancement comes into play

Processor : control , datapath