CS 1541 Introduction

Technology Constraints

Wonsun Ahn

Department of Computer Science

School of Computing and Information

Technology Constraints
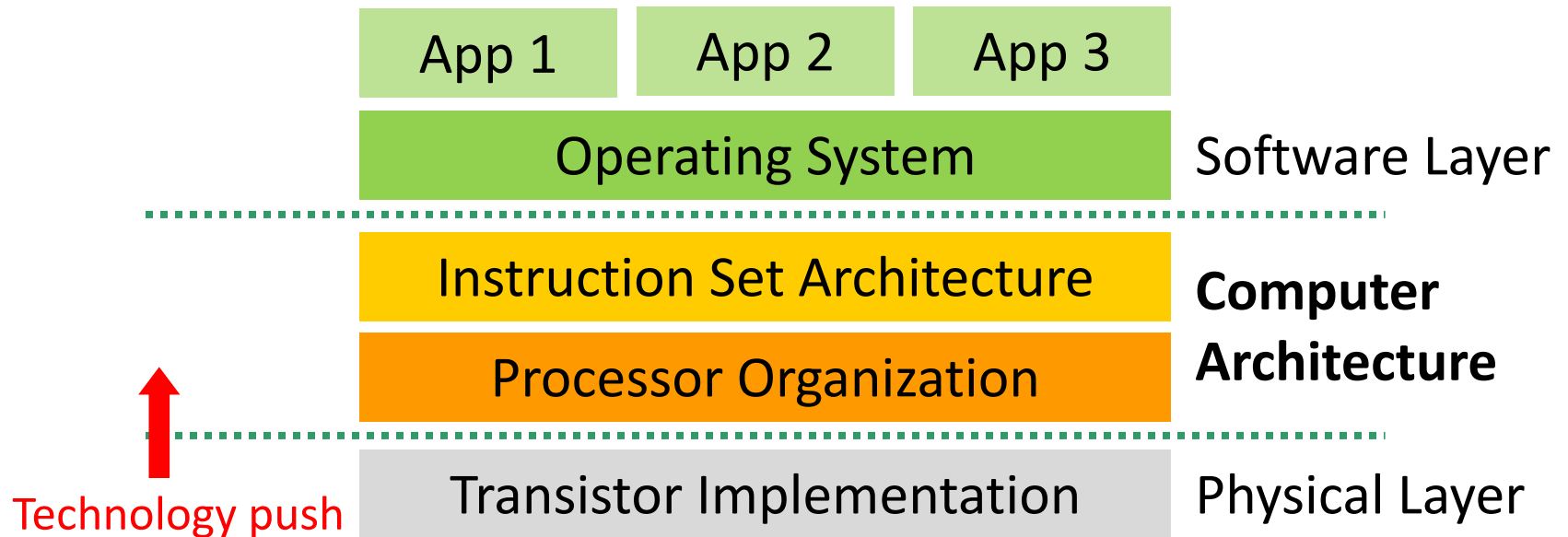
University of Pittsburgh

Technology Constraints

■ Constraints in technology push architecture too

App 1	App 2	App 3

| Operating System | Software Layer |

| Instruction Set Architecture | **Computer Architecture** |
| Processor Organization | |

↑ Technology push

| Transistor Implementation | Physical Layer |

- *Power Wall*: Thermal Design Power (TDP) constraint
- *Memory Wall*: Constraint in bandwidth to memory
- *Variability*: Limits in the precision of manufacturing technology

■ Processor must be designed to meet all constraints

Power Wall

- Power$_{CPU}$ = Power$_{dynamic}$ + Power$_{leakage}$
  Power$_{dynamic}$ $\propto$ A * N * CFV^2
  Power$_{leakage}$ $\propto$ f(N, V, V$_{th}$) $\propto$ N * V * e^{-Vth}
  - Leakage power is also called *static power*
  - This total CPU power cannot exceed TDP

- Moore's Law transistor scaling means two things:
  - N = Number of transistors ($\propto$ 1/transistor size^2) ⇧ ⇧
  - C = Capacitance ($\propto$ transistor size) ⇩
  - Reductions in C does not compensate for increases in N

- Architects must use tricks to keep power in check
 - To keep packing more transistors to increase performance

1. Reducing Dynamic Power

- **Power$_{dynamic}$ ($\propto$ A * N * CFV^2)** + Power$_{leakage}$ ($\propto$ N * V * e^{-Vth})

- Reducing A (Activity): *Clock gating*
  - Disables the clock signal to unused parts of the chip (idle cores)
  - Wake-up is instantaneous (the moment clock signal goes in)

- Reducing F (Frequency) and V (Supply Voltage)
  - When F is reduced, V can also be reduced
    (The RC curve and water pressure, remember?)
  - *Dynamic Voltage Frequency Scaling (DVFS)* done on multi-cores
    - Slow down low-priority cores, speed up high-priority cores

DVFS and Transistor Speed

- RC Charging Curve of V_G
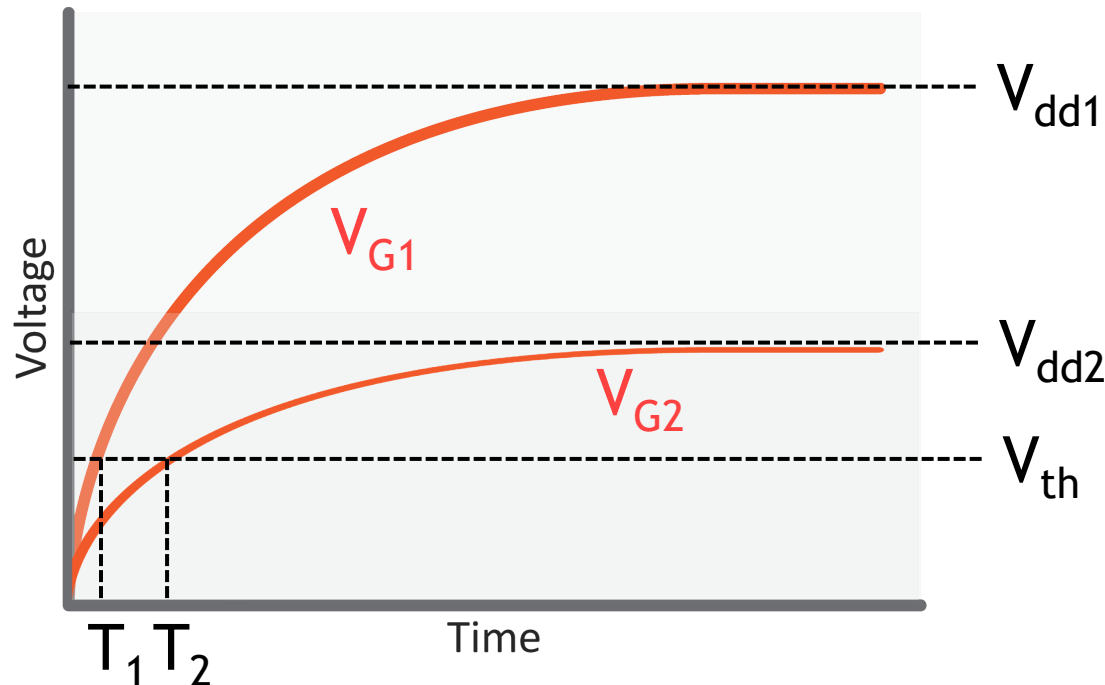
- Delay (T) increases with supply voltage (V_{dd}) decrease
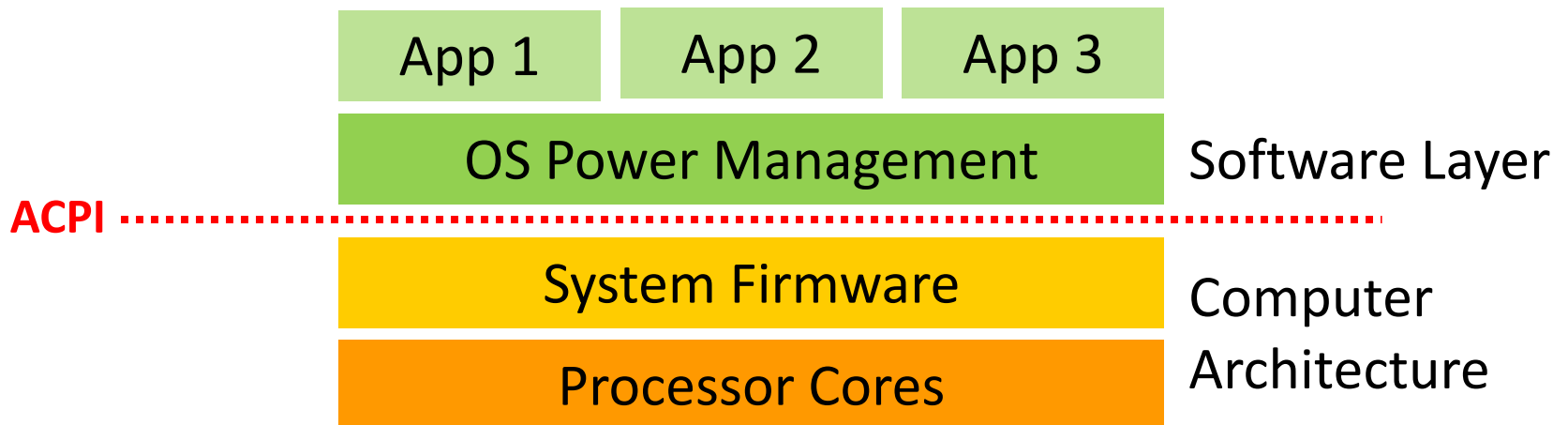 → Frequency (1/T) decreases with V_{dd} decrease

2. Reducing Leakage Power

- $\text{Power}_{dynamic}$ ($\propto A * N * CFV^2$) + $\text{Power}_{leakage}$ ($\propto N * V * e^{-Vth}$)

- Reducing N (Transistor Number): *Power gating*
  - Disables power to unused parts of the chip (unused cores)
  - Eliminates dynamic power *and* leakage power to those parts
  - Drawback: wake-up takes a much longer time than clock gating
    - Delay for supply voltage to stabilize
    - Delay to backup and restore CPU state to/from memory

- Reducing V (Supply Voltage): *DVFS* also helps here
  - Only the voltage scaling part, not the frequency scaling

OS Manages Power

- Who decides which cores to clock gate and power gate?
- Who decides how to apply DVFS to the cores?
- *ACPI (Advanced Configuration and Power Interface)*
  - OS performs power management using this interface
    - OS knows best which threads to prioritize for best user experience
  - Open standard interface to system firmware
    - Firmware sends signals to processor cores to control them

| App 1	App 2	App 3	
OS Power Management			Software Layer

ACPI ┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈┈

System Firmware	Computer
Processor Cores	Architecture

3. Simpler Processor Design

- **Plenty of transistors but not enough power**
  - Power becomes the ultimate currency in processor design

- **Complex logic for performance is power hungry**
  - Not easy to eke out the last bit of performance out of a program
  - Diminishing returns on performance for power investment

- **Push towards simpler architectures:**
  - *Multi-cores*: Run multiple programs (threads) on simple cores
  - *GPUs*: Run each instruction on massively parallel compute units
  - *Caches*: Memory caches are power efficient (low dynamic power)

Memory Wall

- **Refers to both *latency (ns)* and *bandwidth (GB/s)***
  - CPU frequency and overall performance increased dramatically
  - Memory (DRAM) latency and bandwidth have lagged far behind

- **Why?**
  - Limit on the number of CPU / DRAM pins that can be soldered on
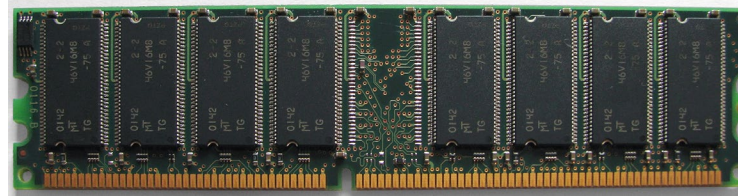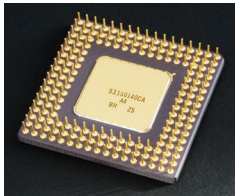
  - DRAM manufacturers have traditionally prioritized capacity
  - DDR1 (1998): 1.6 GB/s → DDR4 (2014): 25.6 GB/s
    (Impressive? Not so much compared to CPU performance)

Memory Wall

Sustained (streaming) Memory Bandwidth is falling behind Peak FLOPS rates, but every other kind of memory access is falling behind even faster....

FLOPS vs Network Latency: ~30x/decade

FLOPS vs Memory Latency: >100x/decade

FLOPS vs Network Bandwidth: 9x/decade

FLOPS vs Memory Bandwidth: 4.5x/decade
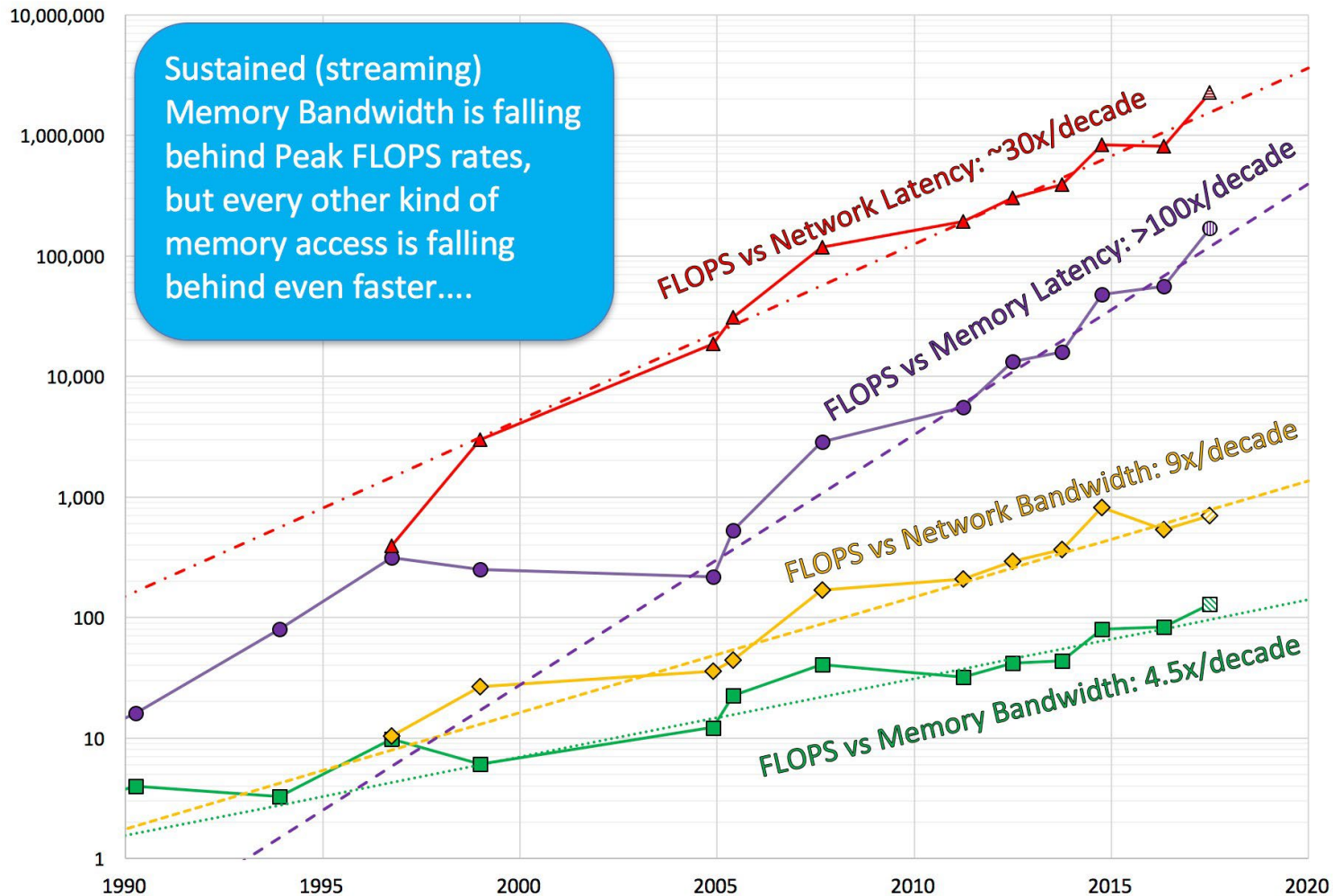
Source: SC16 Invited Talk ""Memory Bandwidth and System Balance in HPC Systems" by John D. McCalpin

- FLOPS = floating point operations per second (performance)

Memory Wall

- Where did the Memory Wall push architecture?

- *Caches*: If hit in cache, no need to go to memory
  - Caching reduces both data access latency/bandwidth

- *3D-Stacked Memory:* Stack CPU on top of memory
  - Drill vias, or holes, through silicon to bond CPU with memory
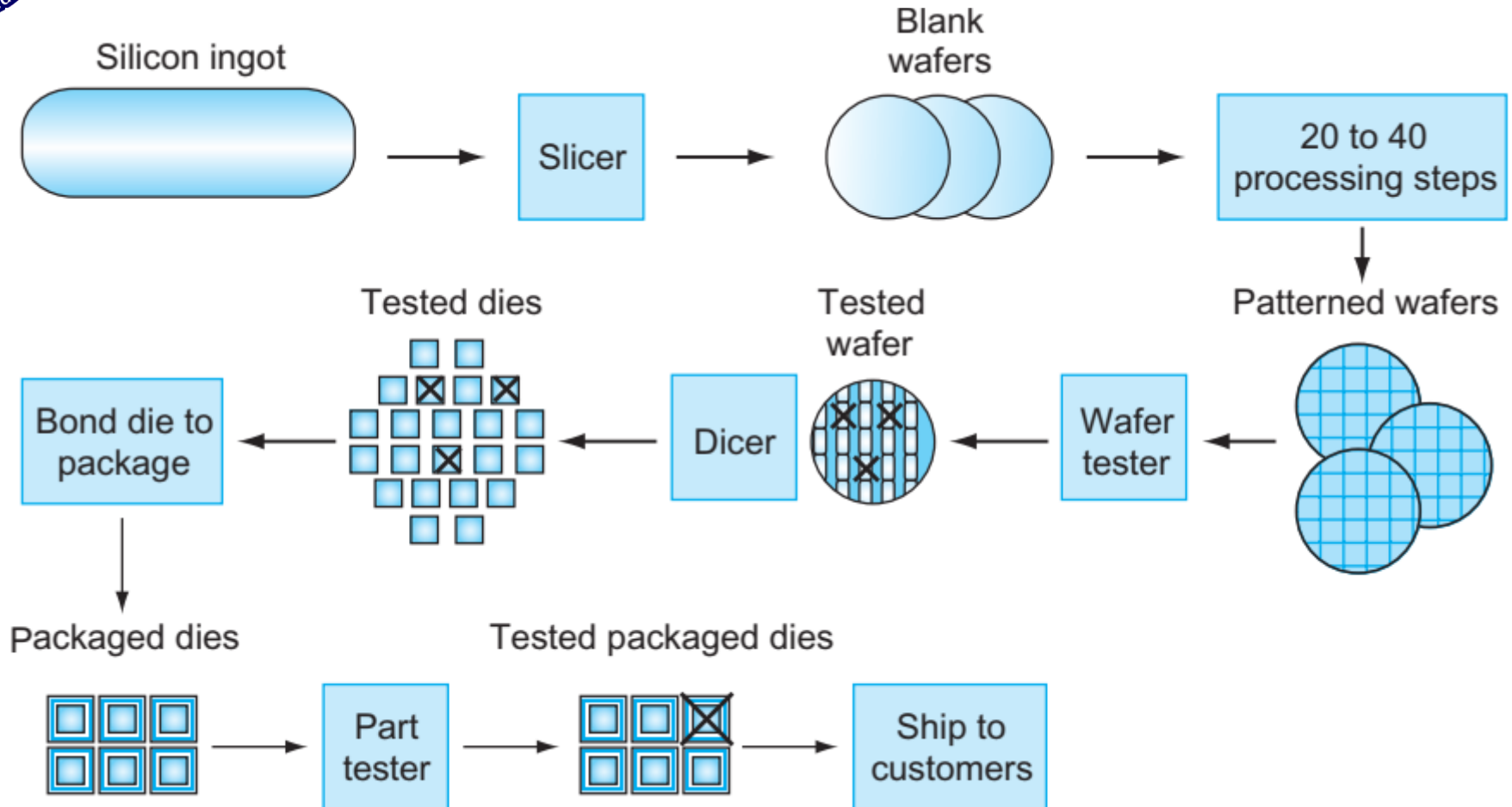 - Through silicon vias (TSVs) have low latency / high bandwidth
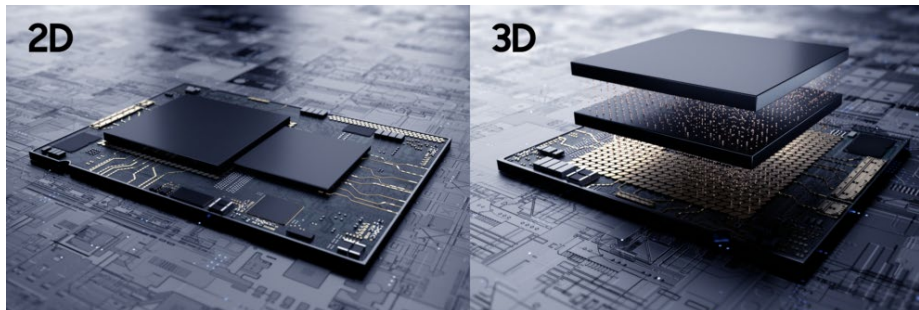
Variability

- *Variability*: differences in speed of individual transistors
  - If fab can't ensure uniformity of transistors, speeds will differ
  - Speed differences mostly come from variations in V_{th}:
    low $V_{th} \rightarrow$ cycle time ⇩ but leakage power ⇧
    high $V_{th} \rightarrow$ cycle time ⇧ but leakage power ⇩

- If unlucky and a logic path has lots of *slow* transistors
  $\rightarrow$ CPU may miss clock cycle time if path is exercised
  $\rightarrow$ CPU must be discarded, since it malfunctions

- If unlucky and a region has too many *fast* transistors
  $\rightarrow$ Region may generate too much heat due to low V_{th}
  $\rightarrow$ CPU must be discarded, due to overheating

- Leads to low chip *yield*

Wafer Yield

Silicon ingot → Slicer → Blank wafers → 20 to 40 processing steps

Patterned wafers

Tested dies ← Dicer ← Tested wafer ← Wafer tester ← Patterned wafers

Bond die to package ← Tested dies

Packaged dies → Part tester → Tested packaged dies → Ship to customers

■ Lower yield leads to higher production cost

Variability

- **Where did Variability push architecture?**

- *Product binning:* Sell slower CPUs at a cheaper "bin"
  - And rate slower CPUs at a lower CPU frequency
  - Instead of discarding them as "malfunctioning"
- *Multi-cores*: Easy to disable one or two buggy cores
  - Compared to single core where subcomponents must be disabled
  - When cores are disabled, sold as a cheaper chip
- *Limited pipelining*: pipelining exacerbates variability
  - With long stages, many transistors so tend to even each other out
  - With short stages, few transistors so probable all are slow

Intel i9 Product Binning

Model	# Cores	# Threads	Base Clock	All Core Turbo	Turbo Boost	Total L3 Cache	PL1 TDP
i9-10900K	10	20	3.7	4.8	5.1	20	125
i9-10900KF	10	20	3.7	4.8	5.1	20	125
i9-10900	10	20	2.8	4.5	5.0	20	65
i9-10900F	10	20	2.8	4.5	5.0	20	65
i9-10900T	10	20	1.9	3.7	4.5	20	35
i7-10700K	8	16	3.8	4.7	5.0	16	125
i7-10700KF	8	16	3.8	4.7	5.0	16	125
i7-10700	8	16	2.9	4.6	7.7	16	65
i7-10700F	8	16	2.9	4.6	4.7	16	65
i7-10700T	8	16	2.0	3.7	4.4	16	35
i5-10600K	6	12	4.1	4.5	4.8	12	125
i5-10600K	6	12	4.1	4.5	4.8	12	125
i5-10600	6	12	3.3	4.4	4.8	12	65
i5-10600T	6	12	2.4	3.7	4.0	12	35
i5-10500	6	12	3.1	4.2	4.5	12	65
i5-10500T	6	12	2.3	3.5	3.8	12	35
i5-10400	6	12	2.9	4.0	4.3	12	65
i5-10400F	6	12	2.9	4.0	4.3	12	65
i5-10400T	6	12	2.0	3.2	3.6	12	35

Why the close to 4X difference? Clock difference is just 2X!

Produced from one wafer

4 out of 10 cores were discarded.

Source: https://www.techspot.com/article/2039-chip-binning/
* TDP is calculated using the Base Clock frequency at a nominal supply voltage

Opportunities for Speed Improvement

- **So Dennard Scaling is dead**
  - Free CPU frequency gains are no longer there

- **And we are walled in by technology constraints**
  - Power wall
  - Memory wall
  - Variability
  - …

- **Where do architects go look for performance?**

Improving Execution Time

- Execution time = $\frac{\text{instructions}}{\text{program}}$ X $\frac{\text{cycles}}{\text{instructions}}$ X $\frac{\text{seconds}}{\text{cycle}}$

- Improving $\frac{\text{seconds}}{\text{cycle}}$:
  - *Pipelining* can lead to higher frequencies

- Improving $\frac{\text{cycles}}{\text{instructions}}$:
  - *Caching* can reduce cycles for memory instructions
  - *Superscalars* can execute multiple instructions per cycle
  - *Multi-cores* execute multi-instructions from multi-threads

- Improving $\frac{\text{instructions}}{\text{program}}$:
  - *GPU*s are *SIMD (Single Instruction Multiple Data)* processors

What about Other Performance Goals?

- We talked a lot about execution speed

- But there are other performance goals such as:
  - Energy efficiency
  - Reliability
  - Security
  - …

- In this class, we will mainly focus on speed
  - Not that other goals are not important
  - We will touch upon other goals when relevant
  - Performance will be used synonymously with speed

Textbook Chapters

- Please review Chapter 1 of the textbook.