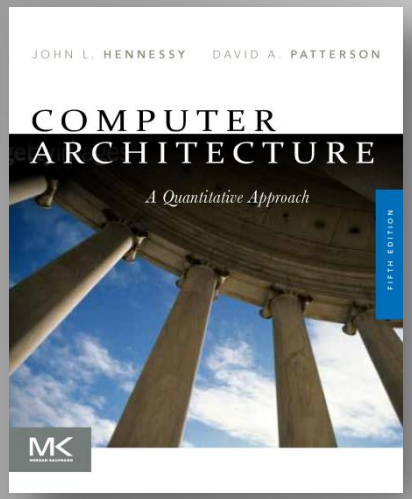


Chapter 1

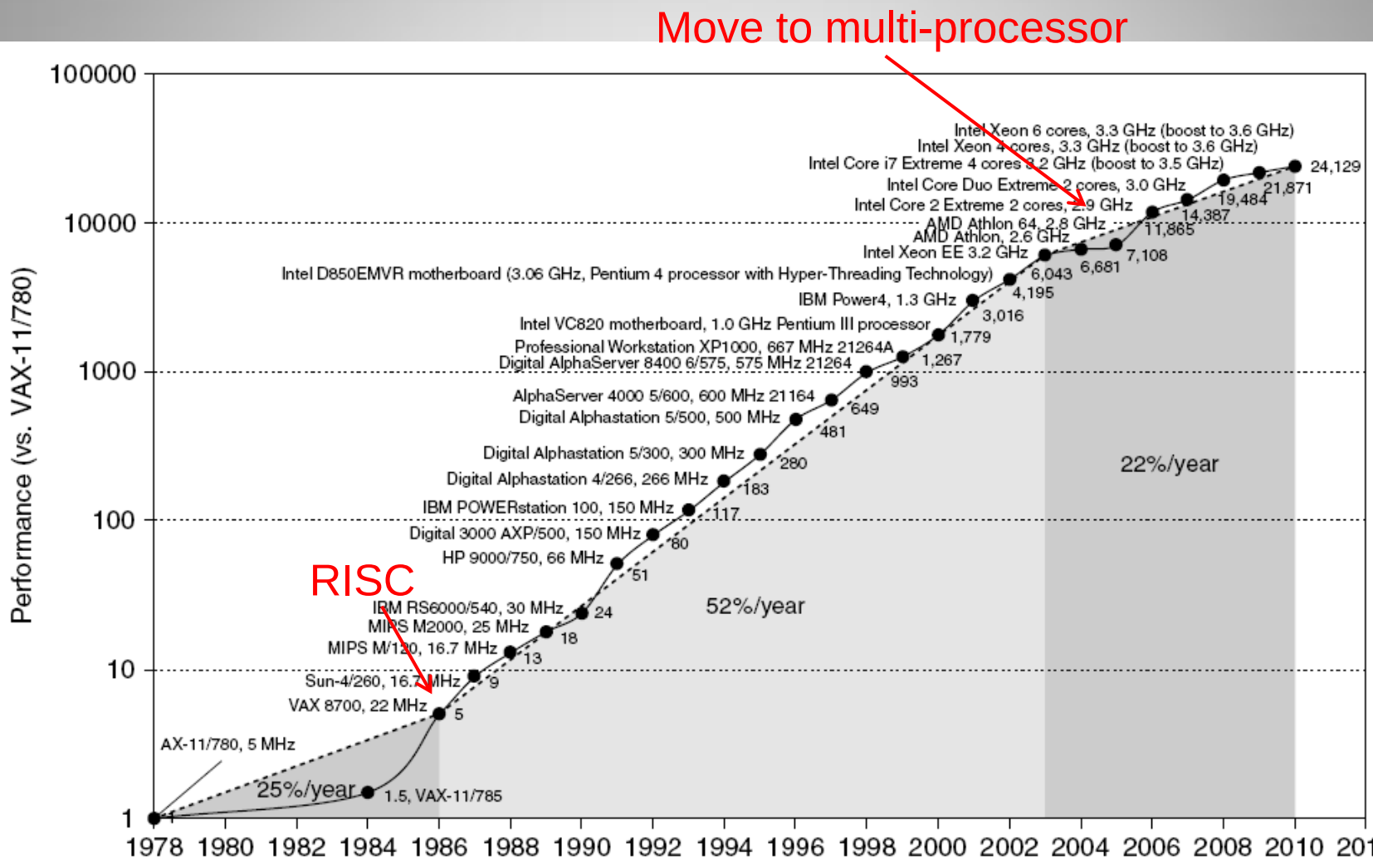
Fundamentals of Quantitative Design and Analysis

Contents

Computer technology

- Performance improvements:
  - Improvements in semiconductor technology
    - Feature size, clock speed
  - Improvements in computer architectures
    - Enabled by HLL compilers, UNIX
    - Lead to RISC architectures
  - Together have enabled:
    - Lightweight computers
    - Productivity-based managed/interpreted programming languages

Single processor performance



Current trends in architecture

- Cannot continue to leverage Instruction-Level parallelism (ILP)
  - Single processor performance improvement ended in 2003. Why?
- New models for performance:
  - Data-level parallelism (DLP)
  - Thread-level parallelism (TLP)
  - Request-level parallelism (RLP)
- The new models for performance require explicit restructuring of the application. No more free lunch for application developers!!!

Classes of computers

1. Personal Mobile Device (PMD)
    1. e.g. smart phones, tablet computers
    2. Emphasis on energy efficiency and real-time
2. Desktop Computers
    1. Emphasis on price-performance
3. Servers
    1. Emphasis on availability, scalability, throughput
4. Clusters / Warehouse Scale Computers (WSCs)
    1. Used for “Software as a Service (SaaS)”
    2. Emphasis on availability and price-performance
    3. Sub-class: Supercomputers, emphasis: floating-point performance and fast internal networks
5. Embedded Computers
    1. Emphasis: price

	Personal Mobile device	Desktop	Server	Cluster WSC	Embedded
Price of system ($)	100 - 1,000	300 – 2,500	5,000 – 10,000,000	100,000 – 200,000,000	10 – 100,000
Price of processor ($)	10 - 100	50 - 500	200- 2,000	50 - 250	0.01 - 100
Critical system design issues	Cost, energy, performance	Cost, energy, performance, graphics	Throughput, availability, Scalability, energy	Price-performance, energy-proportionality	Price, energy, performance

Parallelism

- Application parallelism:
  - Data-Level Parallelism (DLP)
  - Task-Level Parallelism (TLP)

- Architectural parallelism exploits application parallelism:
  - Instruction-Level Parallelism (ILP) → pipelining, speculative execution.
  - Vector architectures/Graphic Processor Units (GPUs) →exploit DLP in SIMD architectures.
  - Thread-Level Parallelism → DLP or TLP of interacting threads.
  - Request-Level Parallelism→ parallelism among decoupled tasks.

Michel Flynn's taxonomy

- *SISD* - Single Instruction stream, Single Data stream
- *SIMD* - Single Instruction stream, Multiple Data streams
  - Vector architectures
  - Multimedia extensions
  - Graphics processor units
- *MIMD* - Multiple Instruction streams, Multiple Data streams
  - Tightly-coupled MIMD
  - Loosely-coupled MIMD
- *MISD* - Multiple Instruction streams, Single Data stream
  - No commercial implementation

Defining computer architecture

- *Old* view of computer architecture:
  - Instruction Set Architecture (ISA) design
  - Decisions regarding:
    - registers, memory addressing,
    - addressing modes,
    - instruction operands,
    - available operations,
    - control flow instructions,
    - instruction encoding.

- *Real* computer architecture:
  - Specific requirements of the target machine
  - Design to maximize performance within constraints: cost, power, and availability
  - Includes ISA, microarchitecture, hardware

MIPS instruction format

- **R-instructions** → all data values are in registers

  OPCODE rd,rs,rt Example: add $s1, $s2, $s3

  rd- destination register

  rs, rt – source registers

- **I-instructions** → operate on an immediate value and a register value. Immediate values may be a maximum of 16 bits long.

  OPCODE rs,rt,Imm

- **J-instructions** → used to transfer control

  OPCODE label

- **FR- instructions** → similar to R-instruction but operating of floating point

  OPCODE fmt,fs,ft,fd,funct

- **FI- instructions** → similar to I-instruction but operating of floating point

  OPCODE fmt,ft,Imm

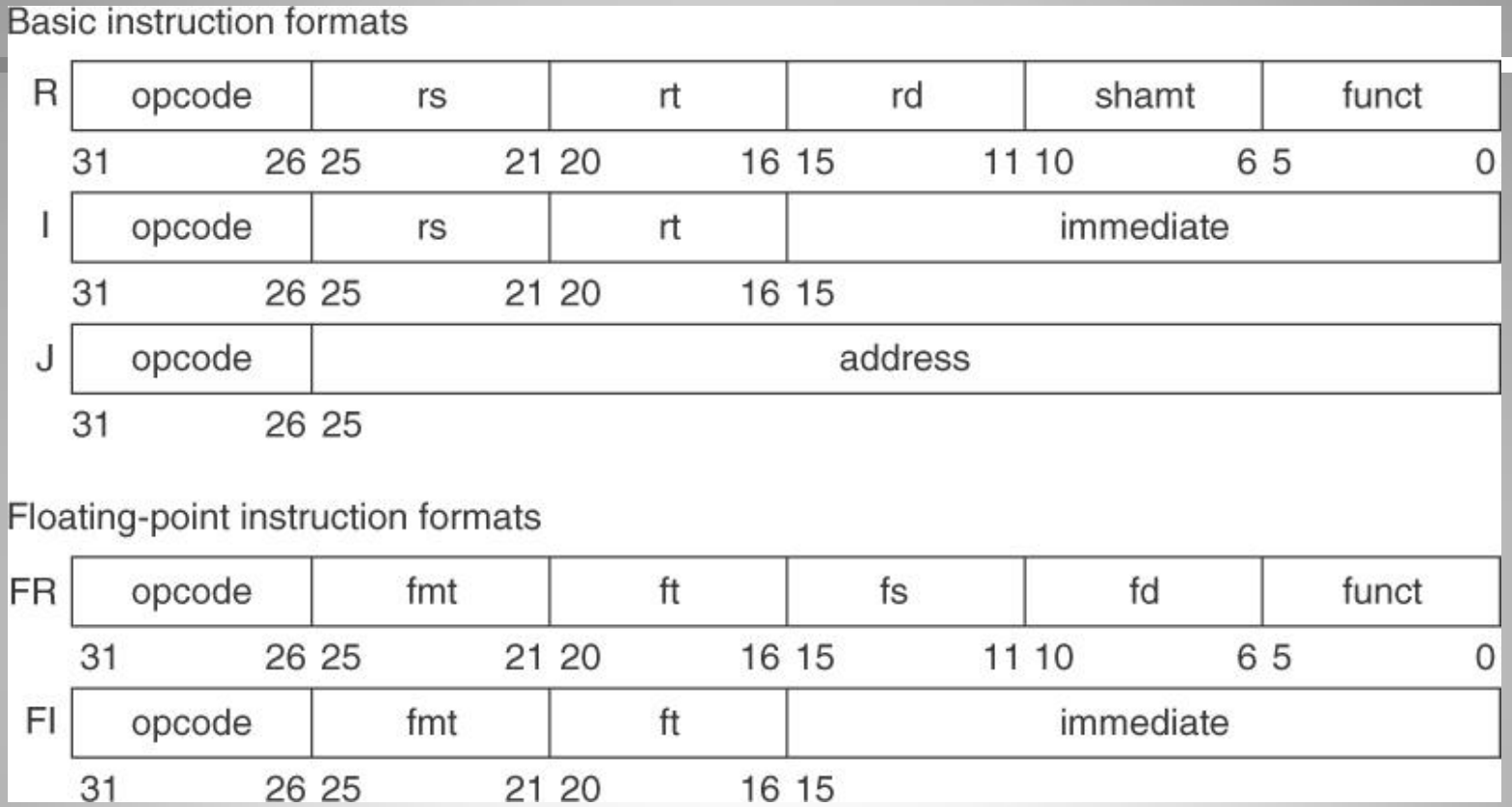


Figure 1.6 MIPS64 instruction set architecture formats. All instructions are 32 bits long. The R format is for integer register-to-register operations, such as DADDU, DSUBU, and so on. The I format is for data transfers, branches, and immediate instructions, such as LD, SD, BEQZ, and DADDIs. The J format is for jumps, the FR format for floating-point operations, and the FI format for floating-point branches.

Mnemonic	Meaning	Type	Opcode	Funct
add	Add	R	0x00	0x20
addi	Add Immediate	I	0x08	NA
addiu	Add Unsigned Immediate	I	0x09	NA
addu	Add Unsigned	R	0x00	0x21
and	Bitwise AND	R	0x00	0x24
andi	Bitwise AND Immediate	I	0x0C	NA
beq	Branch if Equal	I	0x04	NA
bne	Branch if Not Equal	I	0x05	NA
div	Divide	R	0x00	0x1A
divu	Unsigned Divide	R	0x00	0x1B
j	Jump to Address	J	0x02	NA
jal	Jump and Link	J	0x03	NA
jr	Jump to Address in Register	R	0x00	0x08
lbu	Load Byte Unsigned	I	0x24	NA
lhu	Load Halfword Unsigned	I	0x25	NA
lui	Load Upper Immediate	I	0x0F	NA
lw	Load Word	I	0x23	NA
mfhi	Move from HI Register	R	0x00	0x10
mflo	Move from LO Register	R	0x00	0x12
mfc0	Move from Coprocessor 0	R	0x10	NA
mult	Multiply	R	0x00	0x18
multu	Unsigned Multiply	R	0x00	0x19
nor	Bitwise NOR (NOT-OR)	R	0x00	0x27
xor	Bitwise XOR (Exclusive-OR)	R	0x00	0x26
or	Bitwise OR	R	0x00	0x25
ori	Bitwise OR Immediate	I	0x0D	NA
sb	Store Byte	I	0x28	NA
sh	Store Halfword	I	0x29	NA
slt	Set to 1 if Less Than	R	0x00	0x2A
slti	Set to 1 if Less Than Immediate	I	0x0A	NA
sltiu	Set to 1 if Less Than Unsigned Immediate	I	0x0B	NA
sltu	Set to 1 if Less Than Unsigned	R	0x00	0x2B
sll	Logical Shift Left	R	0x00	0x00
srl	Logical Shift Right (0-extended)	R	0x00	0x02
sra	Arithmetic Shift Right (sign-extended)	R	0x00	0x03
sub	Subtract	R	0x00	0x22
subu	Unsigned Subtract	R	0x00	0x23
sw	Store Word	I	0x2B	NA

Computer implementation

- Organization / microarchitecture → high-level aspects of computer design including:
  - Memory system
  - Memory interconnect
  - CPU
- Hardware → detailed logic design and the packaging technology.

Technology improvement rate per year

- Integrated circuit
  - Transistor density: 35% (Moore's law)
  - Die size: 10-20%
  - Integration overall: 40-55%
- DRAM capacity: 25-40% (slowing)
- Flash capacity: 50-60%
  - 15-20X cheaper/bit than DRAM
- Magnetic disk: 40%
  - 15-25X cheaper/bit then Flash
  - 300-500X cheaper/bit than DRAM

Flash memory

- Flash memory - *electronic non-volatile storage medium that can be electrically erased and reprogrammed*.
- NAND flash memory
  - May be written and read in blocks (or pages) which are generally much smaller than the entire device.
  - Used in main memory, memory cards, USB flash drives, solid-state drives for general storage and transfer of data.
- NOR flash memory
  - Allows a single machine word (byte) to be written—to an erased location—or read independently.
  - Allows true random access and therefore direct code execution

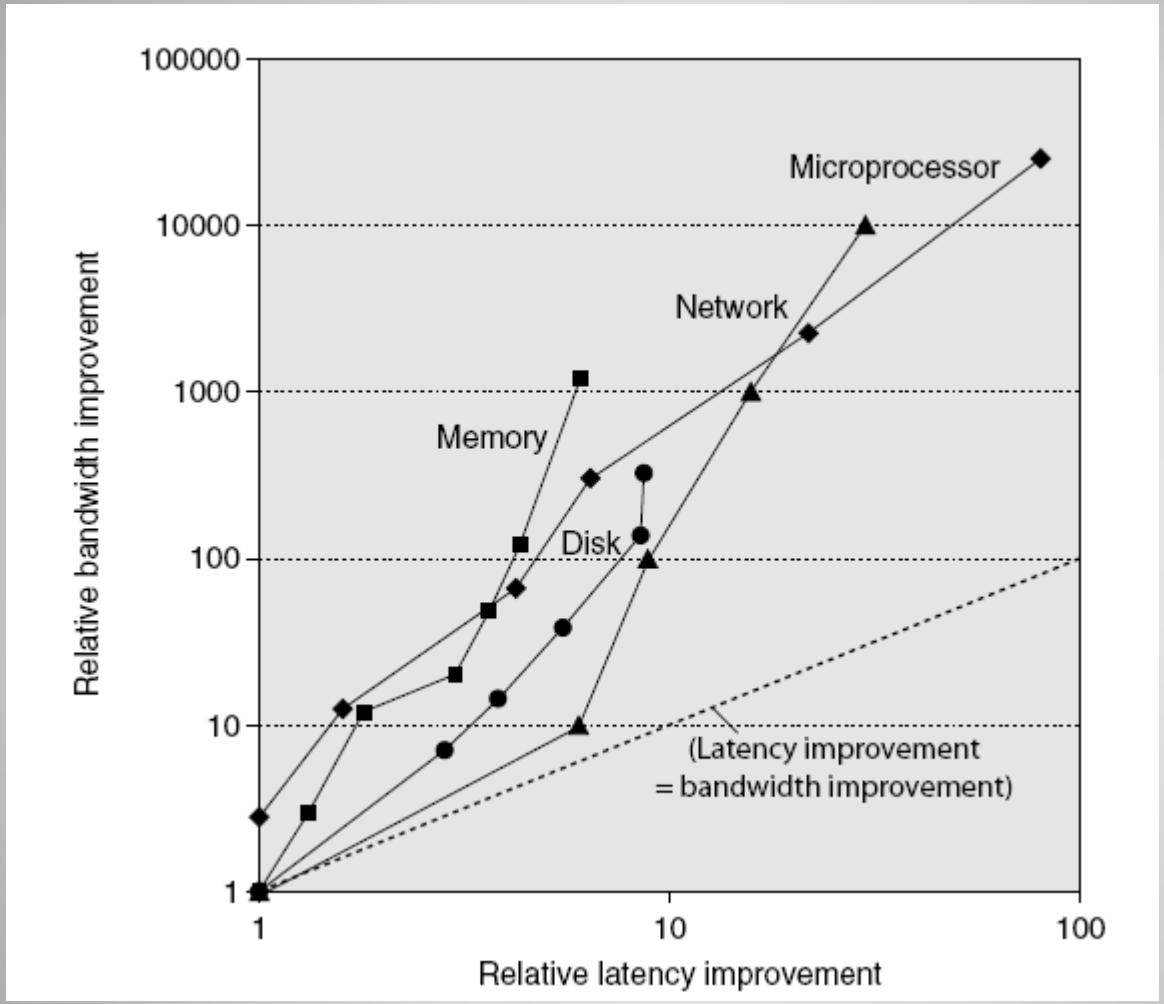
DRAM – dynamic random-access memory

- Stores *each bit in a separate capacitor within an integrated circuit*. The capacitor can be either charged or discharged; these two states are taken to represent the two values of a bit, 0 and 1.
- Dynamic, as opposite to SRAM (static RAM)→*needs to be periodically refreshed as capacitors leak charge*.
- Structural simplicity: *only one transistor and a capacitor are required per bit*, compared to four or six transistors in SRAM. This allows DRAM to reach very high densities.
- Unlike flash memory, *DRAM is volatile memory* since it loses its data quickly when power is removed.

Evolution of bandwidth and latency

- Bandwidth or throughput ➔total work done in a given time
  - improvement for processors →10,000 - 25,000 times
  - improvement for memory and disks → 300 - 1,200 times
- Latency or response time ➔ time between start and completion of an operation
  - improvement for processors → 30 - 80 times
  - improvement for memory and disks → 6 - 8 times
- *Processors have improved at a much faster rate than memory and disks.!!*

Bandwidth and latency



Log-log plot of bandwidth and latency milestones

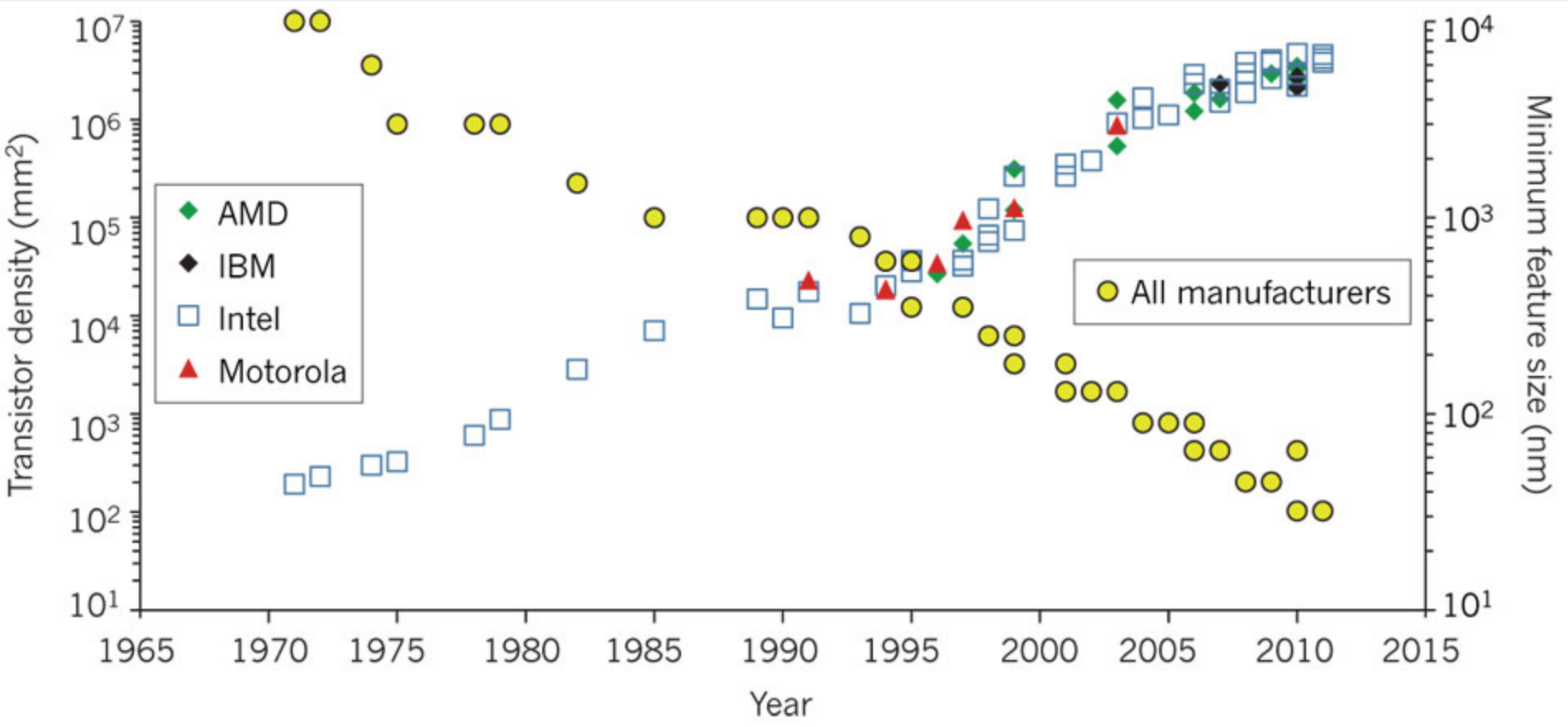
Feature size of transistors and wires

- Feature size ➔Minimum size of transistor or wire in x or y dimension
  - 10.0 microns in 1971
  - 0.32 microns in 2011
- *Transistor performance scales linearly with feature size*.
- *Wire delay does not improve with feature size!*
- Integration density scales quadratically

Moore's Law

- The number of transistors in a dense integrated circuit doubles approximately every two years, 18 months to be exact.
- Gordon E. Moore, co-founder of Intel Corporation, who described the trend in a 1965 paper.
- His prediction has proven to be accurate and the law is now used in the semiconductor industry to guide long-term planning and to set targets for research and development

Feature size of transistors and wires (cont'd)



Between 1970 and 2011, the gate length of MOSFETs shrank from 10 μm to 28 nm (yellow circles; *y* axis, right), and the number of transistors per square millimetre increased from 200 to over 1 million (diamonds, triangles and squares show data for the four main microprocessor manufacturers; *y* axis, left). AMD, Advanced Micro Devices; IBM, International Business Machines.

Application: questions related to Moore's law

(a) The number of transistors on a chip in 2015 should be how many times the number in 2005 based on Moore's law?

(b) In the 90s the increase in clock rate once mirrored the trend. Had the clock rate continued to climb at the same rate fast would the clock rate be in 2015?

(c) At the current rate of increase what are the clock rates projected to be in 2015?

(d) What has limited the growth of the clock rate and what are architects doing with the extra transistors to increase performance?

(e) The rate of growth of DRAM capacity has also slowed down. For 20 years it increased by 60%/year. It dropped to 40%/year and now is in the 25-40%/year . If this trend continues what will be this rate in 2020?

Answers

a. $(1.35)^{10}$ = approximately 20

b. $3200 \times (1.4)^{12}$ = approximately 181,420

c. $3200 \times (1.01)^{12}$ = approximately 3605

d. Power density, which is the power consumed over the increasingly small area, has created too much heat for heat sinks to dissipate. This has limited the activity of the transistors on the chip. Instead of increasing the clock rate, manufacturers are placing multiple cores on the chip.

e. Anything in the 15–25% range would be a reasonable conclusion based on the decline in the rate over history. As the sudden stop in clock rate shows, though, even the declines do not always follow predictions.

Power and energy

- Problem: Get power in, get power out

Power is the rate at which the energy has been converted every second.

- Thermal Design Power (TDP)
  - Characterizes sustained power consumption
  - Used as target for power supply and cooling system
  - Lower than peak power, higher than average power consumption

- Clock rate can be reduced dynamically to limit power consumption

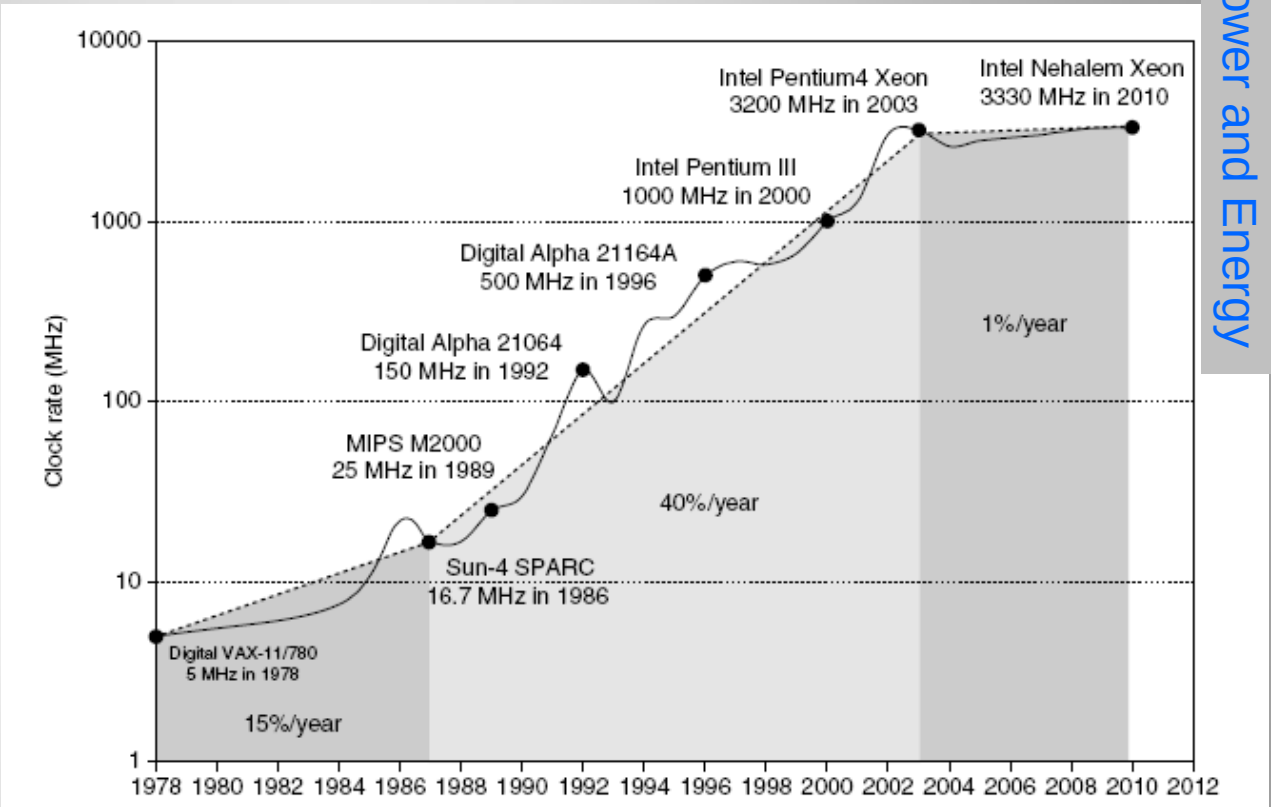
- Energy per task is often a better measurement

Dynamic energy and power

- Dynamic energy – energy to switch the transistor state
  - Transistor switch from 0 → 1 or 1 → 0
  - ½ x Capacitive load x Voltage^2
- Dynamic power – power to switch the transistor state
  - ½ x Capacitive load x Voltage^2 x Frequency switched
- Reducing clock rate reduces power, not energy

Processor power consumption

- Intel 80386 consumed ~ 2 W
- 3.3 GHz Intel Core i7 consumes 130 W
- Heat must be dissipated from 1.5 x 1.5 cm chip
- This is the limit of what can be cooled by air
- Dramatic increase in power consumption!!



Techniques for reducing power

- Do nothing well
- Dynamic Voltage-Frequency Scaling (DVFS)
- Low power state for DRAM, disks
- Over-clocking, turning off cores

Power consumption

- Static power consumption:
  - I x V (Static current x Voltage)
  - Scales with number of transistors
- **Power gating** ➔ technique used in integrated circuit design to reduce power consumption, by shutting off the electric current to blocks of the circuit that are not in use.
- **Clock gating** ➔ technique used in many synchronous circuits for reducing dynamic power dissipation. It saves power by adding more logic to a circuit to disable portions of the circuitry so that the flip-flops in them do not have to switch states. Switching states consumes power. The switching power consumption goes to zero, and only leakage currents are incurred

Trends in cost

- Cost driven down by yield learning curve
- Yield → the ratio of the number of products that can be sold to the number of products that can be manufactured.
- Estimated typical cost of modern 300 mm or 12 inch wafer 0.13 nm process fabrication plant is $2-4 billion. Typical number of processing steps for a modern integrated circuit is more than 150. Typical production cycle-time is over 6 weeks. Individual wafers cost multiple thousands of dollars. Given such huge investments, consistent high yield is necessary for faster time to profit.
- DRAM: price closely tracks cost
- Microprocessors: price depends on volume
  - 10% less for each doubling of volume

Integrated circuit cost

- Integrated circuit

$$\text{Cost of integrated circuit} = \frac{\text{Cost of die} + \text{Cost of testing die} + \text{Cost of packaging and final test}}{\text{Final test yield}}$$

$$\text{Cost of die} = \frac{\text{Cost of wafer}}{\text{Dies per wafer} \times \text{Die yield}}$$

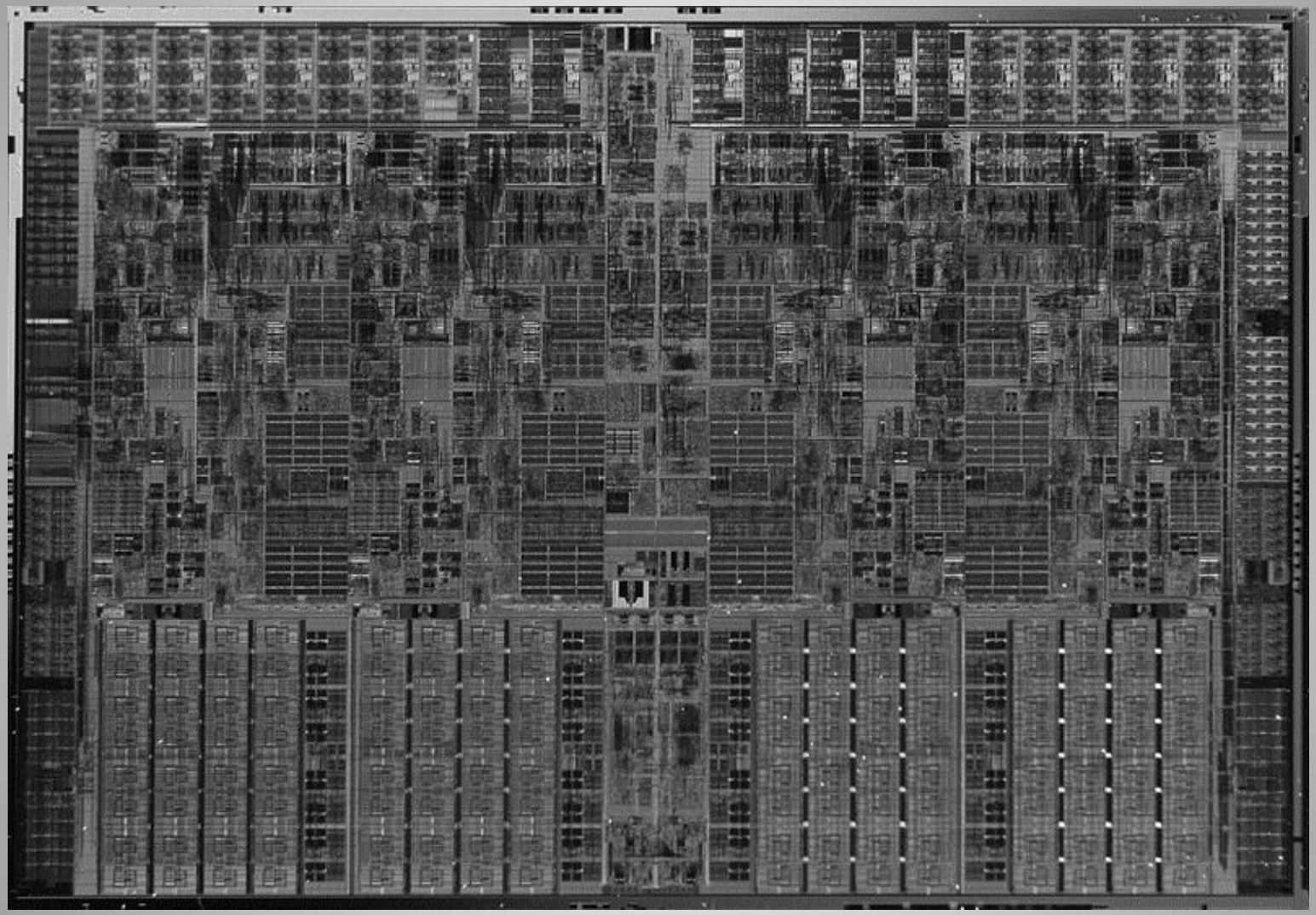
$$\text{Dies per wafer} = \frac{\pi \times (\text{Wafer diameter}/2)^2}{\text{Die area}} - \frac{\pi \times \text{Wafer diameter}}{\sqrt{2 \times \text{Die area}}}$$

- Bose-Einstein formula:

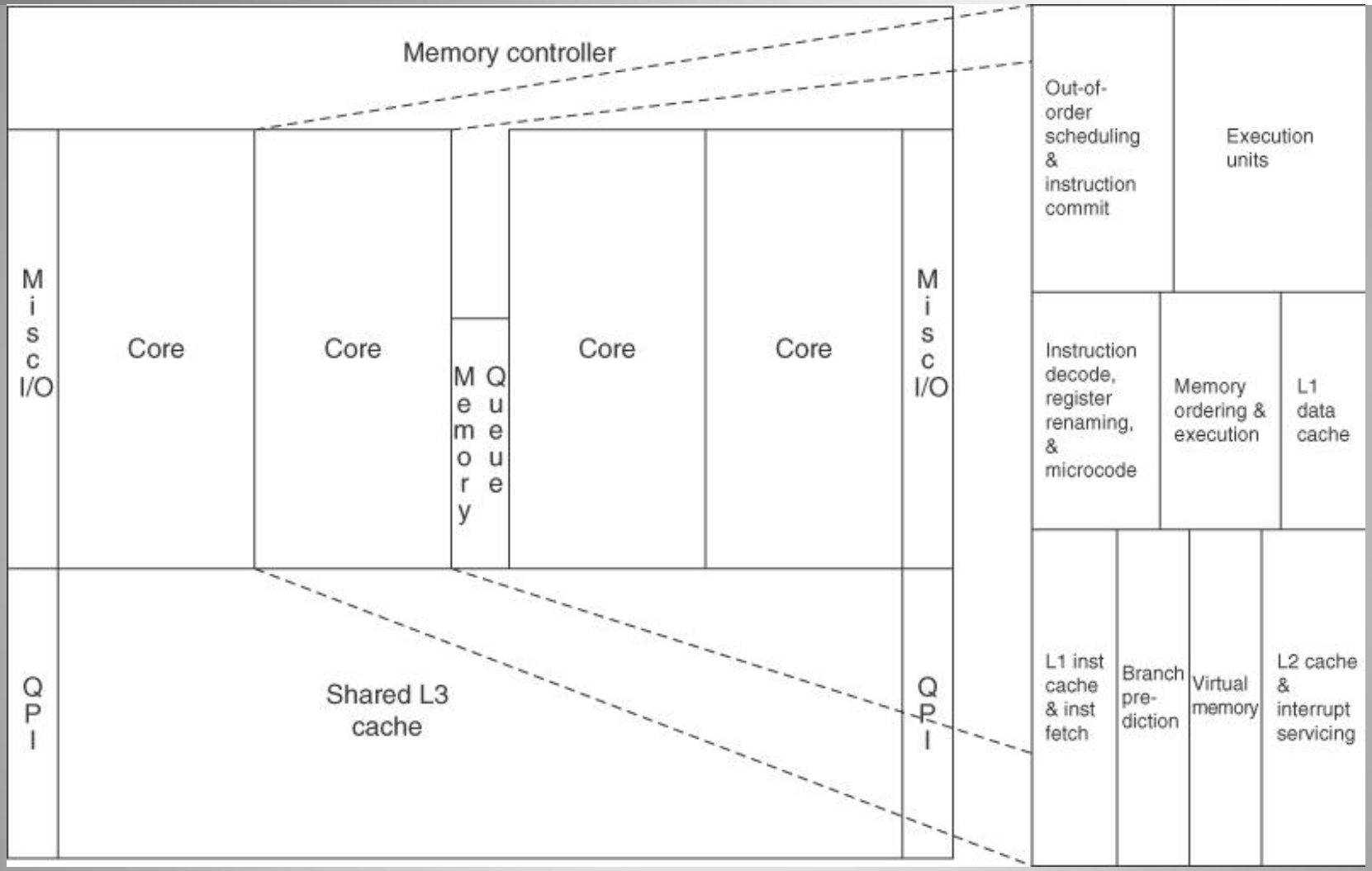
$$\text{Die yield} = \text{Wafer yield} \times 1/(1 + \text{Defects per unit area} \times \text{Die area})^N$$

- Defects per unit area = 0.016-0.057 defects per square cm (2010)
- N = process-complexity factor = 11.5-15.5 (40 nm, 2010)

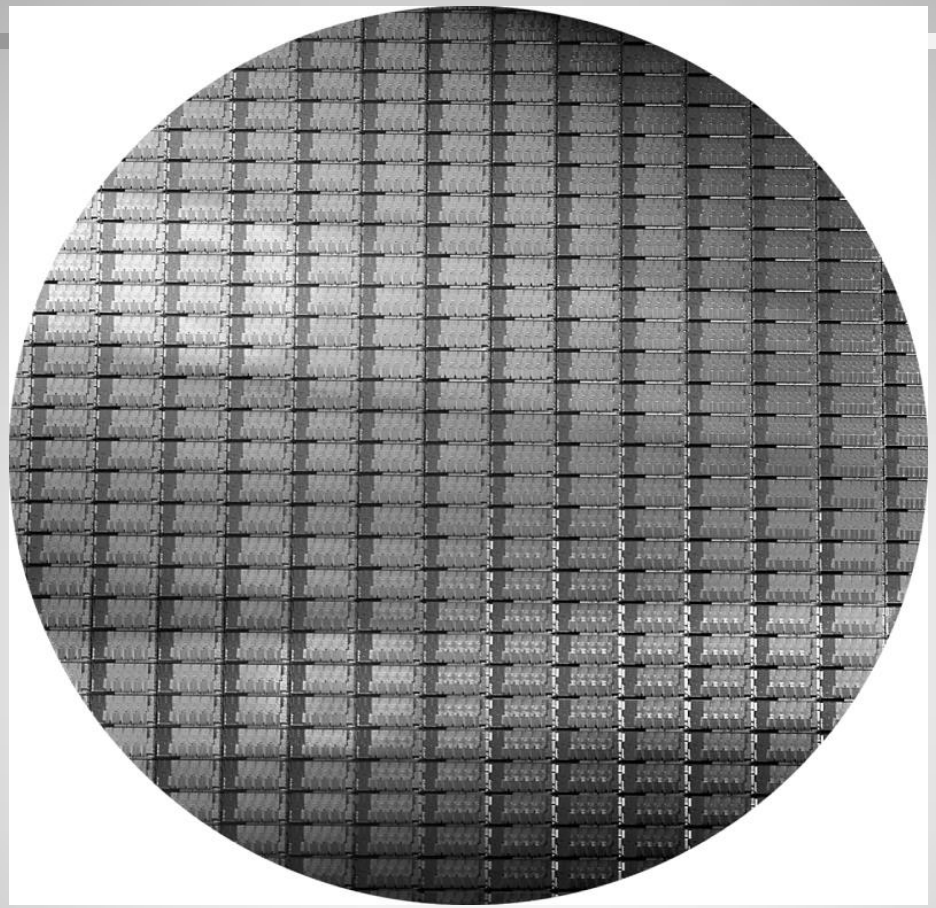
Intel i7 microprocessor



Left - floor plan of Core i7; Right - floor plan of second core



- QPI→ Quick Path Interconnect



This 300 mm wafer contains 280 full Sandy Bridge dies, each 20.7 by 10.5 mm in a 32 nm process. (Sandy Bridge is Intel's successor to Nehalem used in the Core i7.) At 216 mm2, the formula for dies per wafer estimates 282. (Courtesy Intel.)

Case study – chip fabrication costs

	Die size (mm^2)	Estimated defect rate per(cm^2)	Manufacturing size (nm)	Transistors (millions)
IBM Power 5	389	0.3	130	276
Sun Niagara	380	0.75	90	279
AMD Opteron	199	0.75	90	233

Problem

- a. What is the yield for IBM Power 5?
- b. Why does IBM Power 5 have a lower defect rate?

$$\text{Die yield} = \text{Wafer yield} \times 1/(1 + \text{Defects per unit area} \times \text{Die area})^{N}$$

a. $$\text{Yield} = \left(1 + \frac{0.30 \times 3.89}{4.0}\right)^{-4} = 0.36$$

b. It is fabricated in a larger technology, which is an older plant. As plants age, their process gets tuned, and the defect rate decreases.

- Notes: We assumed that the wafer yield is 100/%, no wafers are bad
- N is the process complexity factor. For the 40 nm process it is in the range 11.5 – 15.5. For the 130 nm process we took N=4

More questions

- A new facility uses a fabrication identical with the one for the Power 5 and produces two chips from 300 mm wafers:
  - Woods : 150 mm^2 ; the profit is $20/defect-free chip.
  - Markon: 250 mm^2 ; the profit is $25/defect-free chip
- How much profit can be made for (a) Woods; (b) Markon?
- (c) Which chip should be produced at the new facility?
- (d) If the demand is 50,000 Woods and 25,000 Mackron chips/month and you can fabricate 150 wafers/month , how many wafers should be made for each chip?

a. Dies per wafer $= \frac{\pi \times (30/2)^2}{1.5} - \frac{\pi \times 30}{\text{sqrt}(2 \times 1.5)} = 471 - 54.4 = 416$

$$\text{Yield} = \left(1 + \frac{0.30 \times 1.5}{4.0}\right)^{-4} = 0.65$$

$$\text{Profit} = 416 \times 0.65 \times \$20 = \$5408$$

b. Dies per wafer $= \frac{\pi \times (30/2)^2}{2.5} - \frac{\pi \times 30}{\text{sqrt}(2 \times 2.5)} = 283 - 42.1 = 240$

$$\text{Yield} = \left(1 + \frac{0.30 \times 2.5}{4.0}\right)^{-4} = 0.50$$

$$\text{Profit} = 240 \times 0.50 \times \$25 = \$3000$$

c. The Woods chip

d. Woods chips: 50,000/416 = 120.2 wafers needed
Markon chips: 25,000/240 = 104.2 wafers needed

Therefore, the most lucrative split is 120 Woods wafers, 30 Markon wafers.

Dependability

- Module reliability
  - Mean time to failure (MTTF)
  - Mean time to repair (MTTR)
  - Mean time between failures (MTBF) = MTTF + MTTR
  - Availability = MTTF / MTBF

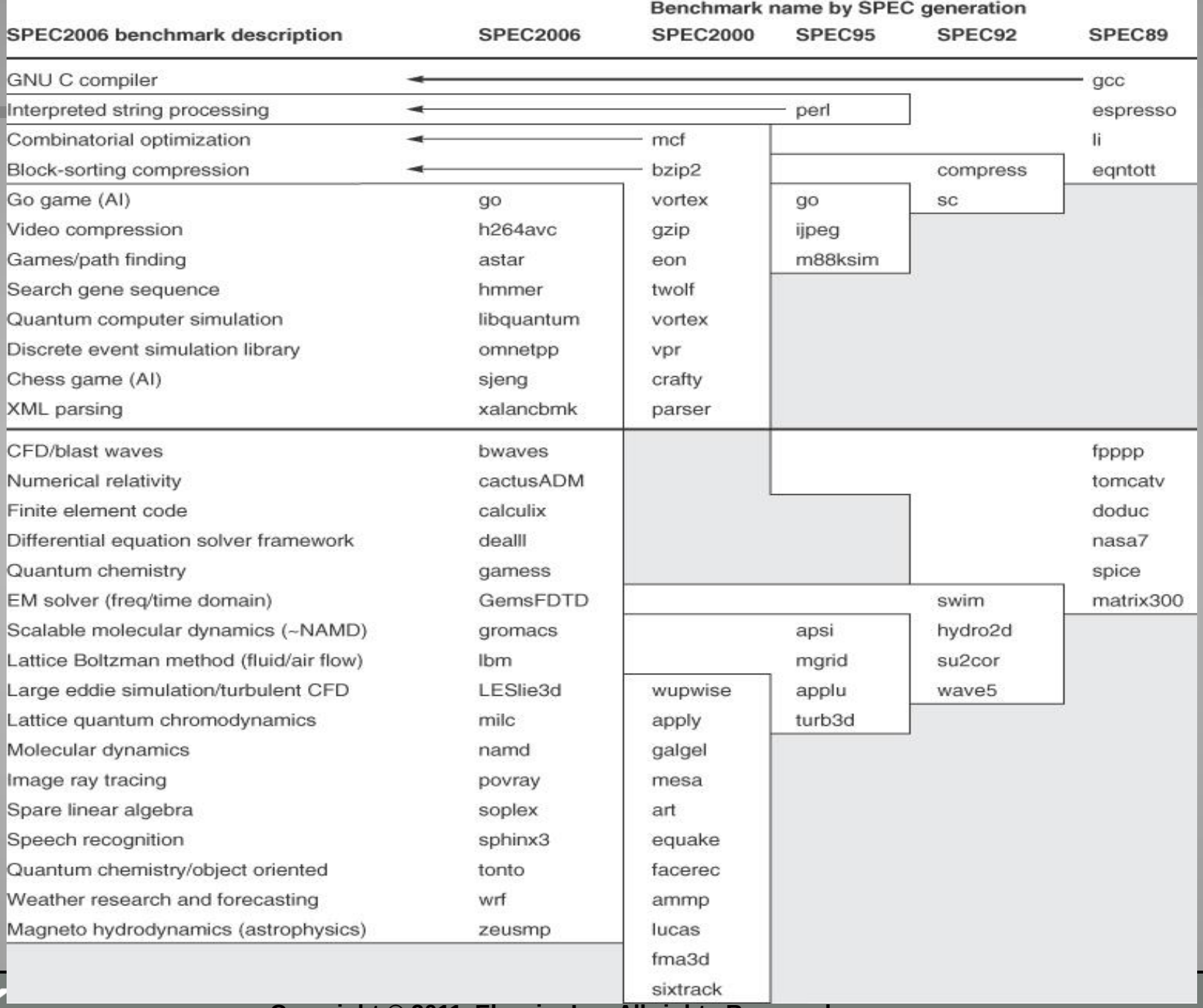
Measuring performance

- Typical performance metrics:
  - Response time
  - Throughput
- Speedup of X relative to Y
  - Execution time$_Y$ / Execution time$_X$
- Execution time
  - Wall clock time: includes all system overheads
  - CPU time: only computation time
- Benchmarks
  - Kernels (e.g. matrix multiply)
  - Toy programs (e.g. sorting)
  - Synthetic benchmarks (e.g. Dhrystone)
  - Benchmark suites (e.g. SPEC06fp, TPC-C)

Evolution of benchmarks over time

- Of the 12 SPEC2006 integer programs, 9 are written in C, and the rest in C++.
- For the floating-point programs, the split is 6 in Fortran, 4 in C++, 3 in C, and 4 in mixed C and Fortran.

- SPEC2006 programs and the evolution of the SPEC benchmarks over time, with integer programs above the line and floating-point programs below the line. The figure shows all 70 of the programs in the 1989, 1992, 1995, 2000, and 2006 releases.
- The benchmark descriptions on the left are for SPEC2006 only and do not apply to earlier versions. Programs in the same row from different generations of SPEC are generally not related; for example, fpppp is not a CFD code like bwaves. Gcc is the senior citizen of the group. Only 3 integer programs and 3 floating-point programs survived three or more generations. Note that all the floating-point programs are new for SPEC2006.
- Although a few are carried over from generation to generation, the version of the program changes and either the input or the size of the benchmark is often changed to increase its running time and to avoid perturbation in measurement or domination of the execution time by some factor other than CPU time.



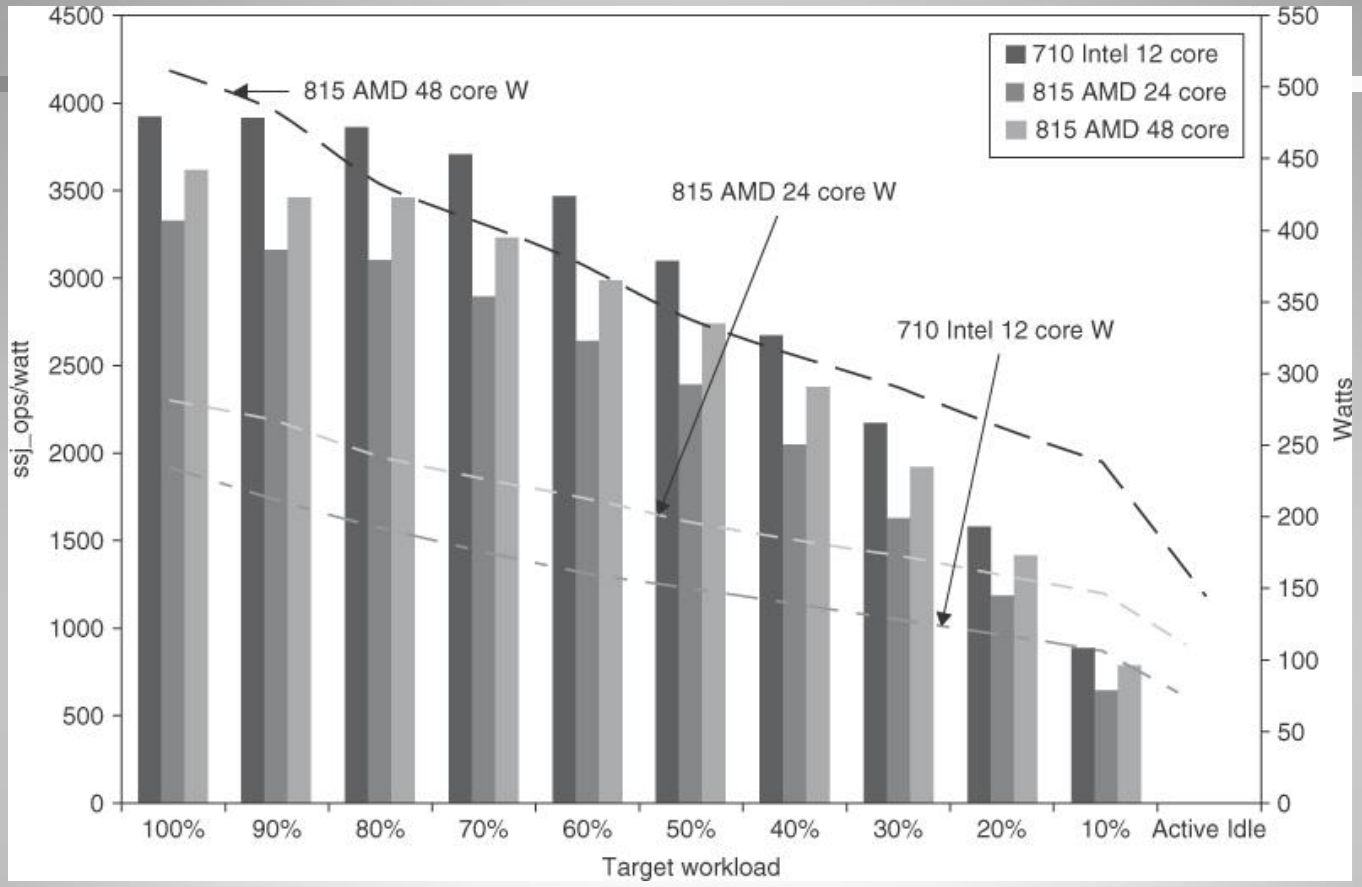


Figure 1.19 Power-performance of the three servers in Figure 1.18. Ssj_ops/watt values are on the left axis, with the three columns associated with it, and watts are on the right axis, with the three lines associated with it. The horizontal axis shows the target workload, as it varies from 100% to Active Idle. The Intel-based R715 has the best ssj_ops/watt at each workload level, and it also consumes the lowest power at each level.

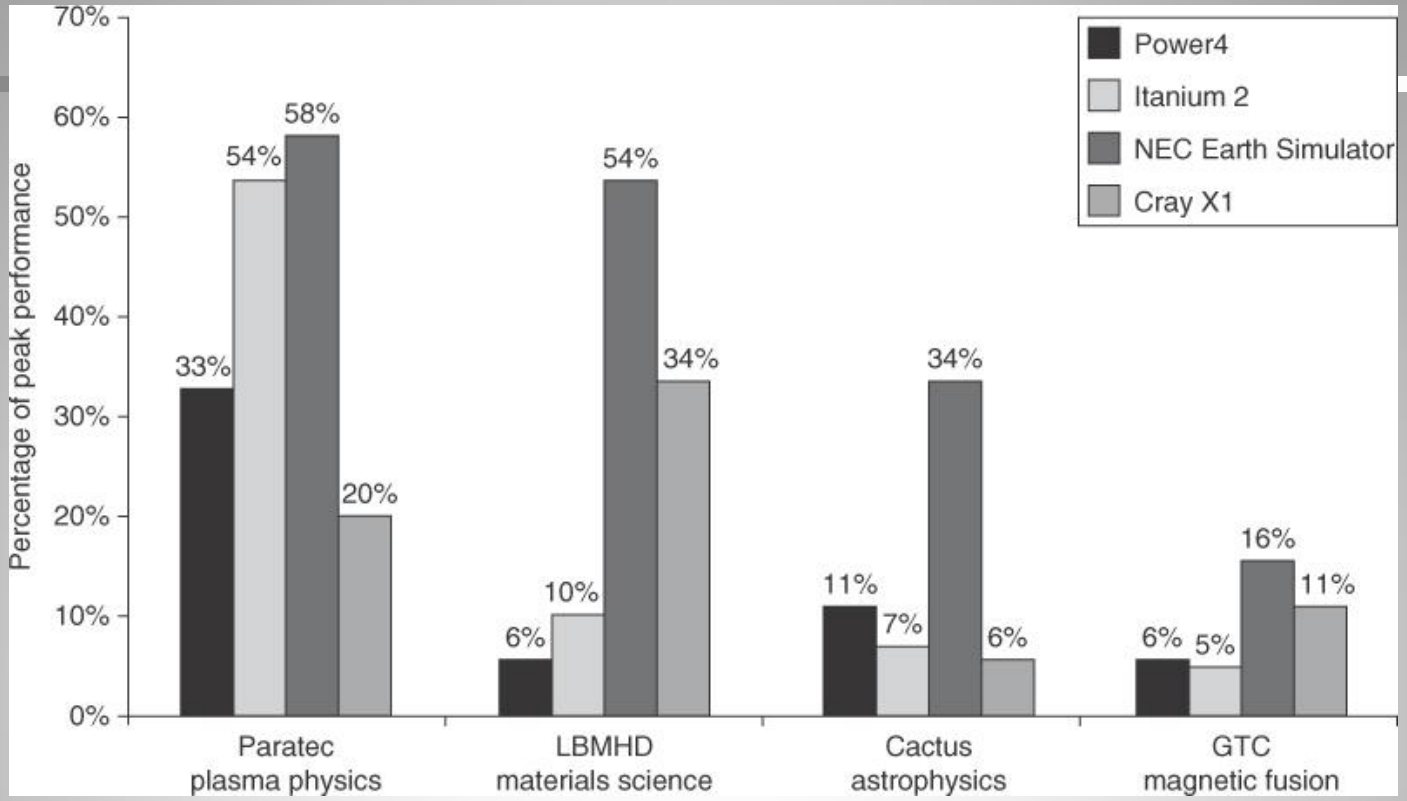


Figure 1.20 Percentage of peak performance for four programs on four multiprocessors scaled to 64 processors. The Earth Simulator and X1 are vector processors (see Chapter 4 and Appendix G). Not only did they deliver a higher fraction of peak performance, but they also had the highest peak performance and the lowest clock rates. Except for the Paratec program, the Power 4 and Itanium 2 systems delivered between 5% and 10% of their peak. From Oliker et al. [2004].

Principles of computer design

- Take Advantage of Parallelism
  - e.g. multiple processors, disks, memory banks, pipelining, multiple functional units
- Principle of Locality
  - Reuse of data and instructions
- Focus on the Common Case
  - Amdahl's Law

$$\text{Execution time}_{\text{new}} = \text{Execution time}_{\text{old}} \times \left((1 - \text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}} \right)$$

$$\text{Speedup}_{\text{overall}} = \frac{\text{Execution time}_{\text{old}}}{\text{Execution time}_{\text{new}}} = \frac{1}{(1 - \text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}}$$

The processor performance equation

$$\text{CPU time} = \text{CPU clock cycles for a program} \times \text{Clock cycle time}$$

$$\text{CPU time} = \frac{\text{CPU clock cycles for a program}}{\text{Clock rate}}$$

$$\text{CPI} = \frac{\text{CPU clock cycles for a program}}{\text{Instruction count}}$$

$$\text{CPU time} = \text{Instruction count} \times \text{Cycles per instruction} \times \text{Clock cycle time}$$

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

Different instruction types have different CPIs

$$\text{CPU clock cycles} = \sum_{i=1}^{n} \text{IC}_i \times \text{CPI}_i$$

$$\text{CPU time} = \left(\sum_{i=1}^{n} \text{IC}_i \times \text{CPI}_i \right) \times \text{Clock cycle time}$$

Fallacies

- Multiprocessors are a silver bullet ➔ *to improve performance replace a high-clock rate single core with multiple lower-clock-rate, efficient cores. The burden is now on application developers to exploit parallelism.*
- *Increasing performance improves energy efficiency.*
- Benchmarks remain valid indefinitely ➔ *almost 70% of the original kernels in the SPEC2000 or earlier were dropped.*
- Accuracy of reported MTTF ➔ *the MTTF of disks as currently reported is almost 140 years!!*
- Peak performance tracks observed performance ➔ *peak performance of different programs on the same processor varies widely.*

Pitfalls

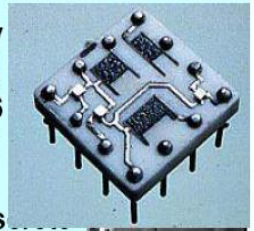
- Ignoring Amdahl's law
  - Optimize a feature before measuring its usage.
  - Dependability depends on the weakest link
- Fault detection can lower availability
  - Some errors, e.g., an error in the branch predictor, could lower the performance but not the availability.

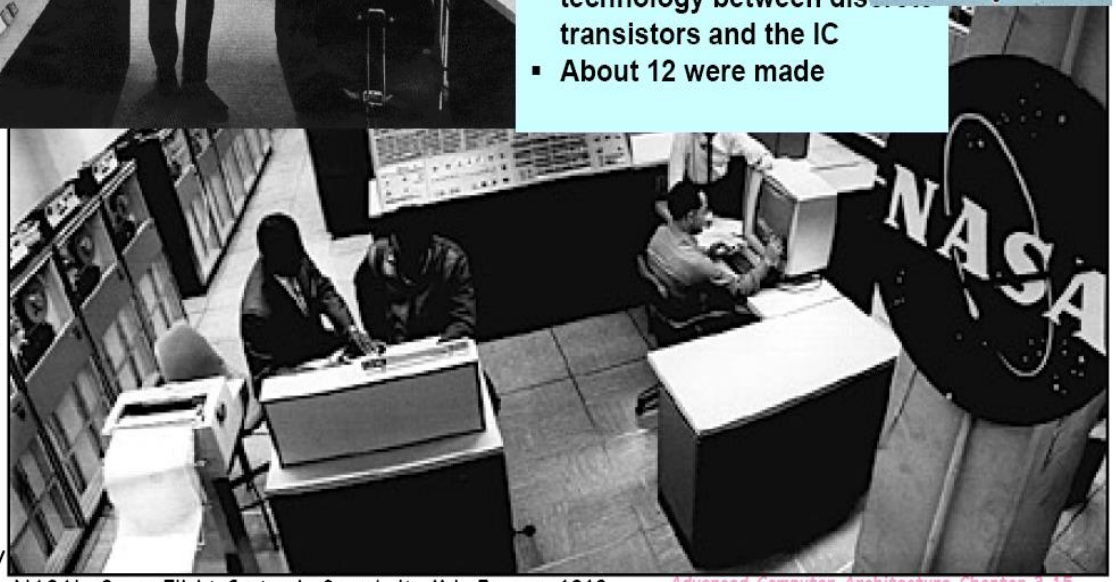
Supercomputers of the late 1960s - IBM 360/91

- Launched in January 1968. Installed at NASA Aimes.
- Primary memory - up to 6 MB interleaved 16 ways.
- Secondary memory – 300 MB (two IBM 2301 drum and 2 IBM 2314 disks).
- The CPU had five highly autonomous execution units:
  - processor storage,
  - storage bus control,
  - instruction processor,
  - fixed-point processor and
  - floating-point processor.
- Only four floating point registers.
- Tomasulo's algorithm for register renaming in 360/91 used in many modern processors for exploiting Instruction Level Parallelism (ILP).

IBM360/91

- CPU cycle time: 60 nanoseconds
- memory cycle time (to fetch and store eight bytes in parallel): 780ns
- Standard memory capacity: 2,097,152B interleaved 16 ways (magnetic cores)
- Up to 6,291,496 bytes of main storage
- Up to 16.6-million additions/second
- Ca.120K gates, ECL
- Solid Logic Technology (SLT), an IBM invention which encapsulated 5-6 transistors into a small module--a transition technology between dis... transistors and the IC
- About 12 were made





NASA Center for Computational Sciences

See:
Some Reflections on Computer Engineering: 30 Years after the IBM System 360 Model 91
Michael J. Flynn
ftp://arith.stanford.edu/tr/micro30.ps.Z

Source:
http://www.columbia.edu/acis/history/36091.html

NASA's Space Flight Center in Greenbelt, Md, January 1968

Supercomputers of late 1960s – CDC 7600

- Designed by Seymour Cray.
  - RISC architecture with a 15-bit instruction word containing a six-bit operation code. Only 64 machine codes; no fixed-point arithmetic in the central processor.
  - Pipelined execution - 10-word instruction stack. All addresses in the stack are fetched, without waiting for the instruction field to be processed.
  - Ten 60-bit read registers and ten 60-bit write registers, each with an address register.
- Clock rate 36.4 MHz (27.5 ns clock cycle). Could deliver about 10 MFLOPS on hand-compiled code, with a peak of 36 MFLOPS.
- 65 Kword primary memory; up to 512 Kword secondary memory.
- Cooled by liquid freon.

Massively parallel systems of the 90s

- Touchstone Delta – prototype developed by Intel in 1990
  - Installed at Caltech for the Concurrent Supercomputer Consortium
  - MIMD architecture with hypercube interconnect; wormhole routing.
  - A node: i860 RISC chip, 60 MFLOPS peak, with 8--16 Mbytes of memory.
  - Peak performance: 32 GFLOPS for a configuration of 484 nodes.
  - LINPACK rating=13.9 GFLOPS; SLALOM benchmark = 5750 patches.
  - Significantly above the Moore curve
- The Paragon
  - Production version of the Touchstone Delta
  - Up to 4,000 nodes
  - A light-weight kernel called SUNMOS
    developed at Sandia National Laboratories
    run on the Paragon's compute processors