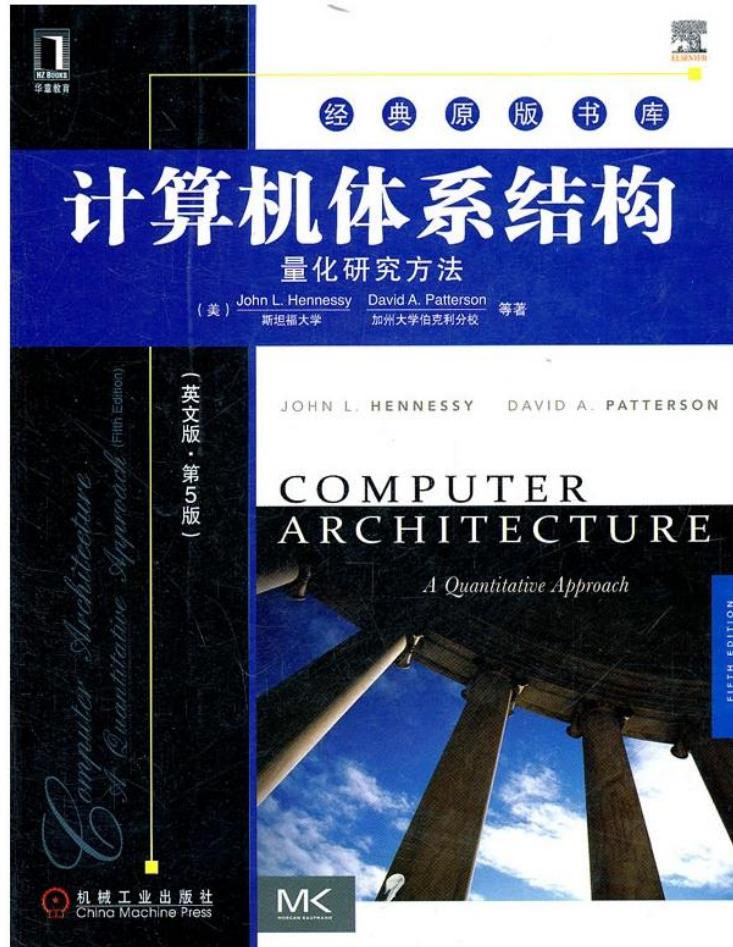


Modern Computer Architecture

with sources that included UPenn & University of Wisconsin slides
by Joe Devietti, Milo Martin & Amir Roth, Mark Hill, Guri Sohi, Jim Smith, and David Wood

Reference book



Schedule

- Introduction and Transistors
- Parallel computing (Isaac D. Scherson, University of California, Irvine, in October)
- ISAs
- Performance
- Pipelining Basic
- Branch Prediction
- Caches
- Virtual Memory
- Out-of-Order Execution
- Multicore multi-thread
- Vectors/GPUs for data parallelism.

Administrivia

- Instructor: **Jianhua LI** (jhli@ecust.edu.cn)
- Three important web sites
 - Course website: syllabus, schedule, lecture notes
 - <http://gschool.ecust.edu.cn>
- **Write a >1000 word paper for any topic of our course.**
- **Homework**
- **Excises**

Grading

- Your grade in this course has the following components:
- Paper Reviews and writing: 15%
- Homework: 5%
- Classroom work and performance: 20%
- Final exam: 60%
- Typical grade distributions
 - ≥ 90 : 10%
 - [70-90]: 80%
 - < 70 : 10%

Prerequisites

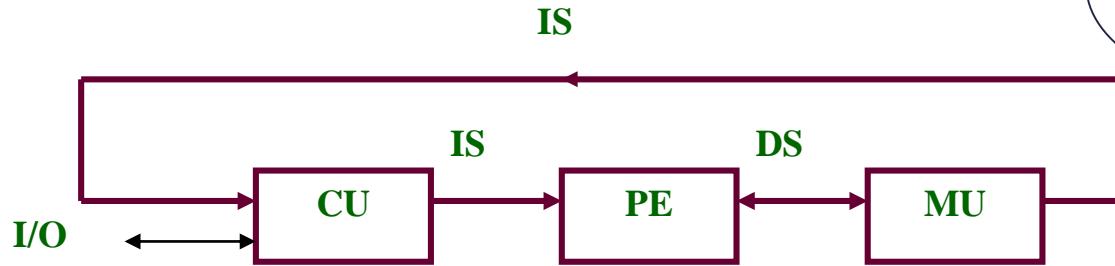
- **Basic computer organization an absolute must**
 - Basic digital logic: gates, boolean functions, latches
 - Binary arithmetic: adders, hardware mul/div, floating-point
 - Basic datapath: ALU, register file, memory interface, muxes
 - Basic control: single-cycle control, microcode
 - **Familiarity with assembly language**
 - “Computer Organization and Design: Hardware/Software Interface”
- **Significant programming experience**
 - You can write code to **simulate hardware**
 - Not difficult if competent programmer; extremely difficult if not

Flynn's Taxonomy

- Single instruction stream, single data stream (SISD)
- Single instruction stream, multiple data streams (SIMD)
 - Vector architectures
 - Multimedia extensions
 - Graphics processor units
- Multiple instruction streams, single data stream (MISD)
 - No commercial implementation
- Multiple instruction streams, multiple data streams (MIMD)
 - Tightly-coupled MIMD
 - Loosely-coupled MIMD

SISD (Single Instruction stream over a Single Data stream)

SISD uniprocessor architecture



The actual machine?

名词:

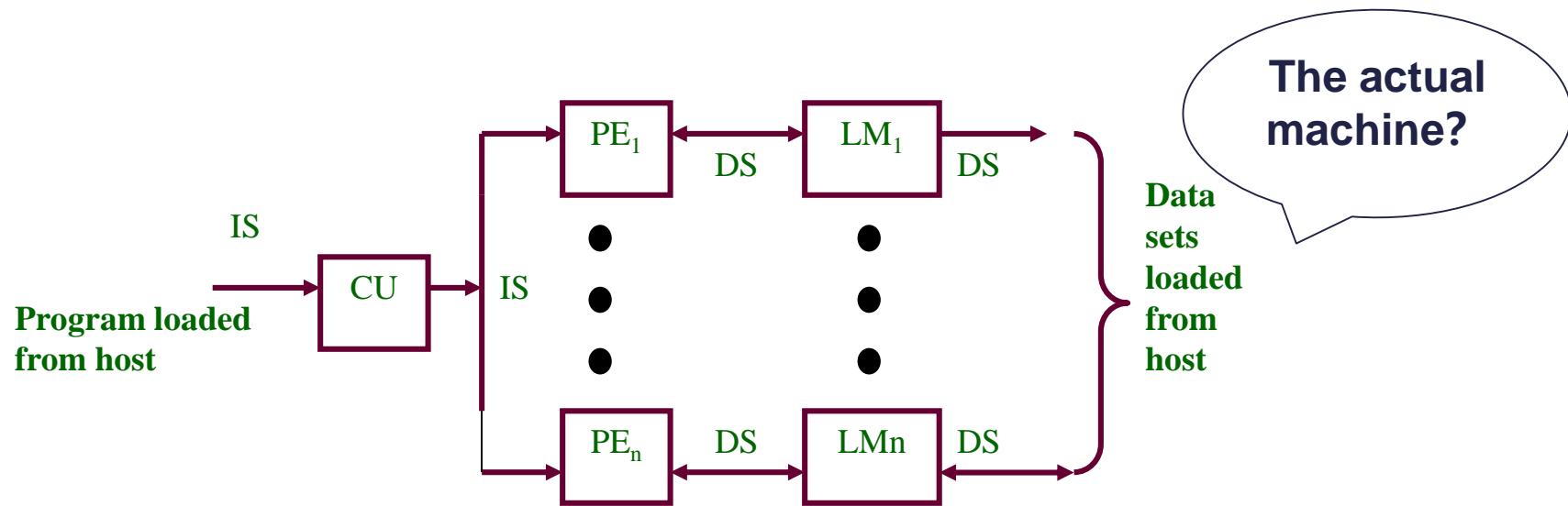
CU = control unit (控制单元)

MU = memory unit (存储单元) **IS = instruction stream** (指令流)

DS = data stream (数据流) **PE = processing element** (处理单元)

LM = Local Memory (局部内存)

SIMD (Single Instruction stream over Multiple Data stream)

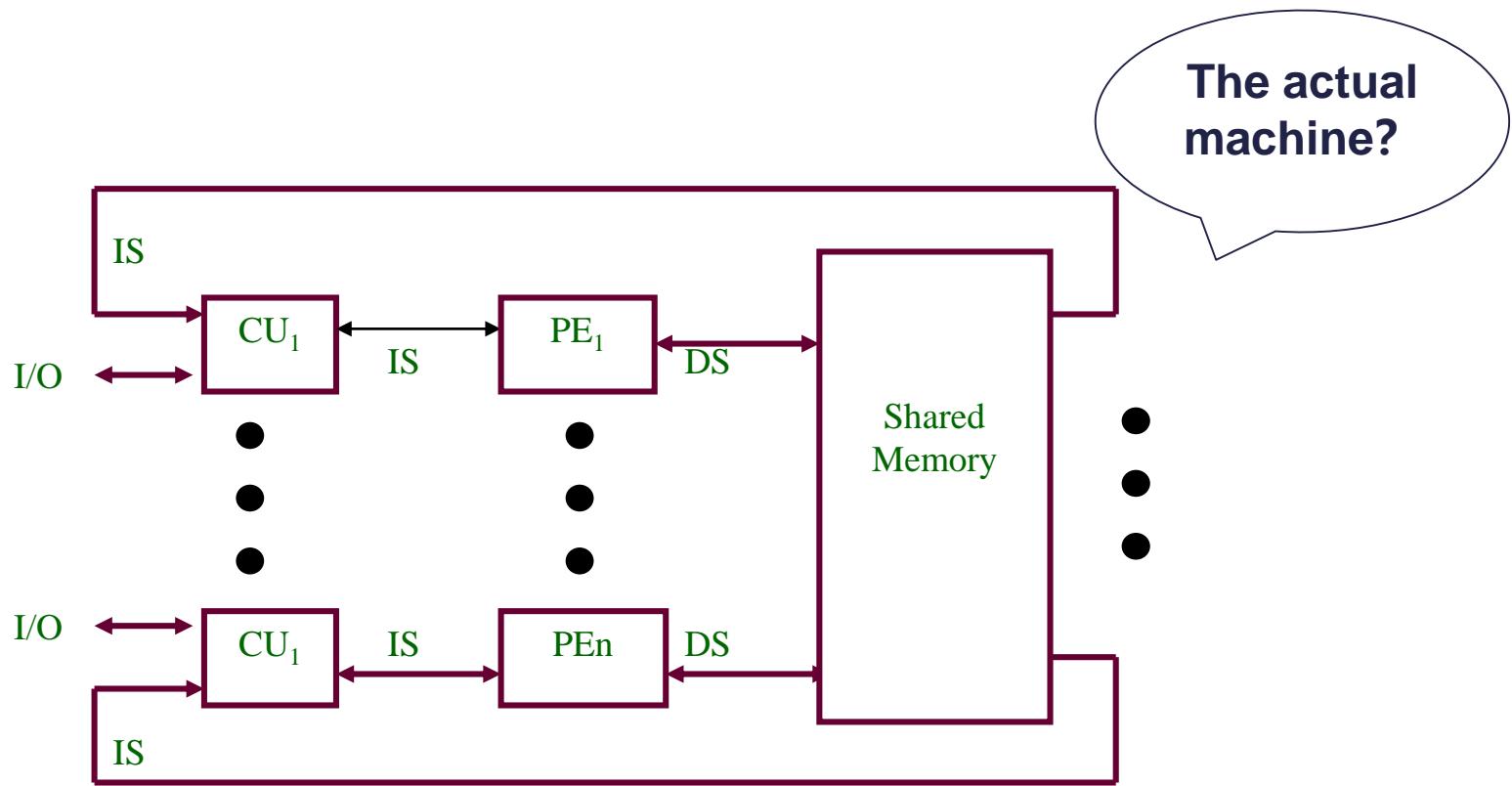


SIMD Architecture

注意：

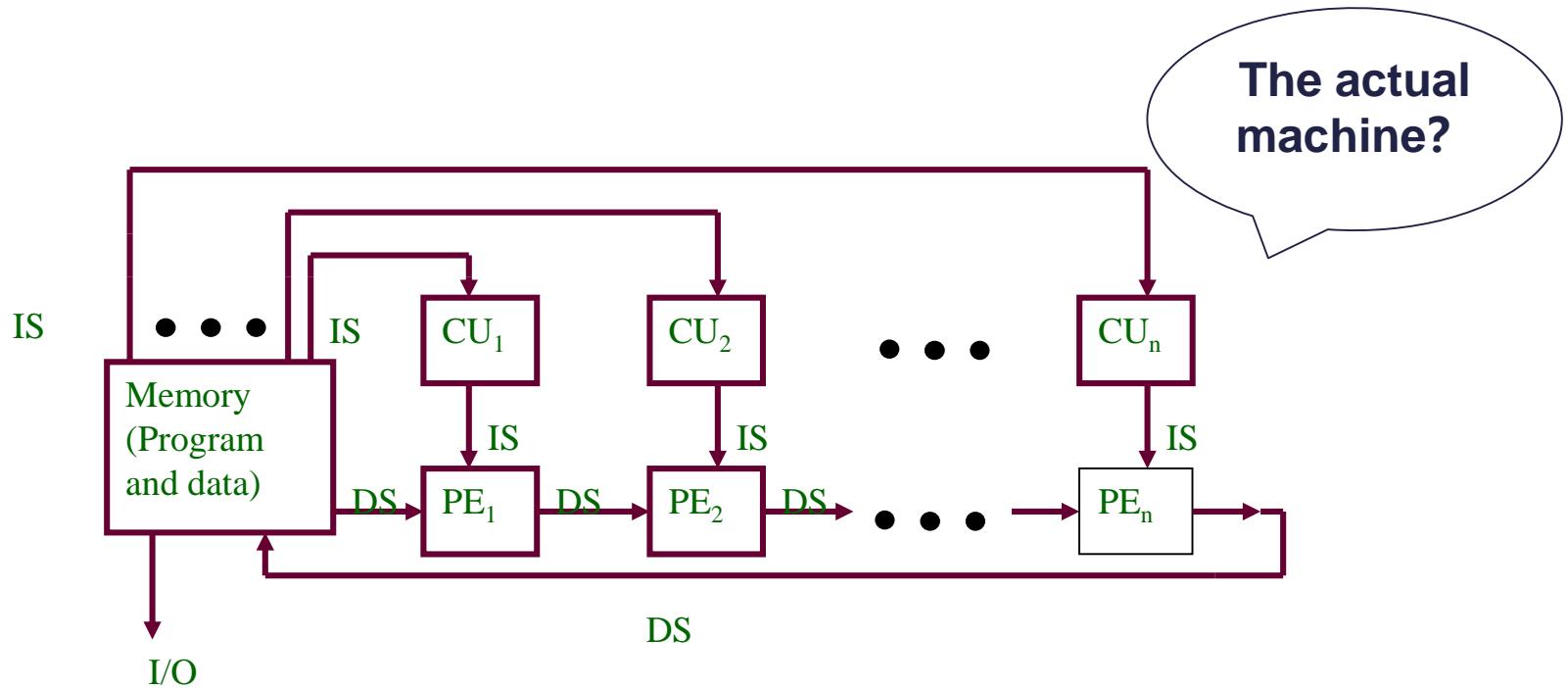
由于CU和PE的分离， SIMD的CPU不能用普通的CPU

MIMD (Multiple Instruction streams over Multiple Data streams)



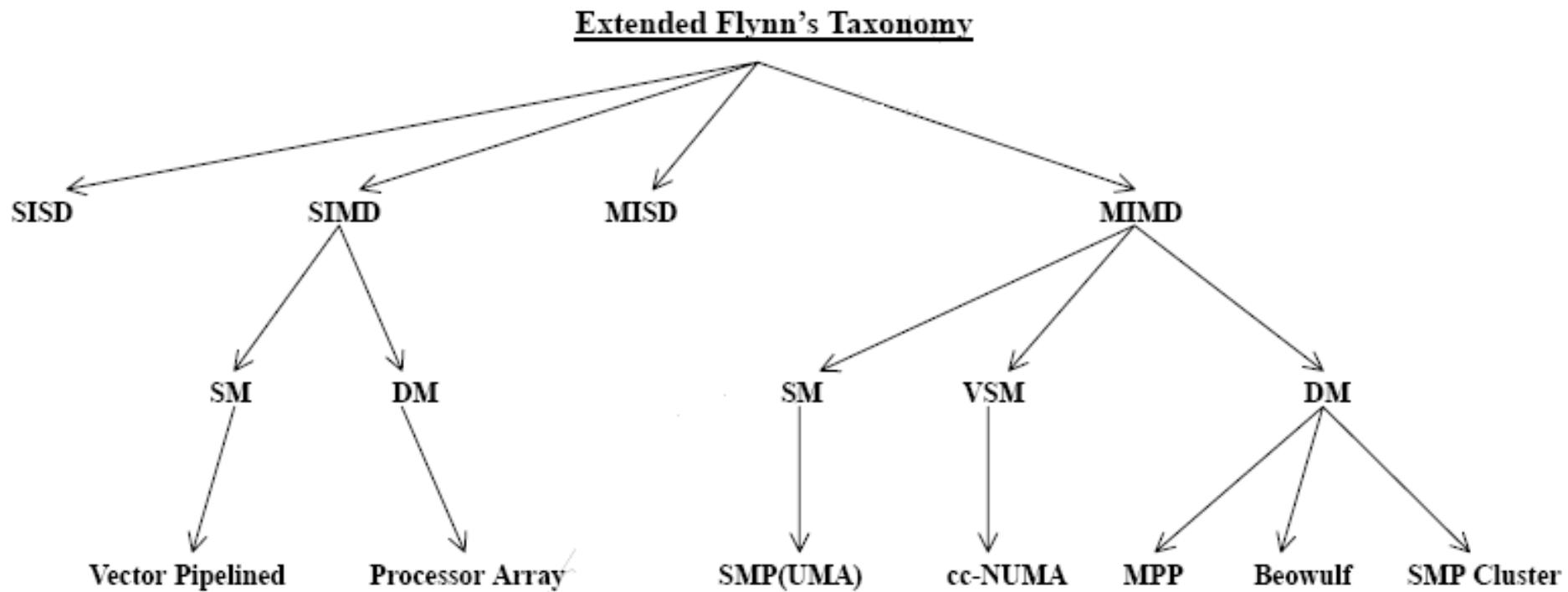
MMD Architecture (with shared memory)

MISD (Multiple Instruction streams over a Single Data streams)



MISD architecture (the systolic array)

Extension of Flynn Taxonomy



Academic Misconduct

- Cheating will **not** be tolerated
- General rule:
 - Anything with your name on it must be **YOUR OWN** work
 - Example: individual work on homework assignments
- Possible penalties
 - Zero on assignment (minimum)
 - Fail course
 - Note on permanent record
 - ...

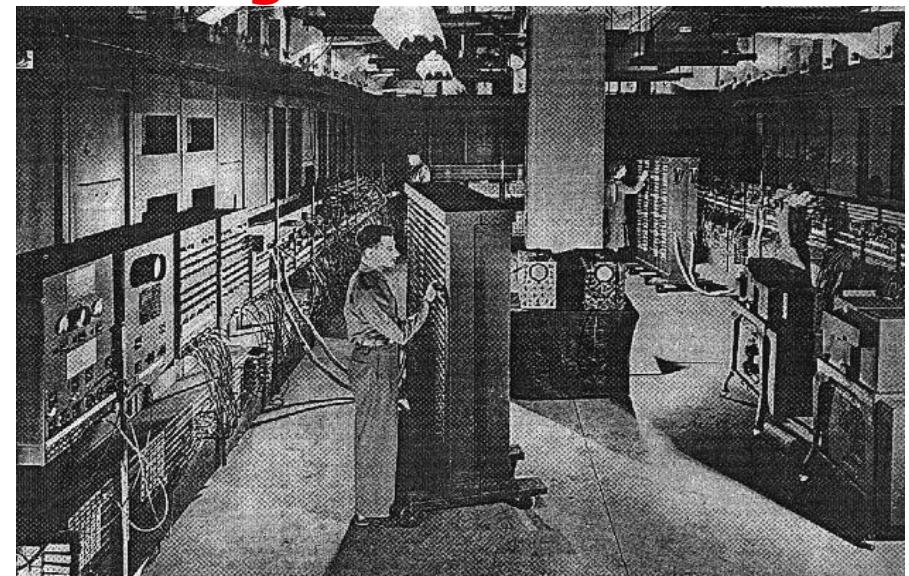
Course Overview

Why Study Computer Architecture?

- **Understand where computers are going**
 - Future capabilities drive the (computing) world
 - Real world-impact: no computer architecture → no computers!
- **Understand high-level design concepts**
 - The best architects understand all the levels
 - Devices, circuits, architecture, compiler, applications
- **Understand computer performance**
 - learn valid experimental methodologies
- **Write better software**
 - The best software designers also understand hardware
 - Need to understand hardware to write fast software
- **Design hardware**
 - At Intel, AMD, IBM, ARM, Qualcomm, Oracle, NVIDIA, Samsung

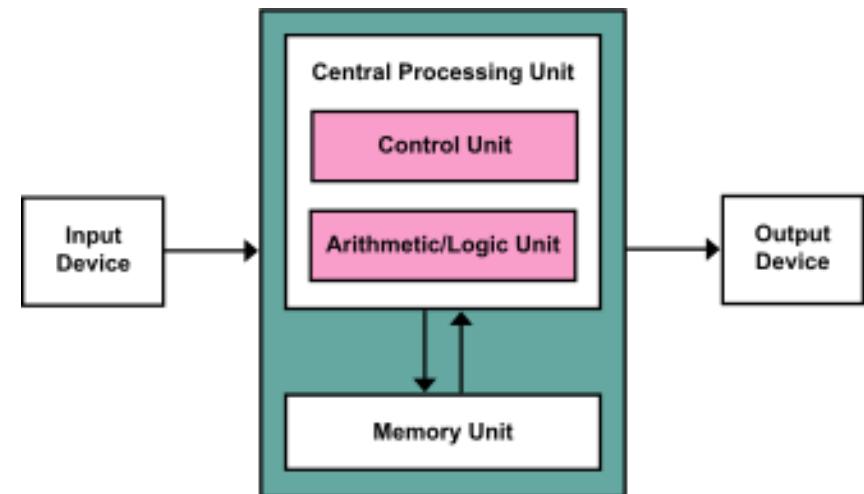
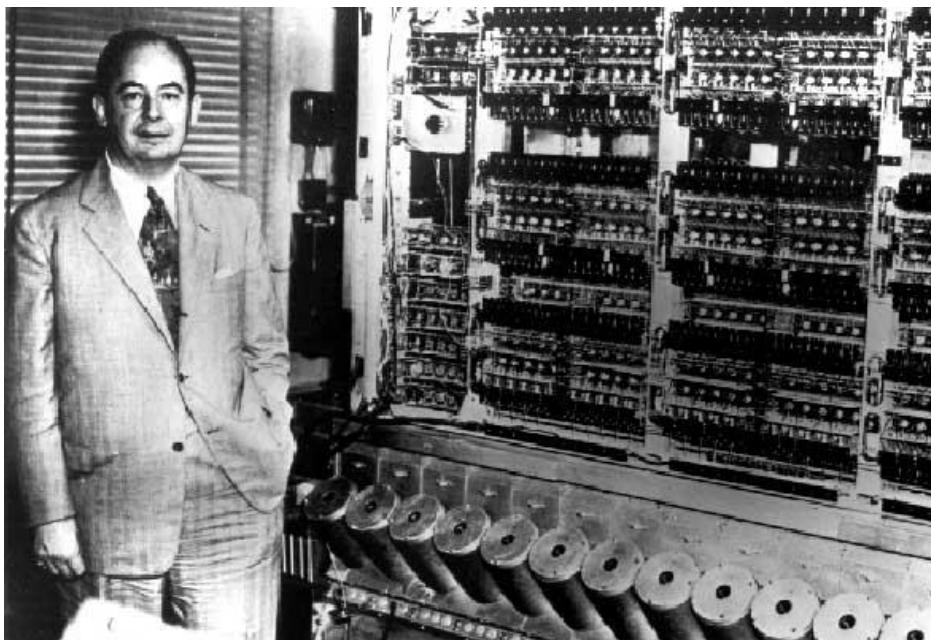
ENIAC

- **ENIAC:** electronic numerical integrator and calculator
 - First operational general-purpose stored-program computer
 - Designed and built here by Eckert and Mauchly from 1943-6
 - ENIAC was designed to calculate artillery firing tables for the US Army's Ballistic Research Laboratory (BRL).
 - Von Neumann architecture? (No)
- **First seminars on computer design**
 - Moore School Lectures, 1946
 - "*Theory and Techniques for Design of Electronic Digital Computers*"

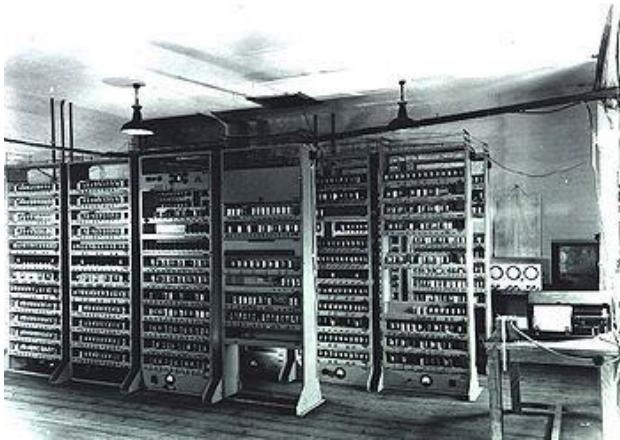


Von Neumann

- The Von Neumann architecture, also known as the Von Neumann model and Princeton architecture.



Von Neumann architecture computer



- **EDSAC (Electronic Delay Storage Automatic Calculator)** was the first practical stored-program electronic computer (May 1949)
- **EDVAC (Electronic Discrete Variable Automatic Computer)**
 - EDVAC's construction in August 1944. EDVAC was delivered to the Ballistics Research Laboratory in August 1949

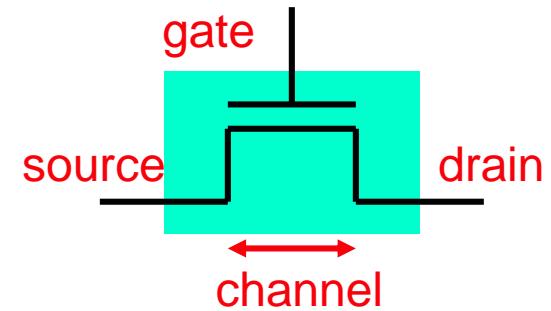
Application Specific Designs

- This class is about **general-purpose CPUs**
 - Processor that can do anything, run a full OS, etc.
 - E.g., Intel Core i7, AMD Athlon, IBM Power, ARM A-series
- In contrast to **application-specific chips**
 - Or **ASICs** (Application specific integrated circuits)
 - Also application-domain specific processors
 - Implement critical domain-specific functionality in hardware
 - Examples: video encoding, 3D graphics
 - General rules
 - Hardware is less flexible than software
 - + Hardware more effective (speed, power, cost) than software
 - + Domain specific more “parallel” than general purpose
 - But general mainstream processors becoming more parallel
- Trend: from **specific** to **general** (for a specific domain)

Technology Trends

“Technology”

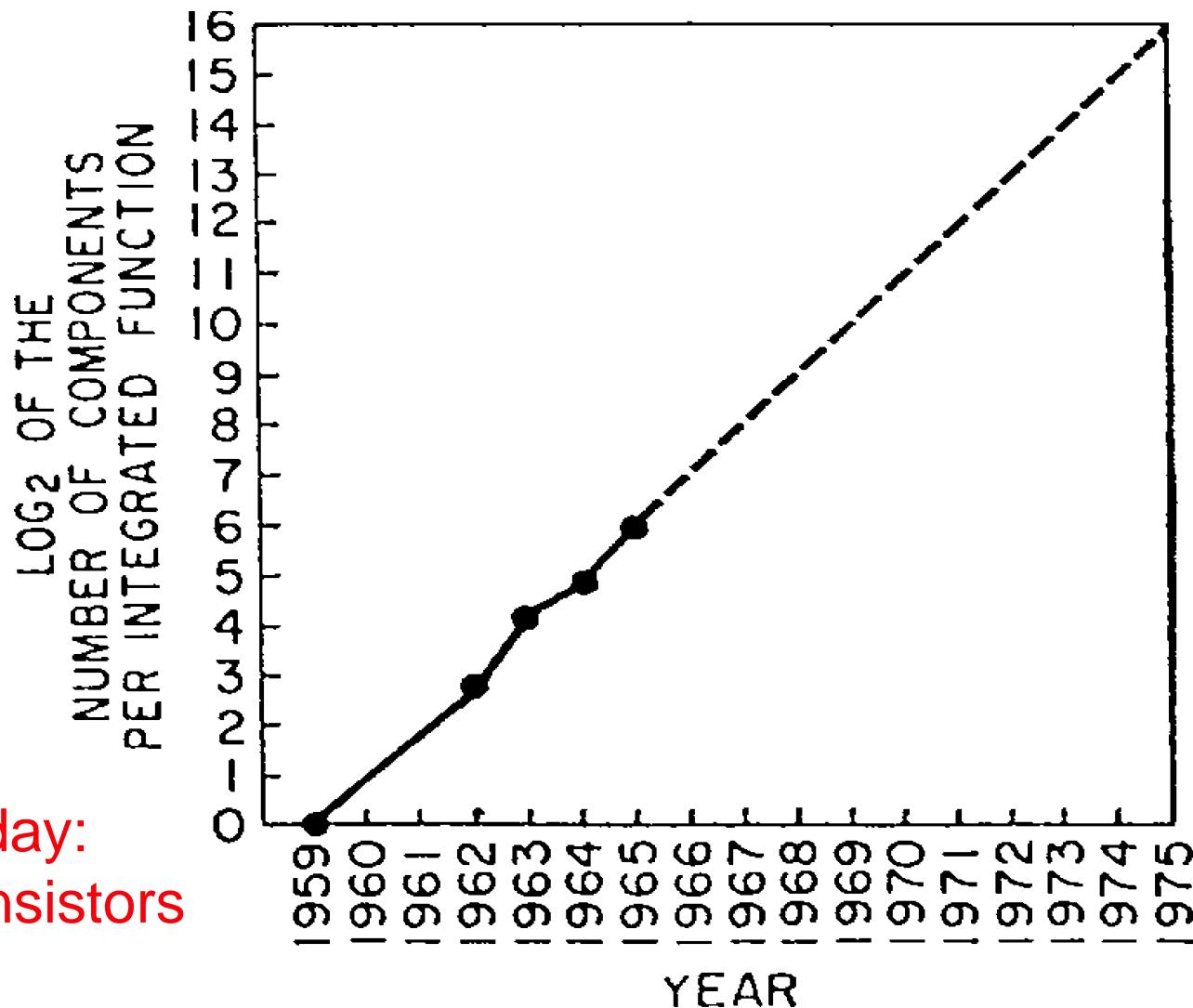
- Basic element
 - Solid-state **transistor**
 - Building block of **integrated circuits (ICs)**
- What's so great about ICs? Everything
 - + High performance, high reliability, low cost, low power
 - + Lever of mass production
- Several kinds of integrated circuit families
 - **SRAM/logic**: optimized for speed (used for processors)
 - **DRAM**: optimized for density, cost, power (used for memory)
 - **Flash**: optimized for density, cost (used for disks)
 - Increasing opportunities for integrating multiple technologies
- Non-transistor storage and inter-connection technologies
 - Disk, optical storage, ethernet, fiber optics, wireless



"Integrated circuits will lead to such wonders as home computers—or at least terminals connected to a central computer—automatic controls for automobiles, and personal portable communications equipment."

- Gordon Moore, 1965

Moore's Law - 1965



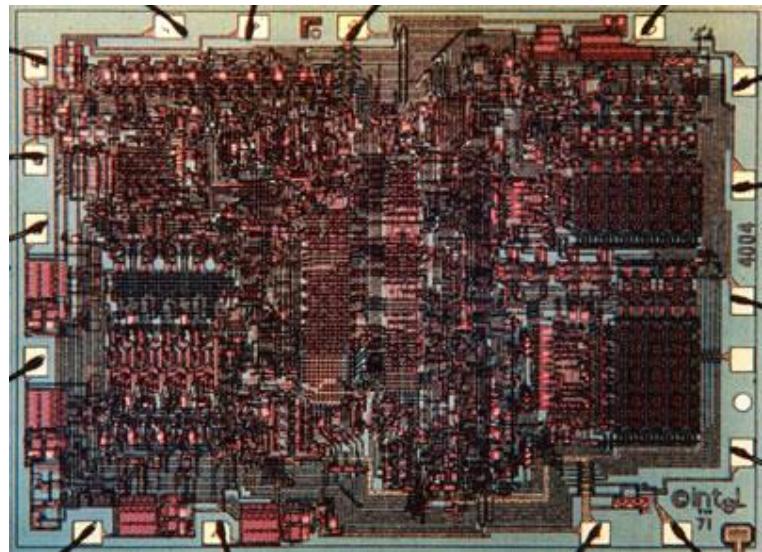
Today:
 2^{32} transistors

Revolution I: The Microprocessor

- **Microprocessor revolution**
 - One significant technology threshold was crossed in 1970s
 - Enough transistors ($\sim 25K$) to put a 16-bit processor on one chip
 - Huge performance advantages: fewer slow chip-crossings
 - Even bigger cost advantages: one “stamped-out” component
- Microprocessors have allowed new market segments
 - Desktops, CD/DVD players, laptops, game consoles, set-top boxes, mobile phones, digital camera, mp3 players, GPS, automotive
- And replaced incumbents in existing segments
 - Microprocessor-based system replaced supercomputers, “mainframes”, “minicomputers”, etc.

First Microprocessor

- Intel 4004 (1971)
 - Application: calculators
 - Technology: 10000 nm
 - 2300 transistors
 - 13 mm²
 - 108 KHz
 - 12 Volts
 - 4-bit data
 - Single-cycle datapath



Tracing the Microprocessor Revolution

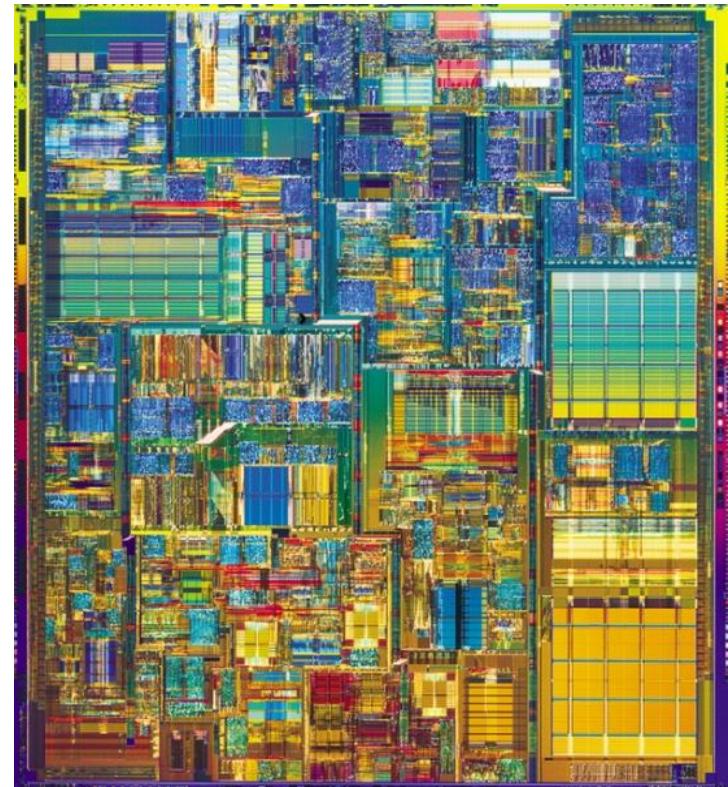
- How were growing transistor counts used?
- Initially to widen the datapath
 - 4004: 4 bits → AMD Opteron (2003): 64 bits
- ... and also to add more powerful instructions
 - To amortize overhead of fetch and decode
 - To simplify assembly programming (which was done by hand then)

Revolution II: Implicit Parallelism

- Then to **extract implicit instruction-level parallelism**
 - Hardware provides parallel resources, figures out how to use them
 - Software is oblivious
- Initially using pipelining ...
 - Which also enabled increased clock frequency
- ... caches ...
 - Which became necessary as processor clock frequency increased
- ... and integrated floating-point
- Then deeper pipelines and branch speculation
- Then multiple instructions per cycle (superscalar)
- Then dynamic scheduling (out-of-order execution)
- We will discuss this in detail

Pinnacle of Single-Core Microprocessors

- Intel Pentium4 (2003)
 - Application: desktop/server
 - Technology: 90nm (1/100th of 4004)
 - 55M transistors (20,000x)
 - 101 mm² (10x)
 - 3.4 GHz (10,000x)
 - 1.2 Volts (1/10th)
 - 32/64-bit data (16x)
 - 22-stage pipelined datapath
 - 3 instructions per cycle (superscalar)
 - Two levels of on-chip cache
 - data-parallel vector (SIMD) instructions, hyperthreading

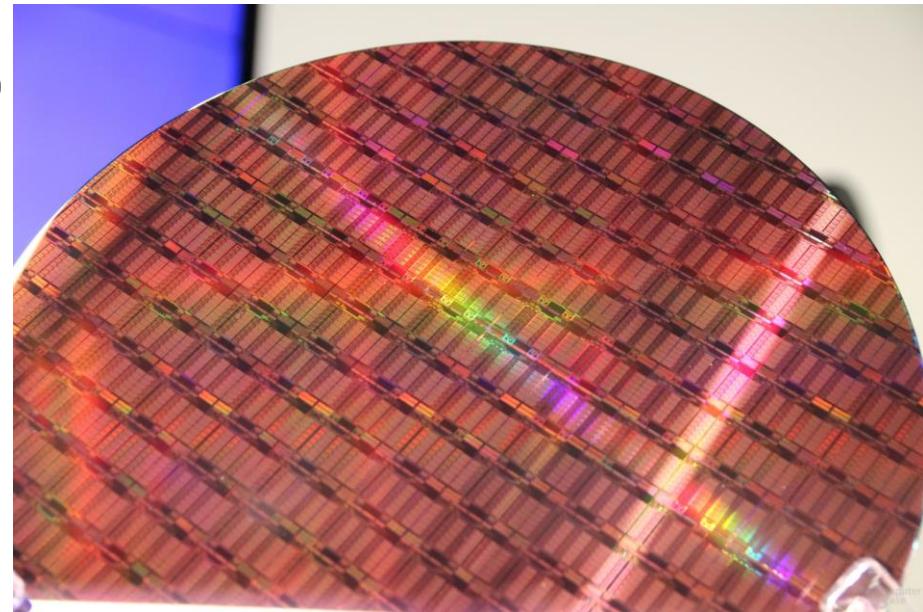


Revolution III: Explicit Parallelism

- Then to support **explicit data & thread level parallelism**
 - Hardware provides parallel resources, software specifies usage
 - Why? diminishing returns on instruction-level-parallelism
- First using (subword) vector instructions..., Intel's SSE
 - One instruction does four parallel multiplies
- ... and general support for multi-threaded programs
 - Coherent caches, hardware synchronization primitives
- Then using support for multiple concurrent threads on chip
 - First with single-core multi-threading, now with multi-core

Modern Multicore Processor

- Intel Xeon E5-2699 V4 (2016)
 - Application: server
 - Technology: 14nm (16% of P4)
 - 7.2B transistors (130x)
 - 456 mm² (4.5x)
 - 2.4 to 3.6 Ghz (~1x)
 - 1.8 Volts (~1x)
 - 256-bit data (2x)
 - 14-stage pipelined datapath (0.5x)
 - 4 instructions per cycle (1x)
 - Three levels of on-chip cache
 - data-parallel vector (SIMD) instructions, hyperthreading
 - **22-core multicore** (22x)



Revolution IV: Heterogeneous Processing

- Combining **multiple** kinds of compute engines in one die
 - not just homogenous collection of cores
 - System-on-Chip (SoC) is one common example in mobile space
- Lots of stuff on the chip beyond just CPUs
 - Graphics Processing Units (GPUs)
 - throughput-oriented specialized multicore processors
 - good for gaming, machine learning, computer vision, ...
 - Special-purpose logic
 - media codecs, cell phone radio, encryption, compression
- Excellent energy efficiency and performance
 - extremely complicated to program!

Technology Disruptions

- Classic examples:
 - The transistor
 - Microprocessor
- More recent examples:
 - Multicore processors
 - Flash-based solid-state storage
- Near-term potentially disruptive technologies:
 - Non-volatile memory (e.g., phase-change memory)
 - Chip stacking (aka “3D die stacking”)
- Disruptive “end-of-scaling”
 - “If something can’t go on forever, it must stop eventually”
 - Can we continue to shrink transistors for ever?
 - Transistor energy efficiency is not improving as rapidly

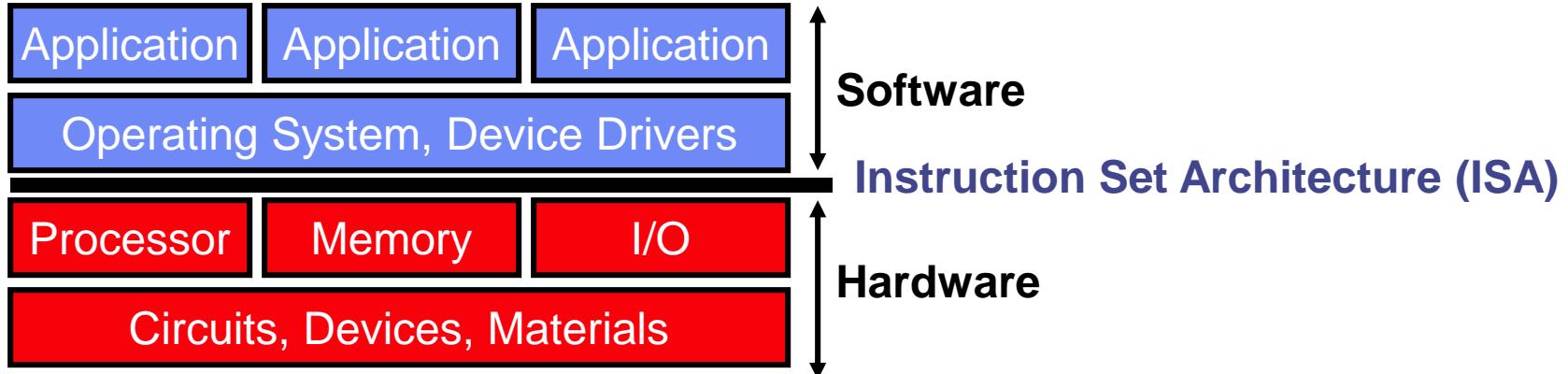
Managing This Mess

- Architect must consider all factors
 - Goals/constraints, applications, implementation technology
- Questions
 - How to deal with all of these inputs?
 - How to manage changes?
- Answers
 - Accrued institutional knowledge (stand on each other's shoulders)
 - Experience, rules of thumb
 - Discipline: clearly defined end state, keep your eyes on the ball
 - **Abstraction and layering**

Pervasive Ideas: Abstraction and Layering

- **Abstraction**: only way of dealing with complex systems
 - Divide world into objects, each with an...
 - **Interface**: knob(s)
 - **Implementation**: “black box”
 - Only specialists deal with implementation, rest of us with interface
- **Layering**: repeated abstraction makes life even simpler
 - Divide objects in system into layers, objects at layer n :
 - are implemented using interfaces of layer $n - 1$
 - don't need to know interfaces of layer $n - 2$ (sometimes helps)
- **Inertia**: the dark side of layering
 - Layer interfaces become entrenched over time (“standards”)
 - Very difficult to change even if benefit is clear (example: Digital TV)
- **Opacity**: hard to reason about performance across layers

Abstraction, Layering, and Computers



- **Computer architecture**
 - Definition of **ISA** to facilitate implementation of software layers
- This course mostly on **computer micro-architecture**
 - Design of chip to implement **ISA**
- Touch on compilers & OS ($n + 1$), circuits ($n - 1$) as well

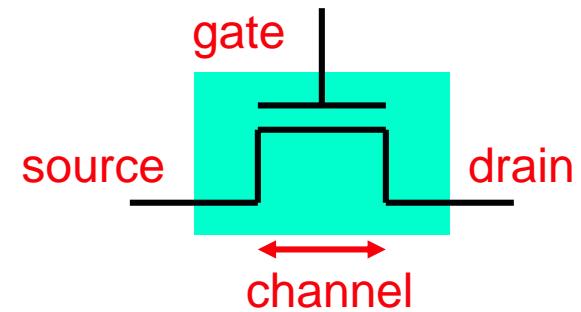
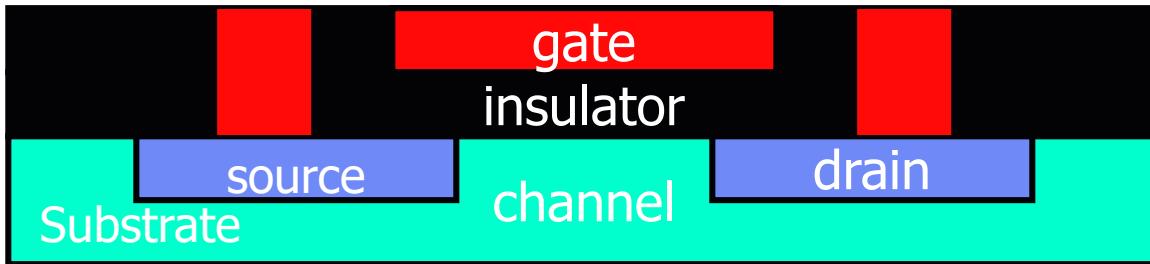
Technology & Energy

Technology & Energy

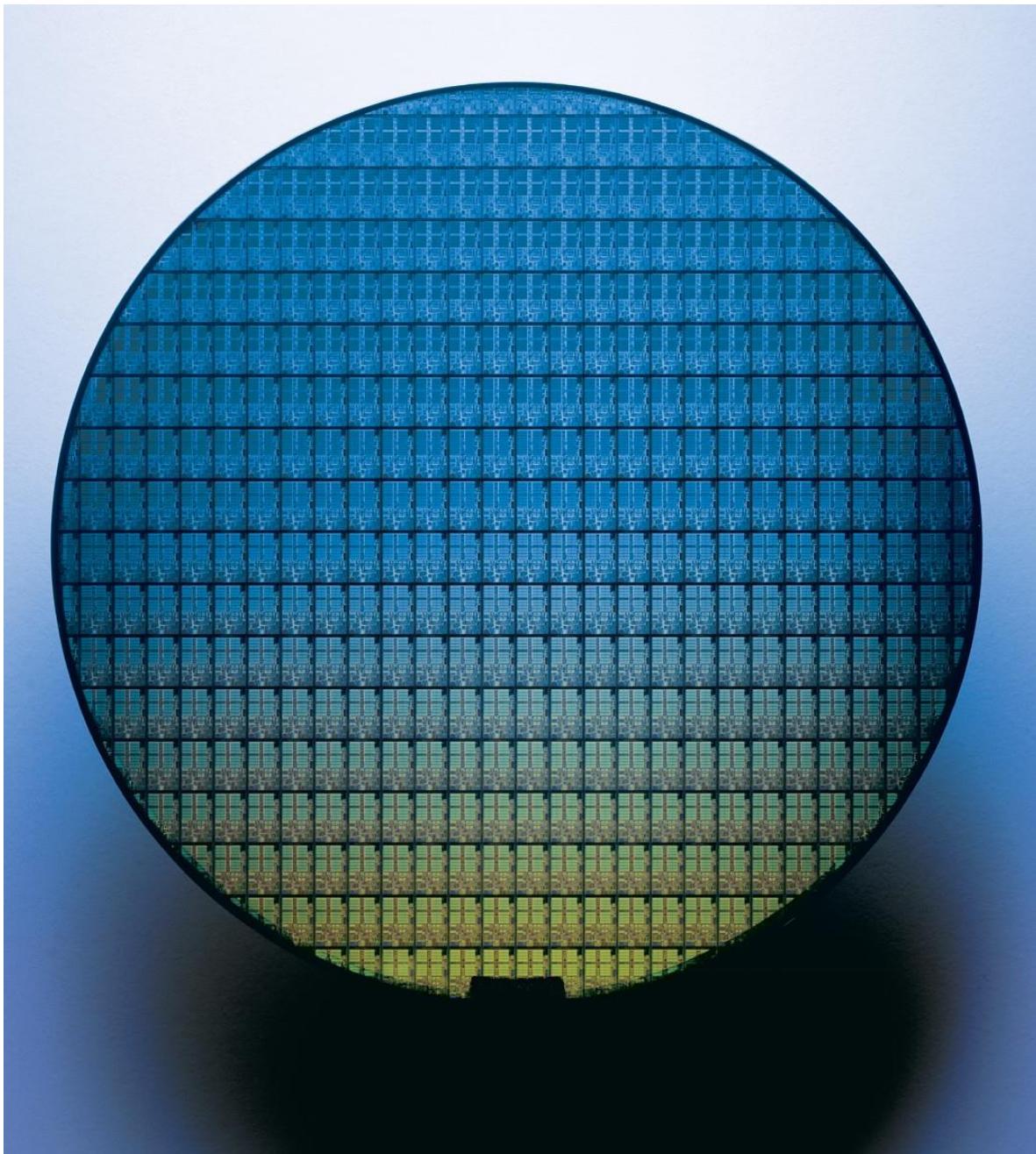
- Technology basis
 - Fabrication (manufacturing) & cost
 - Transistors & wires
 - Implications of transistor scaling (Moore's Law)
- Energy & power

Technology & Fabrication

Semiconductor Technology



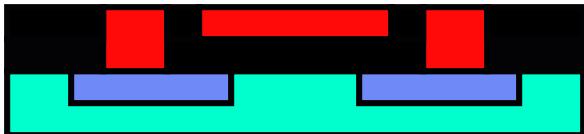
- Basic technology element: **MOSFET**
 - Solid-state component acts like electrical switch
 - **MOS**: metal-oxide-semiconductor
 - Conductor, insulator, semi-conductor
- **FET**: field-effect transistor
 - Channel conducts source→drain only when voltage applied to gate
- **Channel length**: characteristic parameter (short → fast)
 - Aka “feature size” or “technology node”
 - Currently: 14 nanometers (nm)
 - Continued miniaturization (scaling) known as “**Moore’s Law**”
 - Won’t last forever, physical limits approaching (or are they?)



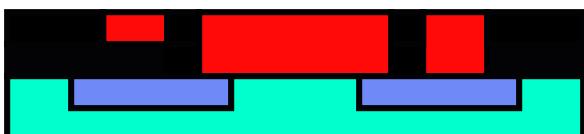
Intel
Pentium M
Wafer

Manufacturing Defects

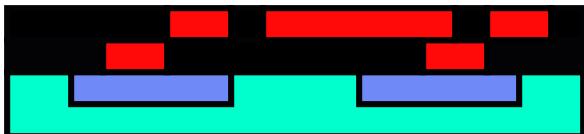
Correct:



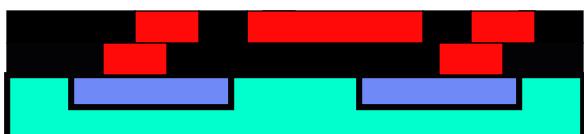
Defective:



Defective:



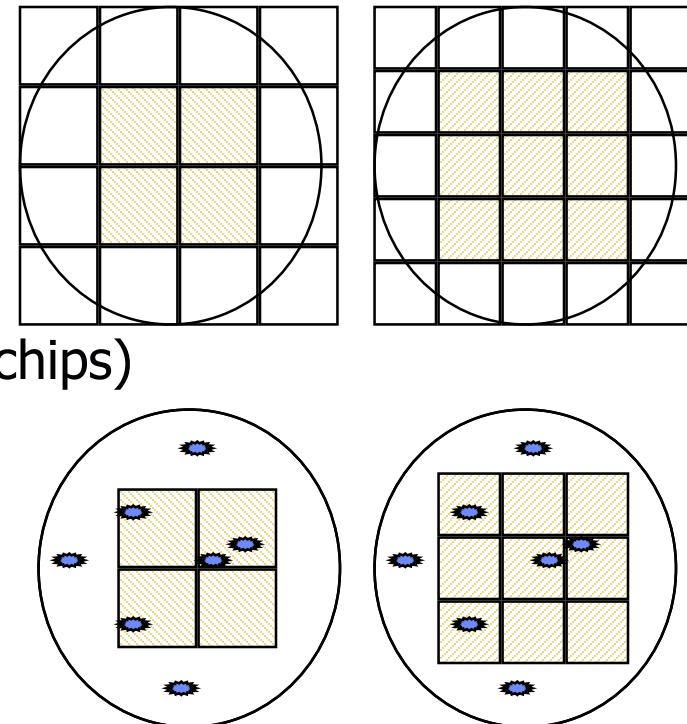
Slow:



- Defects can arise
 - Under-/over-doping
 - Over-/under-dissolved insulator
 - Mask mis-alignment
 - Particle contaminants
- Try to minimize defects
 - Process margins
 - Design rules
 - Minimal transistor size, separation
- Or, tolerate defects
 - Redundant or “spare” memory cells
 - Can substantially improve yield

Cost Implications of Defects

- Chips built in multi-step chemical processes on **wafers**
 - Cost / wafer is constant, $f(\text{wafer size, number of steps})$
- Chip (die) cost is related to **area**
 - Larger chips means fewer of them
- Cost is **superlinear** in area
 - Why? random defects
 - **Larger chip, more chance of defect**
 - Result: lower “yield” (fewer working chips)
- **Wafer yield**: % wafer that is chips
- **Die yield**: % chips that work
- Yield is increasingly non-binary - fast vs slow chips



Manufacturing Cost

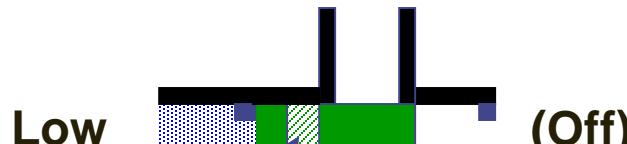
- **Chip cost vs system cost**
 - Cost of memory, storage, display, battery, etc.
- **Cost vs price**
 - Relationship complicated; **microprocessors not commodities**
 - Specialization, compatibility, different cost/performance/power
 - Economies of scale
- **Unit costs:** die manufacturing, testing, packaging, burn-in
 - Die cost based on area & defect rate (yield)
 - Package cost related to heat dissipation & number of pins
- **Fixed costs:** design & verification, fab cost
 - Amortized over “proliferations”, e.g., Core i3, i5, i7 variants
 - Building new “fab” costs billions of dollars today
 - Both getting worse; trend toward “foundry” & “fabless” models

Transistor Switching Speed

A Transistor Analogy: Computing with Air

- Use air pressure to encode values
 - High pressure represents a “1” (blow)
 - Low pressure represents a “0” (suck)
- Valve can allow or disallow the flow of air
 - Two types of valves

N-Valve



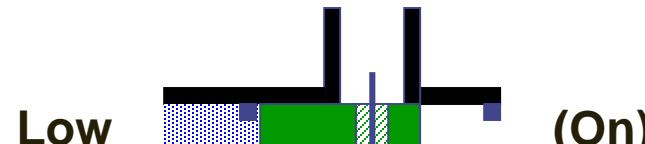
(Off)

Low

hole

High

P-Valve



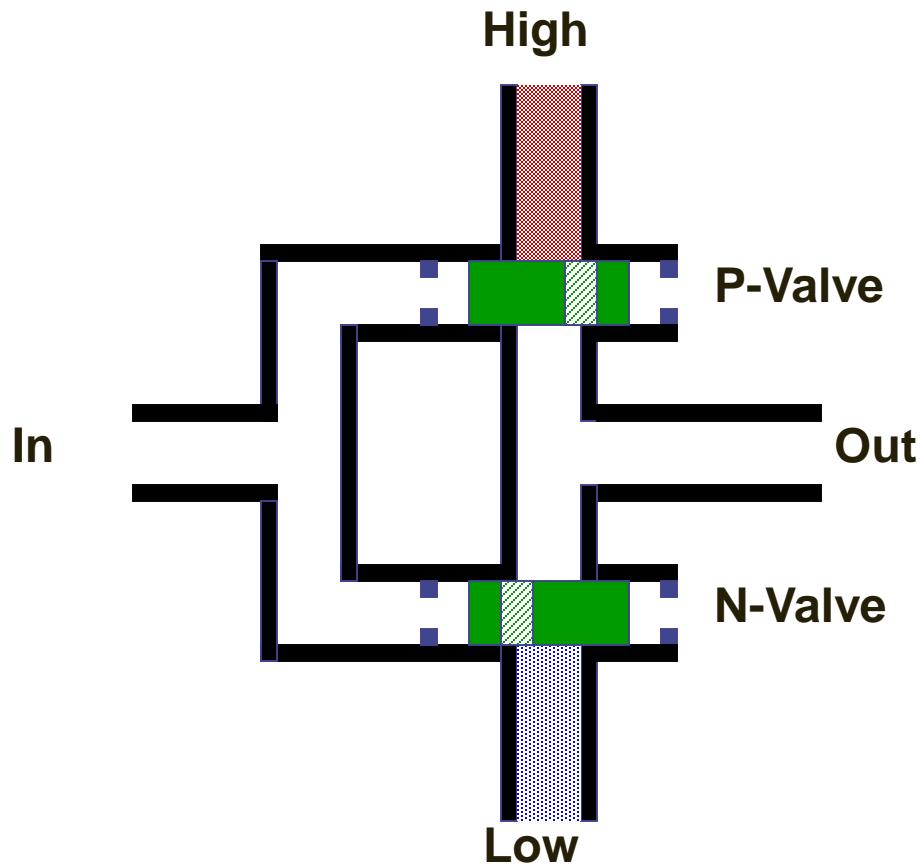
(On)

Low

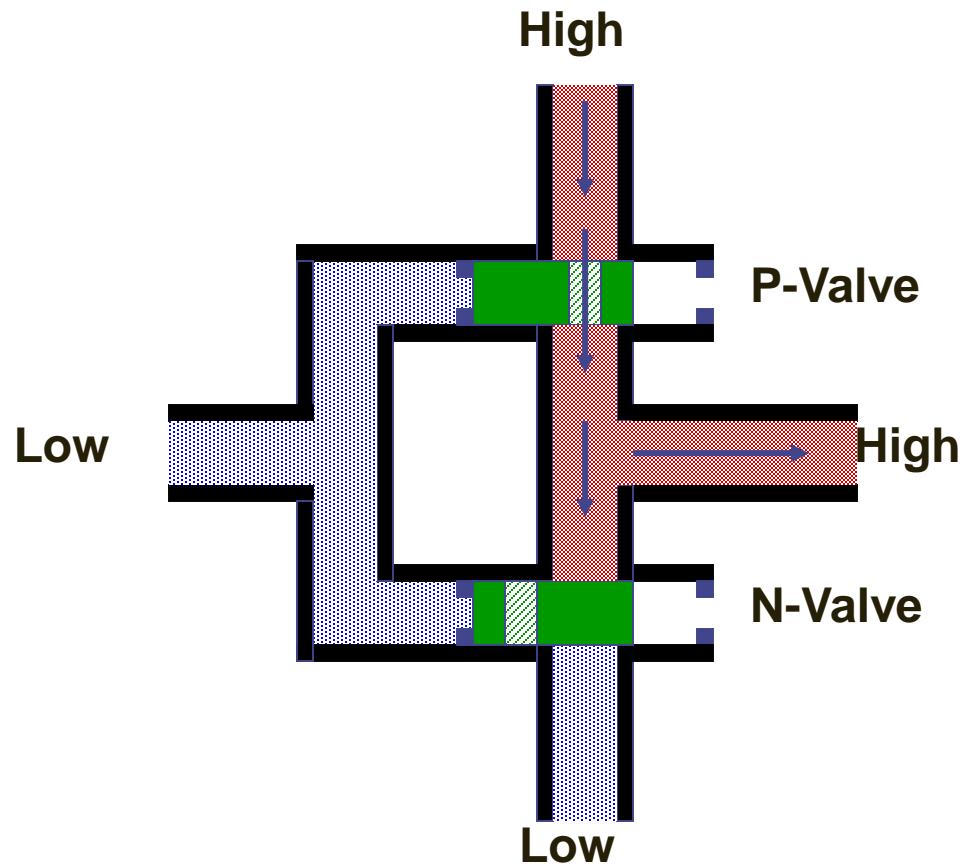
High

(Off)

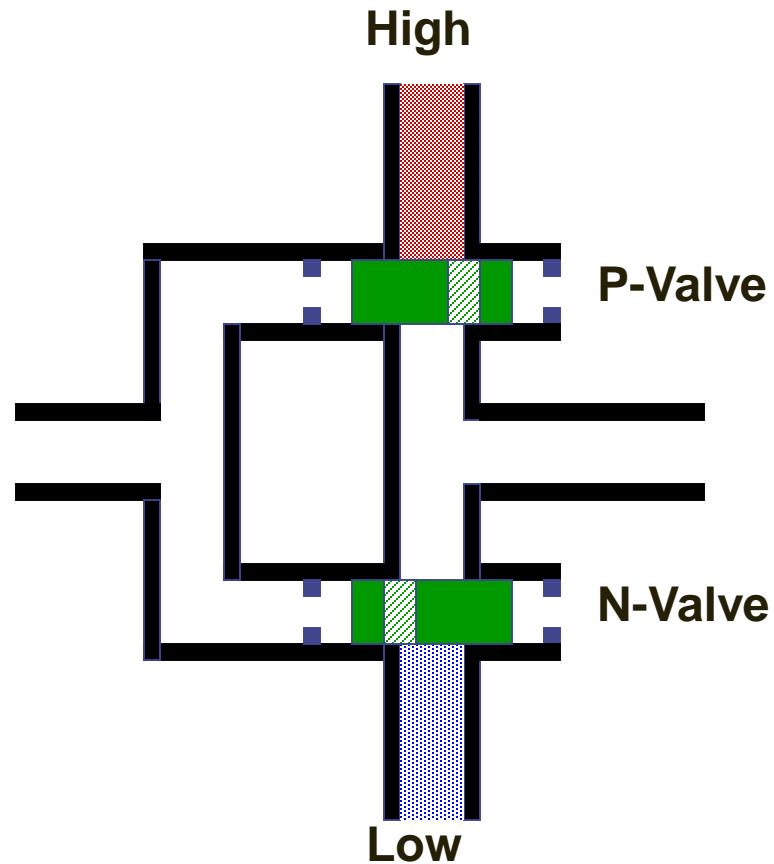
Pressure Inverter



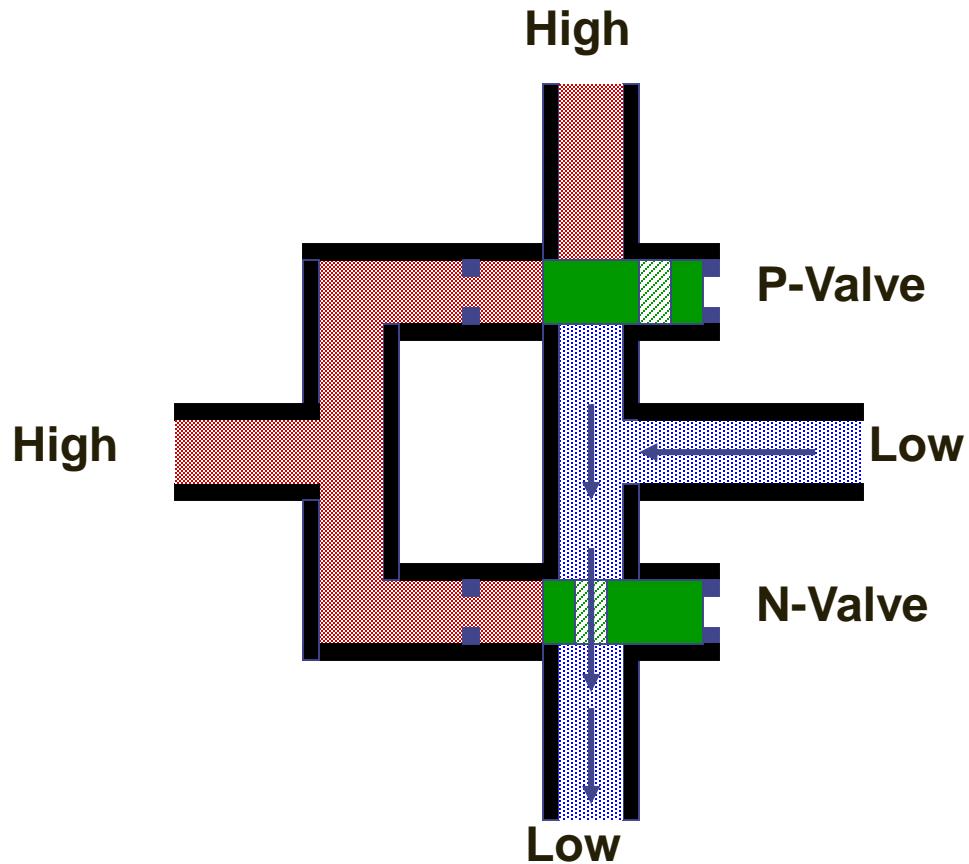
Pressure Inverter (Low to High)



Pressure Inverter



Pressure Inverter (High to Low)

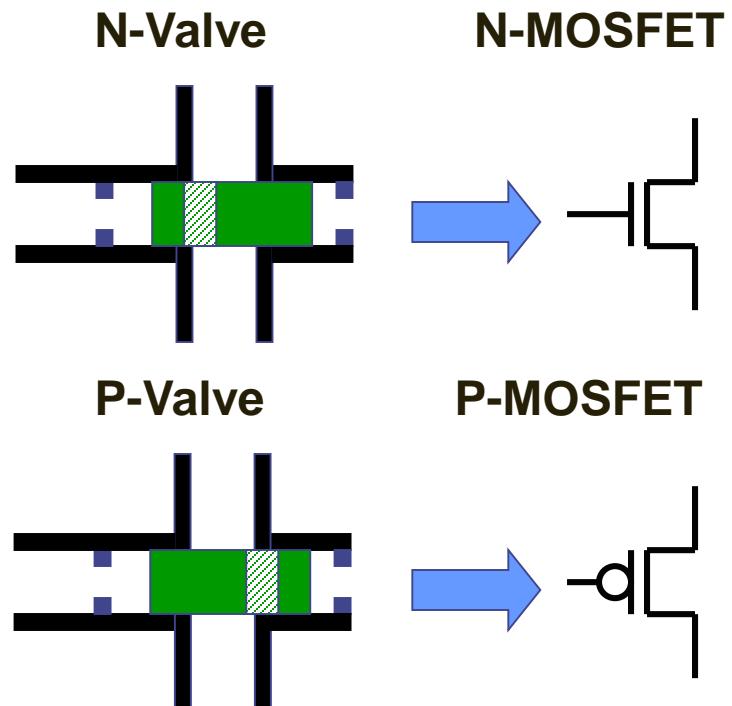


Analogy Explained

- Pressure differential → electrical potential (voltage)
 - Air molecules → electrons
 - Pressure (molecules per volume) → voltage
 - High pressure → high voltage
 - Low pressure → low voltage
- Air flow → electrical current
 - Pipes → wires
 - Air only flows from high to low pressure
 - Electrons only flow from high to low voltage
 - Flow only occurs when changing from 1 to 0 or 0 to 1
- Valve → transistor
 - The transistor: one of the century's most important inventions

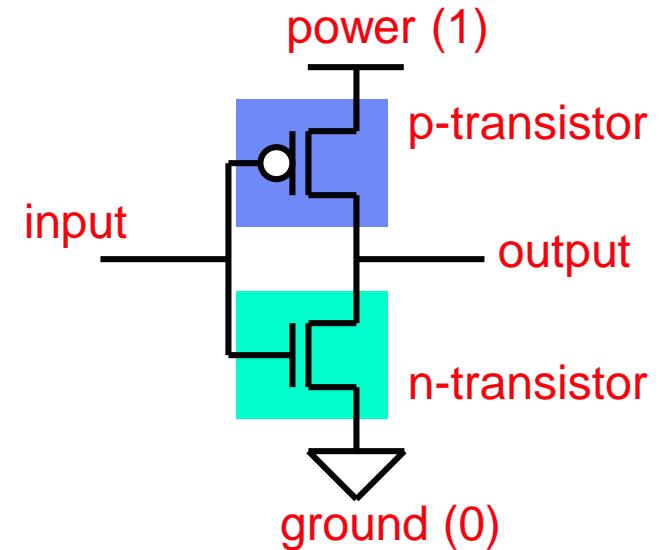
Transistors as Switches

- Two types
 - N-type
 - P-type
- Properties
 - Solid state (no moving parts)
 - Reliable (low failure rate)
 - Small (14nm channel length)
 - Fast (<0.1ns switch latency)



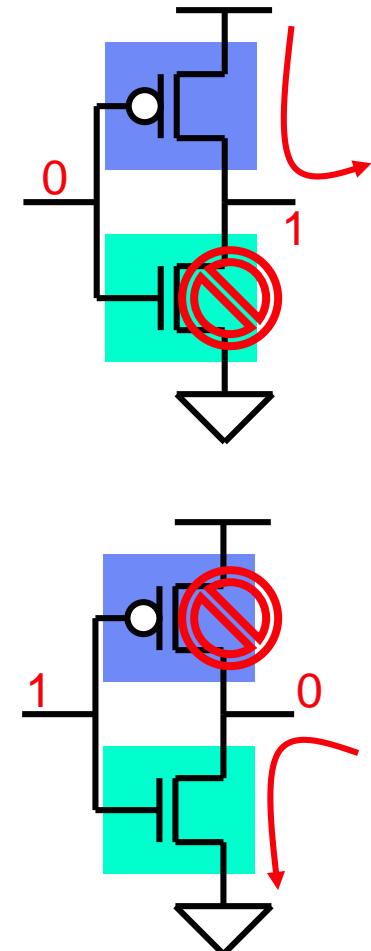
Complementary MOS (CMOS)

- Voltages as values
 - Power (V_{DD}) = "1", Ground = "0"
- Two kinds of MOSFETs
 - **N-transistors**
 - Conduct when gate voltage is 1
 - Good at passing 0s
 - **P-transistors**
 - Conduct when gate voltage is 0
 - Good at passing 1s
- **CMOS**
 - Complementary n/p- networks form boolean logic (i.e., gates)
 - And some non-gate elements too (important example: RAMs)



Basic CMOS Logic Gate

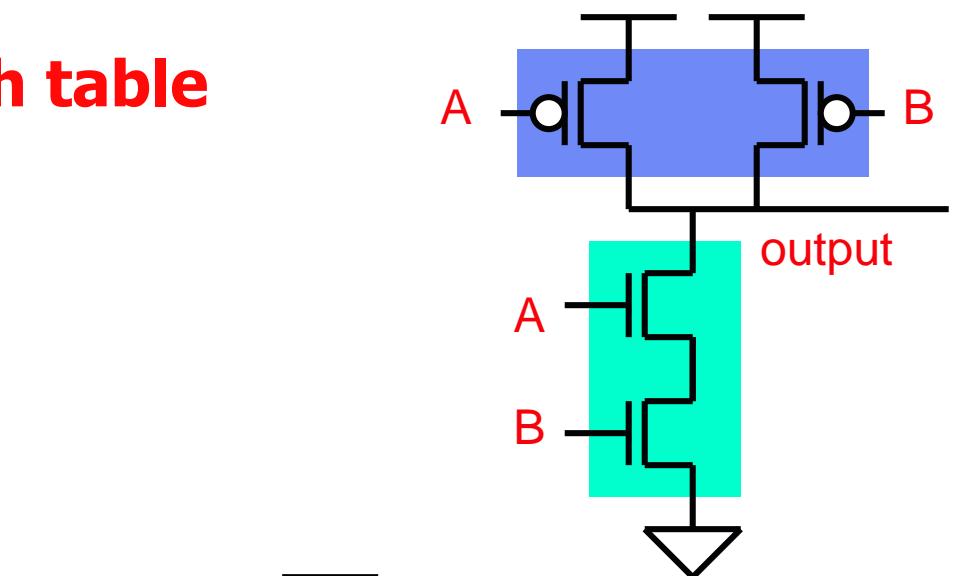
- **Inverter:** NOT gate
 - One p-transistor, one n-transistor
 - Basic operation
 - Input = 0
 - P-transistor closed, n-transistor open
 - Power charges output (1)
 - Input = 1
 - P-transistor open, n-transistor closed
 - Output discharges to ground (0)



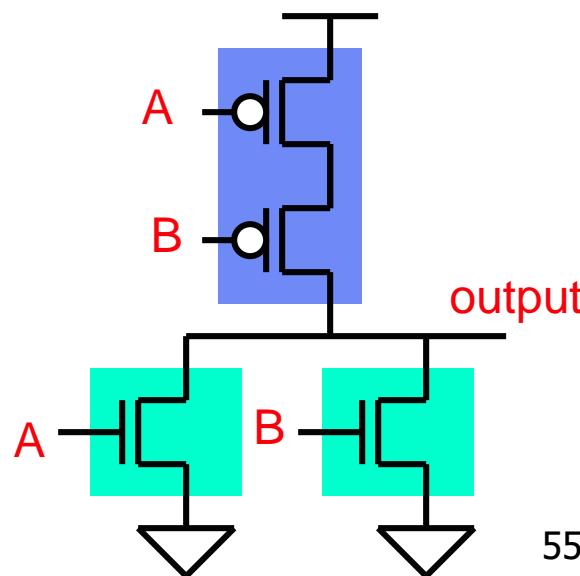
Another CMOS Gate Example

- What is this? Look at **truth table**

- 0, 0 → 1
- 0, 1 → 1
- 1, 0 → 1
- 1, 1 → 0
- Result: **NAND** (NOT AND)
- NAND is “universal”



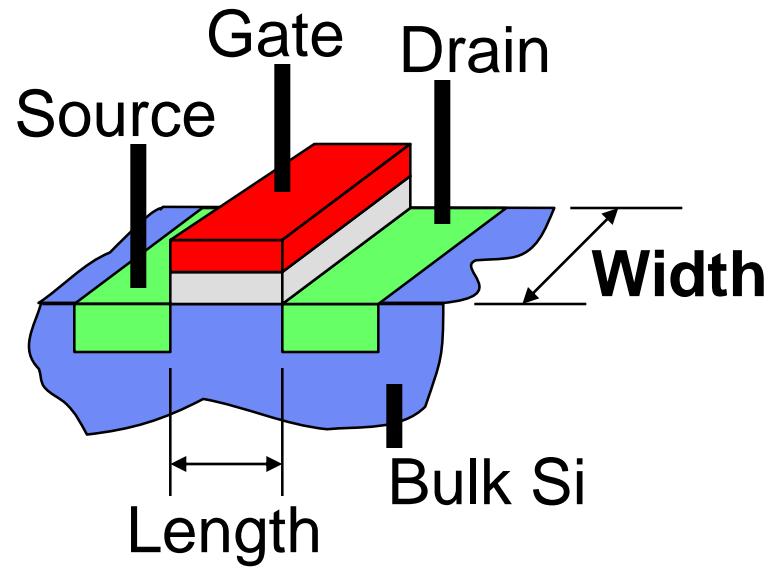
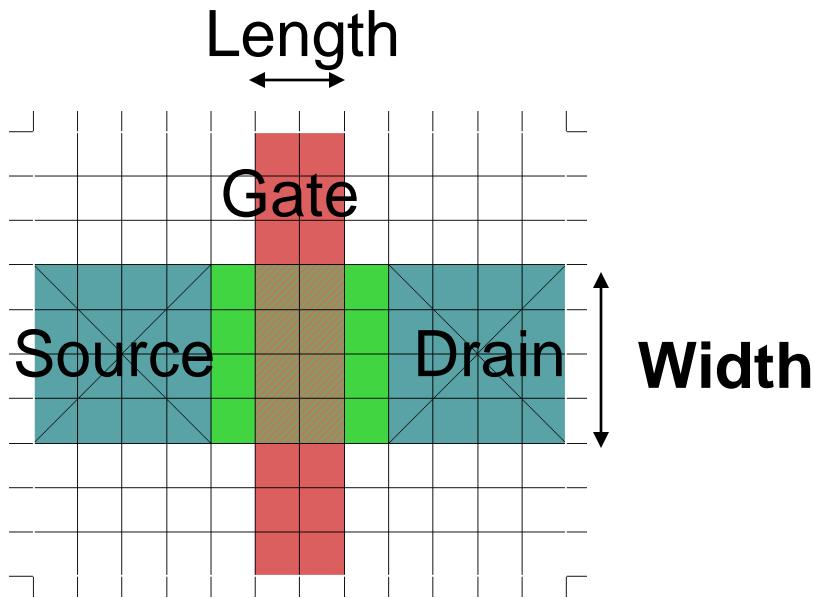
- What function is this?



Technology Basis of Transistor Speed

- Physics 101: delay through an electrical component $\propto \text{RC}$
 - **Resistance (R)**  \propto length / cross-section area
 - Slows rate of charge flow
 - **Capacitance (C)**  \propto length * area / distance-to-other-plate
 - Stores charge
 - **Voltage (V)**
 - Electrical pressure
 - **Threshold Voltage (V_t)**
 - Voltage at which a transistor turns “on”
 - Property of transistor based on fabrication technology
 - **Switching time** $\propto (\mathbf{R} * \mathbf{C}) / (\mathbf{V} - \mathbf{V}_t)$
- Two kinds of electrical components
 - CMOS transistors (gates, sources, drains)
 - Wires

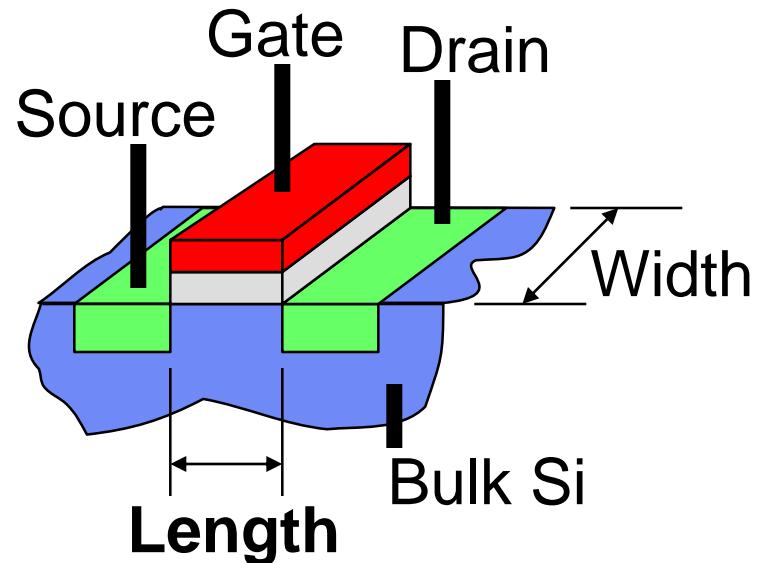
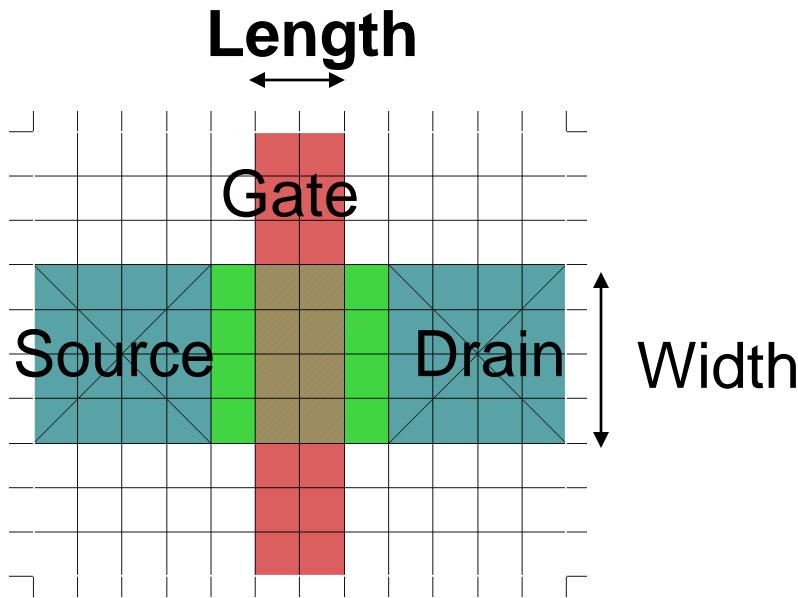
Transistor Geometry: Width



Diagrams © Krste Asanovic, MIT

- **Transistor width**, set by designer for each transistor
- Wider transistors:
 - **Lower resistance** of channel (increases drive strength) – good!
 - But, **increases capacitance** of gate/source/drain – bad!
- Result: set width to balance these conflicting effects

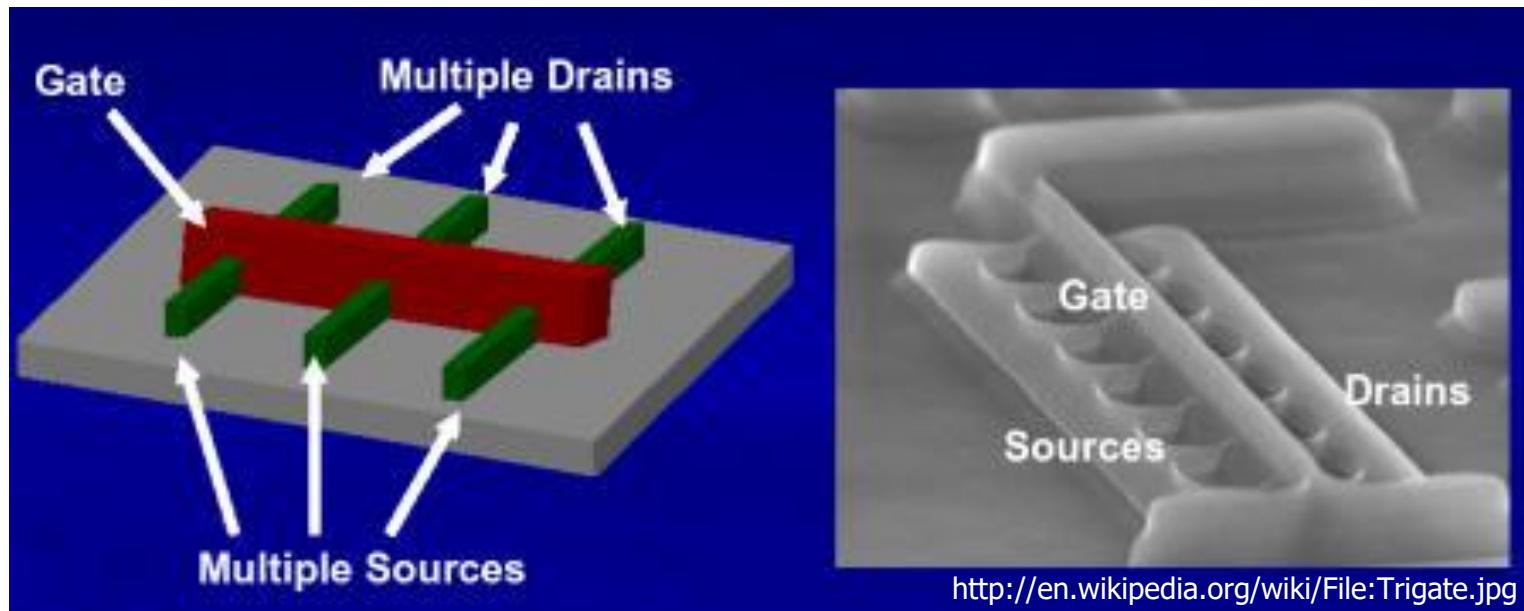
Transistor Geometry: Length & Scaling



Diagrams © Krste Asanovic, MIT

- **Transistor length:** characteristic of “process generation”
 - “22nm” refers to the transistor gate length
- Each process generation shrinks transistor length by 1.4x
 - “Moore’s law” -> roughly 2x improvement in transistor density
 - Roughly linear improvement in switching speeds (lower resistance)

Trigate FinFET Transistors



- nonplanar (or “3D”) transistors
 - trigate: multiple sources/drains/gates
 - FinFET: gate is wrapped around the channel
- lower leakage, faster switching times
- Intel’s trigate design released in mid-2012 (Ivy Bridge)
 - TSMC FinFET in mid-2015, Samsung FinFET in early 2016

Technology Scaling Trends

Dennard Scaling

“Design of ion-implanted MOSFET’s with very small physical dimensions”
Robert H. Dennard, Fritz H. Gaensslen, Hwa-Nien Yu, V. Leo Rideout, Ernest
Bassous, and Andre R. LeBlanc
IEEE Journal of Solid-State Circuits, October 1974

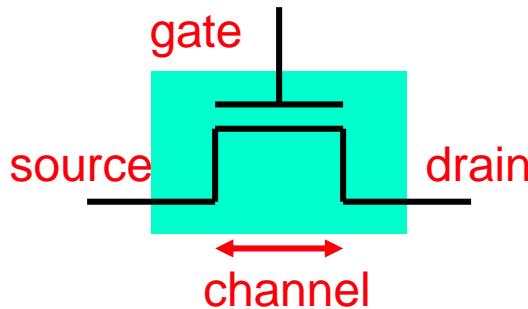
Table 1
Scaling Results for Circuit Performance

Device or Circuit Parameter	Scaling Factor
Device dimension t_{ox} , L , W	$1/\kappa$
Doping concentration N_a	κ
Voltage V	$1/\kappa$
Current I	$1/\kappa$
Capacitance $\epsilon A/t$	$1/\kappa$
Delay time/circuit VC/I	$1/\kappa$
Power dissipation/circuit VI	$1/\kappa^2$
Power density VI/A	1

Dennard Scaling

- stopped in ~2005 due to leakage
 - V close to V_t , transistors never really “on” or “off”
 - gate-oxide leakage due to very small oxide thickness
 - quantum-mechanical electron tunneling
- Moore’s Law still in effect!
 - but power usage goes up with increasing transistor counts

Moore's Law: Technology Scaling



- **Moore's Law:** aka “technology scaling”
 - Continued miniaturization (esp. reduction in channel length)
 - + Improves switching speed, power/transistor, area(cost)/transistor
 - Reduces transistor reliability
 - Literally: DRAM density (transistors/area) doubles every 18 months
 - Public interpretation: performance doubles every 18 months
 - Not quite right, but helps performance in several ways...

Moore's Law in the Future

- Won't last forever, approaching physical limits
 - "If something must eventually stop, it can't go on forever"
 - But betting against it has proved foolish in the past
 - Perhaps will "slow" rather than stop abruptly
- Transistor count will likely continue to scale
 - 3D "die stacking" is arriving
- But transistor performance scaling?
 - Running into physical limits
 - Example: gate oxide < 10 silicon atoms!
 - Can't decrease it much further
 - Power is becoming the limiting factor

The image shows a screenshot of a news article from AnandTech. The header features the AnandTech logo and navigation icons (menu, home, search). Below the header, the breadcrumb navigation shows "Home > CPUs". The main title of the article is "Tick Tock On The Rocks: Intel Delays 10nm, Adds 3rd Gen 14nm Core Product 'Kaby Lake'". A byline indicates the article was written by Brett Howse & Ryan Smith on July 16, 2015 at 10:15 AM EST. There are 136 comments available. The bottom right corner of the image displays the large blue Intel logo.

Power & Energy

Power/Energy Are Increasingly Important

- **Battery life** for mobile devices
 - Laptops, phones, cameras
- **Tolerable temperature** for devices without active cooling
 - Power means temperature, active cooling means **cost**
 - No room for a fan in a cell phone, no market for a hot cell phone
- **Electric bill** for compute/data centers
 - Pay for power twice: once in, once out (to cool)
- **Environmental concerns**
 - IT accounts for growing fraction of electricity consumption

Energy & Power

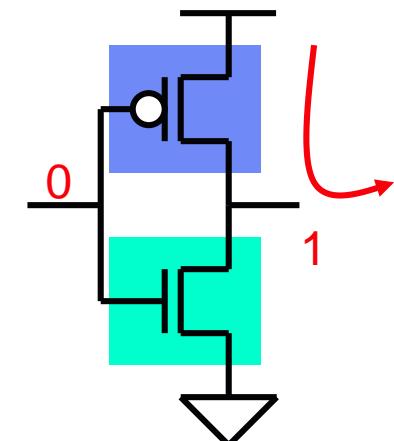
- **Energy:** measured in Joules or Watt-seconds
 - Total amount of energy stored/used
 - Battery life, electric bill, environmental impact
 - Instructions per Joule (car analogy: miles per gallon)
- **Power:** energy per unit time (measured in Watts)
 - Related to “performance” (which is also a “per unit time” metric)
 - Power impacts power supply and cooling requirements (cost)
 - Power-density (Watt/mm²): important related metric
 - Peak power vs average power
 - E.g., camera: power “spikes” when you actually take a picture
 - Joules per second (car analogy: gallons per hour)
- Two sources:
 - **Dynamic power:** active switching of transistors
 - **Static power:** leakage of transistors even while inactive

Recall: Tech. Basis of Transistor Speed

- Physics 101: delay through an electrical component $\propto \text{RC}$
 - **Resistance (R)**  \propto length / cross-section area
 - Slows rate of charge flow
 - **Capacitance (C)**  \propto length * area / distance-to-other-plate
 - Stores charge
 - **Voltage (V)**
 - Electrical pressure
 - **Threshold Voltage (V_t)**
 - Voltage at which a transistor turns “on”
 - Property of transistor based on fabrication technology
 - **Switching time** $\propto (\mathbf{R} * \mathbf{C}) / (\mathbf{V} - \mathbf{V}_t)$

Dynamic Power

- **Dynamic power ($P_{dynamic}$)**: aka switching or active power
 - Energy to switch a gate (0 to 1, 1 to 0)
 - Each gate has capacitance (C)
 - Charge stored $\propto C * V$
 - Energy to charge/discharge a capacitor $\propto C * V^2$
 - Time to charge/discharge a capacitor $\propto V$
 - Result: frequency $\propto V$
 - $P_{dynamic} \approx N * C * V^2 * f * A$
 - N: number of transistors
 - C: capacitance per transistor (size of transistors)
 - V: voltage (supply voltage for gate)
 - f: frequency (transistor switching freq. \propto clock freq.)
 - A: activity factor (not all transistors may switch this cycle)

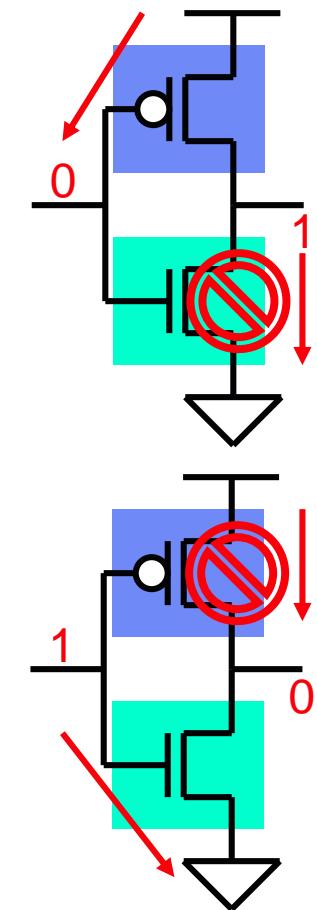


Reducing Dynamic Power

- Target each component: $P_{\text{dynamic}} \approx N * C * V^2 * f * A$
- **Reduce number of transistors (N)**
 - Use fewer transistors and gates
- **Reduce capacitance (C)**
 - Smaller transistors (Moore's law)
- **Reduce voltage (V)**
 - Quadratic reduction in energy consumption!
 - But also slows transistors (transistor speed $\propto V$)
- **Reduce frequency (f)**
 - Slower clock frequency (reduces power but not energy) Why?
- **Reduce activity (A)**
 - “Clock gating” disable clocks to unused parts of chip
 - Don’t switch gates unnecessarily

Static Power

- **Static power (P_{static})**: aka idle or leakage power
 - Transistors don't turn off all the way
 - Transistors are “leaky valves”
 - $P_{\text{static}} \approx N * V * e^{-V_t}$
 - N: number of transistors
 - V: voltage
 - **V_t (threshold voltage)**: voltage at which transistor conducts (begins to switch)
- Switching speed vs leakage trade-off
- The lower the V_t :
 - Faster transistors (linear)
 - Transistor speed $\propto V - V_t$
 - Leakier transistors (exponential)



Reducing Static Power

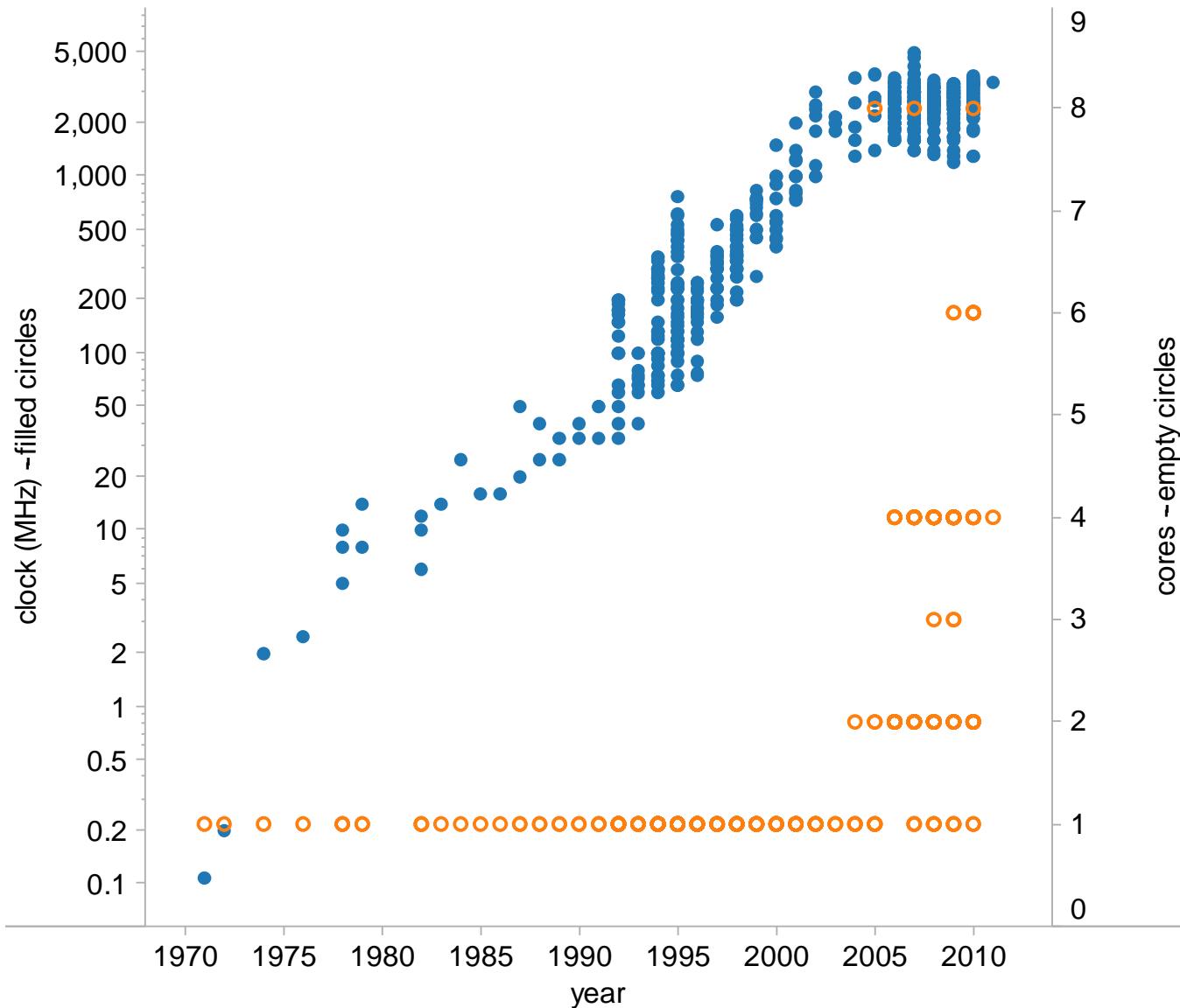
- Target each component: $P_{\text{static}} \approx N * V * e^{-Vt}$
- **Reduce number of transistors (N)**
 - Use fewer transistors/gates
- **Disable transistors** (also targets N)
 - “Power gating” disable power to unused parts (long latency to power up)
 - Power down units (or entire cores) not being used
- **Reduce voltage (V)**
 - Linear reduction in static energy consumption
 - But also slows transistors ($\text{transistor speed} \propto V$)
- **Dual V_t** – use a mixture of high and low V_t transistors
 - Use slow, low-leak transistors in SRAM arrays
 - Requires extra fabrication steps (cost)
- **Low-leakage transistors**
 - High-K/Metal-Gates in Intel’s 45nm process, “tri-gate” in Intel’s 22nm
 - Reducing frequency can hurt energy efficiency due to leakage power

Dynamic Voltage/Frequency Scaling

- **Dynamically trade-off power for performance**
 - Change the voltage and frequency at runtime
 - Under control of operating system and/or hardware
- Recall: $P_{\text{dynamic}} \approx N * C * V^2 * f * A$
 - Because frequency \propto to $V - V_t$...
 - $P_{\text{dynamic}} \propto$ to $V^2(V - V_t) \approx V^3$
- Reduce both voltage and frequency linearly
 - **Cubic decrease in dynamic power**
 - Linear decrease in performance (actually sub-linear)
 - Thus, only about quadratic decrease in energy
 - Linear decrease in static power
 - Thus, static energy can become dominant
- Newer chips can adjust frequency on a per-core basis

Frequency and Core Count

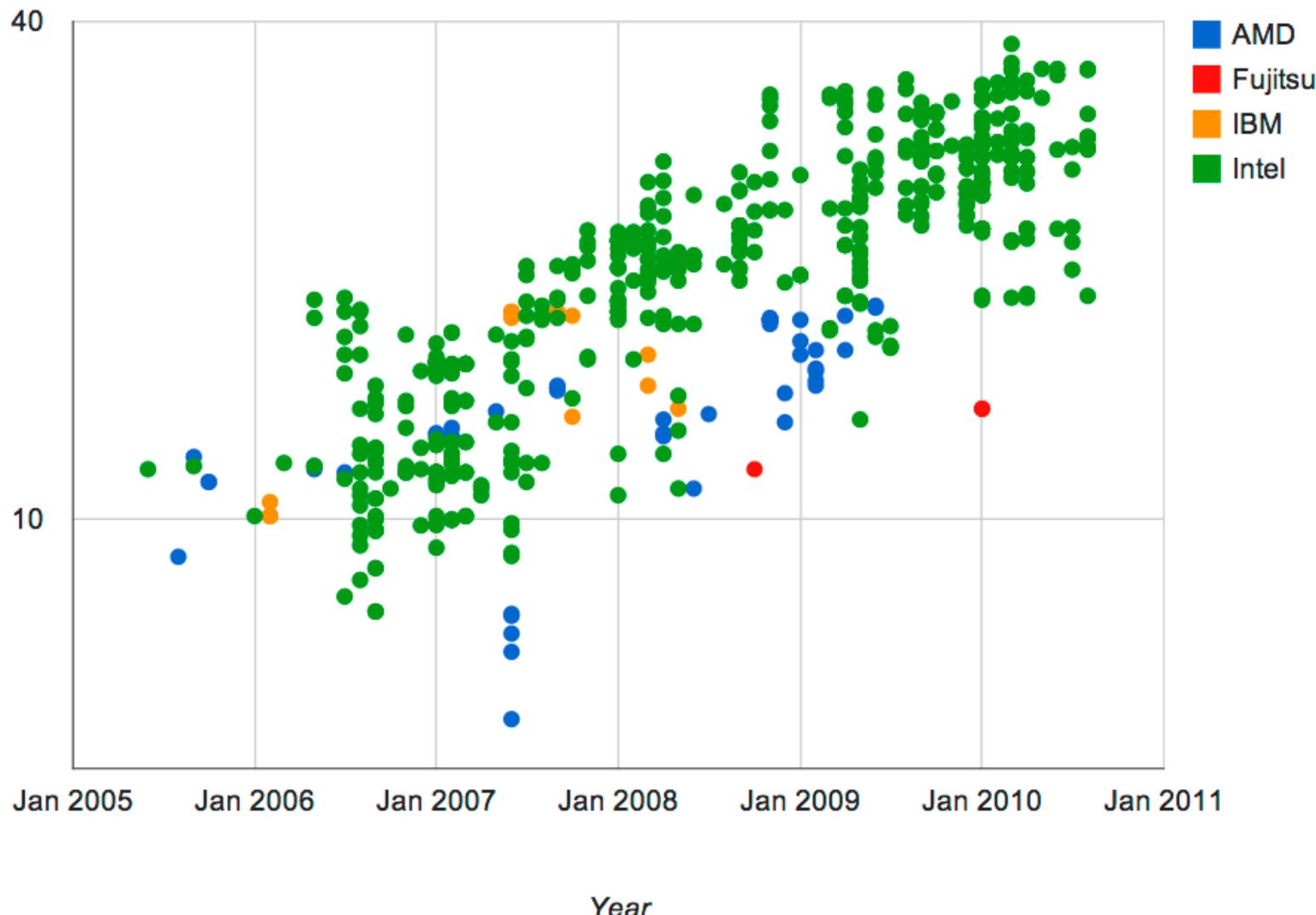
data from
<http://cpudb.stanford.edu>



SpecINT 2006 performance

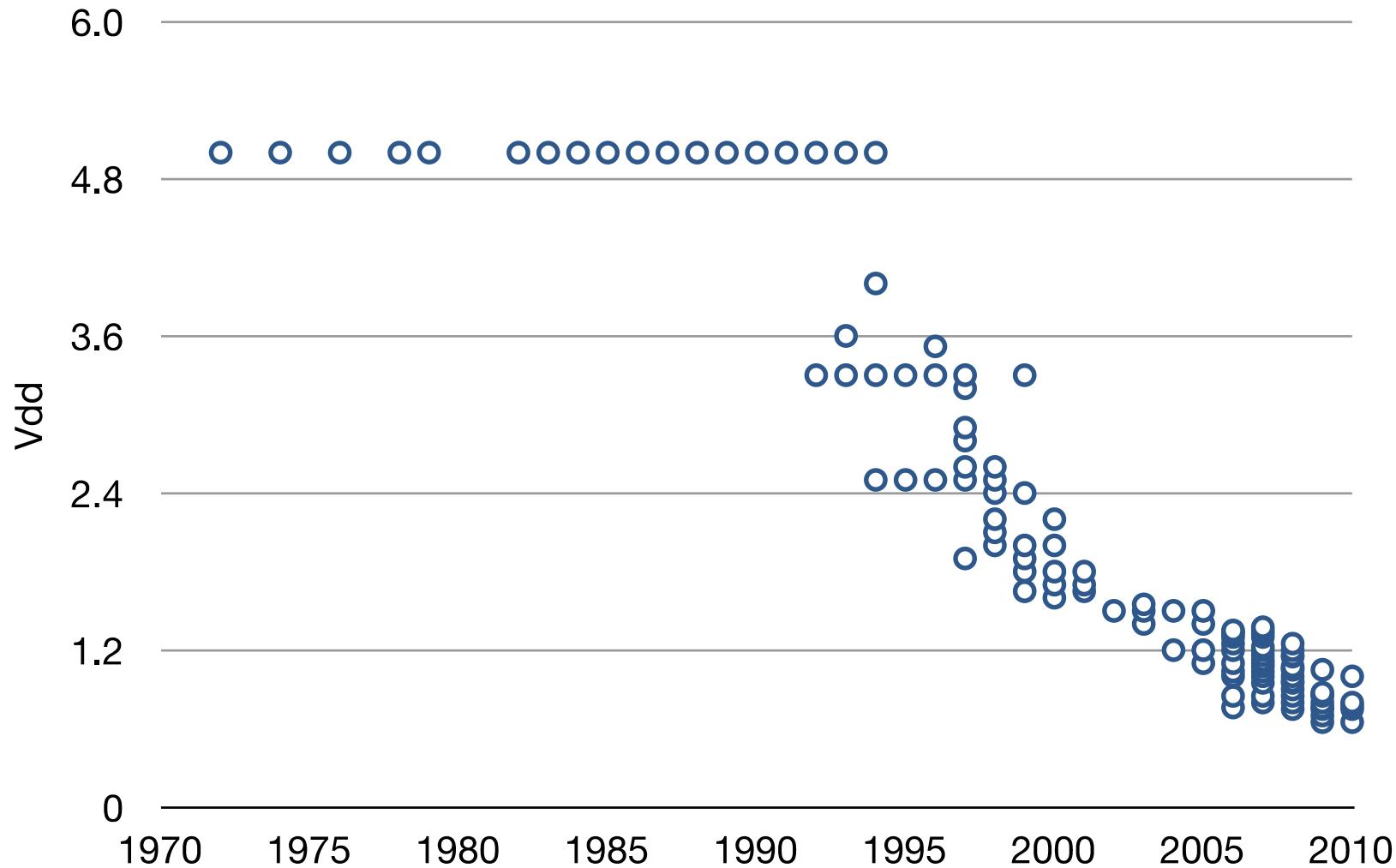
graph from
<http://cpudb.stanford.edu>

Spec Int 2006



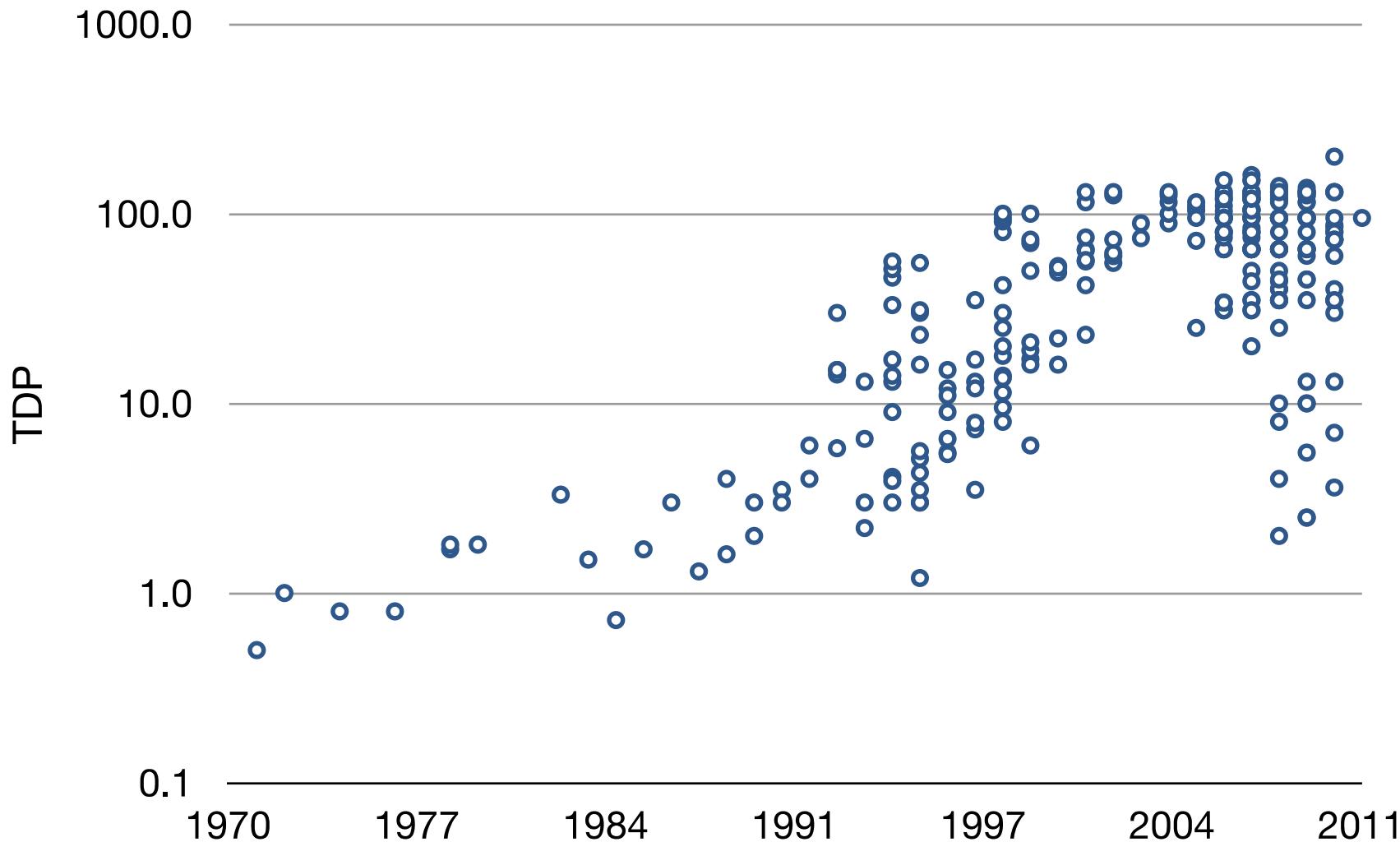
Supply Voltage

data from
<http://cpudb.stanford.edu>



Thermal Design Power

data from
<http://cpudb.stanford.edu>



Moore without Dennard

- + Dennard scaling reduced power/transistor...
 - Required reducing V , which requires a trade-off:
 - Keeping V_t the same and reducing frequency (f)
 - Lowering V_t and increasing leakage exponentially
- + Moore's Law still gives more transistors
 - + Use techniques like high-K/metal gate, dual- V_T , tri-gate
- The end of voltage scaling & "dark silicon"
 - Current projections: power per transistor reduced by 25-35% per technology node
 - What are the implications?

Implications on Software

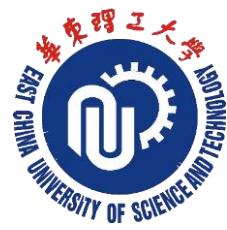
- Software-controlled dynamic voltage/frequency scaling
 - Example: video decoding
 - Too high a clock frequency – wasted energy (battery life)
 - Too low a clock frequency – quality of video suffers
 - “Race to sleep” versus “slow and steady” approaches
- Managing low-power modes
 - Don’t want to “wake up” the processor every millisecond
- Tuning software
 - Faster algorithms can be converted to lower-power algorithms
 - Via dynamic voltage/frequency scaling
- Exploiting parallelism & heterogeneous cores
 - ARM BIG.little design (a few “big” cores & 1 “low power” core)
- Specialized hardware accelerators

Summary

Technology Summary

- Has a first-order impact on computer architecture
 - Performance (transistor delay, wire delay)
 - Cost (die area & defects)
 - **Changing rapidly**
- Most significant trends for architects
 - More and more transistors
 - What to do with them? → integration → **parallelism**
 - Logic is improving faster than memory & cross-chip wires
 - “Memory wall” → caches, more integration
- Power and energy
 - Voltage vs frequency, parallelism, special-purpose hardware
- This unit: a quick overview, just scratching the surface

Rest of course



Thanks!
Q&A

