CS 465: Computer Systems Architecture

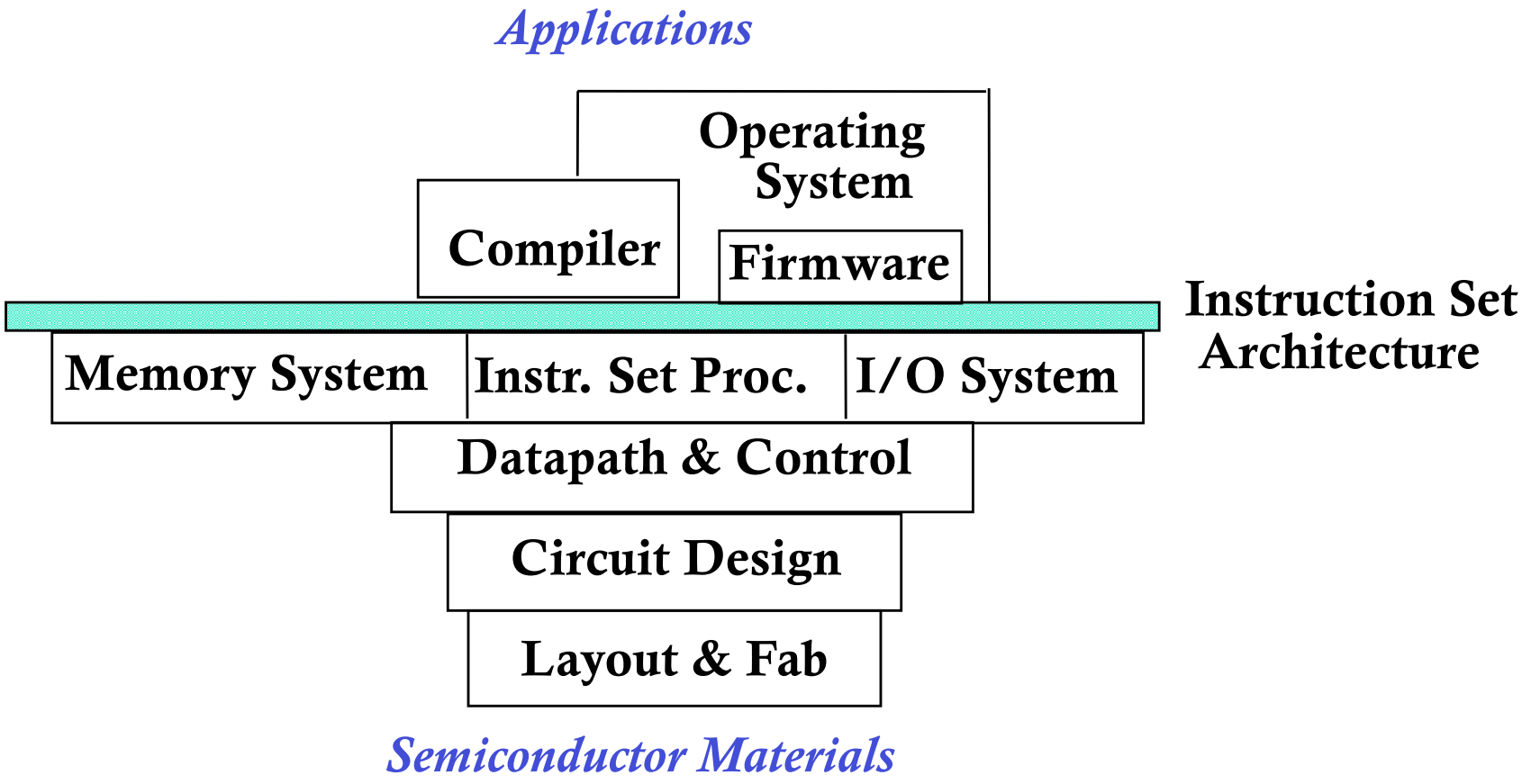
Lecture 1: Introduction

*Slides adapted from Computer Organization and Design by Patterson and Henessey

Roadmap

- Computer systems overview
  - Which 5 major components in a computer?
  - ISA
- Computer performance
  - Metrics
  - Components
  - Evaluation

Computer Systems



Abstractions - ISA

The BIG Picture

- Abstraction helps us deal with complexity
  - Hide lower-level detail
- Instruction set architecture (ISA)
  - The hardware/software interface
- Application binary interface
  - The ISA plus system software interface
- Implementation (hardware obeys instruction)
  - The details underlying and interface

Define ISA

An abstract interface between the hardware and low(est)-level software that encompasses all the information necessary to write a machine language program that will run correctly; including instructions, memory, I/O, registers, etc.

- Instruction Set Architecture
- We will learn/use MIPS in this course

Roadmap

- Overview of computer systems and ISA
- Computer performance
  - Metrics
  - Components
  - Evaluation
- Discussion and conclusion

Computer Performance

- What is important for computer performance?
- How do we determine that one computer is better than another ?

Response Time and Throughput

- 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?

Thought Question

When do we usually care about response time and when about throughput ? Why ?

a) Individual Desktop PC

b) Embedded Processor

c) Mail Server (you are maintaining it …)

We'll focus on response time for now…

Relative Performance

- Define Performance = 1/Execution Time
- "X is n times faster than Y" (n>=1)

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

Relative Performance

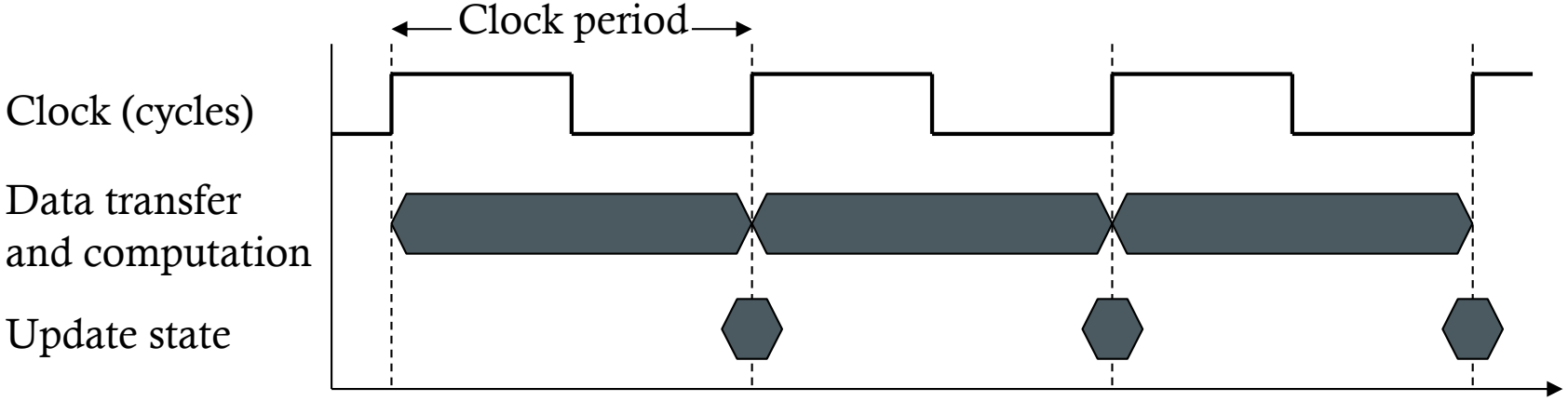
- Example: time taken to run a program
  - 10s on A, 15s on B
  - So A is ???? times faster than B ?
  - Execution Time_B / Execution Time_A = 15s / 10s = 1.5
  - So A is 1.5 times faster than B

Measuring Execution Time

- Elapsed time/Wall clock/Response 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
  - Comprises user CPU time and system CPU time
- Different programs are affected differently by CPU and system performance
  - Focus on CPU performance for now

CPU Clocking

- Operation of digital hardware governed by a constant-rate clock
  - Clock determines when events take place



    - 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

CPU Time

$$\text{CPU Time} = \text{CPU Clock Cycles} \times \text{Clock Cycle Time}$$

$$= \frac{\text{CPU Clock Cycles}}{\text{Clock Rate}}$$

- Performance improved by
  - Reducing number of clock cycles
  - Increasing clock rate
  - Hardware designer often need to trade off either the number of clock cycles required for a program or the length of a clock cycle

CPU Time Example

- Computer A: 2GHz clock, 10s CPU time
- Designing Computer B with the same ISA
  - Aim for 6s CPU time
  - Can do a faster clock, but causes 1.2x clock cycles
- How fast must Computer B clock be?

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

CPU Time Example

- Computer A: 2GHz clock, 10s CPU time
- Designing Computer B with the same ISA
  - Aim for 6s CPU time
  - Can do a faster clock, but causes 1.2x clock cycles
- How fast must Computer B clock be?

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

$$\text{Clock Cycles}_A = \text{CPU Time}_A \times \text{Clock Rate}_A$$

$$= 10\text{s} \times 2\text{GHz} = 20 \times 10^9$$

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

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 ?, ? And ?
- Average cycles per instruction
  - Determined by ?

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

CPI Example

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

$$\text{CPU Time}_A = \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A$$
$$= IC \times 2.0 \times 250\text{ps} = IC \times 500\text{ps}$$

CPI Example

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

$$\text{CPU Time}_A = \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A$$
$$= IC \times 2.0 \times 250\text{ps} = IC \times 500\text{ps}$$

A is faster…

$$\text{CPU Time}_B = \text{Instruction Count} \times \text{CPI}_B \times \text{Cycle Time}_B$$
$$= IC \times 1.2 \times 500\text{ps} = IC \times 600\text{ps}$$

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

…by this much

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 CPIs
    - Average CPI affected by instruction mix

CPI in More Details

- 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

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

CPI Example [Instruction Mix]

- Alternative compiled code sequences using instructions in classes A, B, C. Avg CPI?

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

- Sequence 1, approach 1
  - Avg. CPI = number of clock cycles / number of instructions
  - IC = 2+1+2 = 5
  - Clock Cycles = 2 × 1 + 1 × 2 + 2 × 3 = 10
  - Avg. CPI = 10/5 = 2.0

CPI Example [Instruction Mix]

- Alternative compiled code sequences using instructions in classes A, B, C. Avg CPI?

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

- Sequence 1, approach 2
  - Avg. CPI = $\sum_i CPI_i \times Freq_i$
  - IC = 5
  - $Freq_A$ = 2/5 = 0.4, $Freq_B$ = 1/5 = 0.2, $Freq_C$ = 2/5 = 0.4
  - Avg. CPI = 1*0.4 + 2*0.2 + 3*0.4 = 2.0

CPI Example [Instruction Mix]

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

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

  - Sequence 2:
    - Try it as your exercise

CPU Time with CPI Details

$$\textbf{CPU Time} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{Instruction Count}_i) \times \text{Clock Cycle Time}$$

$$\textbf{CPU Time} = \frac{\sum_{i=1}^{n} (\text{CPI}_i \times \text{Instruction Count}_i)}{\text{Clock Rate}}$$

Performance Summary

The BIG Picture

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

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

Performance Evaluation

- For a computer system, how do we evaluate performance?
- Report?
- Summarize?
- Compare?

Workload & Benchmark

Define Workload: A set of programs run on a computer that is either the actual collection of programs run by a user or constructed from real programs to approximate such a mix. A typical workload specifies both the programs and the relative frequencies.

Define Benchmark: A program selected for use in comparing computer performance.

SPEC CPU Benchmark

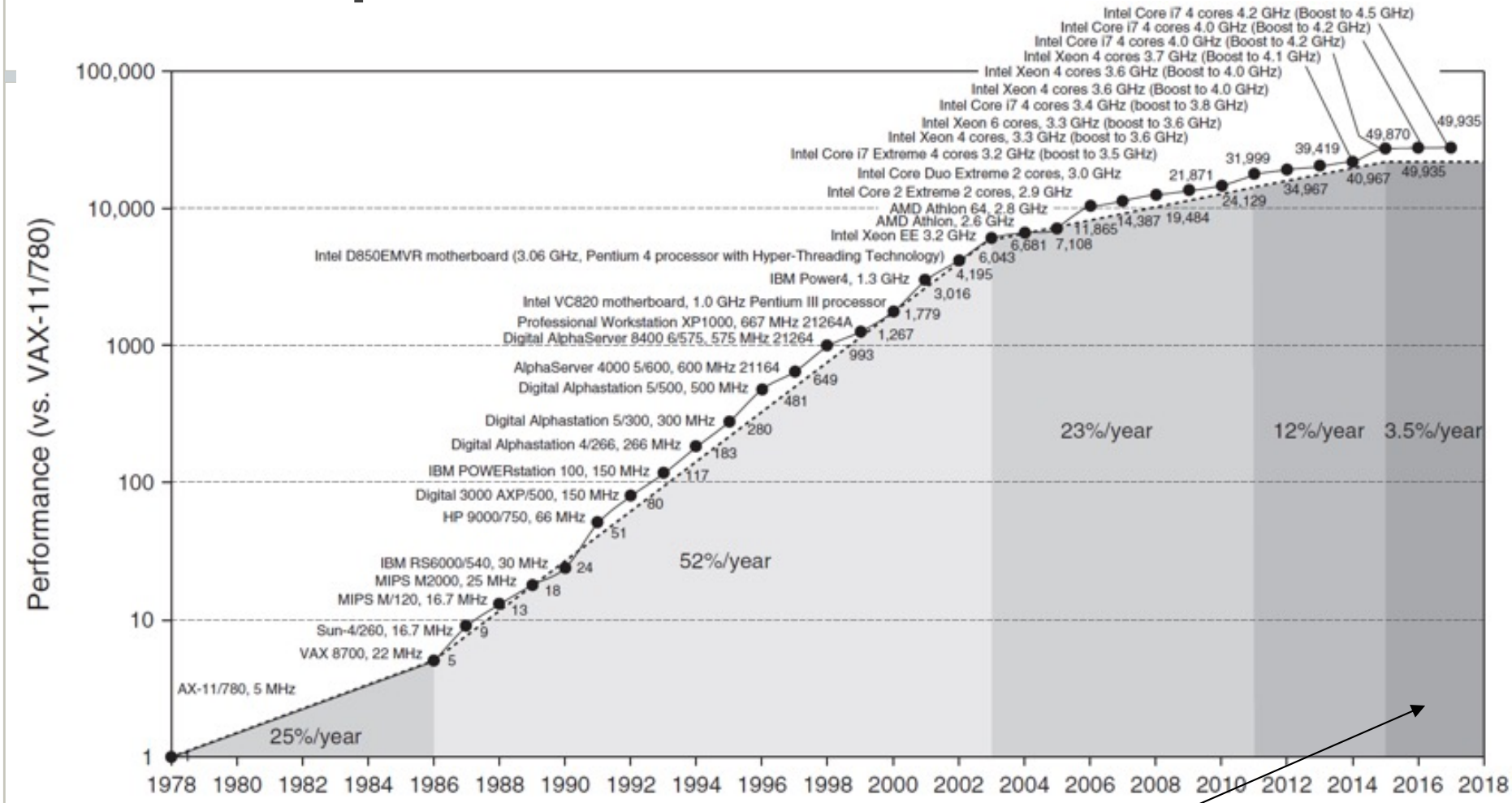
- Standard Performance Evaluation Corp (SPEC)
  - Develops benchmarks for CPU, I/O, Web, …
- SPEC ~~CPU2006~~ CPU2017
  - Elapsed time to execute a selection of programs
    - Negligible I/O, so focuses on CPU performance
  - Normalize relative to reference machine
  - Summarize as geometric mean of performance ratios
    - CINT (integer) and CFP (floating-point)
    - SPEC Speed and SPEC Rate

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

SPECspeed 2017 Integer
Intel Xeon E5-2650L

Description	*Name*	*Instruction Count x 10^9*	*CPI*	*Clock cycle time (seconds x 10^–9)*	*Execution Time (seconds)*	*Reference Time (seconds)*	*SPECratio*
Perl interpreter	perlbench	2684	0.42	0.556	627	1774	2.83
GNU C compiler	gcc	2322	0.67	0.556	863	3976	4.61
Route planning	mcf	1786	1.22	0.556	1215	4721	3.89
Discrete Event simulation - computer network	omnetpp	1107	0.82	0.556	507	1630	3.21
XML to HTML conversion via XSLT	xalancbmk	1314	0.75	0.556	549	1417	2.58
Video compression	x264	4488	0.32	0.556	813	1763	2.17
Artificial Intelligence: alpha-beta tree search (Chess)	deepsjeng	2216	0.57	0.556	698	1432	2.05
Artificial Intelligence: Monte Carlo tree search (Go)	leela	2236	0.79	0.556	987	1703	1.73
Artificial Intelligence: recursive solution generator (Sudoku)	exchange2	6683	0.46	0.556	1718	2939	1.71
General data compression	xz	8533	1.32	0.556	6290	6182	0.98
Geometric mean							2.36

Uniprocessor Performance



Constrained by power, instruction-level parallelism, memory latency