1. History, Technology Trends and Performance Metrics

Note :

- Slides not covered in detail in the class are left as a self-learning exercise

Acknowledgement :

- Text by © Patterson & Hennessy and companion materials by © Mary Jane Irwin
- Text by © Harris & Harris and companion materials
- Image copyrights belong to their respective owners. Details available on request

CG3207 Computer Architecture

Rajesh Panicker, ECE, NUS

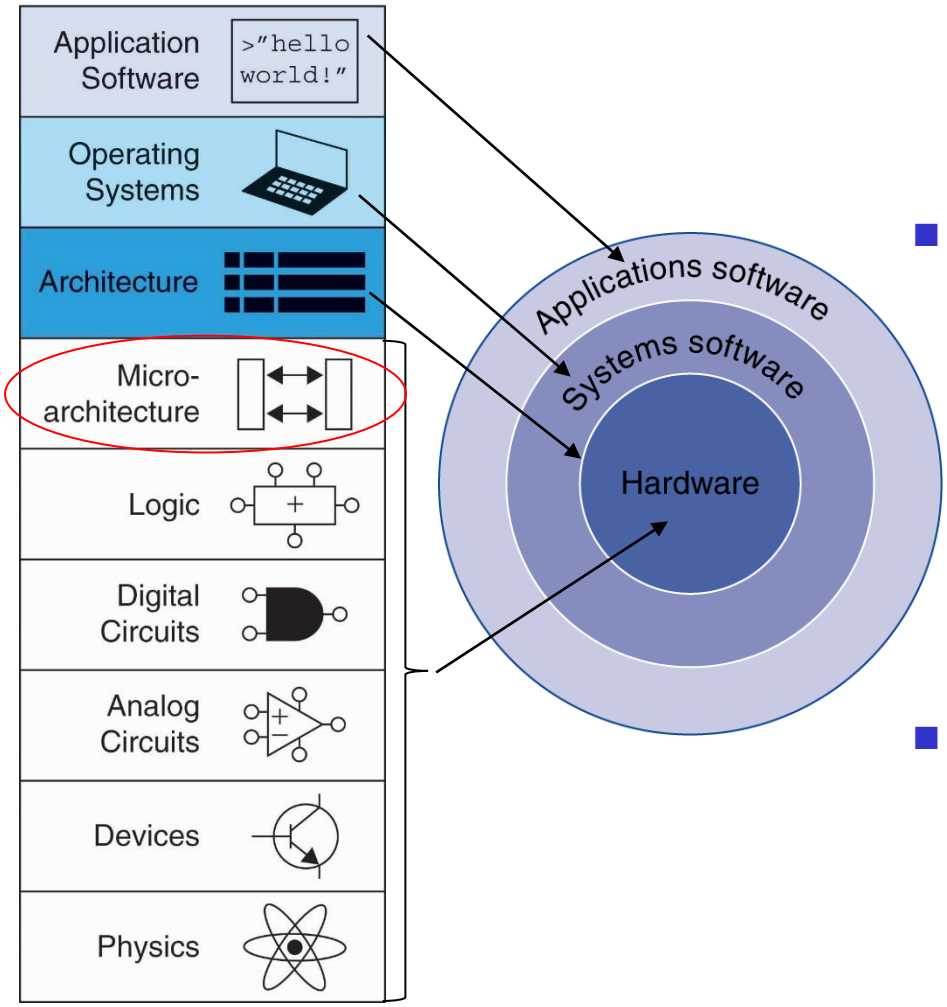
Classes of Computers

- Personal Mobile Device
  - Battery operated
  - Connects to the Internet
  - Hundreds of dollars
  - Smart phones, tablets, electronic glasses
- Embedded / IoT Devices
  - Hidden as components of systems
  - Stringent power/performance/cost constraints
  - Pervasive - Rocket engine control, Car engine, Mobile phone, Set-top box, Industrial control, Air Traffic Control, …..
  - >50 devices per household, increasing – much more than desktops / laptops

Classes of Computers …

- Desktop computers
  - General purpose, variety of software
  - Subject to cost/performance tradeoff
- Server computers
  - Network based
  - High capacity, performance, reliability
  - Range from small servers to building sized
  - Cloud computing
    - Portion of software run on a personal mobile device /embedded computer and a portion run in the Cloud
    - Amazon, Google, Azure

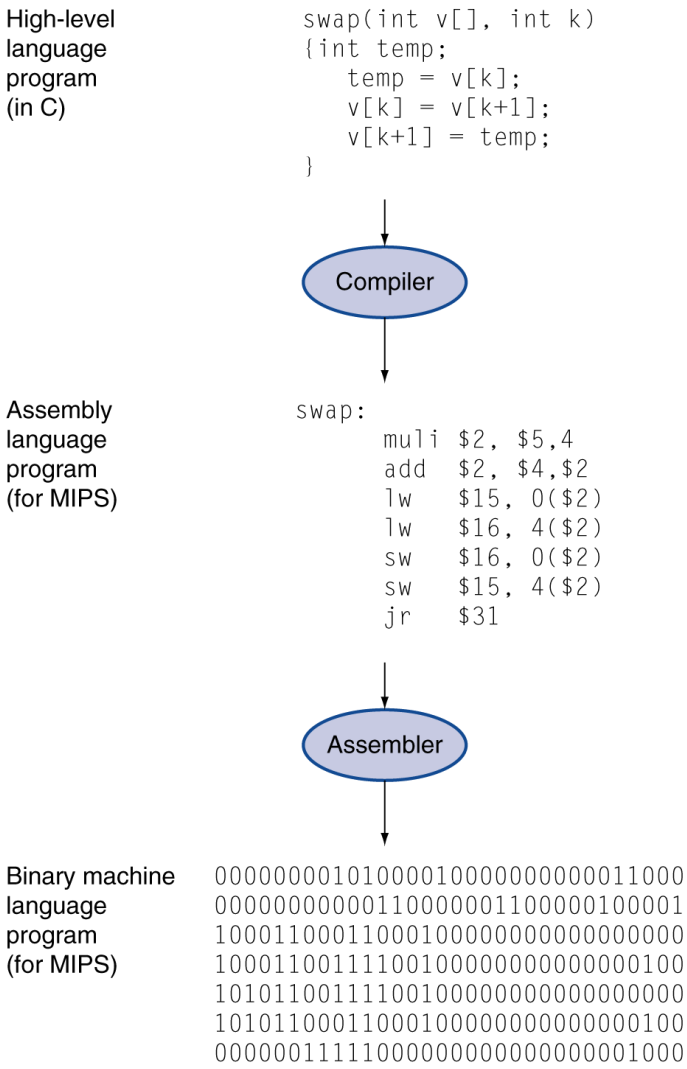
Below Your Program



- Application software
  - Written in high-level language
- System software
  - Operating System: service code
    - Handling input/output
    - Managing memory and storage
    - Scheduling tasks & sharing resources
- Hardware
  - **Processor**, memory, I/O controllers

Levels of Program Code

- High-level language
  - Level of abstraction closer to problem domain
  - Provides for productivity and portability
- Assembly language
  - Textual representation of instructions
- Hardware representation
  - Binary digits (bits)
  - Encoded instructions and data



Instruction Set Architecture (ISA)

- ISA, or simply architecture – the abstract interface between the hardware and the lowest level software that encompasses all the information necessary to write a machine language program, including instructions, registers, memory access, I/O, …
  - Enables implementations of varying cost and performance to run identical software. These implementations are called microarchitectures

    Microarchitecture is often referred to simply as architecture, within a particular ISA.
    ISA = assembly language programmer's / firmware engineer's view of the processor.
    Microarchitecture = hardware engineer's view of the processor.
- The combination of the basic instruction set (the ISA) and the library / operating system interface is called the application binary interface (ABI)
  - An ABI defines how data structures or computational routines (how data is provided as input / parameters to or read as output / return values) are accessed in machine code. Defines a standard for binary portability across computers

To Do : 1) Compare and contrast API and ABI. 2) Read up on 'Calling convention' 3) Which are the popular ISAs?

Digital Hardware Market Segments

- Processor, GPU

- DRAM, Flash memory

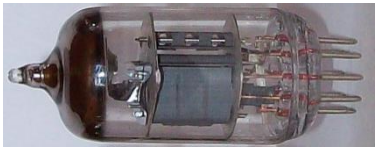
- ASIC (application specific integrated circuit)
  ASSP (application specific standard product)
  FPGA (field programmable gate array)

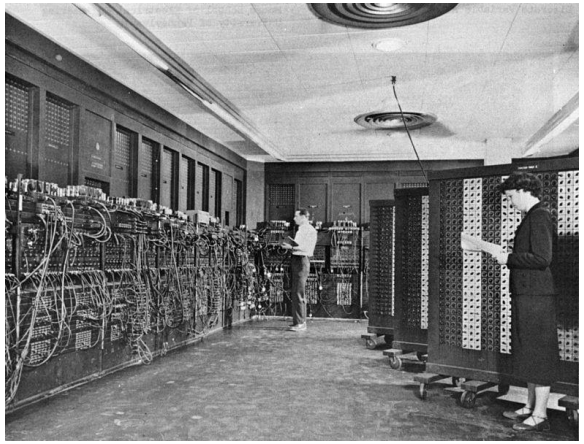
- Convergence as System on Chip (SoC), which may also contain MEMS, analog and radio-frequency functions

History of the Processor

- Vacuum tube

(Lee De Forest, 1906)





- ENIAC

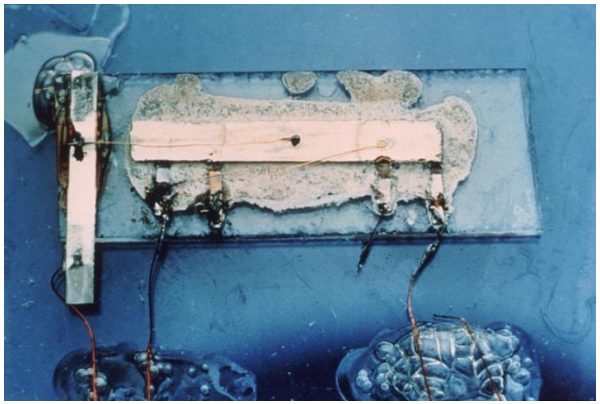
(1946, UPenn)

- Transistor

(1947, Bardeen, Brattain, Shockley)

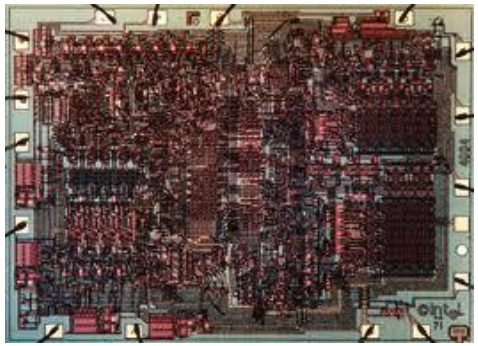
- Integrated circuit

(1958, Jack Kilby)



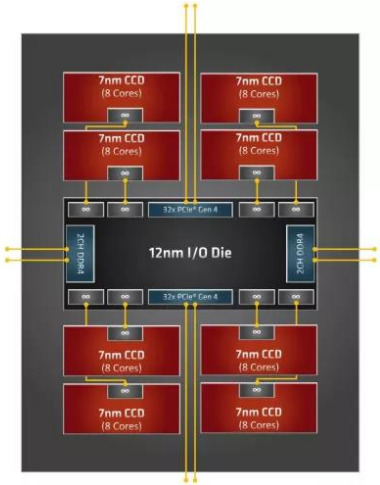
History of Hardware

- Intel 4004
  (1971, 1400 transistors,
  ~$1000)



- 64 Core AMD Ryzen Threadripper
  (2020, ~40 billion transistors,
  total silicon area of ~10 cm^2,
  $2000)

https://www.tomshardware.com/reviews/amd-threadripper-3990x-review

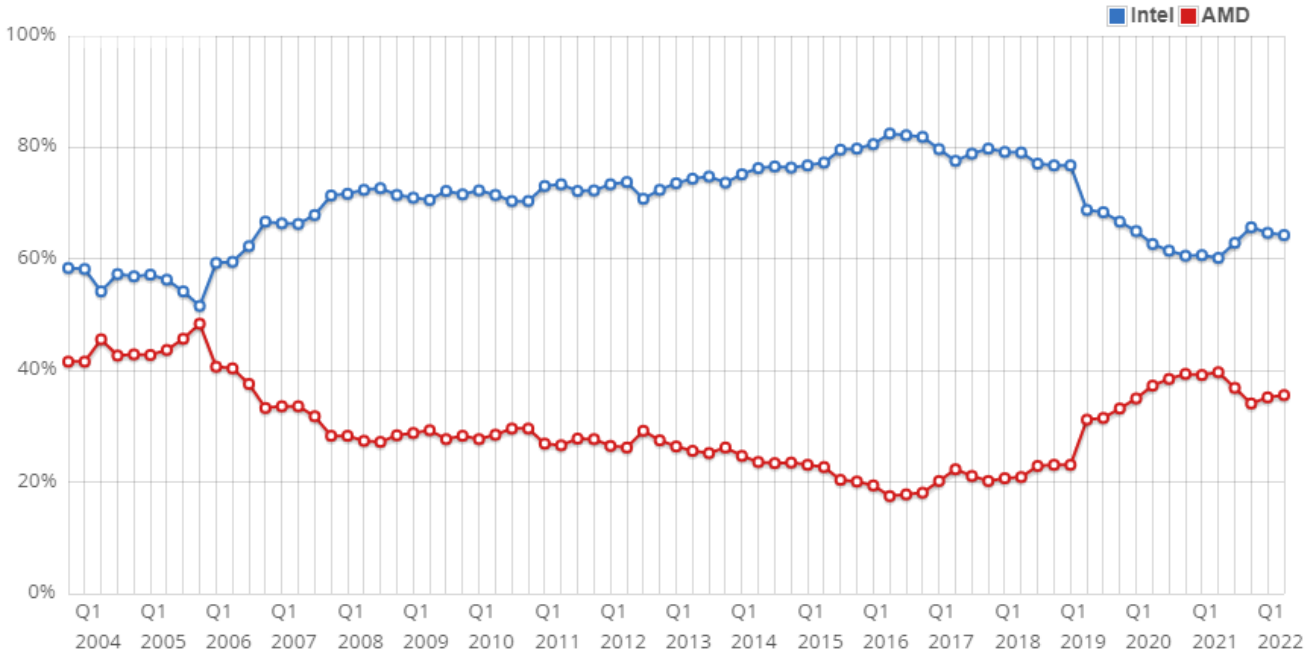


- Very Large Scale Integration (VLSI) – originally defined for chips having transistors in the order of 100,000. Other terms such as ULSI came along, but the usage VLSI remains dominant

https://en.wikichip.org – a website with lot of info on processors, new and old

Desktop Processor Market

Market Share

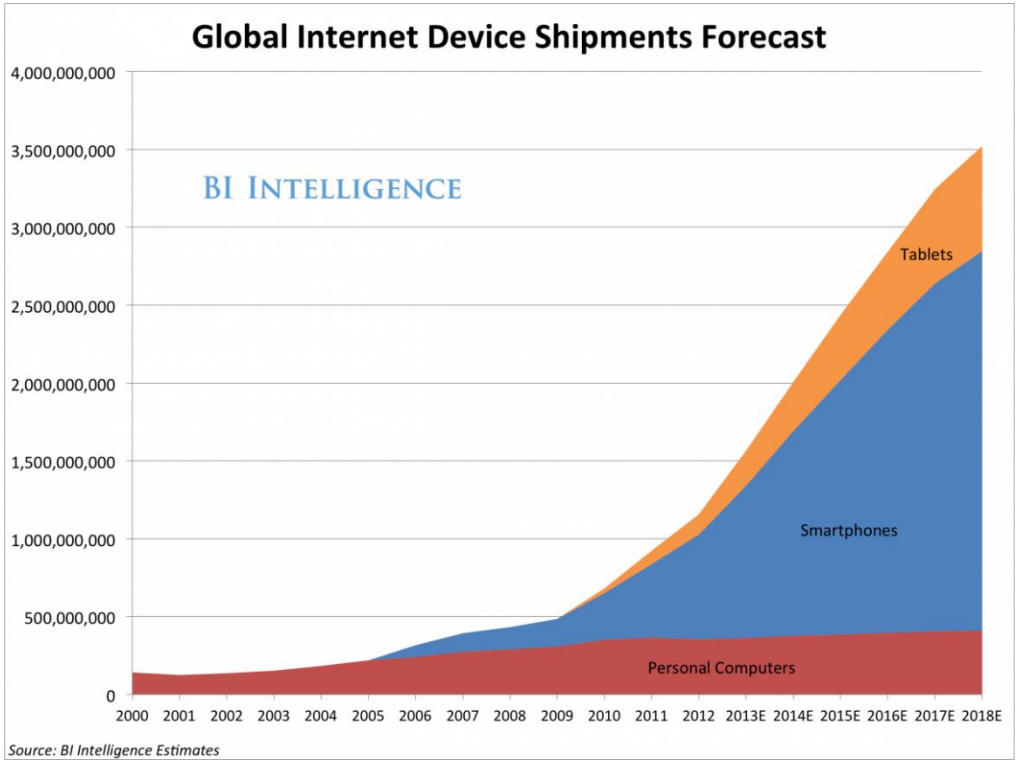


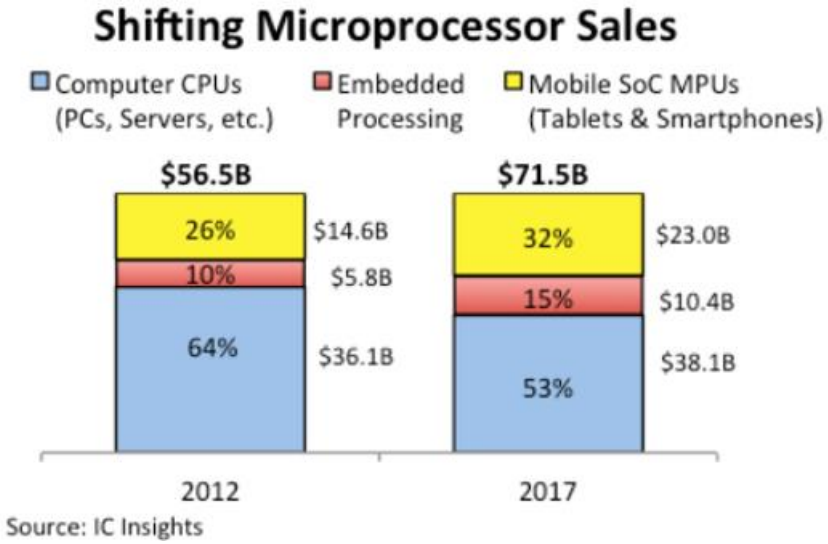
CPU	CPU Mark
AMD Ryzen Threadripper PRO 5995WX	100,012
AMD EPYC 7773X	89,614
Intel Xeon Platinum 8380 @ 2.30GHz	62,318

Performance : PassMark - CPU Mark

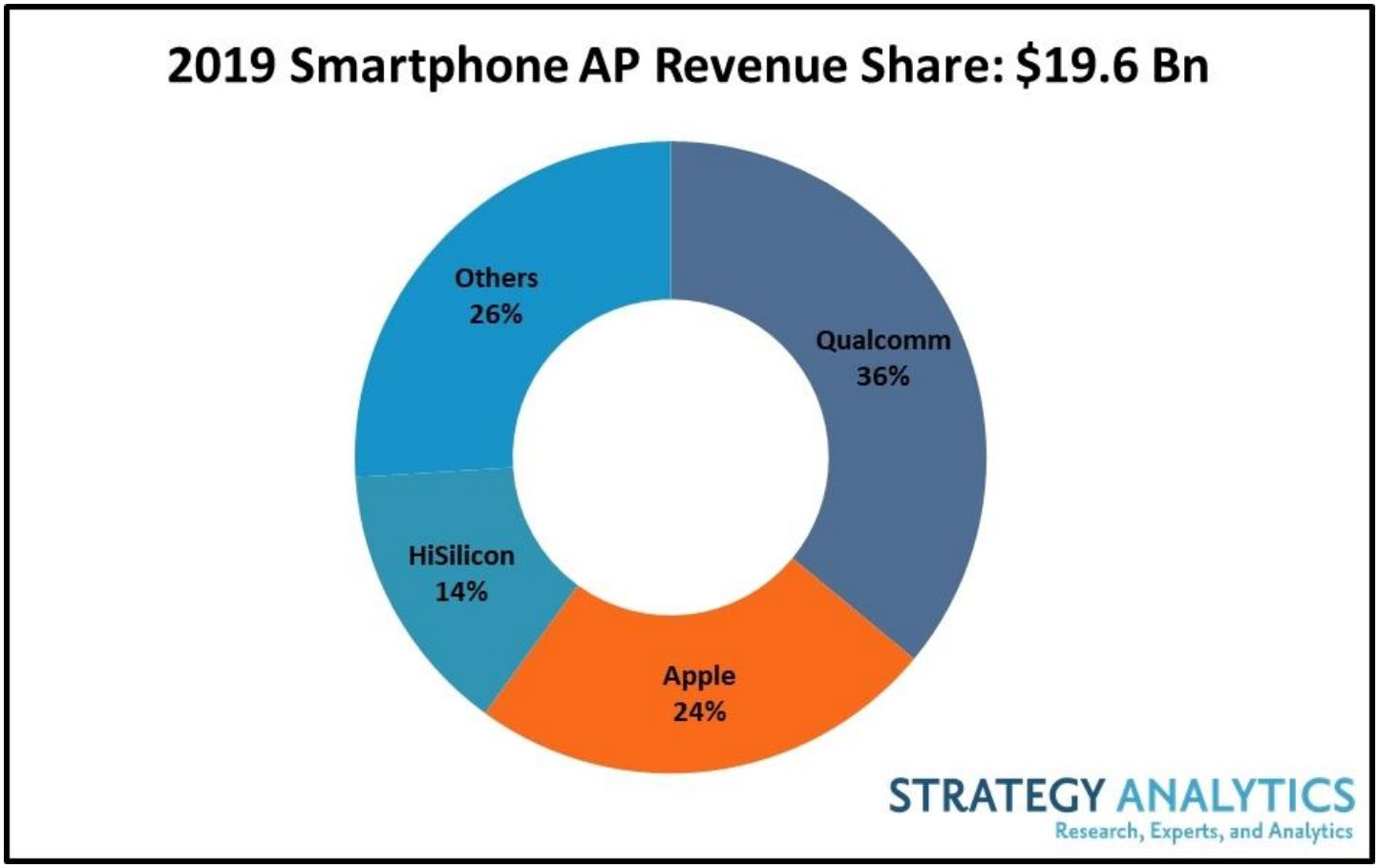
Courtesy: https://www.cpubenchmark.net

Market Trends



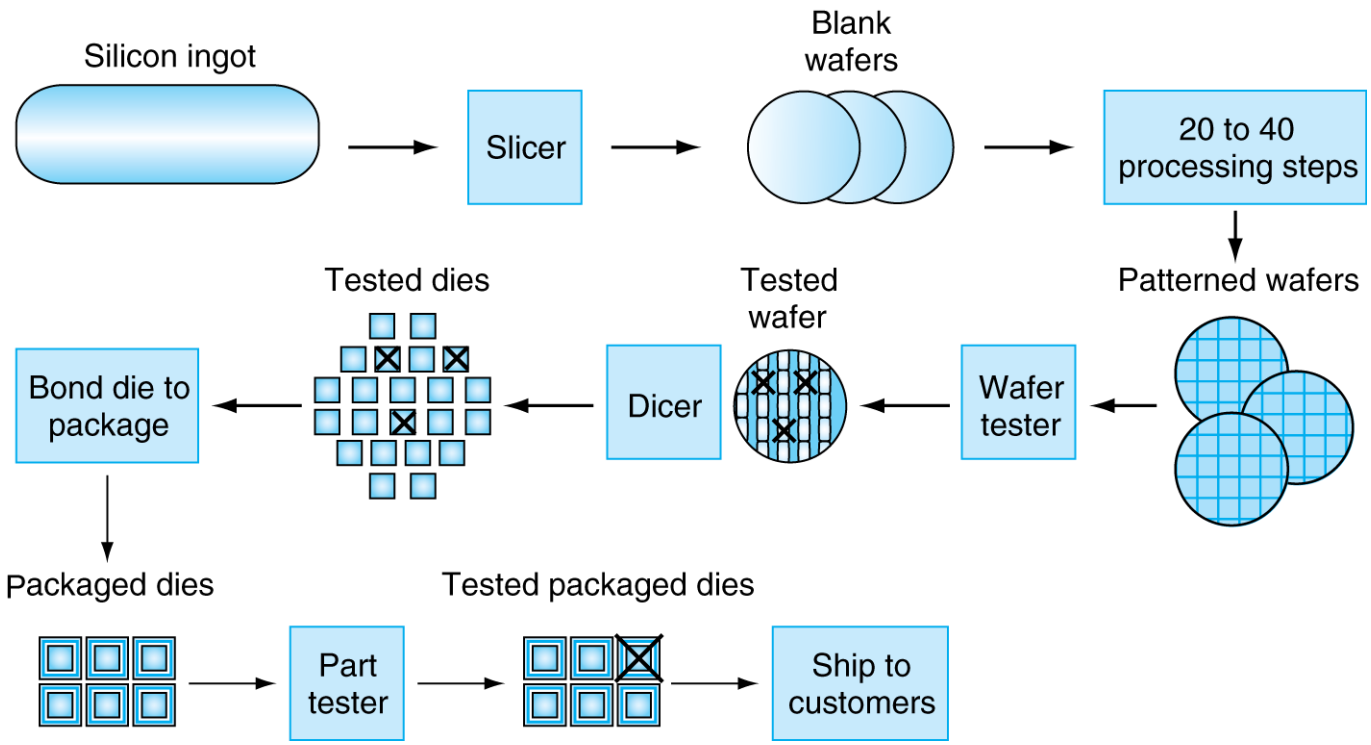


Application Processor Market



Application processors = Processors used in smartphones / tablets (ARMv7A / 8A instruction set)

Manufacturing ICs

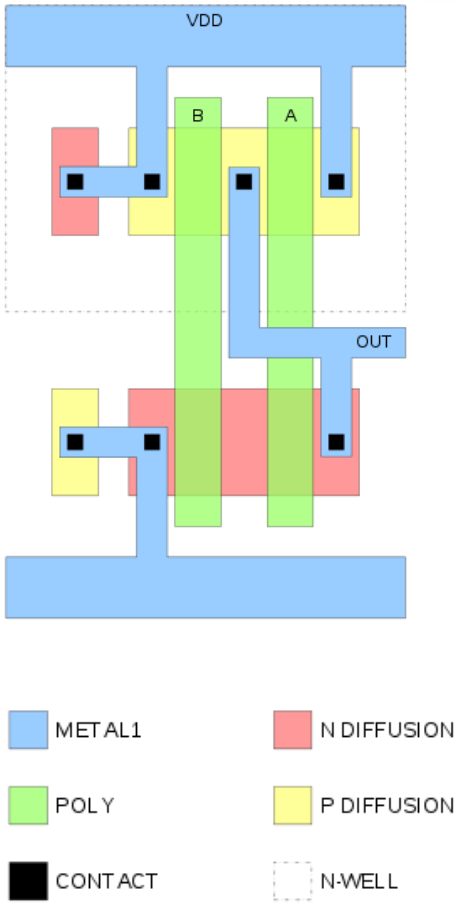


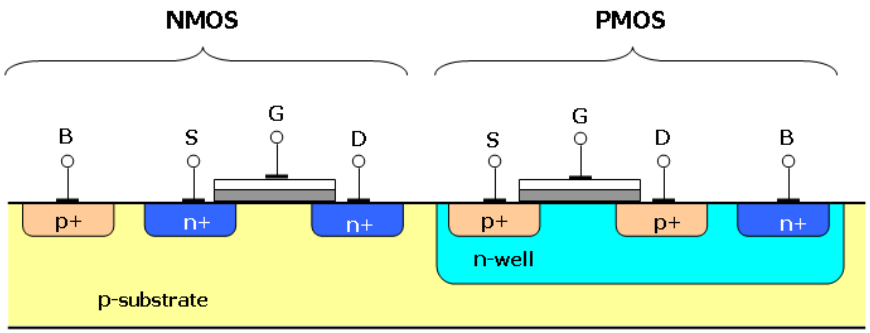
- Yield: proportion of working dies per wafer

To Do: Read up on chip 'binning'

CMOS VLSI (More in EE4415)

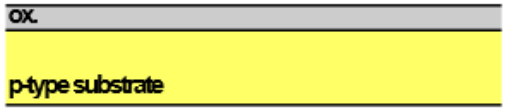
- Complementary metal oxide semiconductor (CMOS) – combination of N-channel MOS (NMOS) and P-channel MOS (PMOS) field effect transistors (FETs)
- The dominant device technology due to power, size and cost considerations, though not as fast as Bipolar Junction Transistor (BJT) based circuits



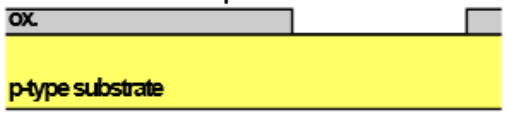


Simplified CMOS Steps (More in EE4415)

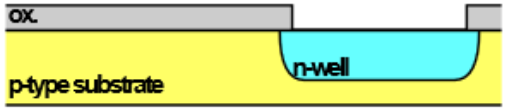
1. Grow field oxide



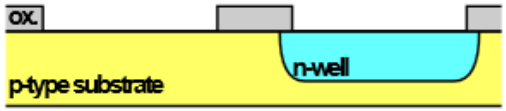
2. Etch oxide for pMOSFET



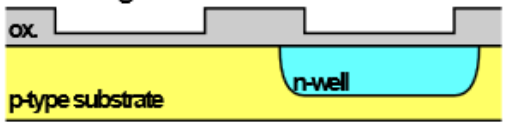
3. Diffuse n-well



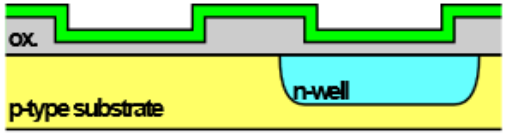
4. Etch oxide for nMOSFET



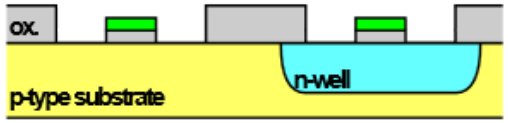
5. Grow gate oxide



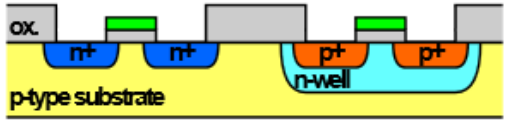
6. Deposit polysilicon



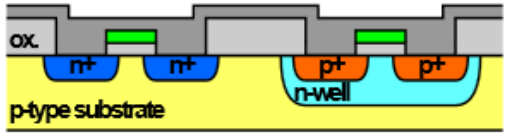
7. Etch polysilicon and oxide



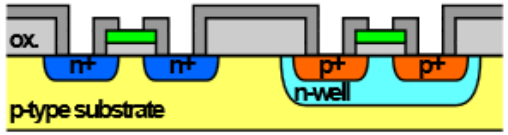
8. Implant sources and drains



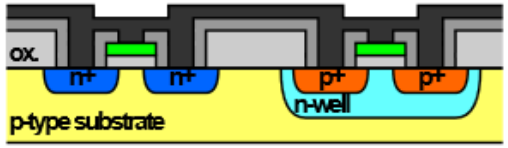
9. Grow nitride



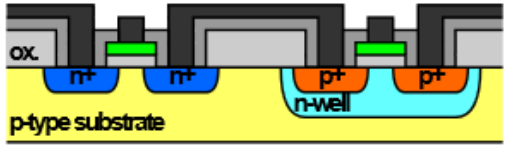
10. Etch nitride



11. Deposit metal

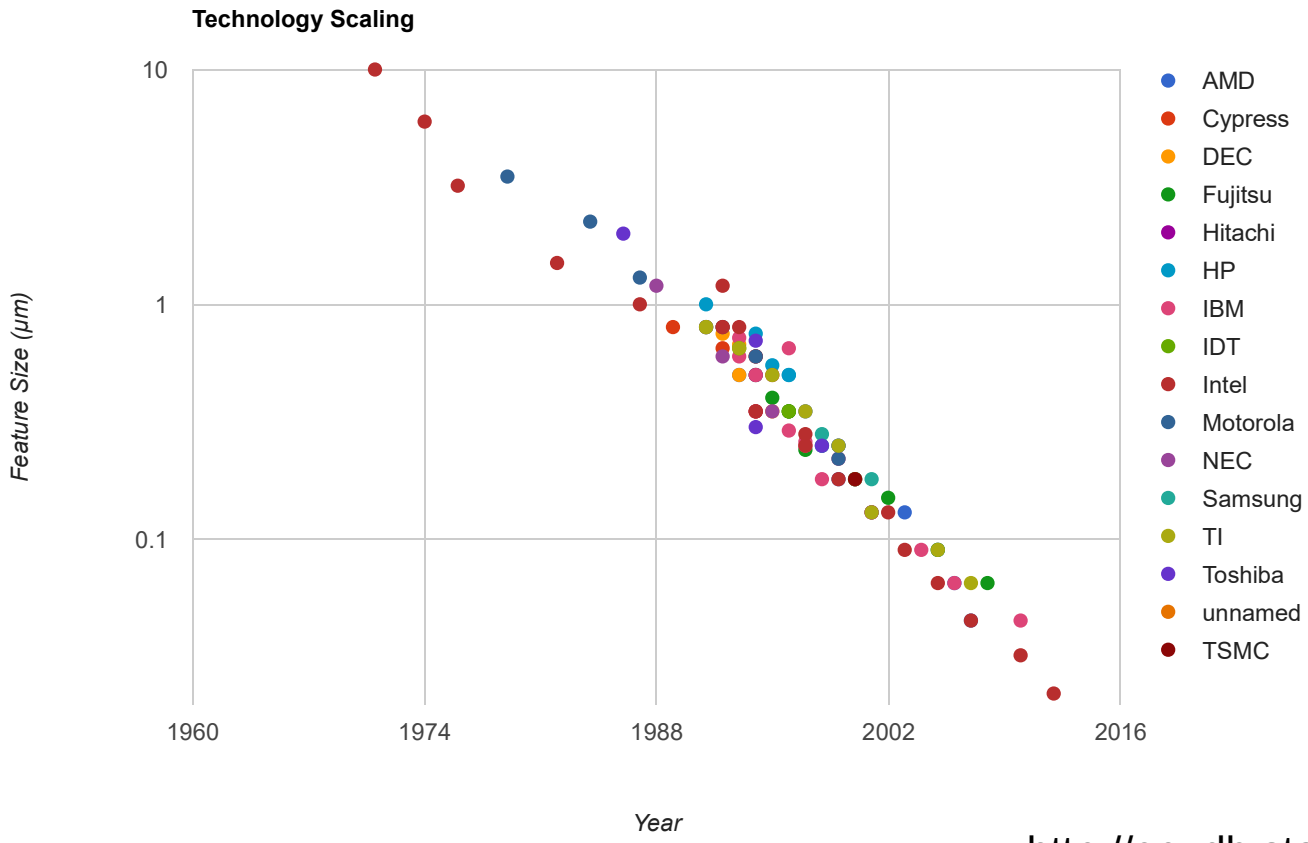


12. Etch metal



Moore's Law

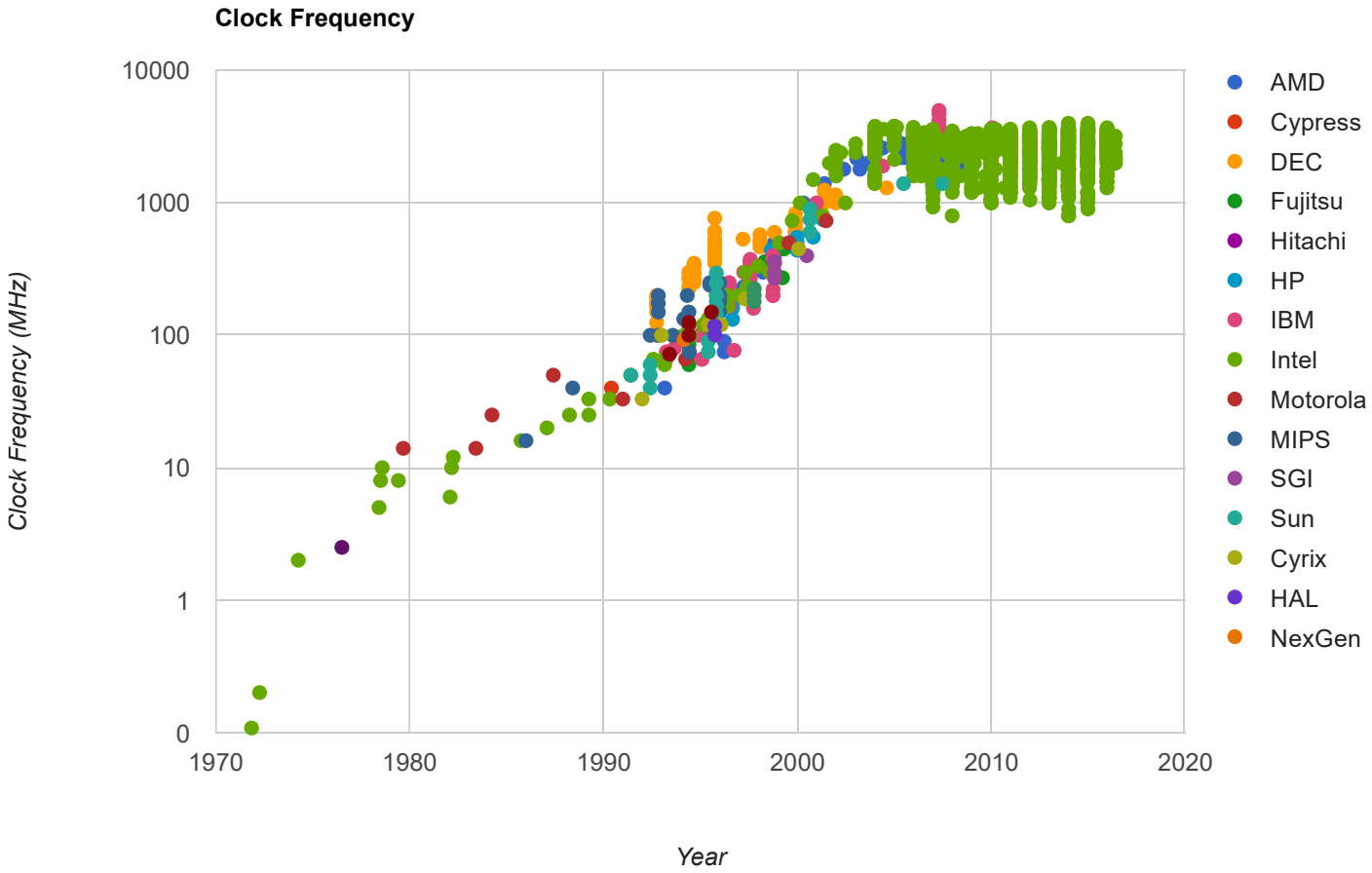
- In 1965, Intel's Gordon Moore predicted that the number of transistors that can be integrated on single chip would double about every two years



http://cpudb.stanford.edu/

Clock Frequencies Over the Years

- Clock frequencies have hit a wall in the 2000s. Designers have to rely on microarchitectural improvements to keep increasing the performance

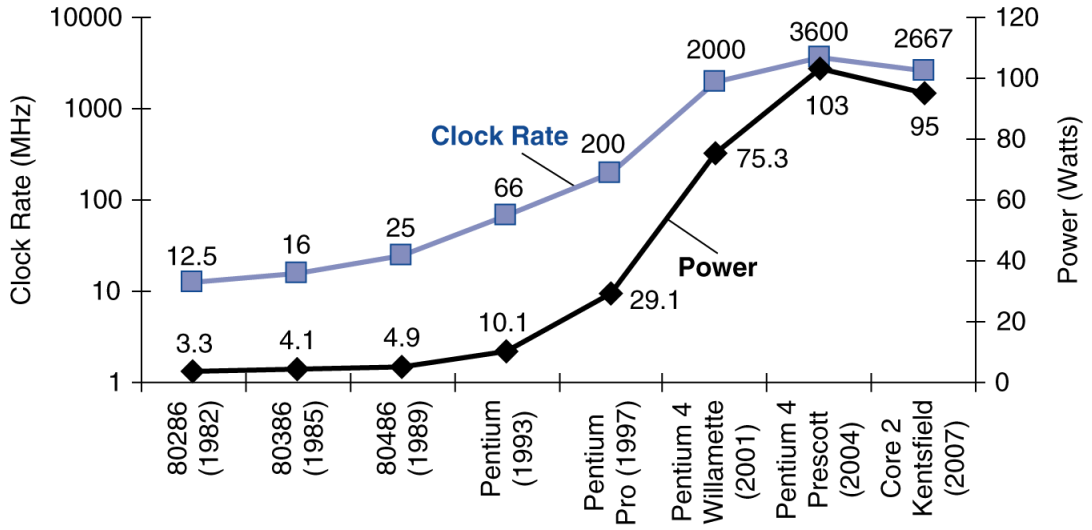


Technology Scaling Road Map (Intel)

Year	2006	2008	2010	2012	2014	2018	2021	2025?
Feature size (nm)	65	45	32	22	14	10	7	5
Codename (Tick/Process)	Presler, Cedar Mill (Netburst), Yonah (P6)	Penryn (Core)	Westmere	Ivy Bridge	Broadwell	Cannon Lake	Alder Lake (12^{th} Gen)	?
Codename (Tock/Arch.)	Conroe/ Merom (Core)	Nehalem	Sandy Bridge	Haswell	Skylake	Ice Lake	Raptor Lake	?
Codename (Optimization)					Kaby Lake (1) Coffee Lake (2) Whiskey Lake (3) Comet Lake (4) Rocket Lake (5)	Tiger Lake	Meteor Lake	?

- Intel used to follow tick (die shrink) and tock (microarchitecture improvement) – now replaced by PAO (process-architecture-optimization).
  - To Do : Identify the processor used by your laptop, mobile phone etc.
- Fun facts about 22nm transistors
  - 100 million can fit on the head of a pin
  - You could fit more than 4,000 across the width of a human hair
  - A 22nm transistor can switch on and off well over 100 billion times/second. You will need 2,000 years to flick a light switch on and off that many times!
  - If car prices had fallen at the same rate as the price of a single transistor has since 1968, a new car today would cost about 1 cent

Power Trends



- In CMOS IC technology

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

×30 | 5V → 1V | ×1000

The thermal design power/point (TDP) is the maximum amount of power the cooling system in a computer is required to dissipate

Reducing Power

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

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

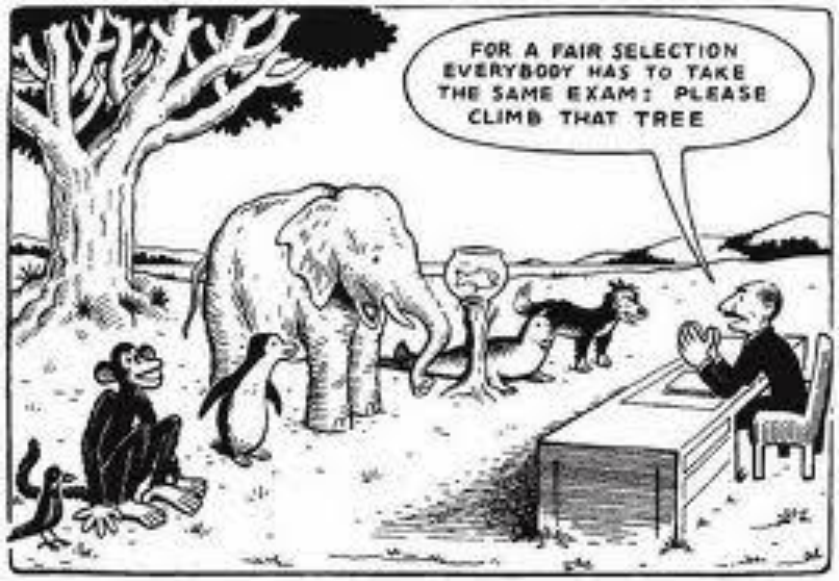
- The power wall
  - We can't reduce voltage further
  - We can't remove more heat
- How else can we improve performance?

Multiprocessors

- Multicore microprocessors
  - More than one processor per chip
- Requires explicitly parallel programming
  - Compare with instruction level parallelism (we'll see it later)
    - Hardware executes multiple instructions at once
    - Hidden from the programmer
  - Hard to do
    - Programming for performance
    - Load balancing
    - Optimizing communication and synchronization

Issues and Modern Trends

- Limited instruction level parallelism (ILP), power issues
  - Cloud computing
  - Hardware accelerators
  - Multi-core / processor systems, clusters
- One will not fit all
  - Heterogeneous systems
  - Application-specific instruction-set processors
  - Hardware/Software co-design
  - Reconfigurable computing



Note : There could be one or two questions in the final exam based on the state of the art in Microprocessor Technology / Market

Issues and Modern Trends …

- The communication bottleneck
  - SoCs, Multi-chip modules
  - 3D ICs, 3D Packaging
  - Optical interconnects

- Leakage current limiting size reduction
  - Multi-gate (3D) FETs – FinFET and gate-all-around (GAA) FETs instead of planar (Intel 14 nm uses tri-gate FinFETs, Samsung uses GAA for 3 nm)
    https://semiengineering.com/new-transistor-structures-at-3nm-2nm
  - Other strategies include channel strain engineering, silicon-on-insulator-based technologies, and high-k/metal gate materials

- Reaching the limits of Silicon
  - Graphene, III-V semiconductors, carbon nanotubes or optical computing?

Performance of Computers

- Algorithm
  - Determines number of operations executed
- Programming language, compiler, architecture
  - Determine number of machine instructions executed per operation
- Processor and memory system
  - Determine how fast instructions are executed
- I/O system (including OS)
  - Determines how fast I/O operations are executed

Throughput versus Response Time

- Response time (execution time) – the time between the start and the completion of a task
  - Important to individual users
- Throughput (bandwidth) – the total amount of work done in a given time
  - Important to data center managers
- Will need different performance metrics as well as a different set of applications to benchmark personal mobile devices, embedded and desktop computers, which are more focused on response time, versus servers, which are more focused on throughput

Relative Performance

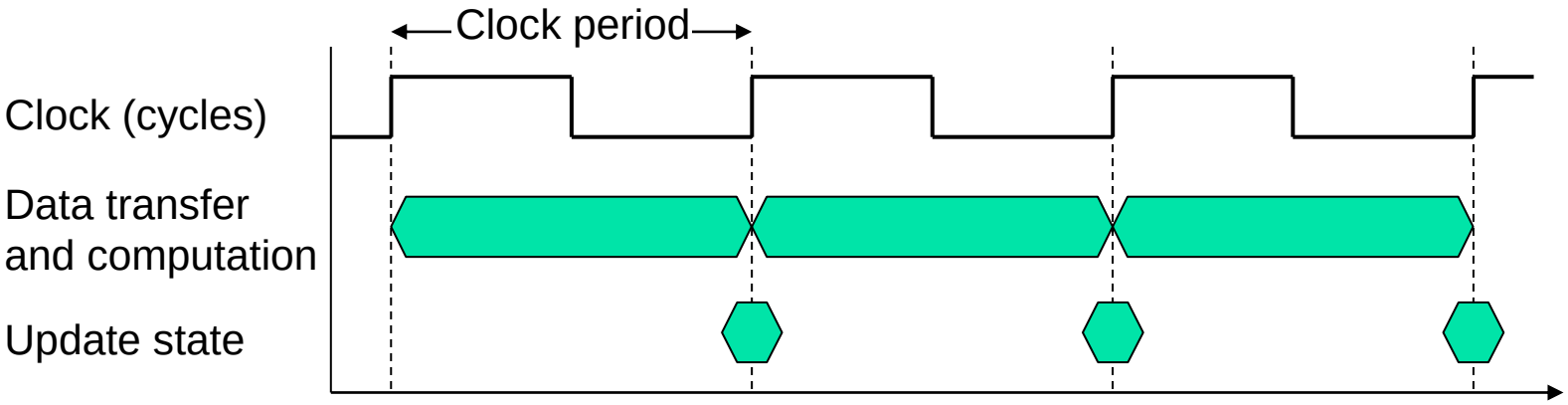
- Define Performance = 1/Execution Time
- "X is n time faster than Y"

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

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

CPU Clocking

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



  - 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 Example

- Computer A: 2GHz clock, 10s CPU time
- Designing Computer B
  - Aim for 6s CPU time
  - Can do faster clock, but causes 1.2 × 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

$$\text{Clock Cycles} = \text{Instruction Count} \times \text{Cycles per Instruction (CPI)}$$

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

$$= \frac{\text{CPU Time}}{\text{Clock Cycle Time}}$$

$$\text{CPU Time} = \text{Instruction Count} \times \text{CPI} \times \text{Clock Cycle Time}$$

$$= \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$$

- Instruction Count for a program
  - Determined by program, ISA and compiler
- Cycles per instruction (CPI)
  - Determined by CPU hardware
  - If different instructions have different CPI
    - Average CPI affected by instruction mix

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
    = 2×1 + 1×2 + 2×3
    = 10
  - Avg. CPI = 10/5 = 2.0
- Sequence 2: IC = 6
  - Clock Cycles
    = 4×1 + 1×2 + 1×3
    = 9
  - Avg. CPI = 9/6 = 1.5

CPI in More Detail

- If different instruction classes take different numbers of cycles

$$\text{Clock Cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{Instructio n Count}_i)$$

  - Weighted average CPI

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

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?

$$\text{CPU Time}_A = \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A$$

$$= I \times 2.0 \times 250\text{ps} = I \times 500\text{ps}$$

A is faster…

$$\text{CPU Time}_B = \text{Instruction Count} \times \text{CPI}_B \times \text{Cycle Time}_B$$

$$= I \times 1.2 \times 500\text{ps} = I \times 600\text{ps}$$

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

…by this much

Performance Summary

- 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

SPEC CPU Benchmark

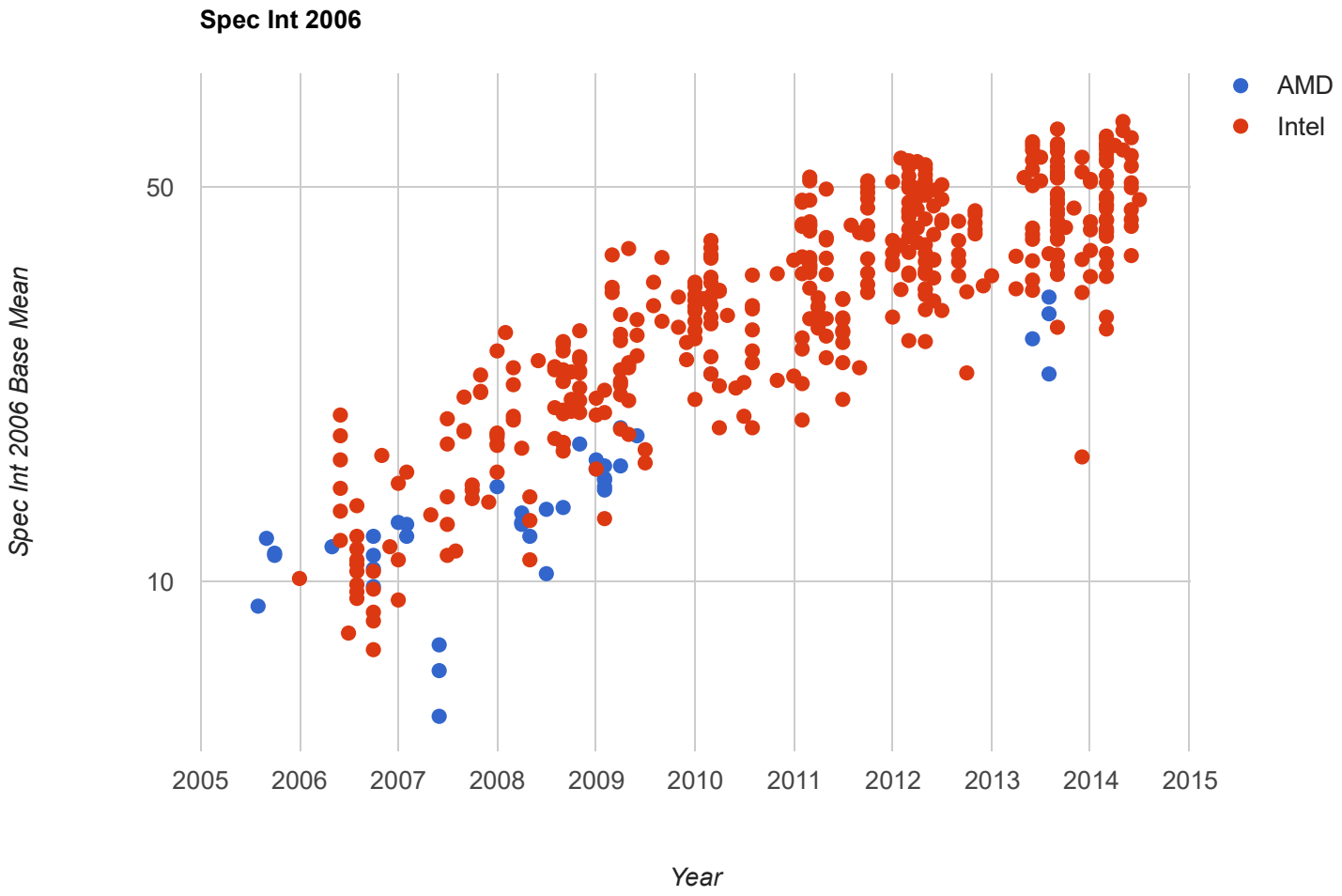
- Programs used to measure performance
  - Supposedly typical of actual workload
- Standard Performance Evaluation Corp (SPEC)
  - Develops benchmarks for CPU, I/O, Web, …
- SPEC CPU2006
  - Elapsed time to execute a selection of programs
    - Negligible I/O, so focuses on CPU performance
  - Normalize relative to reference machine
  - Summarize as geometric mean of performance ratios
    - CINT2006 (integer) and CFP2006 (floating-point)

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

CINT2006 for Intel Core i7 920

Description	Name	Instruction Count x 10^9	CPI	Clock cycle time (seconds x 10^{-9})	Execution Time (seconds)	Reference Time (seconds)	SPECratio
Interpreted string processing	perl	2252	0.60	0.376	508	9770	19.2
Block-sorting compression	bzip2	2390	0.70	0.376	629	9650	15.4
GNU C compiler	gcc	794	1.20	0.376	358	8050	22.5
Combinatorial optimization	mcf	221	2.66	0.376	221	9120	41.2
Go game (AI)	go	1274	1.10	0.376	527	10490	19.9
Search gene sequence	hmmer	2616	0.60	0.376	590	9330	15.8
Chess game (AI)	sjeng	1948	0.80	0.376	586	12100	20.7
Quantum computer simulation	libquantum	659	0.44	0.376	109	20720	190.0
Video compression	h264avc	3793	0.50	0.376	713	22130	31.0
Discrete event simulation library	omnetpp	367	2.10	0.376	290	6250	21.5
Games/path finding	astar	1250	1.00	0.376	470	7020	14.9
XML parsing	xalancbmk	1045	0.70	0.376	275	6900	25.1
Geometric mean	–	–	–	–	–	–	25.7

System Performance Over the Years



SPEC Power Benchmark

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

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

ssj_ops - server side Java operations

SPECpower_ssj2008 for Xeon X5650

Target Load %	Performance (ssj_ops)	Average Power (Watts)
100%	865,618	258
90%	786,688	242
80%	698,051	224
70%	607,826	204
60%	521,391	185
50%	436,757	170
40%	345,919	157
30%	262,071	146
20%	176,061	135
10%	86,784	121
0%	0	80
Overall Sum	4,787,166	1,922
Σssj_ops/Σpower =		2,490

Pitfall: Amdahl's Law

- Improving an aspect of a computer and expecting a proportional improvement in overall performance

$$T_{improved} = \frac{T_{affected}}{improvement\ factor} + T_{unaffected}$$

- Example: multiply accounts for 80s/100s
  - How much improvement in multiply performance to get 5× overall?

$$20 = \frac{80}{n} + 20$$

  - Can't be done!

- Corollary: make the common case fast

Fallacy: Low Power at Idle

- Look back at i7 power benchmark
  - At 100% load: 258W
  - At 50% load: 170W (66%)
  - At 10% load: 121W (47%)
- Google data center
  - Mostly operates at 10% – 50% load
  - At 100% load less than 1% of the time
- Consider designing processors to make power proportional to load

Pitfall: MIPS as a Performance Metric

- MIPS: Millions of Instructions Per Second
  - Doesn't account for
    - Differences in ISAs between computers
    - Differences in complexity between instructions

$$\text{MIPS} = \frac{\text{Instruction count}}{\text{Execution time} \times 10^6}$$

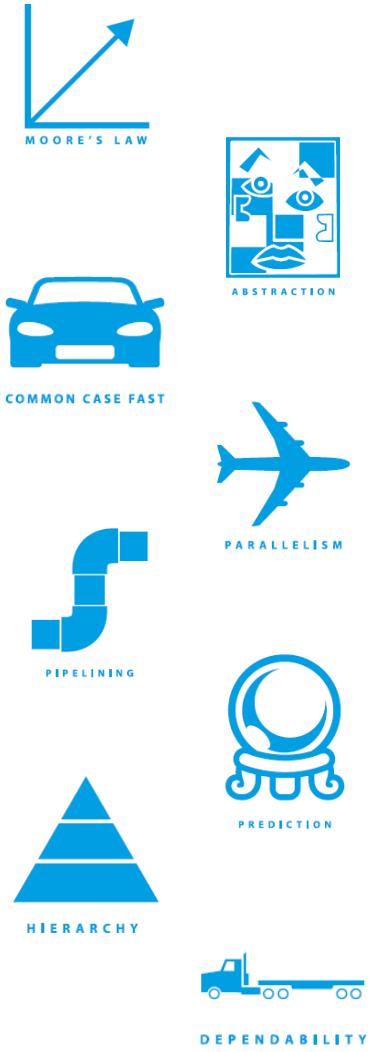
$$= \frac{\text{Instruction count}}{\frac{\text{Instruction count} \times \text{CPI}}{\text{Clock rate}} \times 10^6} = \frac{\text{Clock rate}}{\text{CPI} \times 10^6}$$

  - CPI varies between programs on a given CPU

Not to be confused with MIPS ISA, popular in set-top boxes, wireless routers etc.

Eight Great Ideas

- Design for **Moore's Law**
- 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



Concluding Remarks

- Cost/performance is improving
  - Due to underlying technology development
- Hierarchical layers of abstraction
  - In both hardware and software
- Instruction set architecture
  - The hardware/software interface
- Power is a limiting factor
  - Use parallelism to improve performance
- Execution time: the best performance measure