Performance Verstehen

[Adapted from Mary Jane Irwin for
Computer Organization and Design,
Patterson & Hennessy, © 2005, UCB]

Performance Metrik

- Kaufentscheide
  - Bei einer Auswahl an Maschinen, welche hat
    - Beste Performance ?
    - Geringsten Kosten ?
    - Bestes Kosten/Performance Verhältnis?
- Designentscheide
  - Konfrontiert mit Designoptionen, welche hat
    - Beste Performanceverbesserung ?
    - Geringste Kosten ?
    - Bestes Kosten/Performance Verhältnis?
- Beide benötigen
  - Vergleichsgrundlage
  - Evaluationsmetrik
- Unser Ziel ist es zu verstehen welche Faktoren der Architektur zur Gesamtsystemleistung beitragen und den jeweiligen relativen Anteil (und Kosten) dieser Faktoren

Definition (Geschwindigkeit) Performance

- Reduzieren möchte man normalerweise:
  - Antwortzeit (Response time) – die Zeit zwischen Start und Beendigung eines Programms
    - Wichtig für die einzelnen Benutzer
  - Daher, zur Maximierung der Performance, muss die Antwortzeit minimiert werden

$$\text{Performance}_X = 1 / \text{Ausführungszeit}_X$$

Wenn X n mal schneller als Y ist, dann

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

  - Durchsatz (Throughput) – Die totale Menge an Arbeit welcher in einer gegeben Zeit erledigt wird
    - Wichtig für Data Center Manager
  - Kürzere Antwortzeit verbessert fast immer den Durchsatz

Performance Faktoren

- Müssen unterscheiden zwischen total verstrichener Zeit und Zeit welche für ein Programm verwendet wurde
- CPU Ausführungszeit (**CPU Zeit**) – Zeit welche die CPU an **einem** Programm arbeitet
  - Beinhaltet keine I/O Wartezeiten oder Arbeit an anderen Programmen

$$\text{CPU Ausführungszeit für ein Programm} = \text{Benötigte Taktzyklen} \times \text{Taktperiode}$$

or

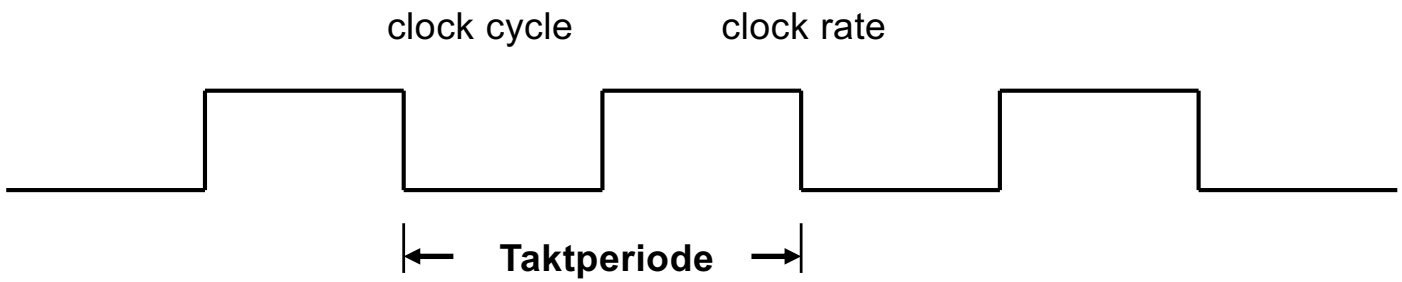
$$\text{CPU Ausführungszeit für ein Programm} = \frac{\text{Benötigte Taktzyklen}}{\text{Taktrate}}$$

- Performance kann verbessert werden durch reduzieren der Taktperiode oder der Anzahl Taktzyklen welche ein Programm benötigt

Review: Taktrate / Taktperiode

- Taktrate (MHz, GHz) ist invers zur Taktperiode

$$CC = 1 / CR$$



10 nsec Taktperiode => 100 MHz Taktrate

5 nsec Taktperiode => 200 MHz Taktrate

2 nsec Taktperiode => 500 MHz Taktrate

1 nsec Taktperiode => 1 GHz Taktrate

500 psec Taktperiode => 2 GHz Taktrate

250 psec Taktperiode => 4 GHz Taktrate

200 psec Taktperiode => 5 GHz Taktrate

Taktzyklen pro Befehl (CPI)

- Nicht alle Befehle benötigen gleichviel Ausführungszeit
  - Eine **Möglichkeit Ausführungszeit zu verstehen** ist, die Anzahl benötigter Befehle multipliziert mit der durchschnittlichen Ausführungszeit eines Befehls

Taktzyklen für ein Programm	=	Befehle für ein Programm	x	Durchschnittliche Taktzyklen pro Befehl

- Zyklen pro Befehl (Clock Cycles Per Instruction, CPI) – **Durchschnittliche** Anzahl der Taktzyklen, die pro Befehl für ein Programm erforderlich sind
  - CPI ist nicht die Anzahl Takte um einen einzelnen Befehl auszuführen (sondern der **Durchschnitt** eines Sets an Befehlen)
  - Eine Möglichkeit zwei verschiedene Implementationen derselben Befehlssatzarchitektur (ISA) zu vergleichen

Effektive CPI

- Um die effektive Gesamt-CPI (für ein Befehlsmix) zu bestimmen werden die verschieden Befehlsklassen und deren durchschnittlichen Taktzyklen betrachtet

$$\text{Effektive Gesamt-CPI} = \sum_{i=1}^{n} (CPI_i \times IC_i)$$

  - Wobei IC_i (instruction count) der Anteil/Prozentsatz der ausgeführter Befehle von der Klasse i entspricht
  - CPI_i ist die mittlere Anzahl Taktzyklen pro Befehl der entsprechenden Klasse von Befehlen
  - n ist die Anzahl der Befehlsklassen

- Die effektive Gesamt-CPI variiert mit dem Befehlsmix (instruction mix) – ein Mass der dynamischen Häufigkeit von Befehlen über ein oder mehrere Programme.

DIE Performance Gleichung

- Unsere Grundgleichung für die Performance ist nun:

$$\text{(User) CPU Zeit} = IC \times CPI \times \text{Taktperiode}$$

or

$$\text{(User) CPU Zeit} = \frac{IC \times CPI}{\text{Taktrate}}$$

- Diese Gleichungen zeigen die drei Schlüsselfaktoren welche Einfluss auf die Performance haben
  - Messen der CPU Zeit durch ausführen des Programm
  - Die Taktrate ist normalerweise gegeben
  - Messen der gesamten Anzahl an Befehlen durch Profiler/Simulatoren ohne alle Implementationsdetails zu kennen
  - CPI variiert je nach Befehlstyp und ISA Implementation. Hier müssen wir die Implementationsdetails kennen

Bestimmung der CPU Performance

CPU Zeit = IC x CPI x Taktperiode

	IC	CPI	Taktperiode
Algorithmus	X	X	
Programmier-sprache	X	X	
Compiler	X	X	
ISA	X	X	X
Prozessor Organisation		X	X
Technologie			X

Ein einfaches Beispiel

Op	Anteil Befehle	CPI_i	Anteil x CPI_i			
ALU	50%	1	.5	.5	.5	.25
Laden	20%	5	1.0	.4	1.0	1.0
Speichern	10%	3	.3	.3	.3	.3
Sprung	20%	2	.4	.4	.2	.4
			Σ = 2.2	1.6	2.0	1.95

- Wie viel schneller wäre der Rechner wenn ein besserer Daten Cache das laden auf 2 Zyklen reduzieren könnte?

  CPU Zeit neu = 1.6 x IC x CC also 2.2/1.6 bedeutet 37.5% schneller

- Wie steht das im Vergleich zur Möglichkeit mit einer besseren Sprungvorhersage 1 Zyklus sparen zu können?

  CPU Zeit neu = 2.0 x IC x CC also 2.2/2.0 bedeutet 10% schneller

- Was wenn 2 ALU Befehle gleichzeitig ausgeführt werden können?

  CPU Zeit neu = 1.95 x IC x CC also 2.2/1.95 bedeutet 12.8% schneller

Performance vergleichen und zusammenfassen

- Wie fassen wir gemessene Performance Metriken in einer einzigen Zahl zusammen?
  - Die durchschnittliche Ausführungszeit (direkt proportional zur totalen Ausführungszeit) ist das **arithmetische Mittel** (AM)

$$AM = 1/n \sum_{i=1}^{n} Time_i$$

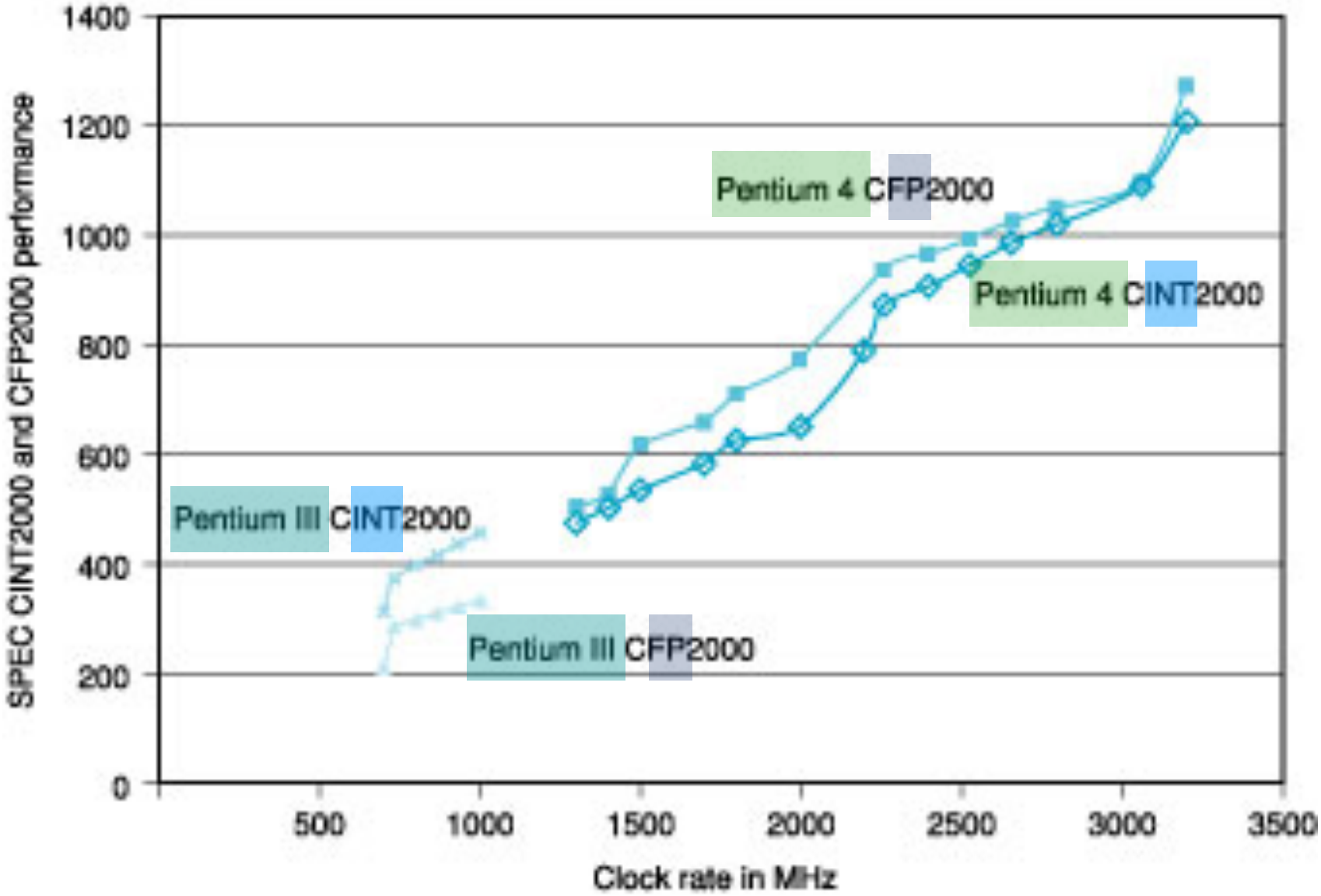
  - Wobei $Time_i$ die Ausführungszeit für Programm i von total n Programmen (gesamte Arbeitsbelastung) ist.
  - Ein kleinerer Mittelwert zeigt einen kleinere durchschnittliche Ausführungszeit und steht daher für eine bessere Performance
- Grundprinzip bei Performancemessungen ist die Reproduzierbarkeit – Beschreibe alles was nötigt ist, dass jemand Anders deine Messung wiederholen kann. (OS Version, Compiler Konfiguration, Input Daten, Computer Konfiguration wie Taktrate, Grösse und Geschwindigkeit des Cache und Hauptspeichers, usw.)

SPEC Benchmarks www.spec.org

Integer benchmarks		FP benchmarks	
gzip	compression	wupwise	Quantum chromodynamics
vpr	FPGA place & route	swim	Shallow water model
gcc	GNU C compiler	mgrid	Multigrid solver in 3D fields
mcf	Combinatorial optimization	applu	Parabolic/elliptic pde
crafty	Chess program	mesa	3D graphics library
parser	Word processing program	galgel	Computational fluid dynamics
eon	Computer visualization	art	Image recognition (NN)
perlbmk	perl application	equake	Seismic wave propagation simulation
gap	Group theory interpreter	facerec	Facial image recognition
vortex	Object oriented database	ammp	Computational chemistry
bzip2	compression	lucas	Primality testing
twolf	Circuit place & route	fma3d	Crash simulation fem
		sixtrack	Nuclear physics accel
		apsi	Pollutant distribution

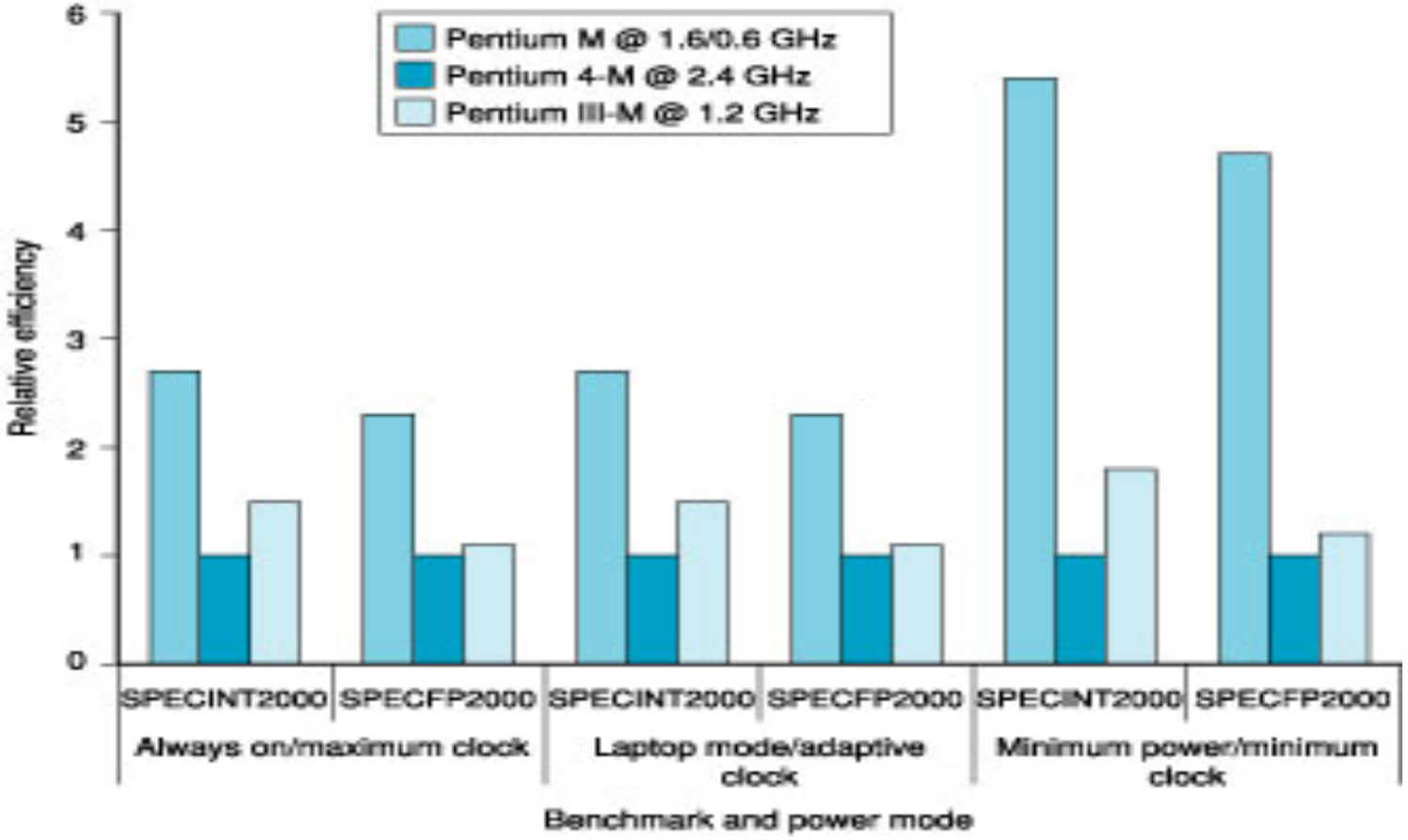
SPEC: System Performance Evaluation Cooperative

Example SPEC Ratings



Andere Performance Metriken | e.g. SPEC Power®

- Energieverbrauch – speziell bei Batteriebetriebenen embedded Geräten (und passiver Kühlung)
  - Für Geräte mit beschränkter Stromversorgung, ist die wichtigste Metrik die Energieeffizienz (Power Benchmarking)



Zusammenfassung: Evaluating ISAs

- Metriken zur Entwurfszeit:
  - Kann das implementiert werden, in welcher Zeit, zu welchen Kosten?
  - Kann das programmiert werden? Leichtere Kompilierung?
- Statische Metriken:
  - Wie viele Bytes benötigt das Programm im Hauptspeicher?
- Dynamische Metriken :
  - Wie viele Befehle werden werden für ein Programm ausgeführt? Wie viele Bytes muss der Prozessor laden um das Programm auszuführen?
  - Wie viele Taktzyklen benötigt ein Befehl?
  - Wie “kurz" darf ein Taktzyklus sein?

Beste Metrik: Zeit die Programm benötigt!

Abhängig vom Befehlssatz, Prozessor Aufbau und Kompilierungstechniken.