

Chapter 1

Performance

Tien-Fu Chen

Dept. of Computer Science and
Engineering

National Chiao Tung Univ.

Material source:
COD4 slides

Execution Time

■ Elapsed Time

- counts everything (*disk and memory accesses, I/O , etc.*)
- a useful number, but often not good for comparison purposes

■ CPU time

- doesn't count I/O or time spent running other programs
- can be broken up into system time, and user time

■ Our focus: user CPU time

- time spent executing the lines of code that are "in" our program

■ Performance

- **$\text{Performance}_x = 1 / \text{Execution time}_x$**

Performance

■ Performance Measures

– Elapsed time

– CPU time

`%time`

`90.7u 12.9s 2:39 65%`

– Cycle time

– Law of Performance

– CPI

– MIPS

– MFLOPS

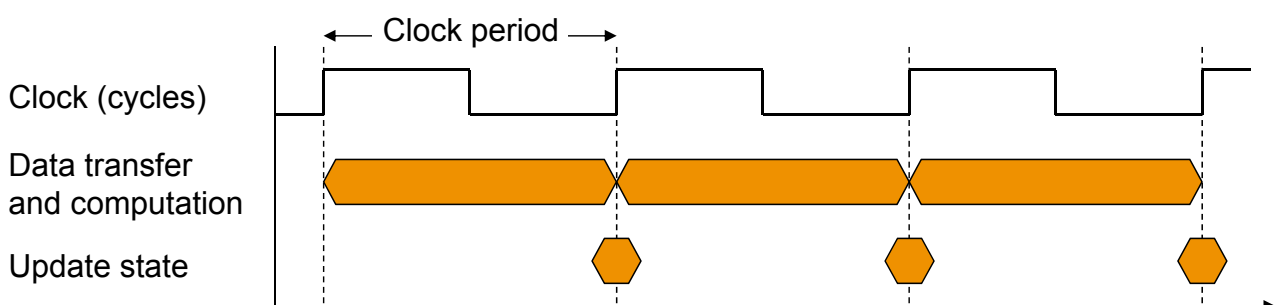
– SPECmarks

■ Benchmarks

■ Averaging

CPU Clocking

■ Operation of digital hardware governed by a constant-rate clock



■ Clock frequency (rate): cycles per second

– e.g., $4.0\text{GHz} = 4000\text{MHz} = 4.0 \times 10^9\text{Hz}$

■ Clock period of 4GHz: duration of a clock cycle

– e.g., $250\text{ps} = 0.25\text{ns} = 250 \times 10^{-12}\text{s}$

Metric prefix

Factor Name Symbol

10^{24}	yotta	Y
10^{21}	zetta	Z
10^{18}	exa	E
10^{15}	peta	P
10^{12}	tera	T
10^9	giga	G
10^6	mega	M
10^3	kilo	K
10^1	deka	da

10^{-1}	deci	d
10^{-2}	centi	c
10^{-3}	milli	m
10^{-6}	micro	μ
10^{-9}	nano	n
10^{-12}	pico	p
10^{-15}	femto	f
10^{-18}	atto	a
10^{-21}	zepto	z
10^{-24}	yocto	y

Law of Performance

Factoring CPU time

$$cpu\ time = \left[\frac{instrn}{program} \right] \times \left[\frac{cycles}{instrn} \right] \times \left[\frac{time}{cycle} \right]$$

- Instruction per program
 - ISA and Compiler
- Cycle per Instruction (CPI)
 - ISA and architecture
- Time per cycle (cycle time)
 - cycle time = time between ticks = seconds per cycle
 - clock rate (frequency) = cycles per second (1 Hz. = 1 cycle/sec)
 - organization, hardware
 - technology, layout

CPU Time Example

- Computer A: 2GHz clock, 10s CPU time
- Designing Computer B
 - Aim for 6s CPU time
 - Can do faster clock, but causes $1.2 \times$ clock cycles
- How fast must Computer B clock be?

$$\text{Clock Rate}_B = \frac{\text{Clock Cycles}_B}{\text{CPU Time}_B} = \frac{1.2 \times \text{Clock Cycles}_A}{6s}$$

$$\begin{aligned} \text{Clock Cycles}_A &= \text{CPU Time}_A \times \text{Clock Rate}_A \\ &= 10s \times 2\text{GHz} = 20 \times 10^9 \end{aligned}$$

$$\text{Clock Rate}_B = \frac{1.2 \times 20 \times 10^9}{6s} = \frac{24 \times 10^9}{6s} = 4\text{GHz}$$

CPI Example

- Computer A: Cycle Time = 250ps, CPI = 2.0
- Computer B: Cycle Time = 500ps, CPI = 1.2
- Same ISA
- Which is faster, and by how much?

$$\begin{aligned} \text{CPU Time}_A &= \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A \\ &= 1 \times 2.0 \times 250\text{ps} = 1 \times 500\text{ps} \end{aligned}$$

A is faster...

$$\begin{aligned} \text{CPU Time}_B &= \text{Instruction Count} \times \text{CPI}_B \times \text{Cycle Time}_B \\ &= 1 \times 1.2 \times 500\text{ps} = 1 \times 600\text{ps} \end{aligned}$$

$$\frac{\text{CPU Time}_B}{\text{CPU Time}_A} = \frac{1 \times 600\text{ps}}{1 \times 500\text{ps}} = 1.2$$

...by this much

CPI Example

- Alternative compiled code sequences using instructions in classes A, B, C

Class	A	B	C
CPI for class	1	2	3
IC in sequence 1	2	1	2
IC in sequence 2	4	1	1

- Sequence 1: IC = 5

- Clock Cycles
= $2 \times 1 + 1 \times 2 + 2 \times 3$
= 10
- Avg. CPI = $10/5 = 2.0$

- Sequence 2: IC = 6

- Clock Cycles
= $4 \times 1 + 1 \times 2 + 1 \times 3$
= 9
- Avg. CPI = $9/6 = 1.5$

Performance Summary

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

- 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

MIPS - Million instructions per second

$$\begin{aligned} \text{MIPS} &= \left[\frac{\text{instrn}}{\text{program}} \right] \times \left[\frac{\text{program}}{\text{time}} \right] \times 10^{-6} \\ &= \frac{\text{inst count}}{\text{exec time} \times 10^6} = \frac{\text{clock rate}}{\text{CPI} \times 10^6} \end{aligned}$$

■ Problems:

- Depend on instruction set
- Depend on different programs
- What if execute only more faster instructions? So it can vary inversely with performance

■ = Meaningless Indicator of Performance??

MIPS example

- Two different compilers are being tested for a 4 GHz. machine with three different classes of instructions: Class A, Class B, and Class C, which require one, two, and three cycles (respectively). Both compilers are used to produce code for a large piece of software.

1st compiler: 5M Class A instructions, 1M Class B instructions, and 1M Class C instructions.

2nd compiler: 10 M Class A instructions, 1M Class B instructions, and 1 M Class C instructions.

- Which sequence will be faster according to MIPS?
- Which sequence will be faster according to execution time?

Benchmarks

- Performance best determined by running a real application
 - Use programs typical of expected workload
 - Or, typical of expected class of applications
e.g., compilers/editors, scientific applications, graphics, etc.
- Small benchmarks
 - nice for architects and designers
 - easy to standardize
 - can be abused
- SPEC (System Performance Evaluation Cooperative)
 - companies have agreed on a set of real program and inputs
 - valuable indicator of performance (and compiler technology)
 - can still be abused

Summarizing Results: Average

- Arithmetic Mean
 - means for times, CPI

$$\overline{time} = \frac{1}{n} \sum_{i=1}^n time_i$$

- Harmonic mean
 - means for rates, MIPS, MFLOPS

$$\overline{rate} = \frac{n}{\sum_{i=1}^n \frac{1}{rate_i}}$$

- Geometric mean
 - normalize numbers

$$\overline{ratio} = \left[\prod_{i=1}^n ratio_i \right]^{1/n}$$

SPECmark

■ SPEC benchmarks

System Performance Evaluation Cooperative

SPEC89 4 integer 6 FP benchmarks

SPEC92 6 integer 14 FP

ex. Espresso, Eqntott, Gcc, Spice, Ddodc,....

SPEC95 8 integer, 10 FP programs

■ SPECmark

aggregate speedup of a vendor's platform relative to a SPARC 10 which has a SPEC mark of 1.0

$$\overline{\text{SPECmark}} = \left[\prod_{i=1}^{10} (\text{individual speedup})_i \right]^{1/10}$$

SPEC CPU2000

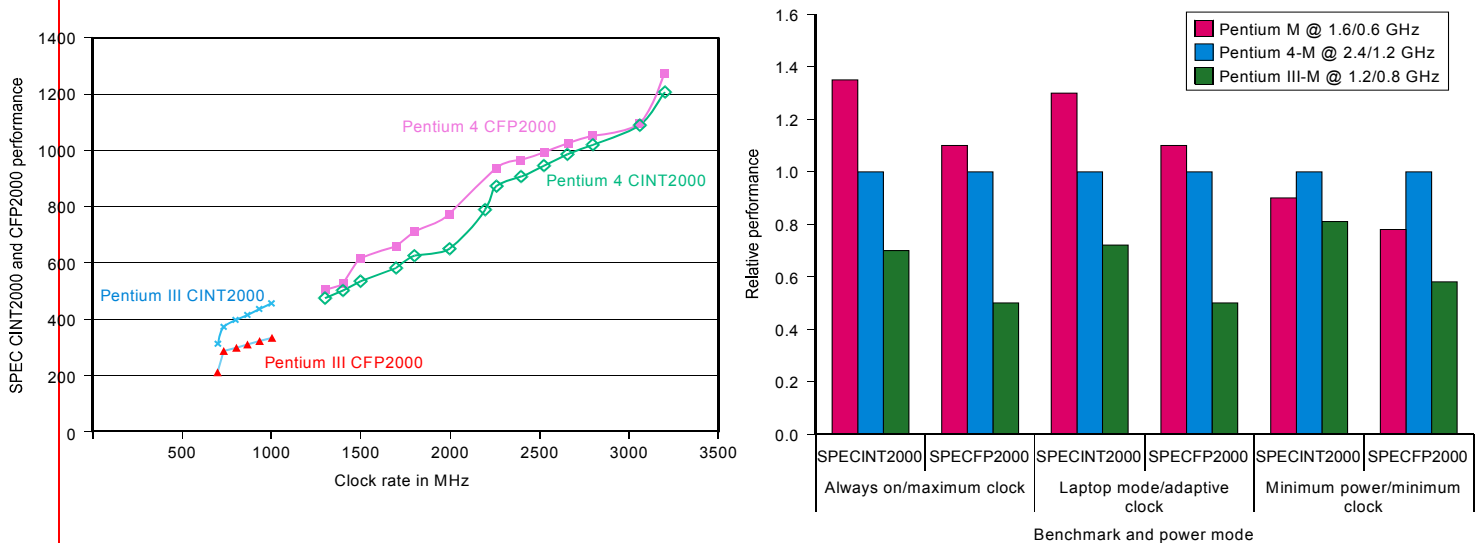
Integer benchmarks		FP benchmarks	
Name	Description	Name	Type
gzip	Compression	wupwise	Quantum chromodynamics
vpr	FPGA circuit placement and routing	swim	Shallow water model
gcc	The Gnu C compiler	mgrid	Multigrid solver in 3-D potential field
mcf	Combinatorial optimization	applu	Parabolic/elliptic partial differential equation
crafty	Chess program	mesa	Three-dimensional graphics library
parser	Word processing program	galgel	Computational fluid dynamics
eon	Computer visualization	art	Image recognition using neural networks
perlbnk	perl application	equake	Seismic wave propagation simulation
gap	Group theory, interpreter	facerec	Image recognition of faces
vortex	Object-oriented database	ammp	Computational chemistry
bzip2	Compression	lucas	Primality testing
twolf	Place and rote simulator	fma3d	Crash simulation using finite-element method
		sixtrack	High-energy nuclear physics accelerator design
		apsl	Meteorology: pollutant distribution

FIGURE 4.5 The SPEC CPU2000 benchmarks. The 12 integer benchmarks in the left half of the table are written in C and C++, while the floating-point benchmarks in the right half are written in Fortran (77 or 90) and C. For more information on SPEC and on the SPEC benchmarks, see www.spec.org. The SPEC CPU benchmarks use wall clock time as the metric, but because there is little I/O, they measure CPU performance.

SPEC 2000

Does doubling the clock rate double the performance?

Can a machine with a slower clock rate have better performance?



Amdahl's Law

Execution Time After Improvement =

Execution Time Unaffected +(Execution Time Affected / Amount of Improvement)

■ Example:

"Suppose a program runs in 100 seconds on a machine, with multiply responsible for 80 seconds of this time. How much do we have to improve the speed of multiplication if we want the program to run 4 times faster?"

How about making it 5 times faster?

■ Principle: Make the common case fast

Speedup

■ Speedup

$$\text{Speedup} = \frac{\text{old time}}{\text{new time}} = \frac{\text{new rate}}{\text{old rate}}$$

■ Amdahl's Law

Improvement to be gained from using faster mode is limited by the fraction of the time the faster mode can be used

Consider an enhancement x
speedups fraction f_x of a task by S_x

$$\text{Speedup}_{\text{overall}} = \frac{1}{(1 - f_x) + (f_x / S_x)}$$

■ Examples

$$f_x = 90\%, S_x = 5 \rightarrow$$

$$\text{Speedup}_{\text{overall}} = \frac{1}{(1 - .90) + (.90 / 5)} = 3.6$$

Amdahl's Law Corollary

■ Increasing speedup

$$f_x = 50\%,$$

$$S_x = 1.10 \rightarrow \text{Speedup}_{\text{overall}} = \frac{1}{(1 - .5) + (.5 / 1.1)} = 1.047$$

$$S_x = 10 \rightarrow \text{Speedup}_{\text{overall}} = \frac{1}{(1 - .5) + (.5 / 10)} = 1.818$$

$$S_x = \infty \rightarrow \text{Speedup}_{\text{overall}} = \frac{1}{(1 - .5) + (.5 / \infty)} = 2.0$$

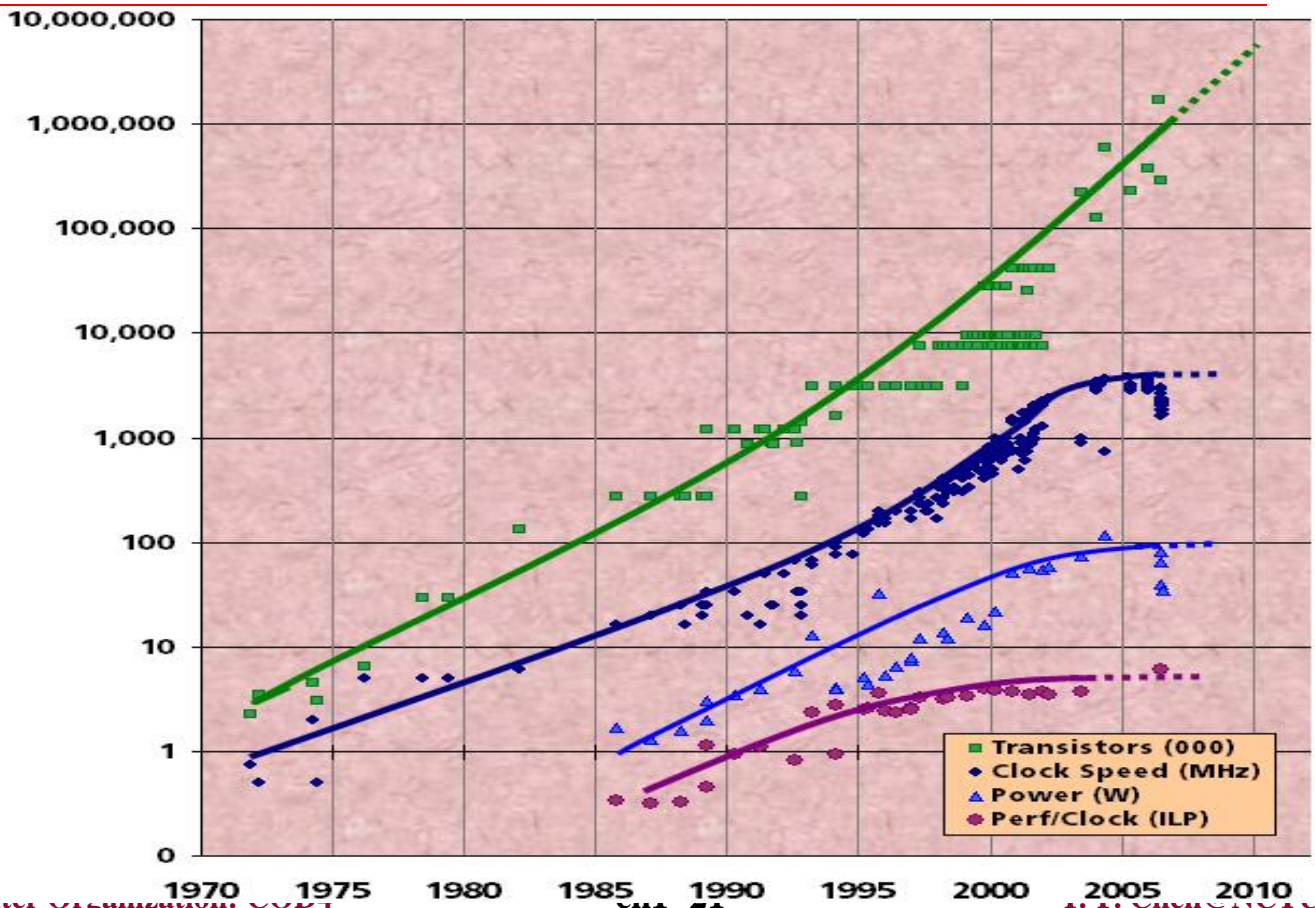
■ Bound

$$\text{Speedup}_{\text{overall}} < \frac{1}{1 - f_x}$$

■ Examples

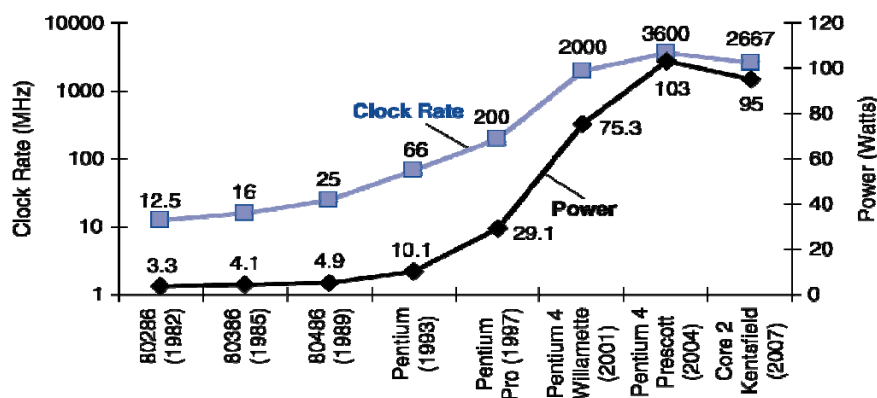
f_x	1%	5%	10%	20%	50%
$1/(1 - f_x)$	1.01	1.05	1.11	1.25	2.00

Power is the first class design issue



Computer Organization: COD4 ch1- 22 T.-F. Chen@NCTU CSIE

Power Trends



- In CMOS IC technology

$$\text{Power} = \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency}$$

×30

5V → 1V

×1000

Reducing Power

- Suppose a new CPU has
 - 85% of capacitive load of old CPU
 - 15% voltage and 15% frequency reduction

$$\frac{P_{\text{new}}}{P_{\text{old}}} = \frac{C_{\text{old}} \times 0.85 \times (V_{\text{old}} \times 0.85)^2 \times F_{\text{old}} \times 0.85}{C_{\text{old}} \times V_{\text{old}}^2 \times F_{\text{old}}} = 0.85^4 = 0.52$$

- The power wall
 - We can't reduce voltage further
 - We can't remove more heat
- How else can we improve performance?