

Chapter 1

Computer Abstractions and Technology

Adapted by Prof. Gheith Abandah

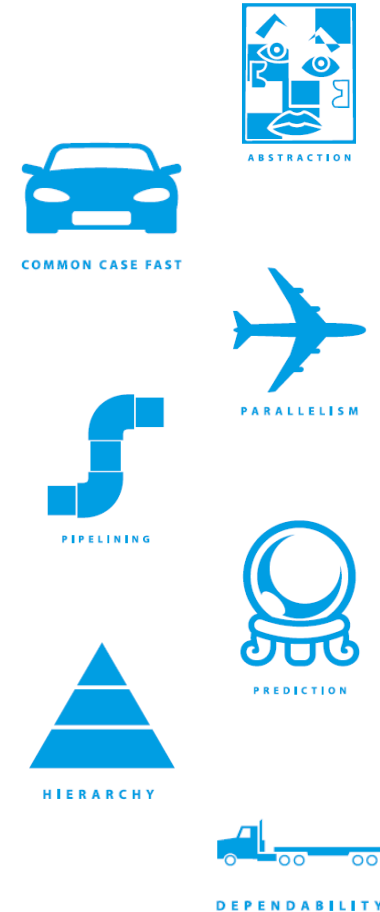
Last updated 8/10/2026

Content

- 1.2 Seven Great Ideas in Computer Architecture (*Review*)
- 1.5 Technologies for Building Processors and Memory
- 1.7 The Power Wall
- 1.8 The Sea Change: The Switch from Uniprocessors to Multiprocessors
- 1.9 Real Stuff: Benchmarking the Intel Core i7

Seven Great Ideas

- Use **abstraction** to simplify design
- Make the **common case fast**
- Performance *via* **parallelism**
- Performance *via* **pipelining**
- Performance *via* **prediction**
- **Hierarchy** of memories
- **Dependability** *via* redundancy

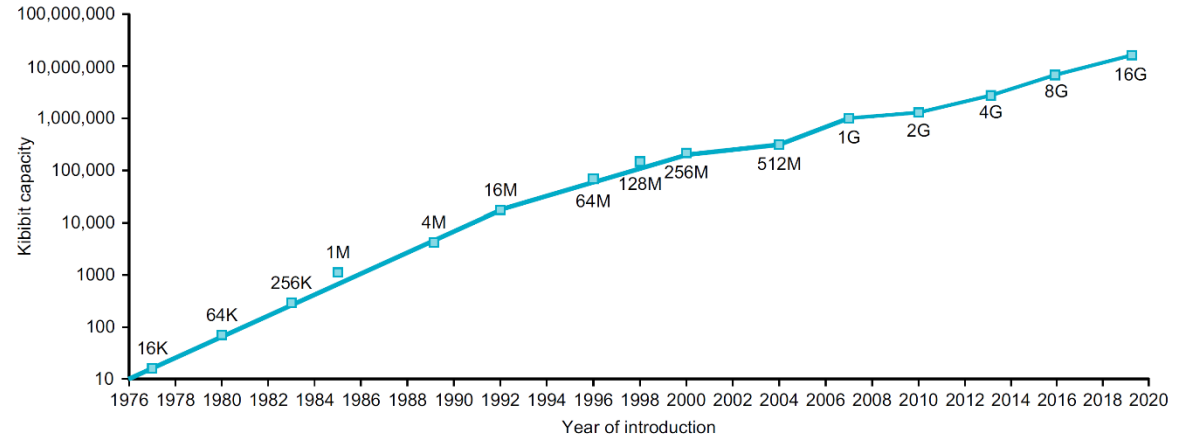


Content

- 1.2 Seven Great Ideas in Computer Architecture (*Review*)
- 1.5 Technologies for Building Processors and Memory
- 1.7 The Power Wall
- 1.8 The Sea Change: The Switch from Uniprocessors to Multiprocessors
- 1.9 Real Stuff: Benchmarking the Intel Core i7

Technology Trends

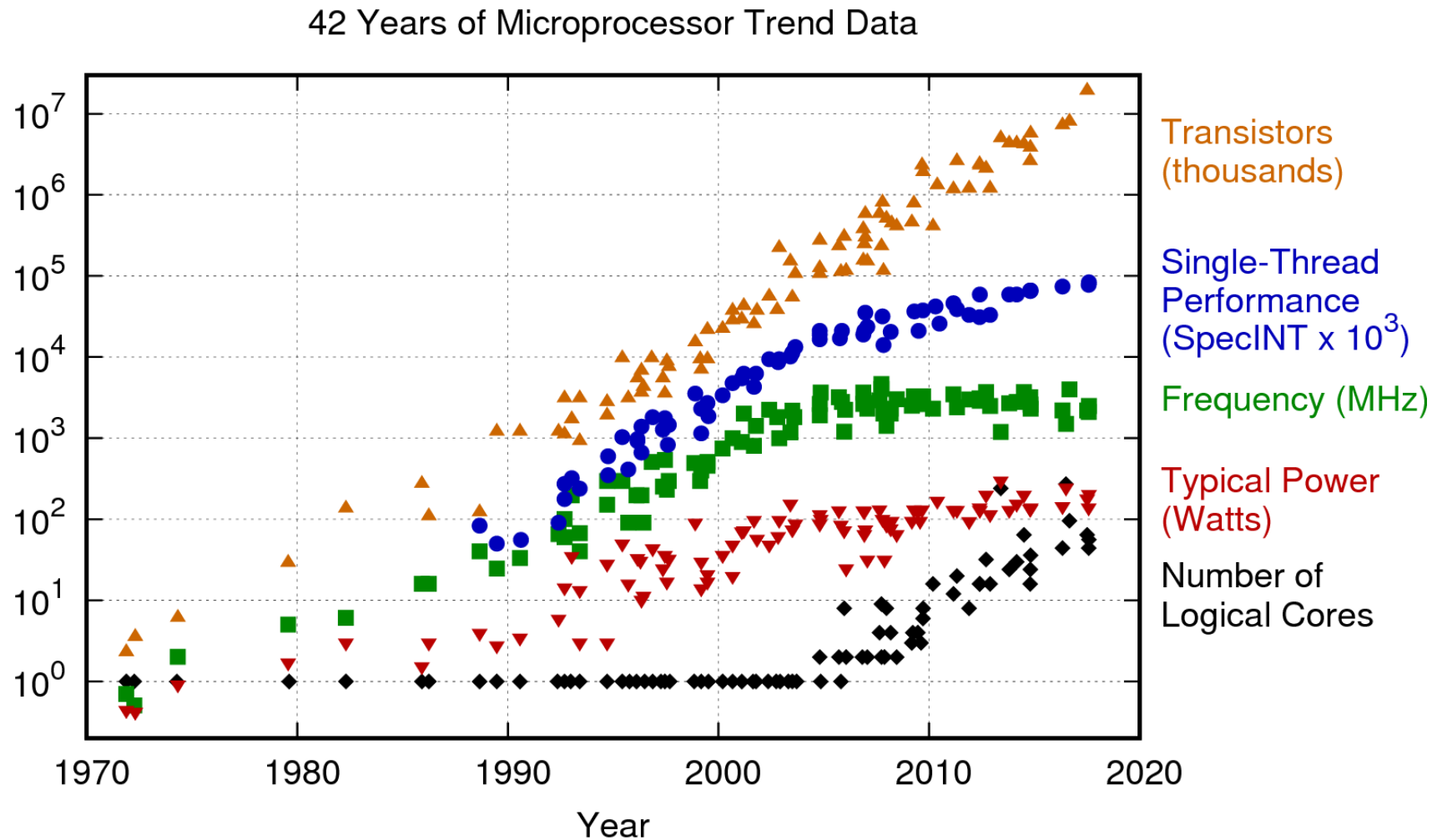
- Electronics **technology continues to evolve**
 - Increased **capacity** and **performance**
 - Reduced **cost**



DRAM capacity

Year	Technology	Relative performance/cost
1951	Vacuum tube	1
1965	Transistor	35
1975	Integrated circuit (IC)	900
1995	Very large-scale IC (VLSI)	2,400,000
2020	Ultra large-scale IC	500,000,000,000

Technology Trends



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp

Technology Trends

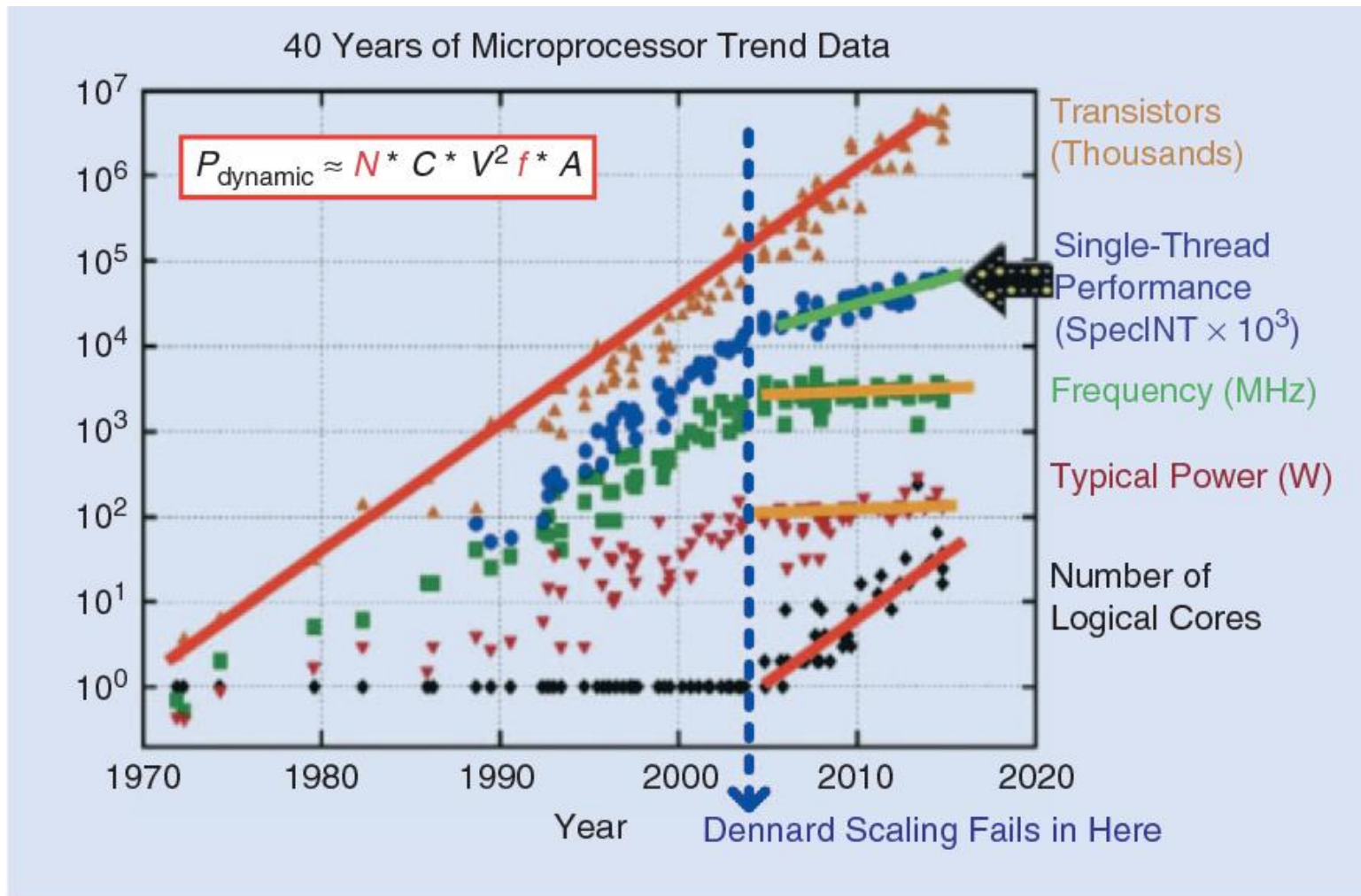


FIGURE 1: The Dennard scaling failed around the middle of the 2000s [24].

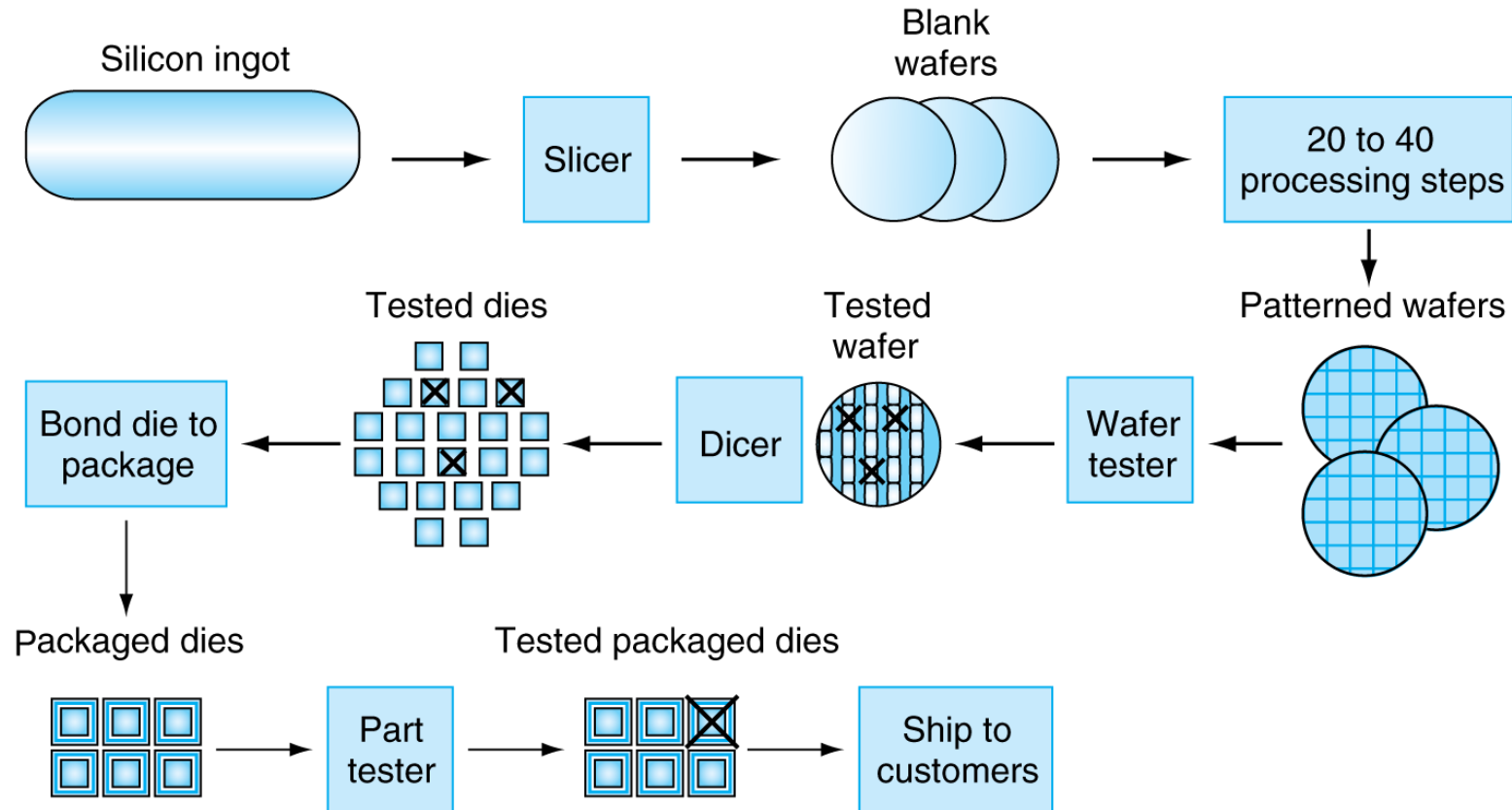
- NVIDIA Blackwell (2024) contains **208 billion transistors**, while Rubin (2026), the largest publicly documented non-wafer-scale GPU by transistor count, contains **336 billion**. Both use two compute dies in one processor package.
- **Exercise:** Treat 2024–2026 as a two-year interval. Calculate the annual compound growth rate and estimate the transistor count of a comparable leading GPU in **2030**, assuming this rate continues.

- NVIDIA Blackwell (2024) contains **208 billion transistors**, while Rubin (2026), the largest publicly documented non-wafer-scale GPU by transistor count, contains **336 billion**. Both use two compute dies in one processor package.
- **Exercise:** Treat 2024–2026 as a two-year interval. Calculate the annual compound growth rate and estimate the transistor count of a comparable leading GPU in **2030**, assuming this rate continues.
- **Annual growth rate:**
$$r = \left(\frac{336}{208} \right)^{1/2} - 1 \approx \boxed{27.1\%}$$
- **Projected count after four years:**
$$N_{2030} = 336(1 + r)^4 \approx \boxed{877 \text{ billion transistors}}$$
- **Note:** This is a recent GPU-generation growth estimate, not a universal industry rate. The 2030 value is an extrapolation. Here, “chip” means one processor package, rather than a single silicon die.

Semiconductor Technology

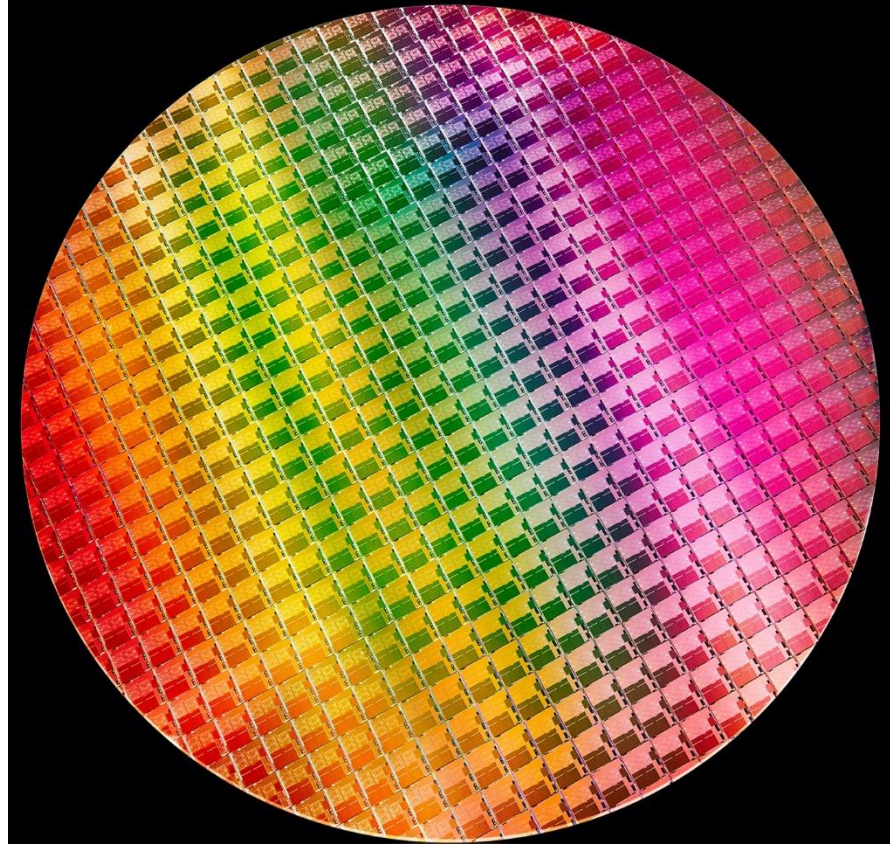
- **Silicon:** semiconductor
- Add materials to transform properties:
 - Conductors
 - Insulators
 - Switch
- **YouTube Video:** VLSI Fabrication Process
<https://youtu.be/g8Qav3vIv9s>

Manufacturing ICs



- **Yield:** proportion of working dies per wafer

Intel Core 10th Gen



- 300mm wafer, 506 chips, 10nm technology
- Each chip is 11.4 x 10.7 mm

Integrated Circuit Cost

$$\text{Cost per die} = \frac{\text{Cost per wafer}}{\text{Dies per wafer} \times \text{Yield}}$$

$$\text{Dies per wafer} \approx \text{Wafer area} / \text{Die area}$$

$$\text{Yield} = \frac{1}{(1 + (\text{Defects per area} \times \text{Die area} / 2))^2}$$

- Nonlinear relation to area and defect rate
 - Wafer cost and area are fixed
 - Defect rate determined by manufacturing process
 - Die area determined by architecture and circuit design

Integrated Circuit Cost

Exercise

- A square die measuring **10 mm × 10 mm** costs **\$20** to manufacture. Assume the cost per die is proportional to the **square of its area**.
- **Exercise:** Estimate the manufacturing cost of a larger die measuring **15 mm × 15 mm**.

Integrated Circuit Cost

Solution

- A square die measuring **10 mm × 10 mm** costs **\$20** to manufacture. Assume the cost per die is proportional to the **square of its area**.
- **Exercise:** Estimate the manufacturing cost of a larger die measuring **15 mm × 15 mm**.
- **Solution:**

$$A_1 = 10 \times 10 = 100 \text{ mm}^2, \quad A_2 = 15 \times 15 = 225 \text{ mm}^2$$

Since $C = kA^2$:

$$C_2 = C_1 \left(\frac{A_2}{A_1} \right)^2 = 20 \left(\frac{225}{100} \right)^2 = \boxed{\$101.25}$$

- The die area increases by **2.25×**, so its estimated cost increases by **5.0625×**.

Content

- 1.2 Seven Great Ideas in Computer Architecture (*Review*)
- 1.5 Technologies for Building Processors and Memory
- 1.7 The Power Wall
- 1.8 The Sea Change: The Switch from Uniprocessors to Multiprocessors
- 1.9 Real Stuff: Benchmarking the Intel Core i7

Power Consumption in ICs

- **Total power = Static power + Dynamic power**

$$P_{\text{total}} = P_{\text{static}} + P_{\text{dynamic}}$$

- **Static power:** Caused by leakage current, even when transistors are not switching.

$$P_{\text{static}} = V I_{\text{leakage}}$$

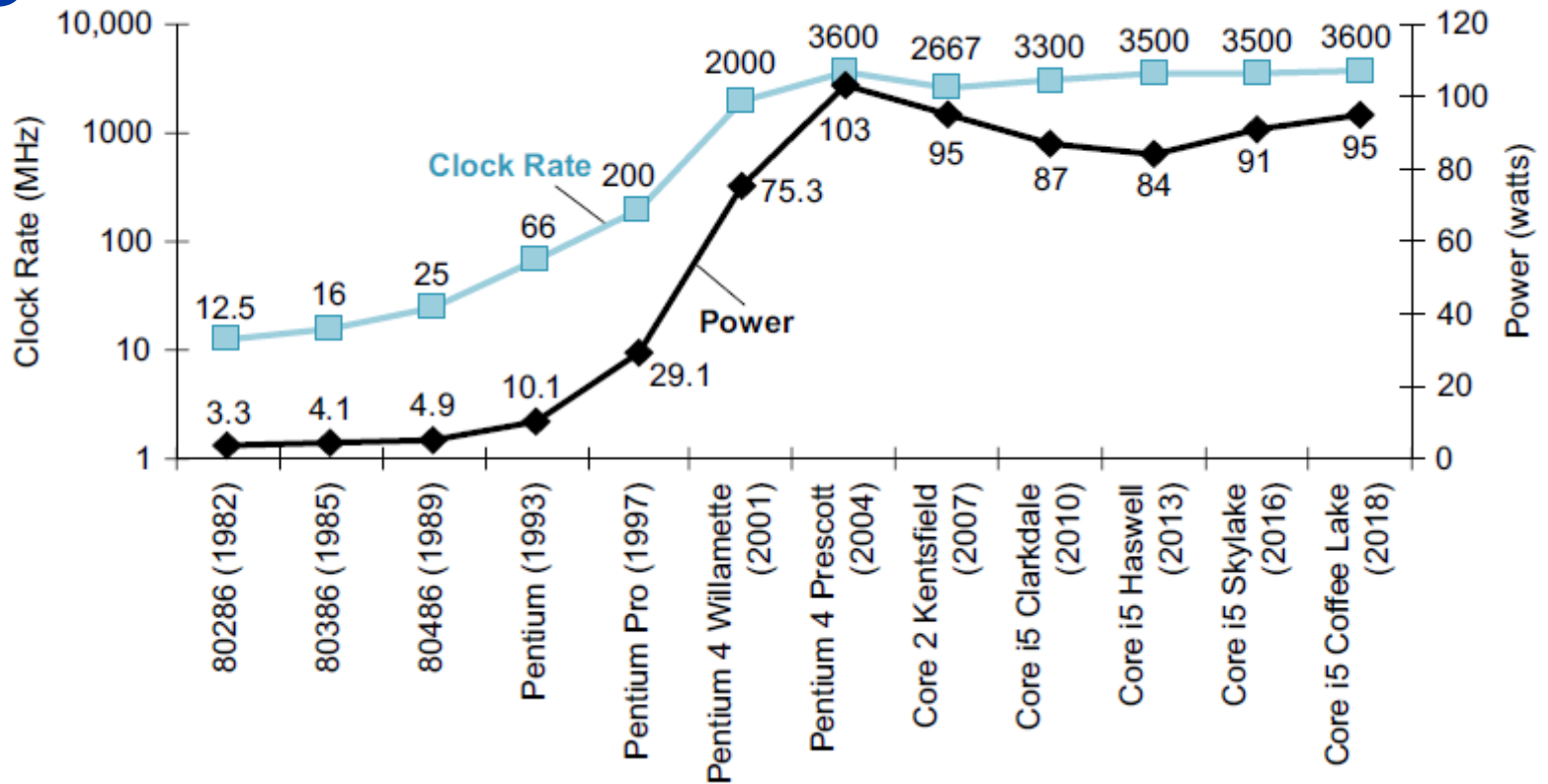
- **Dynamic power:** Caused mainly by charging and discharging capacitances during switching.

$$P_{\text{dynamic}} = \alpha C V^2 f$$

where α is switching activity, C is effective capacitance, V is supply voltage, and f is clock frequency.

- **Key point:** At fixed voltage and switching activity, reducing frequency lowers dynamic power, while leakage power remains.

Power Trends



- In CMOS IC technology

$$\text{Power} = \frac{1}{2} \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency}$$

×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?

Power Consumption in ICs

Exercise

- A modern processor operates at $V_1 = 1.0$ V and $f_1 = 3.0$ GHz, consuming $P_{total} = 100$ W. Static power accounts for **20 W**, and dynamic power accounts for **80 W**.
- The frequency increases to $f_2 = 4.0$ GHz and voltage to $V_2 = 1.1$ V . Estimate the new total power, assuming unchanged switching activity and capacitance, and constant leakage current.

Power Consumption in ICs

Solution

- A modern processor operates at $V_1 = 1.0$ V and $f_1 = 3.0$ GHz, consuming $P_{total} = 100$ W. Static power accounts for **20 W**, and dynamic power accounts for **80 W**.
- The frequency increases to $f_2 = 4.0$ GHz and voltage to $V_2 = 1.1$ V. Estimate the new total power, assuming unchanged switching activity and capacitance, and constant leakage current.

- Dynamic power scales with $V^2 f$:
$$P_{dynamic,2} = 80 \left(\frac{1.1}{1.0} \right)^2 \left(\frac{4.0}{3.0} \right) \approx 129.1 \text{ W}$$

- With constant leakage current, static power scales with voltage:

$$P_{static,2} = 20 \left(\frac{1.1}{1.0} \right) = 22 \text{ W}$$

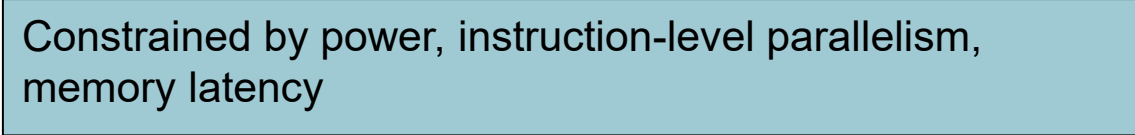
- Therefore:

$$P_{total,2} = 22 + 129.1 = \boxed{151.1 \text{ W}}$$

- The frequency increases by **33.3%**, while total power increases by approximately **51.1%**. *In practice, leakage current also changes with voltage and temperature.*

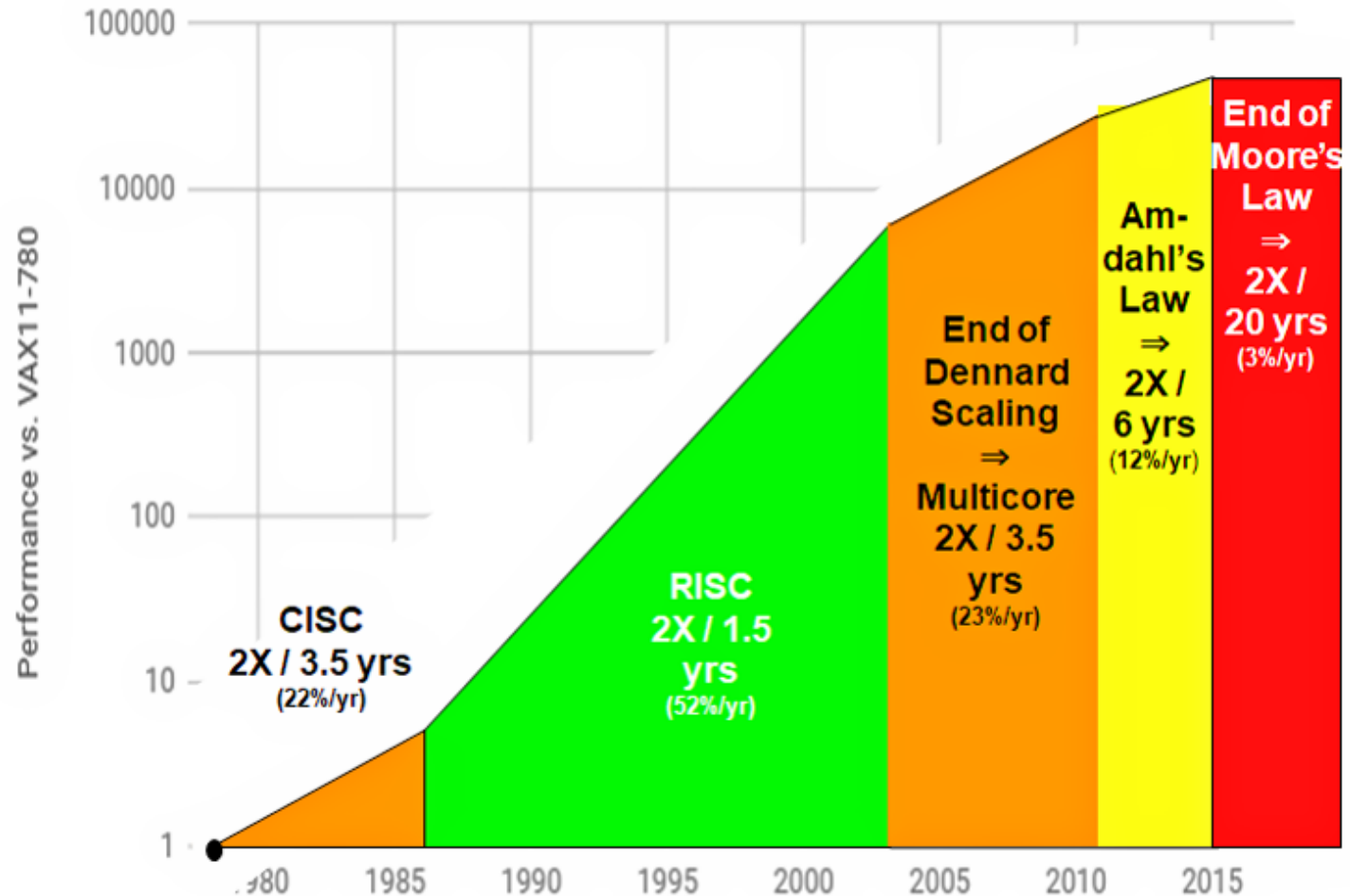
Content

- 1.2 Seven Great Ideas in Computer Architecture (*Review*)
- 1.5 Technologies for Building Processors and Memory
- 1.7 The Power Wall
- 1.8 The Sea Change: The Switch from Uniprocessors to Multiprocessors**
- 1.9 Real Stuff: Benchmarking the Intel Core i7**



Uniprocessor Performance

40 years of Processor Performance



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

Content

- 1.2 Seven Great Ideas in Computer Architecture (*Review*)
- 1.5 Technologies for Building Processors and Memory
- 1.7 The Power Wall
- 1.8 The Sea Change: The Switch from Uniprocessors to Multiprocessors
- 1.9 Real Stuff: Benchmarking the Intel Xeon**

SPEC CPU Benchmark

- Programs used to measure performance
 - Supposedly typical of actual workload
- Standard Performance Evaluation Corp (SPEC)
 - Develops benchmarks for CPU, I/O, Web, ...
- SPEC 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
 - CINT2017 (integer) and CFP2017 (floating-point)

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

SPECspeed 2017 Integer benchmarks on a 1.8 GHz 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

SPEC Power Benchmark

- Power consumption of server at different workload levels
 - Performance: ssj_ops/sec
 - Power: Watts (Joules/sec)

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

SPECpower_ssj2008 for Xeon E5-2650L

Target Load %	Performance (ssj_ops)	Average Power (watts)
100%	4,864,136	347
90%	4,389,196	312
80%	3,905,724	278
70%	3,418,737	241
60%	2,925,811	212
50%	2,439,017	183
40%	1,951,394	160
30%	1,461,411	141
20%	974,045	128
10%	485,973	115
0%	0	48
Overall Sum	26,815,444	2,165
$\sum \text{ssj_ops} / \sum \text{power} =$		12,385

Summary

- 1.2 Seven Great Ideas in Computer Architecture (*Review*)
- 1.5 Technologies for Building Processors and Memory
- 1.7 The Power Wall
- 1.8 The Sea Change: The Switch from Uniprocessors to Multiprocessors
- 1.9 Real Stuff: Benchmarking the Intel Core i7