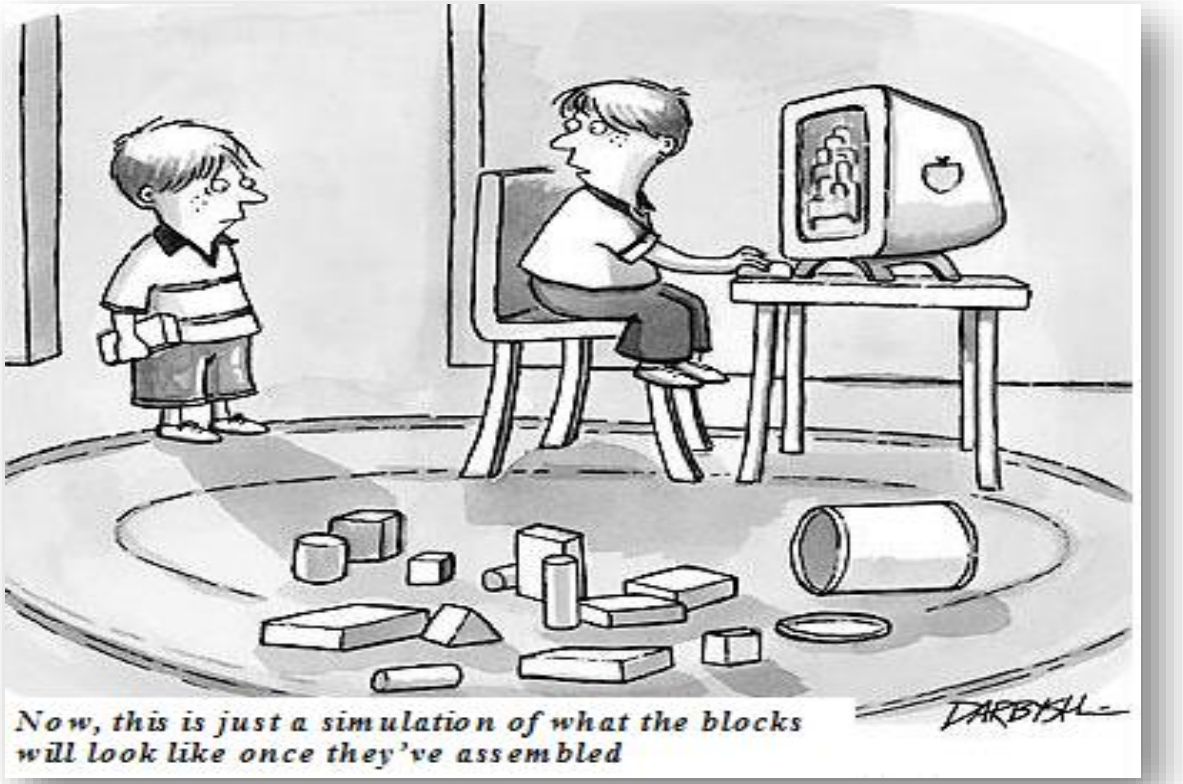
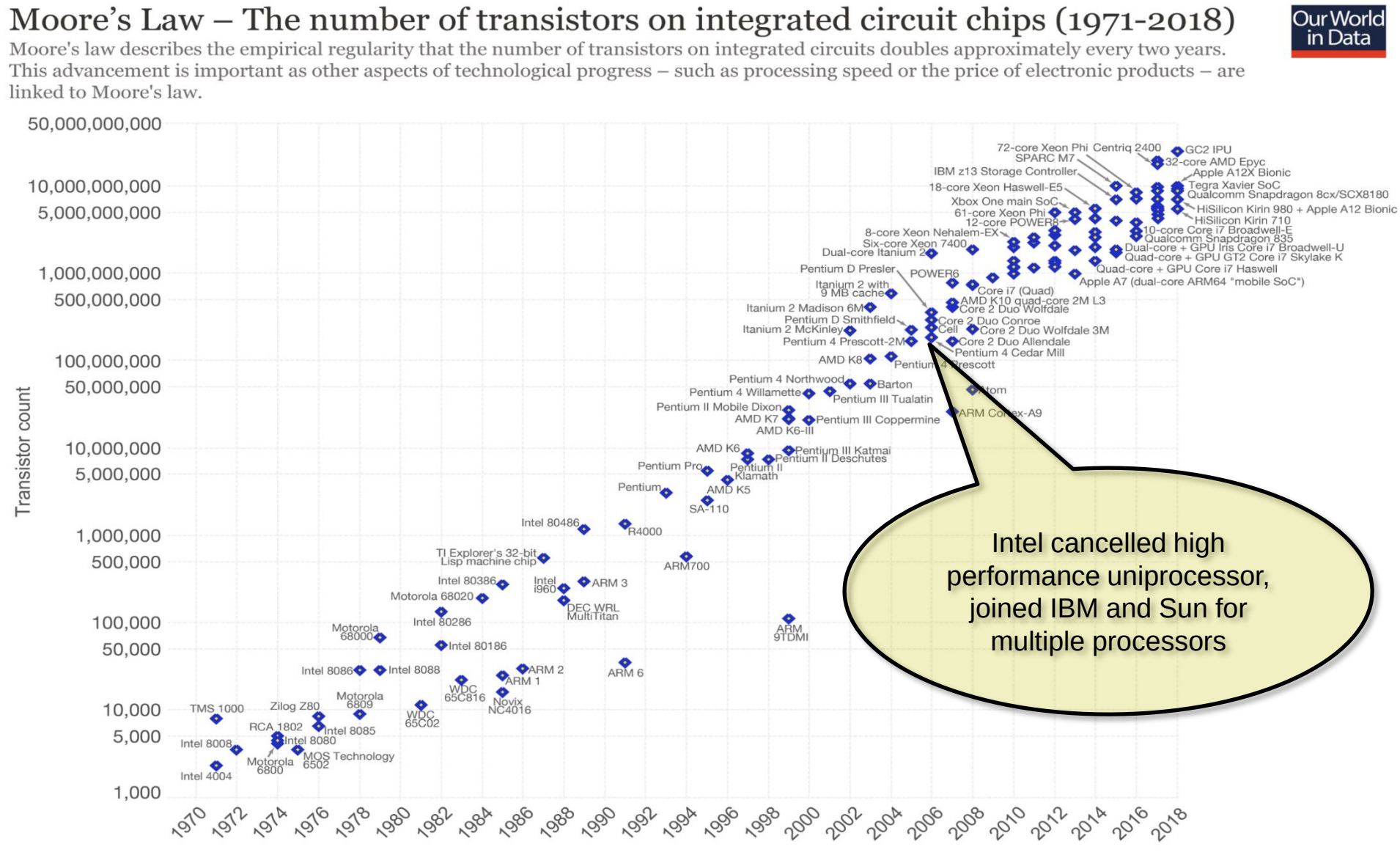
ECE569



- Computing Trends and Course Overview

End of Uniprocessor



Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

Processor Technology Trends

- Shrinking of transistor sizes: 250nm (1997) → 130nm (2002) → 65nm (2007) → 32nm (2010)

→28nm(2011, AMD GPU, Xilinx FPGA)

→22nm(2011, Intel Ivy Bridge, die shrink of the Sandy Bridge architecture)

→ 14nm (2014)

→ Intel Core M, Core I7, **iPhone 6s Plus, NVIDIA GPUs**

→ **12nm (2017)**

→ **Nvidia Titan V (Volta) (5120 Cores)**

→ **10nm (2017)**

→ **Samsung Galaxy S8, Apple iPad Pro 2nd Gen tablets**

→ **7nm (2018)**

→ **Apple A12 ARM SoC iPhone XS, iPhone 11**

→ **5nm (2020)**

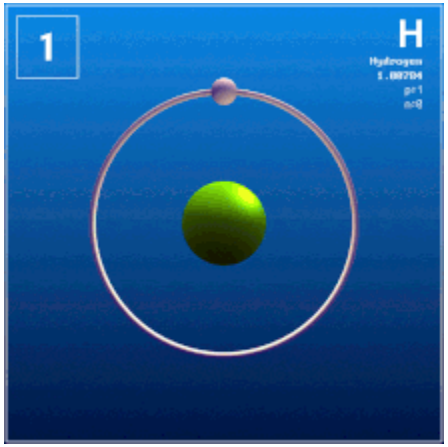
→ **2nm (May 2021) IBM's new 2-nm chips have transistors smaller than a strand of DNA**

→ **Moore's Law**

- Transistor density increases by 35% per year and die size increases by 10-20% per year… more cores!

Putting into perspective

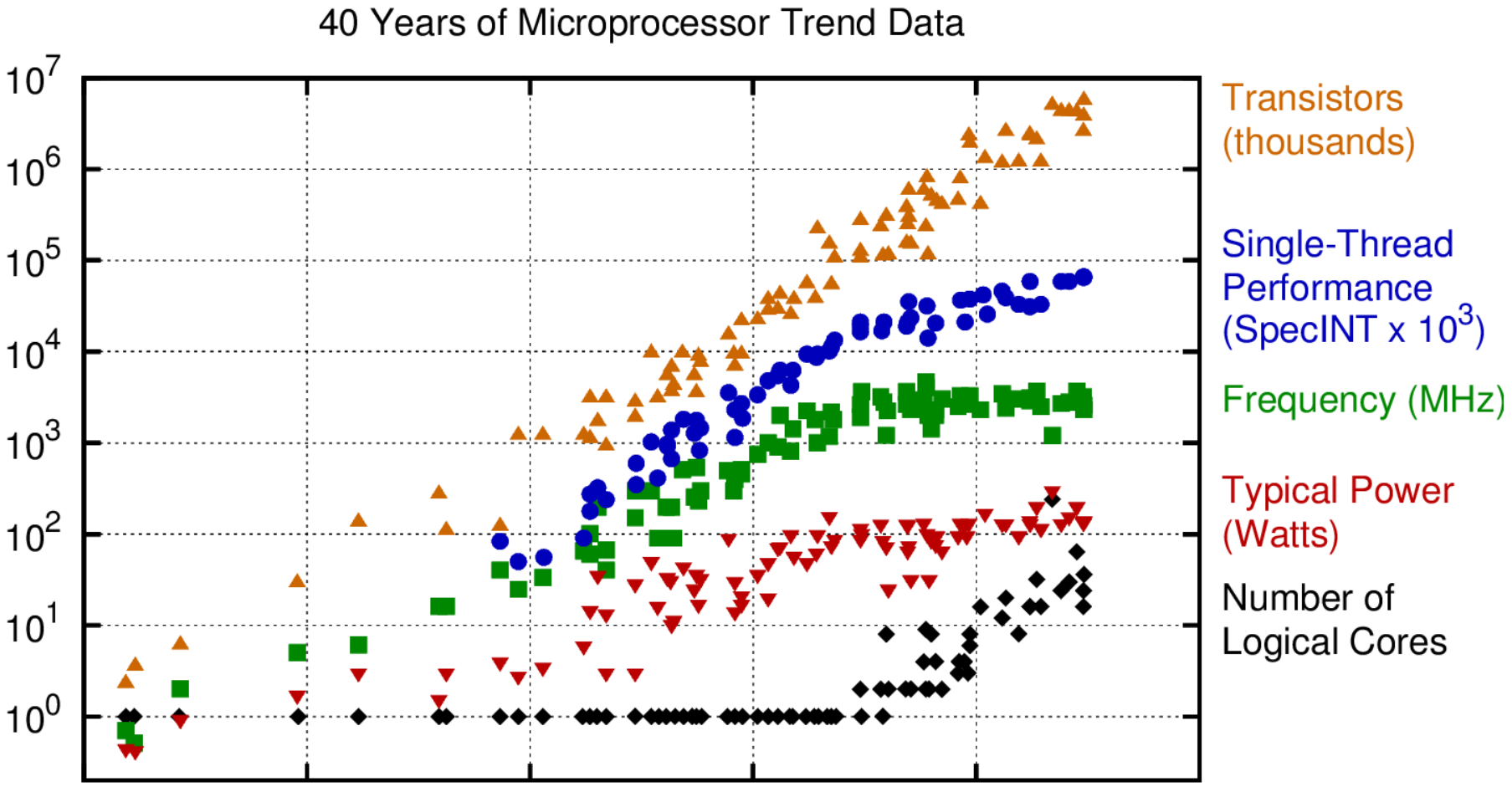
- One nanometer (nm) is one billionth of a meter.
- A hydrogen **atom**, is about 0.1 nanometers
- Silicon's atomic size is about 0.2 nanometers
  - 35 atoms for a 7nm transistor
  - Tesla V100: 815mm square, 21 billion transistors, 5120 cores, 1530MHz,



Technology Scaling and Processor Performance

- **Are processors today getting faster because:**
  - ❑We are clocking transistors faster
  - ❑We have more transistors available for computation

Historical Trends (Frequency!)

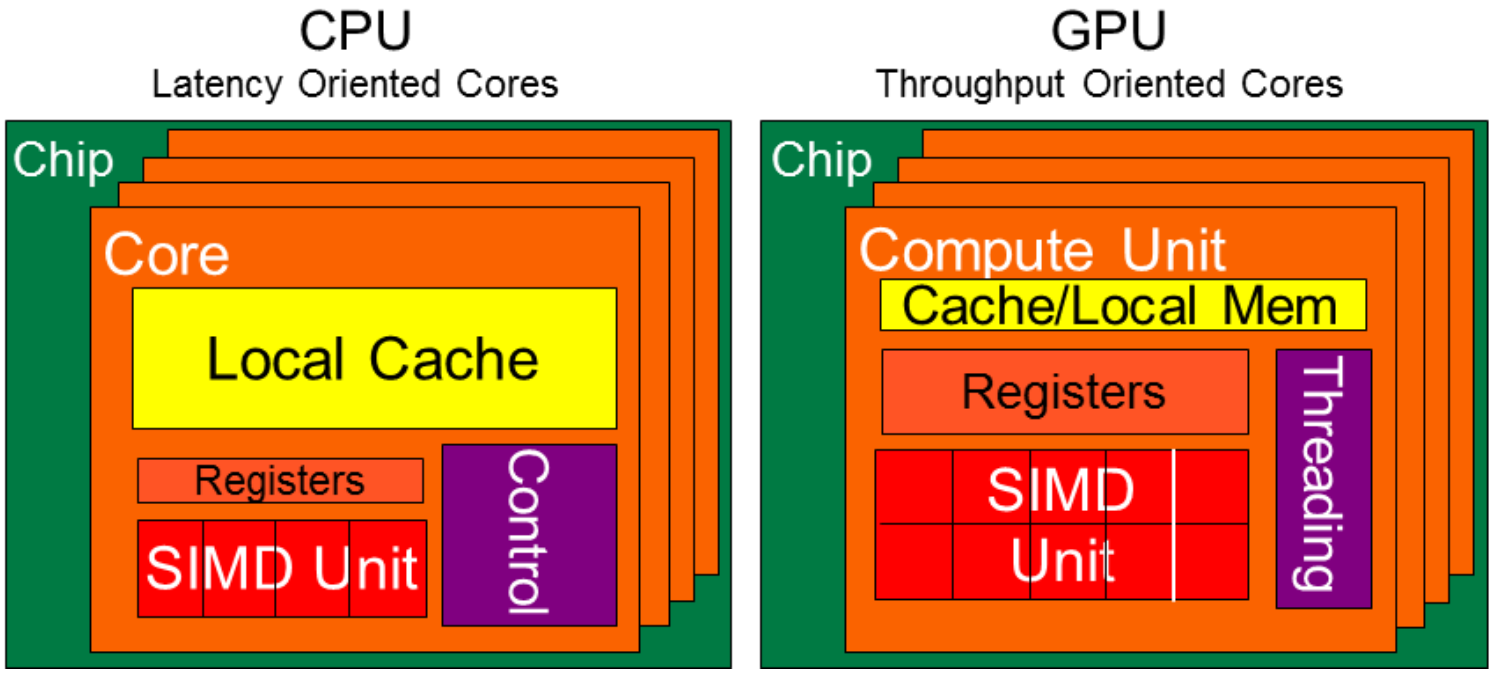


Q1. Why don't we keep increasing clock speed?

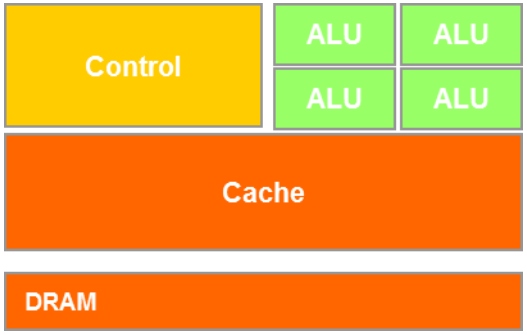
Q2. What are the main trajectories for designing microprocessors?

Latency Devices vs Throughput Devices

- CPUs and GPUs are designed very differently
  - CPUs larger cache and GPUs more registers
  - CPUs complex control

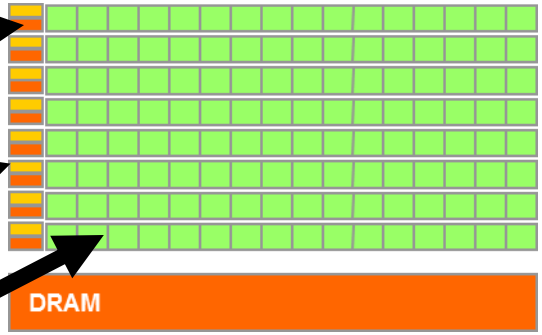


Latency Oriented Devices (CPUs)



- Powerful ALU
  - Reduced arithmetic operation latency
- Large caches
  - Convert long latency memory accesses to short latency cache accesses
    - Spatial and temporal locality
- Sophisticated control
  - Branch prediction for reduced branch latency
  - Data forwarding for reduced data latency
- Neither control logic nor cache memories contribute to the peak calculation throughput.

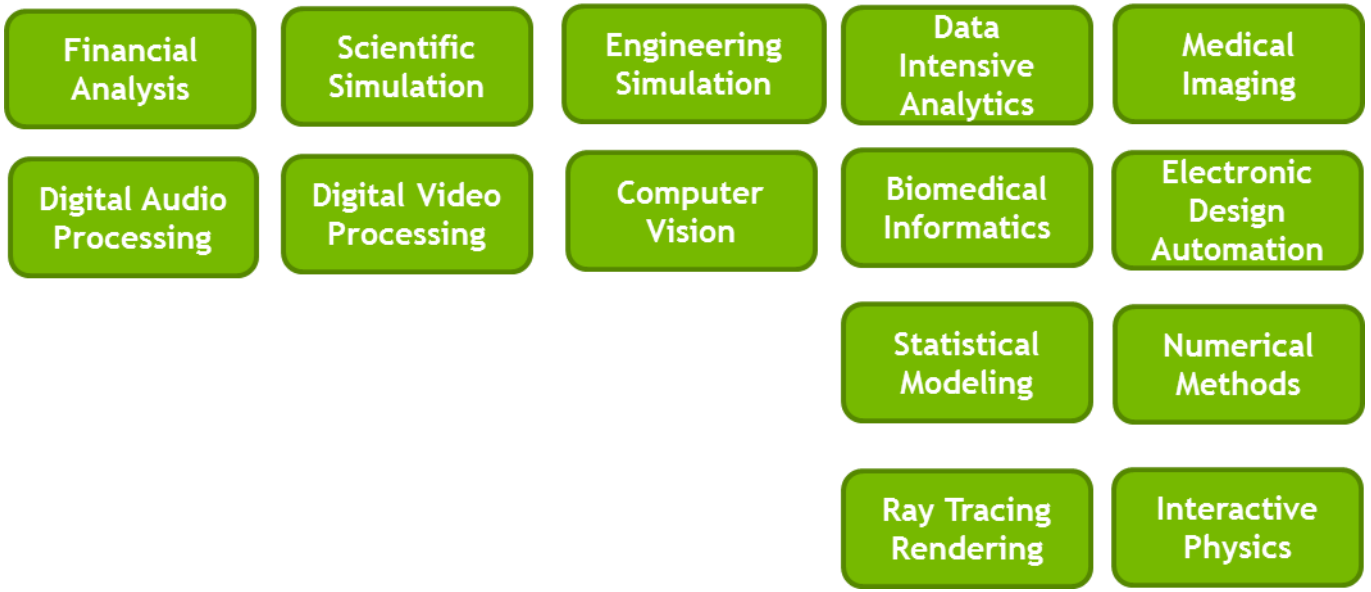
Throughput Oriented Devices (GPUs)



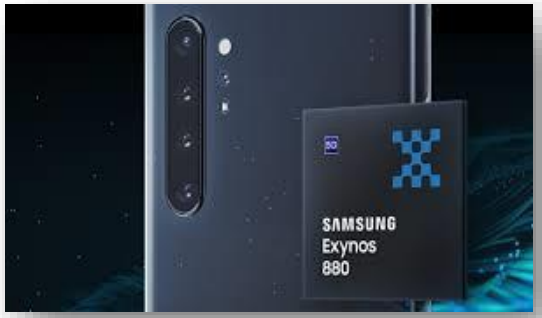
- Small caches
  - To boost memory throughput
- Simple control
  - No branch prediction
  - No data forwarding
- Energy efficient ALUs
  - Many, long latency but heavily pipelined for high throughput
    - Many high throughpout arithmetic pipeline
- Require massive number of threads to tolerate latencies
  - Threading logic
  - Thread state

Best Strategy: Use Both CPU and GPU

- CPUs for sequential parts where latency matters
  - CPUs can be 10X+ faster than GPUs for sequential code
- GPUs for parallel parts where throughput wins
  - GPUs can be 10X+ faster than CPUs for parallel code



Latency Devices vs Throughput Devices

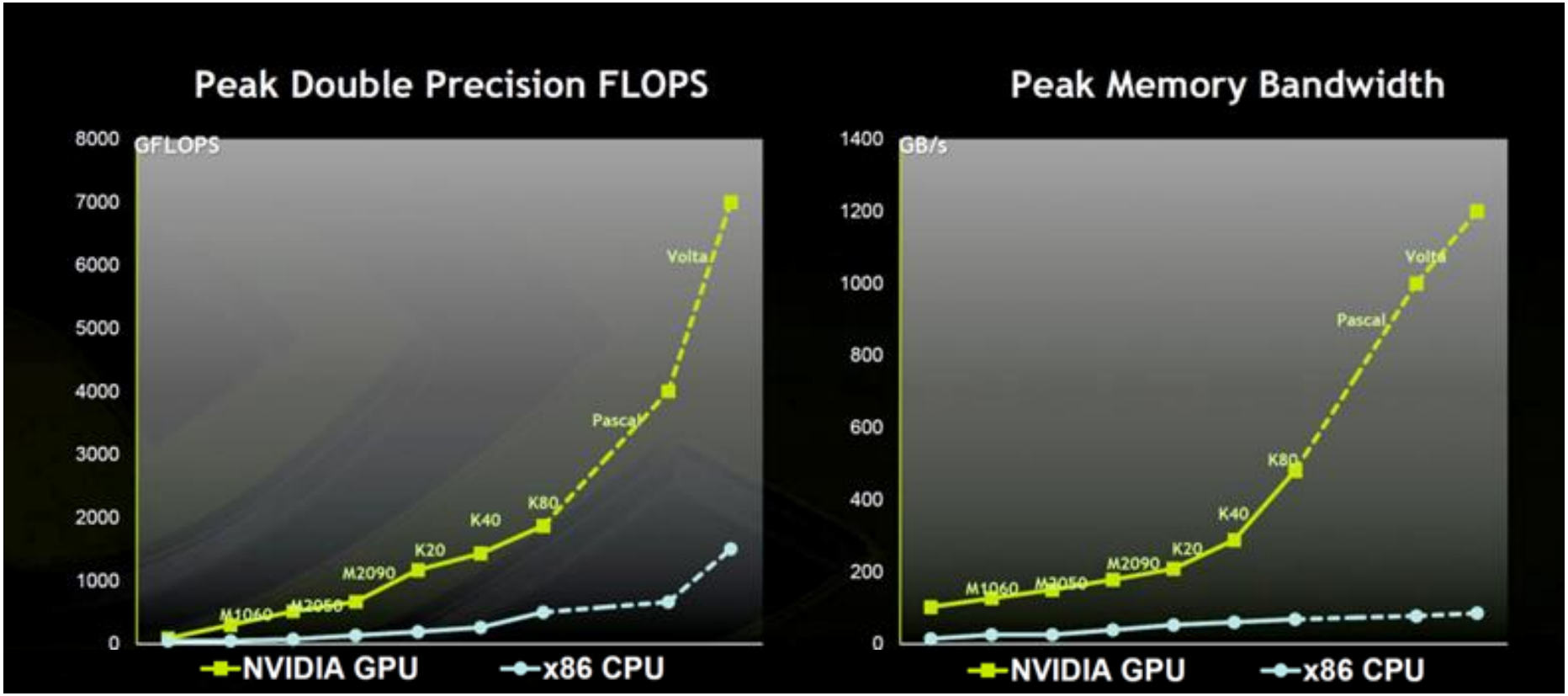


Mix of DSP, Programmable, CPU and GPU cores, cellular radios, image processors, audio/video decoders
Quad core - Cortex A73,
Quad core - Cortex A53,
ARM Mali GPU,
LTE modem, WiFi, Bluetooth, GPS,

Jetson AGX Xavier - 32 TOPs
512-core Volta GPU
8-core ARM

- Heterogenous Parallel Computing
  - Developer must use the best match for the job

Need for Software Developers to Learn Parallel Programming



- Dramatically escalated incentive for parallel program development!
- Parallel programming is not limited to few applications running on expensive computing systems anymore
- Number of applications that need to be developed as parallel programs increased dramatically

Challenges in Parallel Programming

80 processors
64 cores/processor
5120 cores
2048 Threads/processor
163,840 Threads/GPU

Bright news