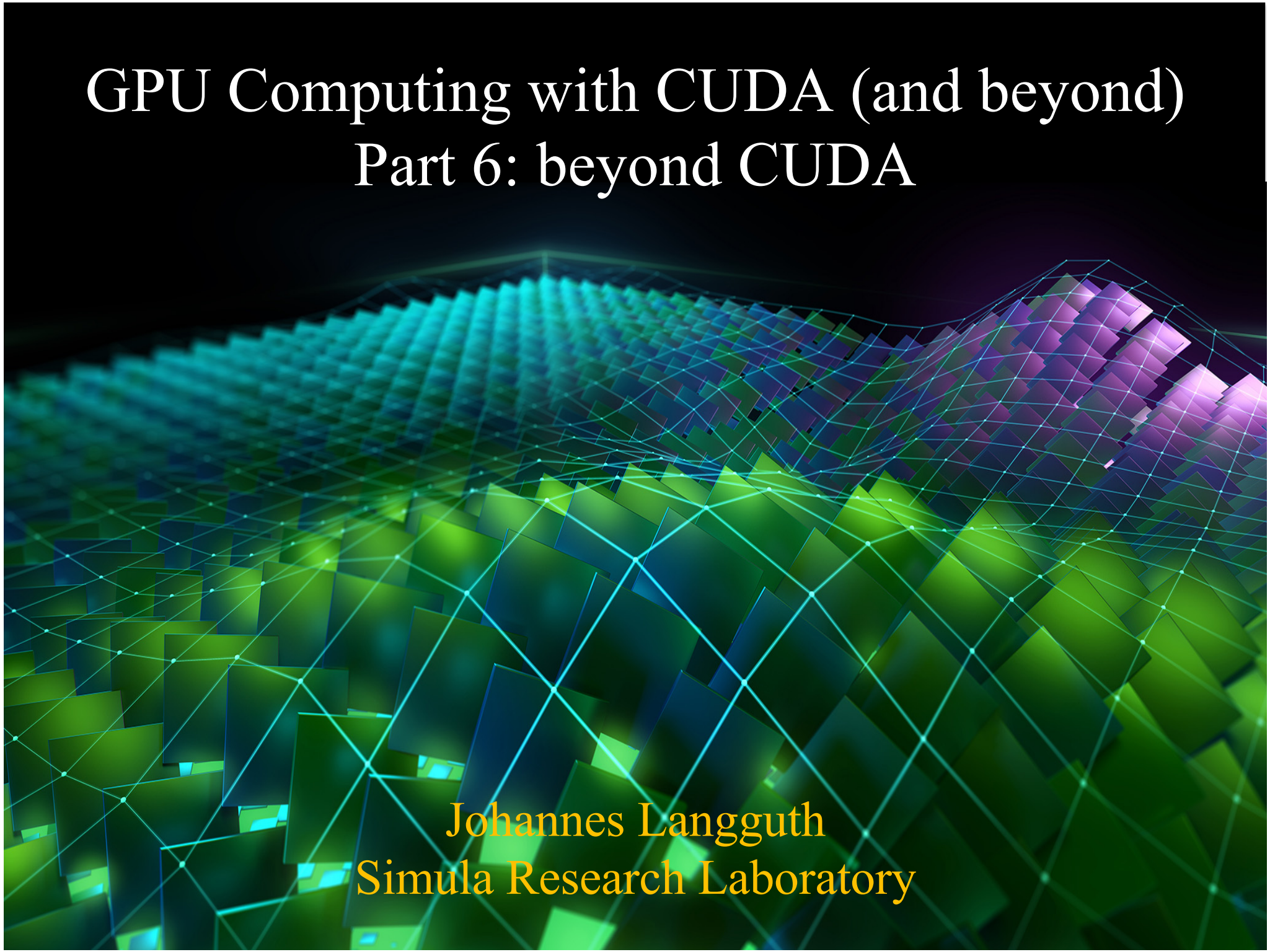
GPU Computing with CUDA (and beyond)
Part 6: beyond CUDA

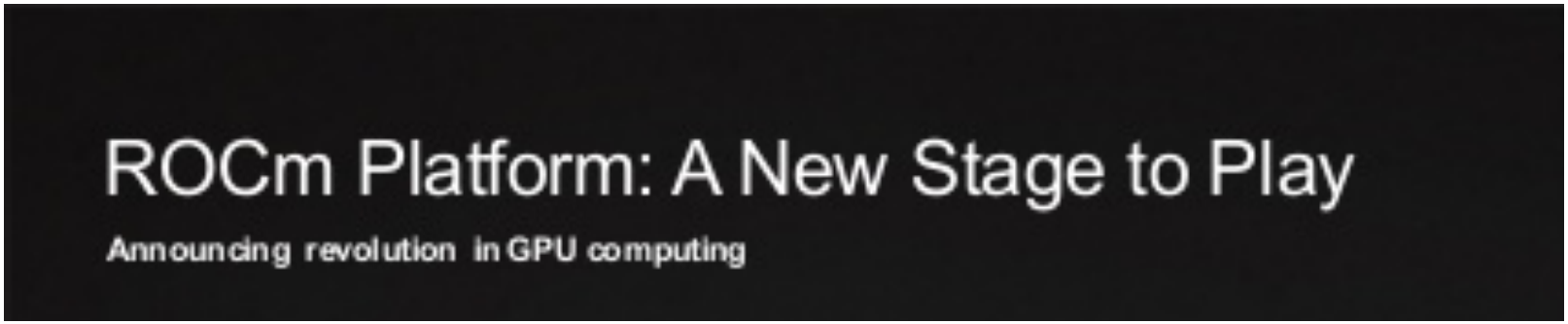


Johannes Langguth
Simula Research Laboratory

AMD GPUs

	AMD Radeon VII	NVIDIA Volta V100
Cores	64	80
Base frequency (Ghz)	1.8	1.6
TFLOPS	3.46	7.5
Memory Bandwidth (GB/s)	1000	900
Cache (last level in MB)	4	6
Memory	16	32
Price (USD)	700	10000

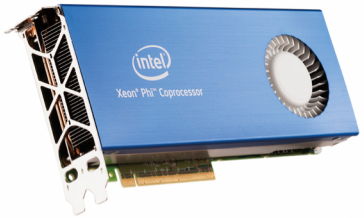
How to Program AMD GPUs

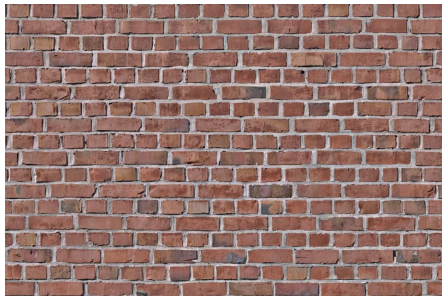


- AMD provides the Radeon Open Compute (ROC)
- Alternative to CUDA
- Currently not very mature
- OpenCL may be preferable

Intel: Alternatives to GPUs

Intel MIC / Manycore / Xeon Phi





2013
KNC: Knight's Corner

2016
KNL: Knight's Landing

2018
The End

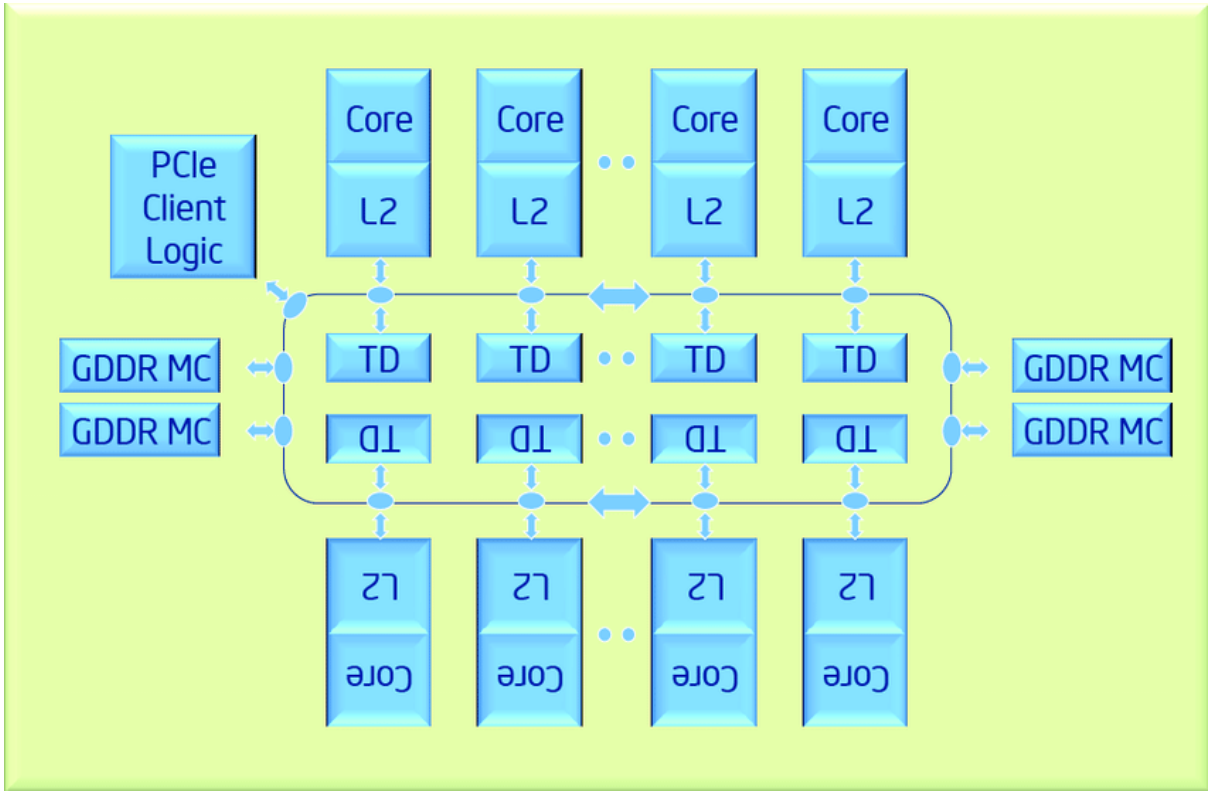
- Combined advantages of CPU and GPU
- Product line was terminated
- Xeon Phi failure shows why GPUs work

Intel: Alternatives to GPUs

Intel MIC / Manycore / Xeon Phi

2013
KNC: Knight's Corner

- 57 – 61 Pentium cores
- Ring bus



- Xeon Phi followed CPU model, but in order execution
- Cache coherency and parallel threads
- Cache traffic overwhelmed the ring bus

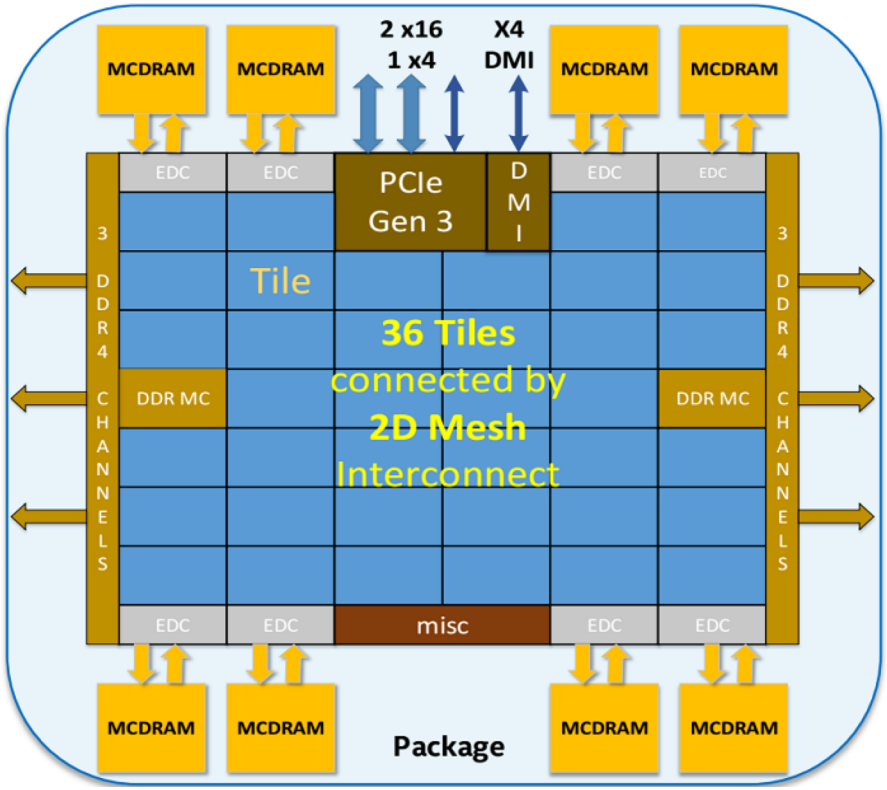
Intel: Alternatives to GPUs

Intel MIC / Manycore / Xeon Phi

2016

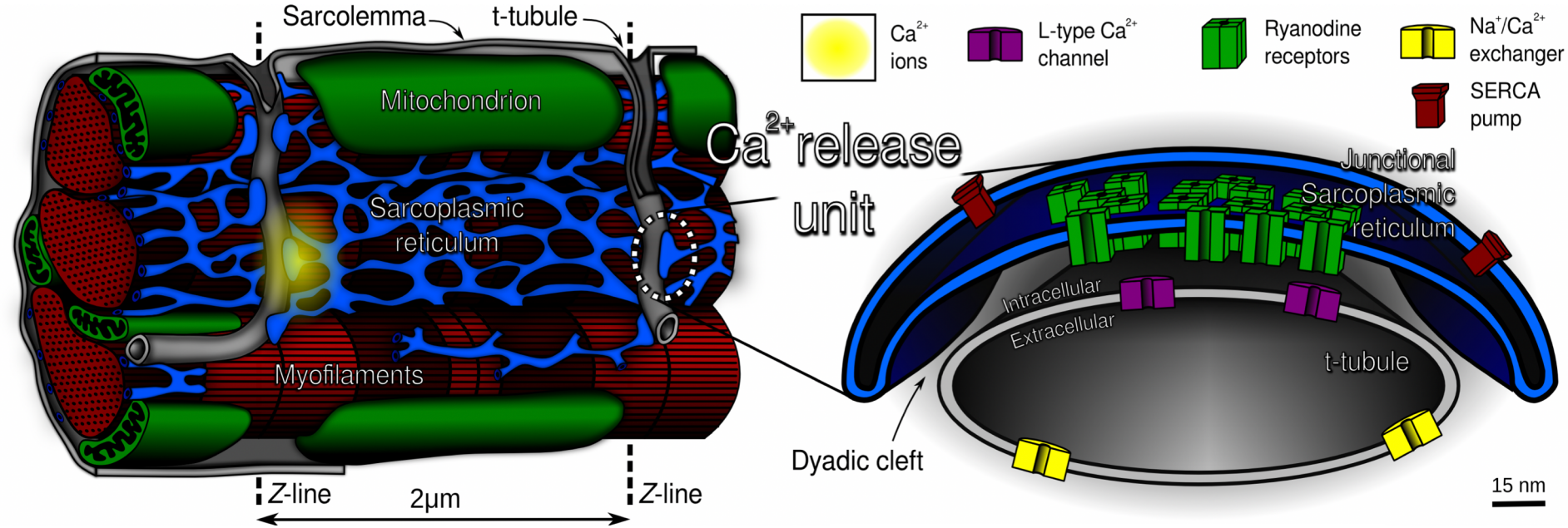
KNL: Knight's Landing

- 64 – 72 Atom cores@1.4 GHz
- 2D Lattice
- 16 GB MCDRAM@500 GB/s

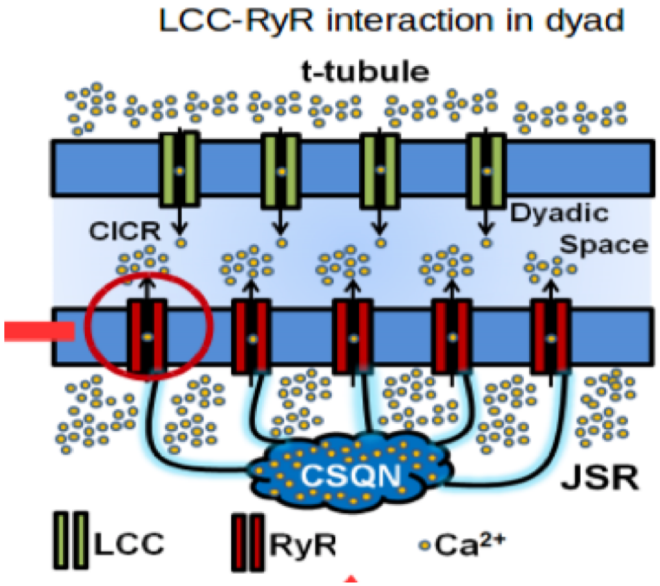


- No major design flaws but
- Too similar to CPUs
- 64 Core CPUs are now available

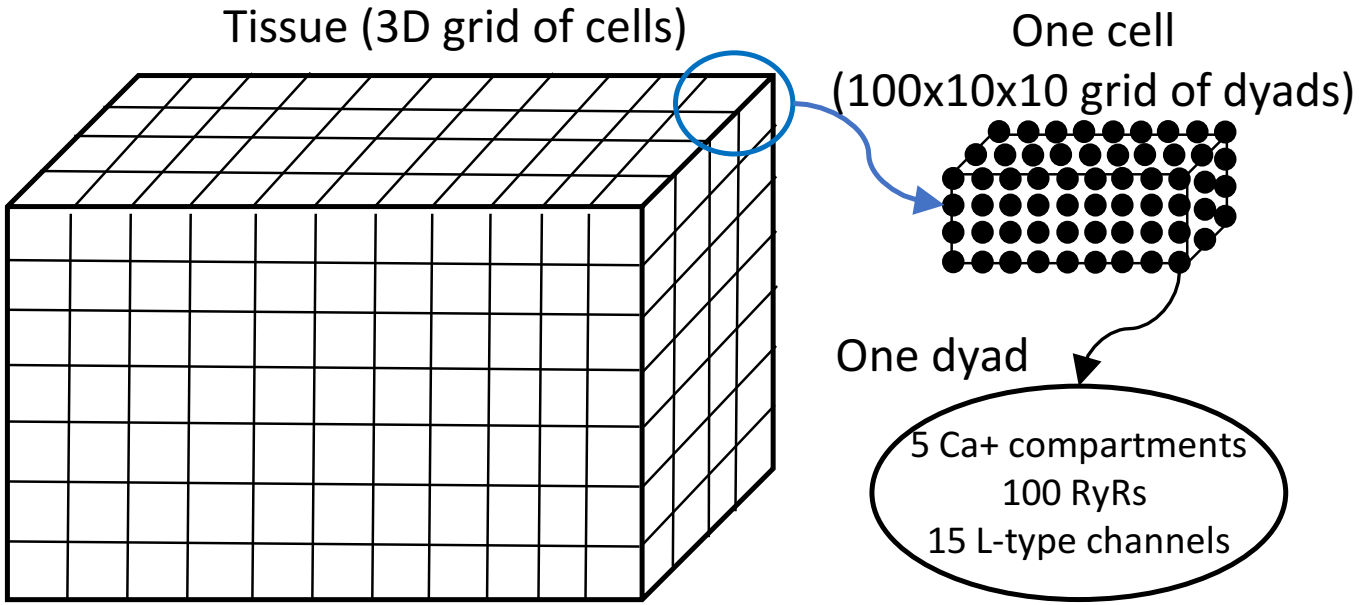
Simulating Calcium Handling in the Heart



Extremely expensive simulation. Requires all the performance we can get.



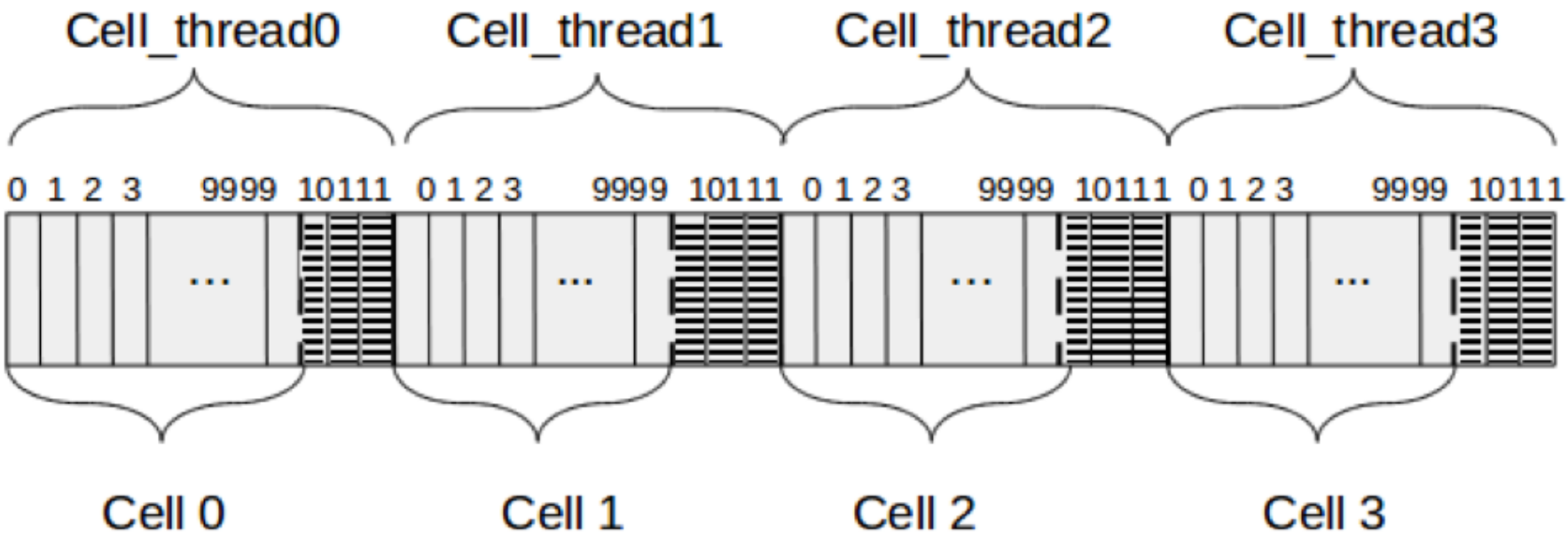
Computational Scope



- $2 * 10^9$ Cells in the heart
- 10^4 Dyads per cell
- 10^2 Ryanodine Receptors (RyRs) per dyad
- 10^4 Time steps per heartbeat

10^{19} possible state transitions

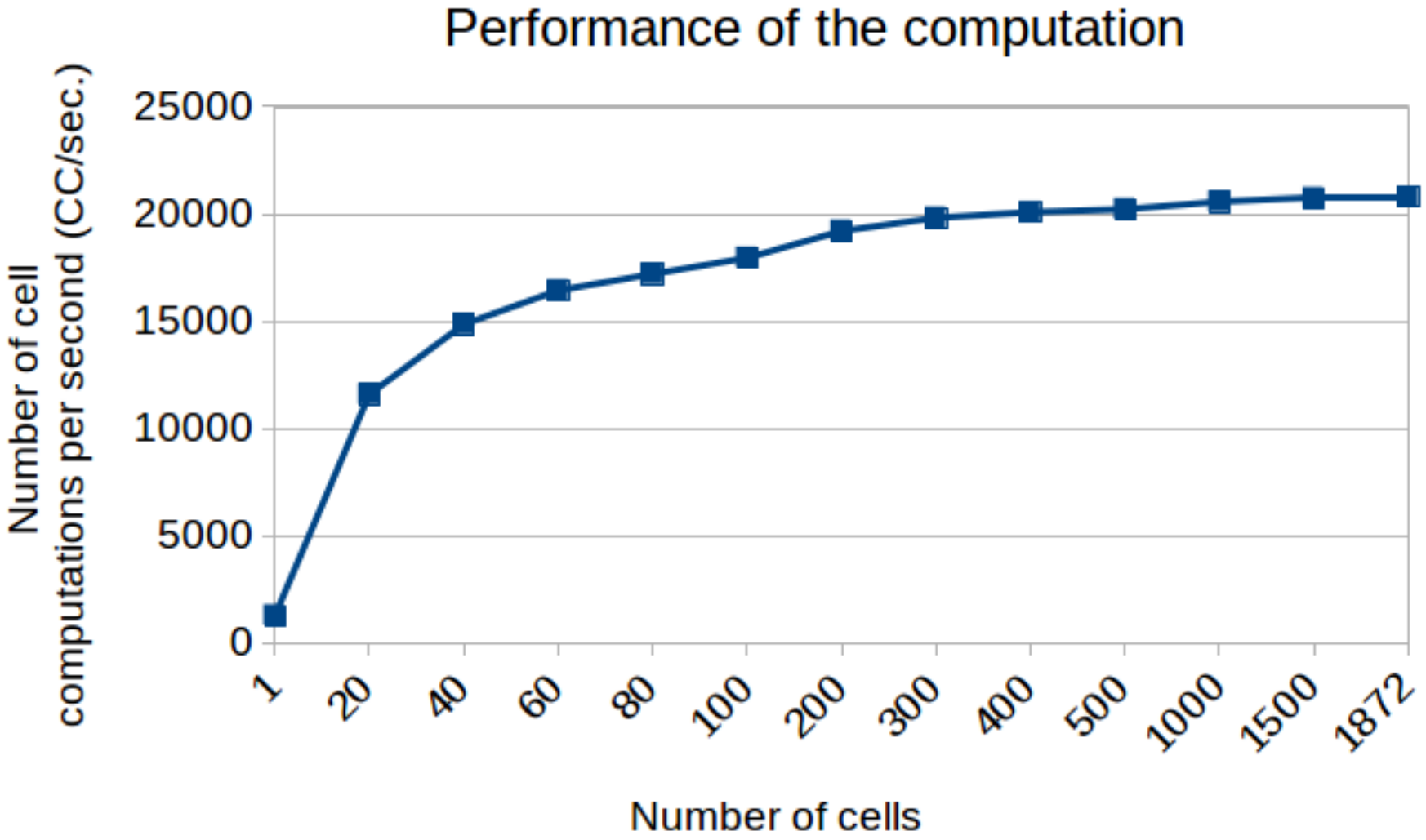
Cell Assignment



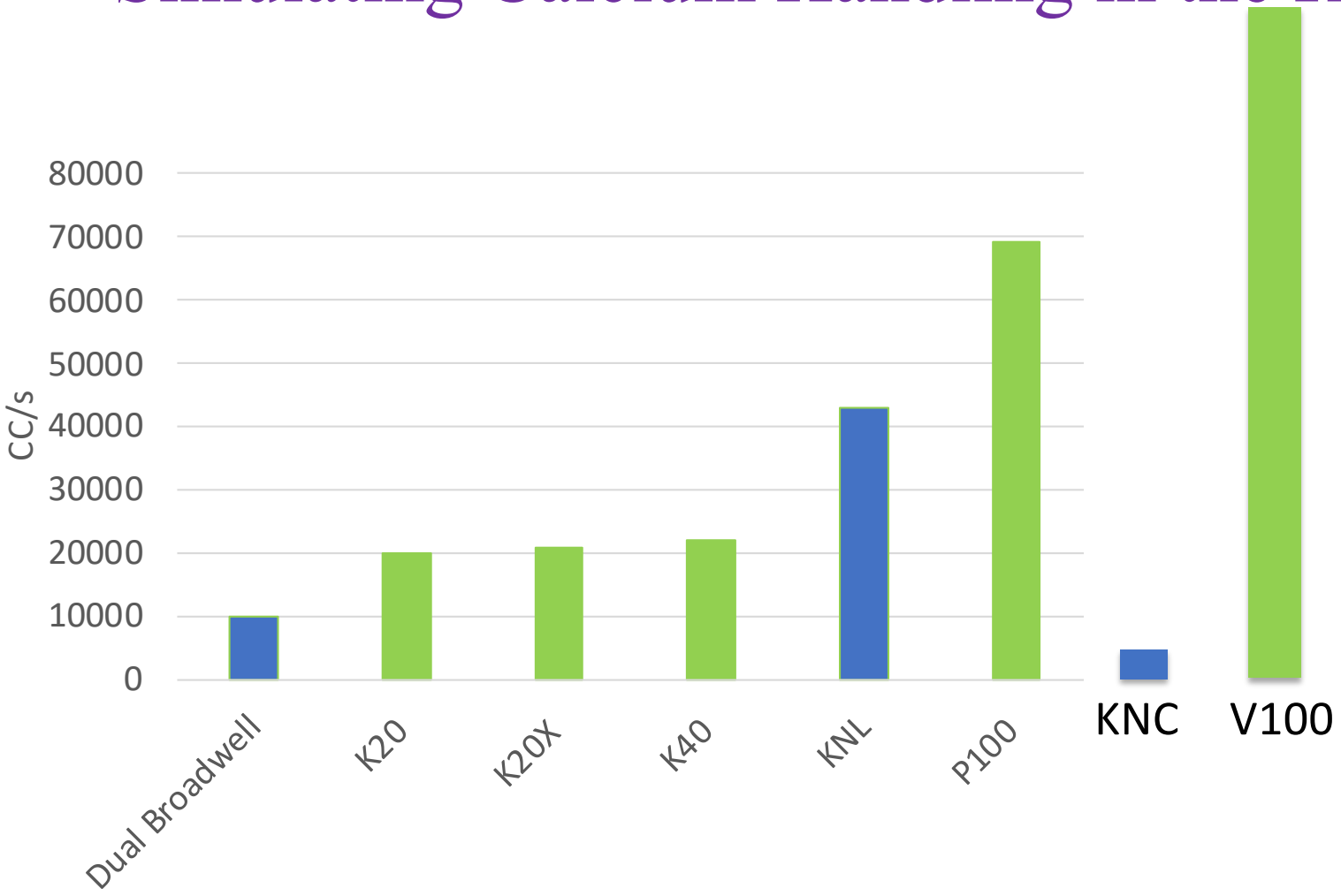
Relevant metric: Cell computations/s (CC/s)

Best performance at 128 threads/block

Amortizing Kernel Launch Overheads

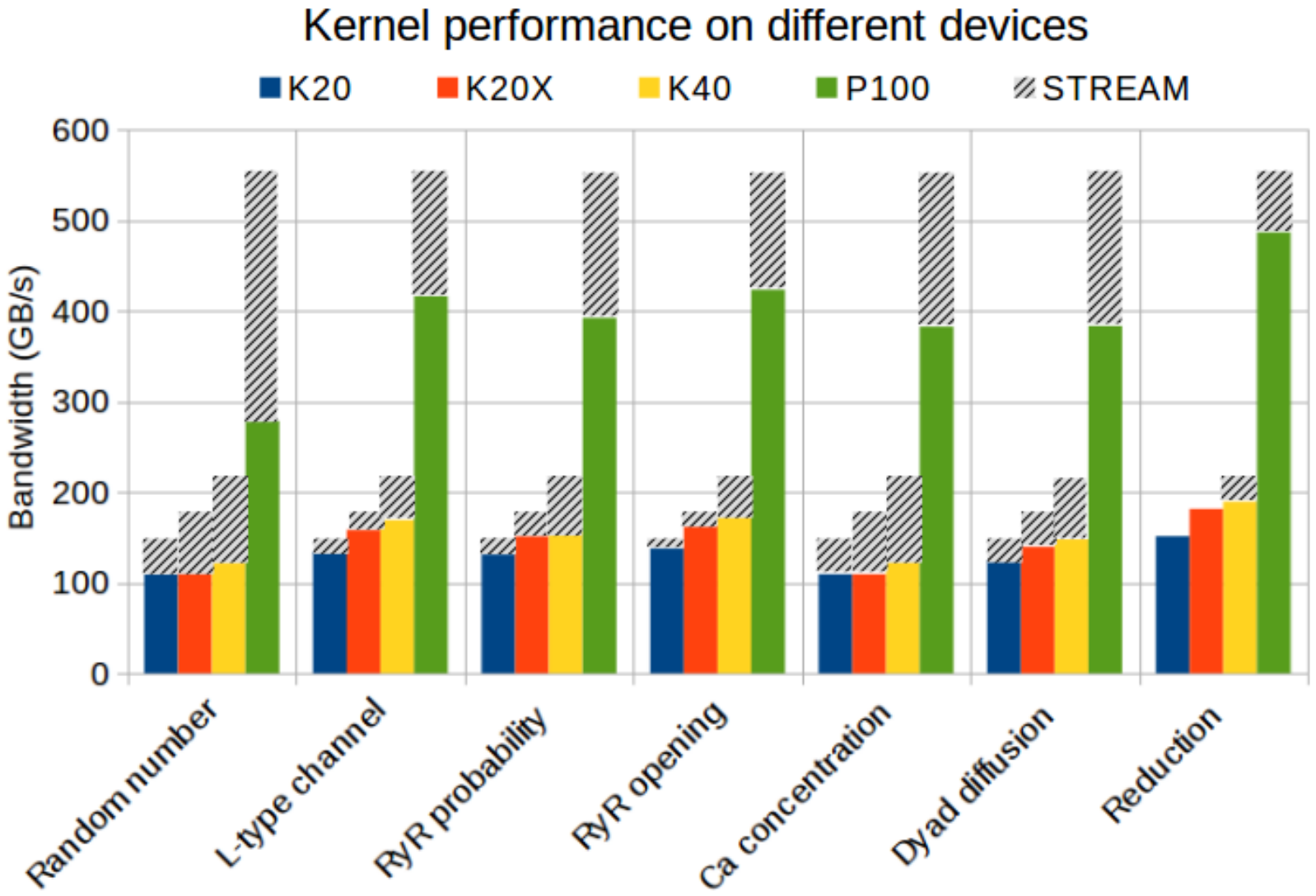


Simulating Calcium Handling in the Heart



- Irregularities in the code
- GPUs still better

Simulating Calcium Handling in the Heart

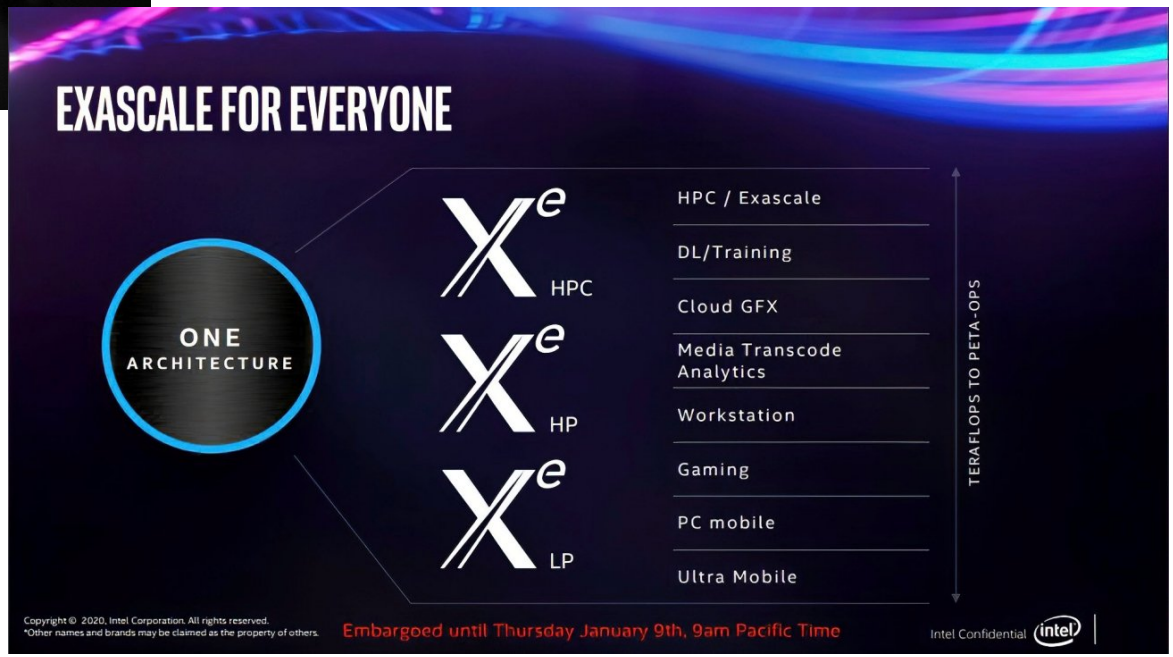


GPU code is close to STREAM bandwidth

Intel: Making GPUs since 2020

- Xeon Phi is discontinued
- Intel under contract for Exascale
- Solution: make a GPU

- Not many details yet
- Aimed at HPC
- Programing: **One API**



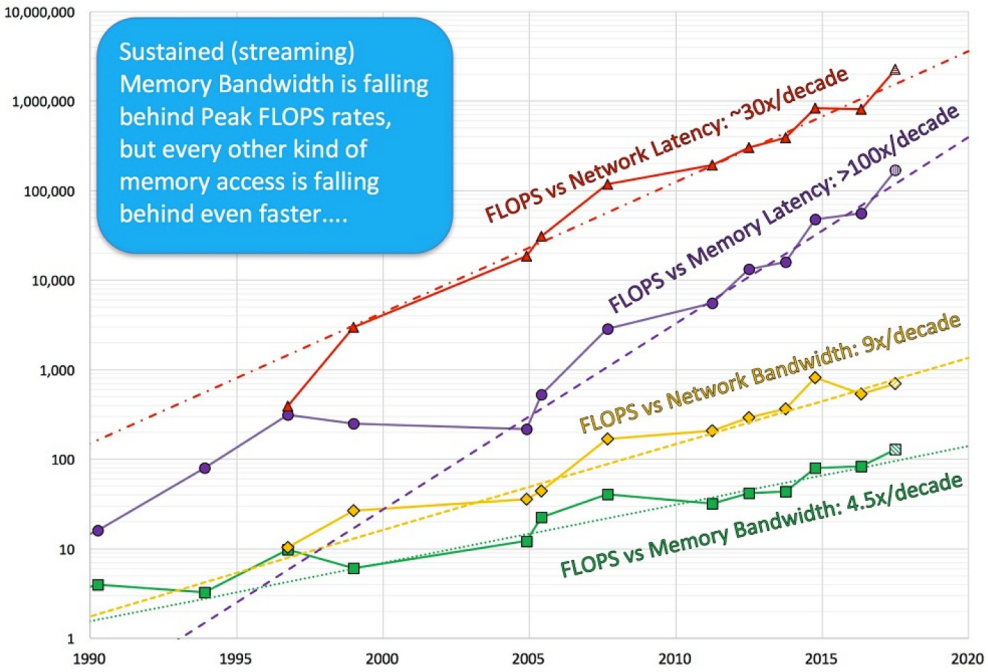
Intel: **one**API

- Data Parallel C++
- Khronos Group SYCL:

Single-source Heterogeneous Programming for OpenCL

Technological Trends of Scalable Systems

- Performance growth primarily at the node level
  - Growing number of cores, hybrid processors
  - Peak FLOPS/socket growing at **50% - 60%** / year
- Communication/computation cost is growing
  - Memory bandwidth increasing at **~23%/yr**
  - Interconnect bandwidth at **~20%/yr**
  - Memory/core is shrinkin



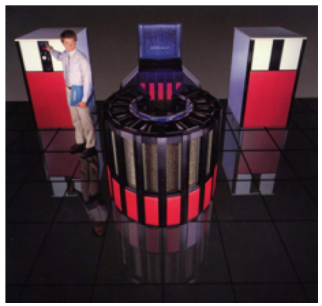
Technological Trends of Scalable Systems

Supercomputer performance growth: $\times 10^3$/decade

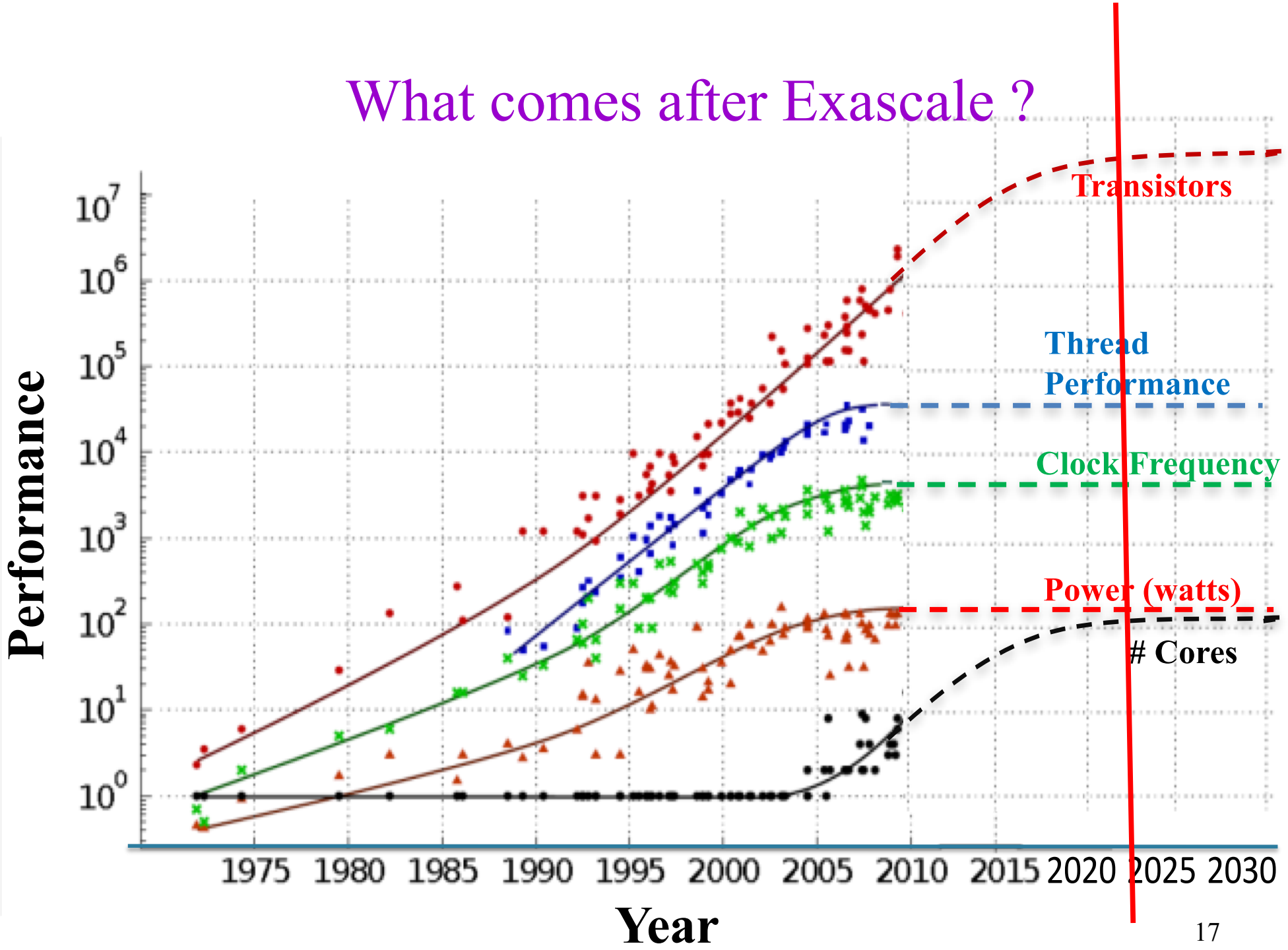
240 Mflops	1TFlop	1 Pflop	148 PFlops
1976	1996	2008	2018
1 core	4510	122K	2.28 M
115KW	850KW	2.35 MW	8.8 MW
2KF/W	1MF/W	0.4GF/W	13.8 GF/W
Cray-1	**ASCI Red**	**Roadrunner**	**Summit**

10^{18}
2021
$\leq$ 30MW

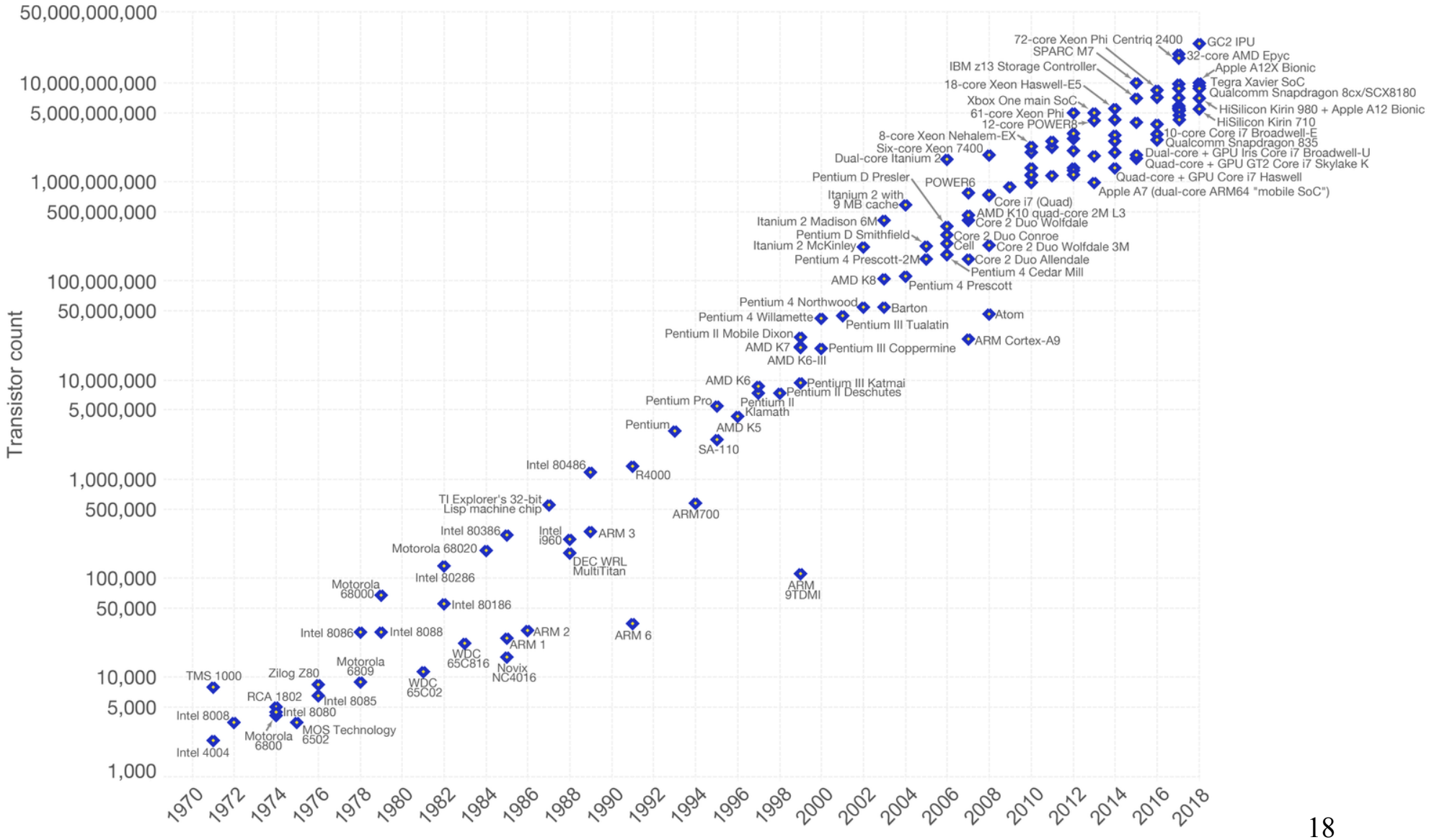
Cray-2
1985-90, 1.9 GFlops
195 KW (10 KF/W)
Ipad-Pro today:
1.6 Gflops



What comes after Exascale ?



The End of Moore's Law ?

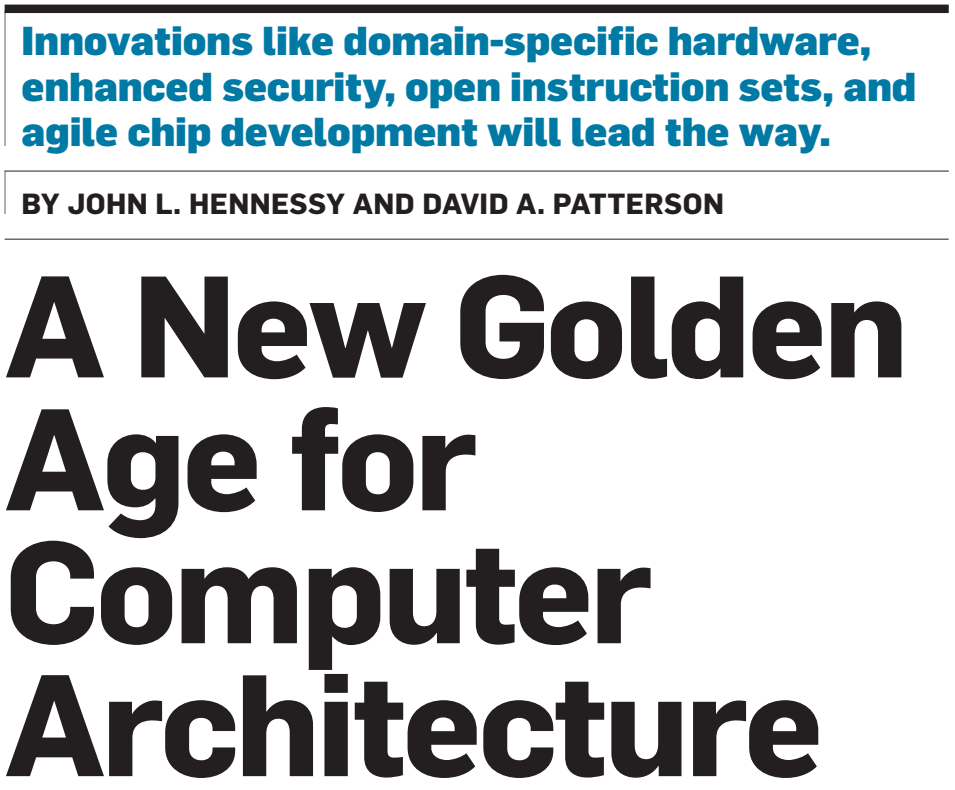


The End of Moore's Law ?

Name	Transistors per mm^2	Year	Process	Maker
Intel 10 nm	100,760,000	2018	10 nm	Intel
5LPE	126,530,000	2018	5 nm	Samsung
N7FF+	113,900,000	2019	7 nm	TSMC
CLN5FF	171,300,000	2019	5 nm	TSMC

Number of Transistors is increasing for now.
Not clear how to best use them though.

On the Horizon: More and More Architectures



Innovations like domain-specific hardware, enhanced security, open instruction sets, and agile chip development will lead the way.

BY JOHN L. HENNESSY AND DAVID A. PATTERSON

A New Golden Age for Computer Architecture

- Clock Frequency has stopped increasing
- Moores Law is coming to an end
- More cores means higher power consumption
- The way out: **adapt architecture to workload**.

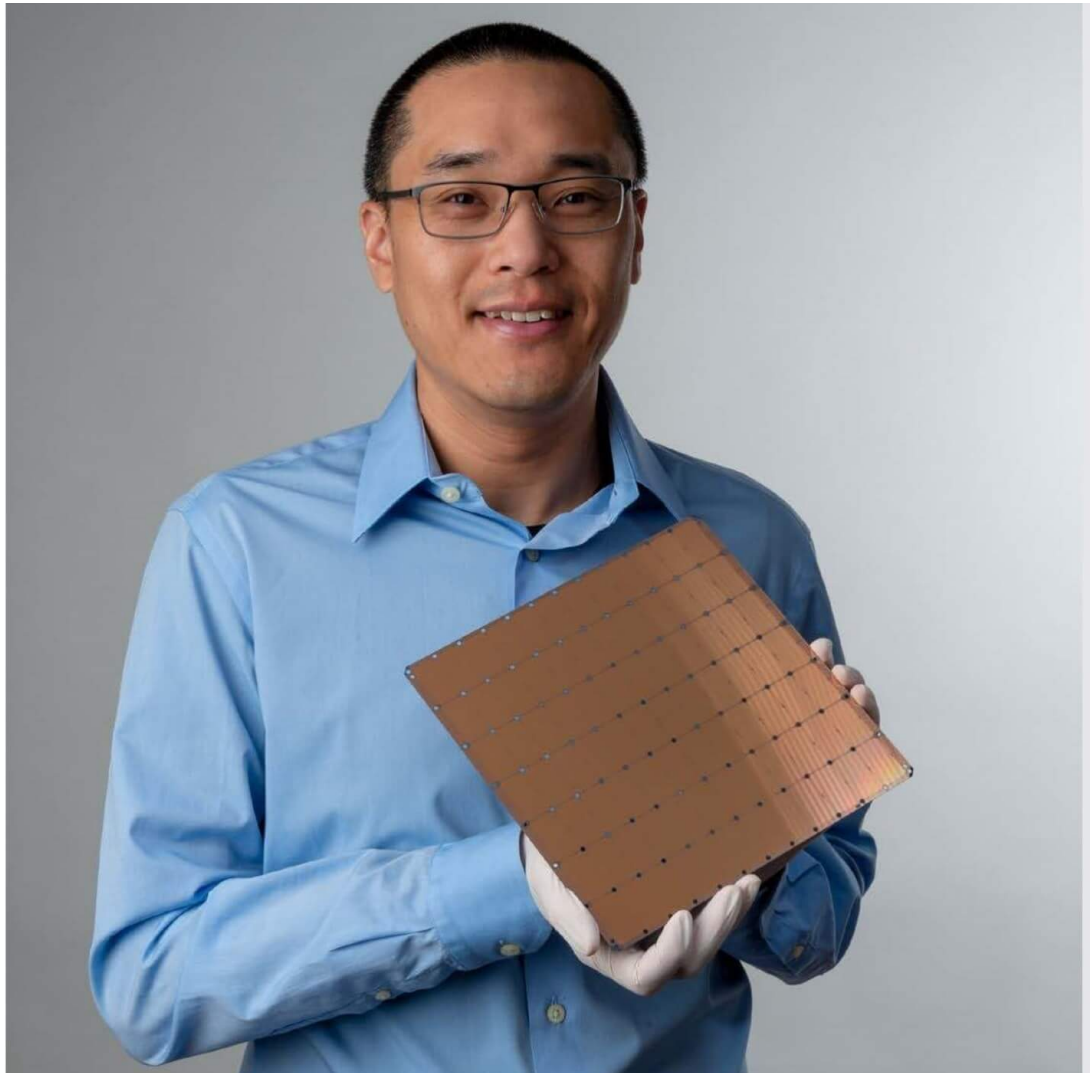
On the Horizon: More and More Architectures

- Lots of startups are working on new processors right now.
- Mostly aimed at AI, but they might still be useful.

Cerebras: Supercomputer on a (big) Chip

Largest Chip Ever Built

- 46,225 mm^2 silicon
- 1.2 trillion transistors
- 400,000 AI optimized cores
- 18 Gigabytes of On-chip Memory
- 9 PByte/s memory bandwidth
- 100 Pbit/s fabric bandwidth
- TSMC 16nm process



cerebras

(not yet available)

Graphcore IPUs. Getting away from SIMD

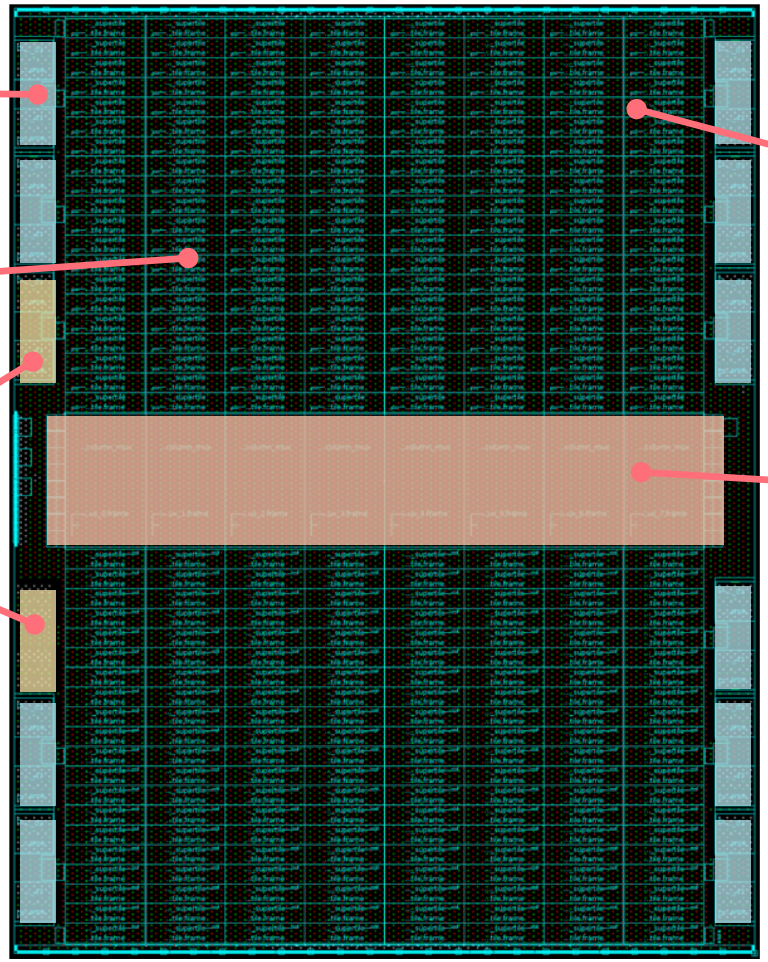
- 1216 Cores per chip / 6 Threads per core
- True MIMD
- Uses SRAM as memory, no DRAM

Graphcore IPUs. Getting away from SIMD

10x IPU-LINKS™
320GB/s chip-to-chip bandwidth

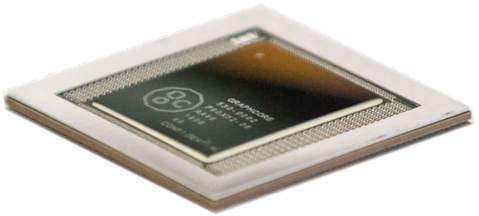
300MB IN-PROCESSOR-MEMORY™
45TB/s memory bandwidth per chip
the whole model held inside the processor

PCIe Gen4 x16
64GB/s bi-directional host
communication bandwidth

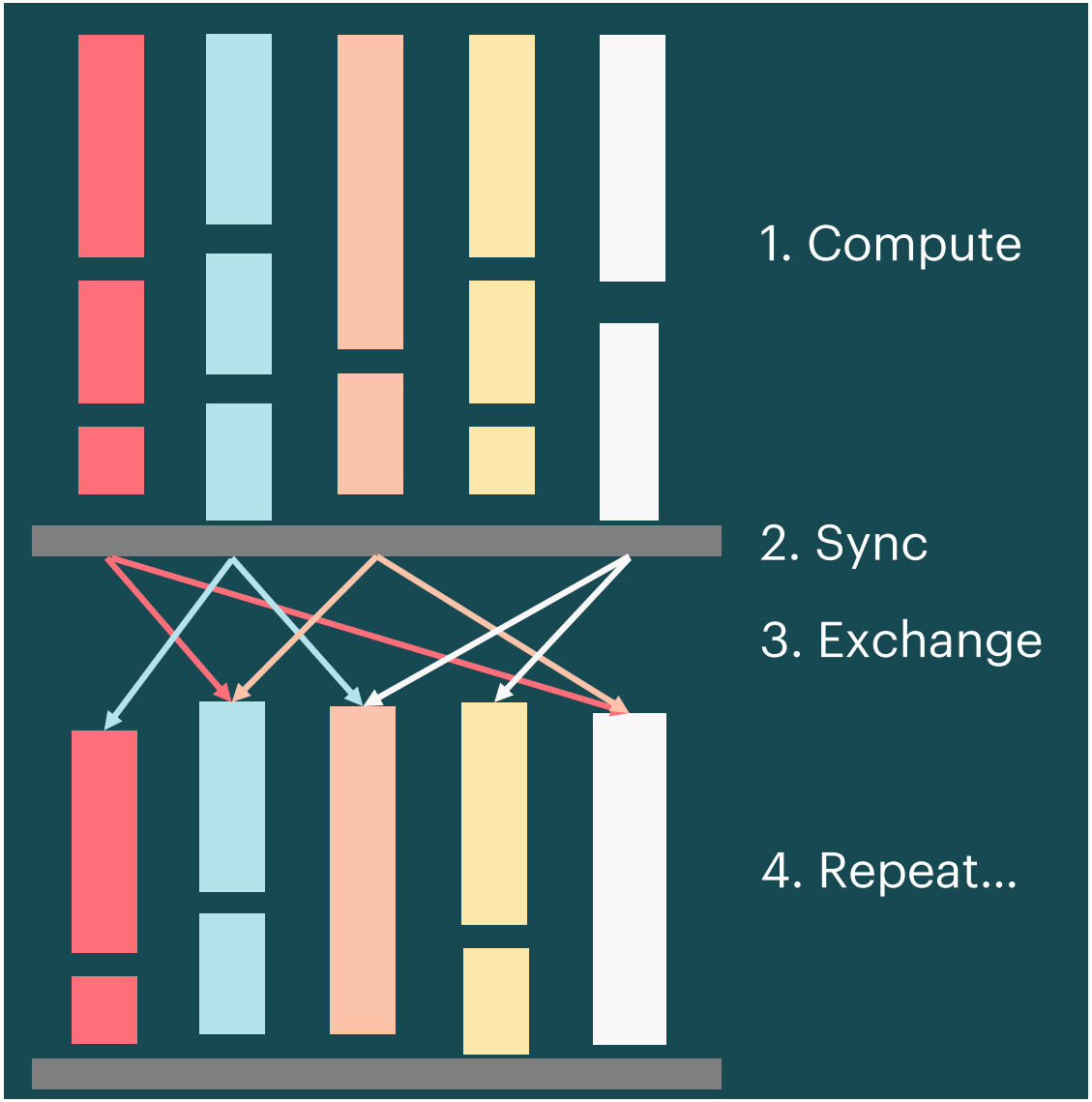


1,216 independent IPU-CORES™
each with IN-PROCESSOR-MEMORY™ tile
> 100GFLOPS per IPU-CORE™
> 7,000 programs executing in parallel

8TB/s all to all IPU-EXCHANGE™
non-blocking, any communication pattern



IPU uses Bulk Synchronous Processing on chip



Graphcore IPUs. What Changes ?

- "Flops are cheap" - still true (at least single precision)
- "Bandwidth is expensive" - no longer true (if program fits)
- "Latency is physics" - no longer relevant
- But: **memory is small**.

Need to use a large number of IPUs for large problems

Lots of IPUs

Graphcore IPU Characteristics

- 6 Threads per core
- 6 Cycles memory latency
- 31.1 Single precision FLOPS
- No double units
- Memory Bandwidth: 7.5 - 30 TB/s

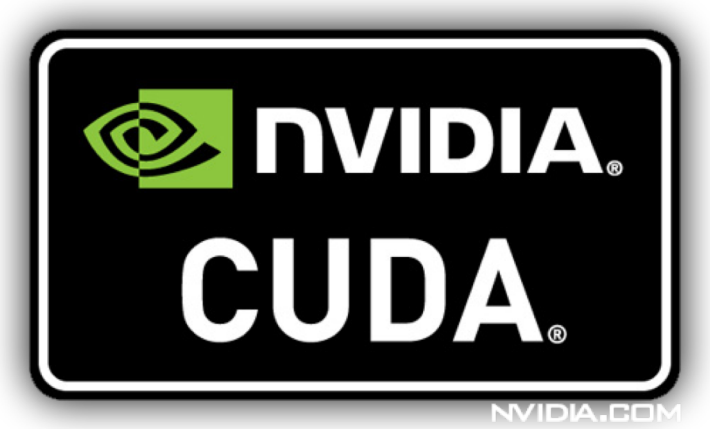
- Tensor core type units
- 1.6 Ghz Clock frequency
- 6.3 GB/s between cores
- Multi-IPU transparent to programmer

But will it work for Scientific Computing ?

- Need double precision units
- Memory is low, but 64 * 300 MB = 19.2 GB
- Codes will need to use a large number of IPUs
- Connection between them and partitioning of data will be crucial

Summary

- GPU programing with CUDA offers a lot of performance
- Acceptable extra effort, but requires understanding new concepts
- Lots of mature software for free
- Multi GPU gets progressively harder – use as needed
- We may see more novel architectures in the future



References

Altanaite, N., & Langguth, J. (2018, February). Gpu-based acceleration of detailed tissue-scale cardiac simulations. In *Proceedings of the 11th Workshop on General Purpose GPUs* (pp. 31-38).

Langguth, J., Lan, Q., Gaur, N., & Cai, X. (2017). Accelerating detailed tissue-scale 3D cardiac simulations using heterogeneous CPU-Xeon Phi computing. *International Journal of Parallel Programming*, *45*(5), 1236-1258.

Langguth, J., Lan, Q., Gaur, N., Cai, X., Wen, M., & Zhang, C. Y. (2016, December). Enabling tissue-scale cardiac simulations using heterogeneous computing on Tianhe-2. In *2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS)* (pp. 843-852). IEEE.

Credit: Lecture contains NVIDIA material available at https://developer.nvidia.com/cuda-zone
Image source: wikipedia.org, anandtech.com
Contains material from ACACES 2018 summer school, originally designed by Scott Baden
IPU material provided by Graphcore Norway.