Google

Architectures for the AI Era: From a TPU to the Extreme Scale

Norman P. Jouppi, with contributions from the Google TPU team

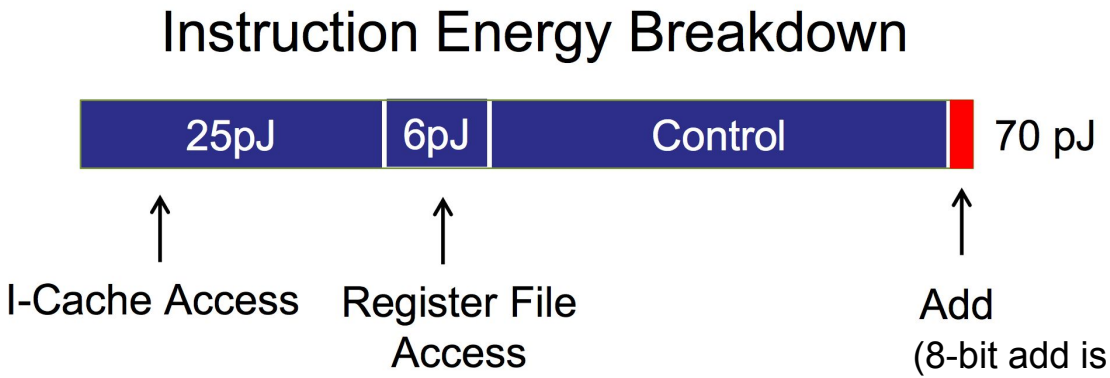
June 23, 2025

The Big

https://www.youtube.com/watch?v=x_x-JAAKSvU

Key Insight #1

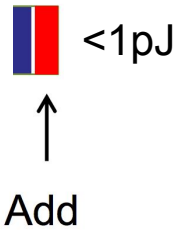
- Energy for control logic, SRAM, and register accesses needed by matrix multiply dominates in CPUs
- Example from Mark Horowitz's ISSCC 2014 Keynote, slide 33: "Computing's Energy Problem: (and what we can do about it)":



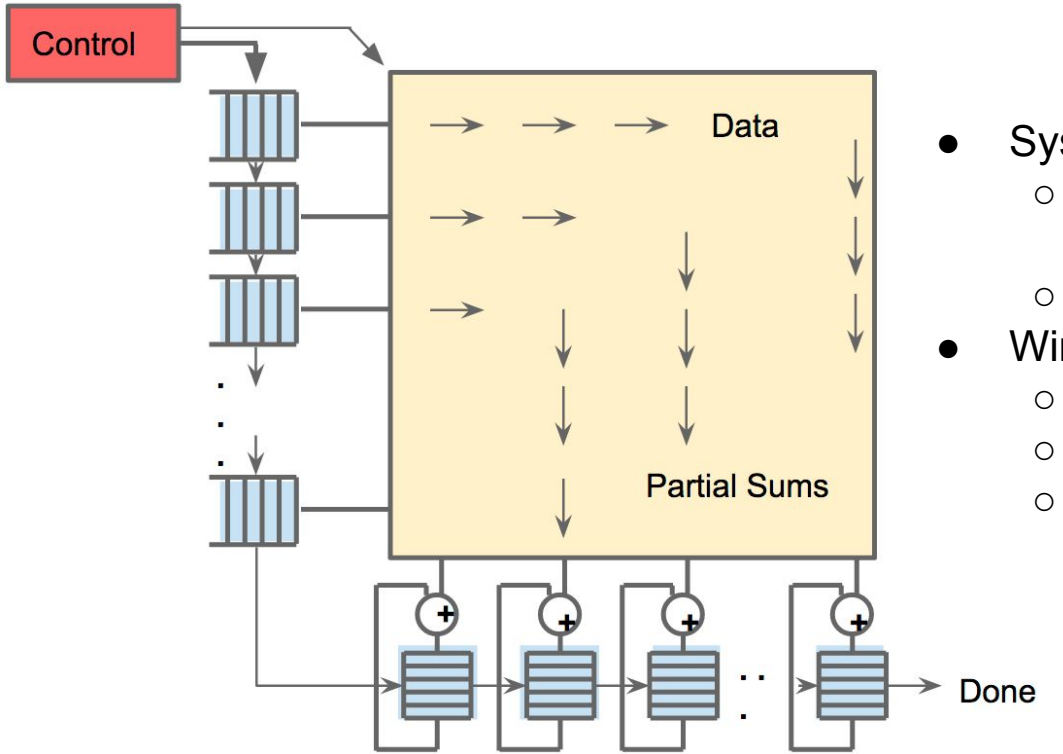
Key Insight #1

- Solution: matrix operations on a 256x256 systolic array
  - Eliminate complex control logic (use pipelined enable bit)
  - Reuse fetched memory and register data >100X
  - Reduce energy overhead per compute by >10X

Instruction Energy Breakdown



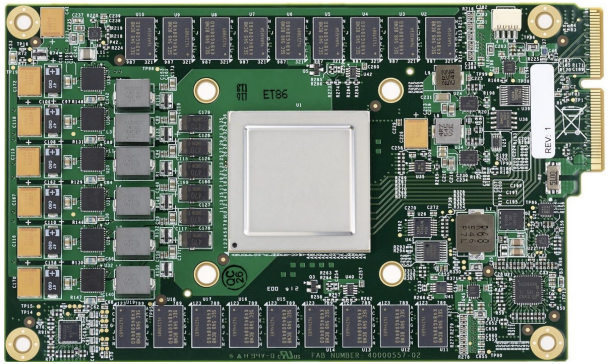
Systolic Execution: Data is Pipelined

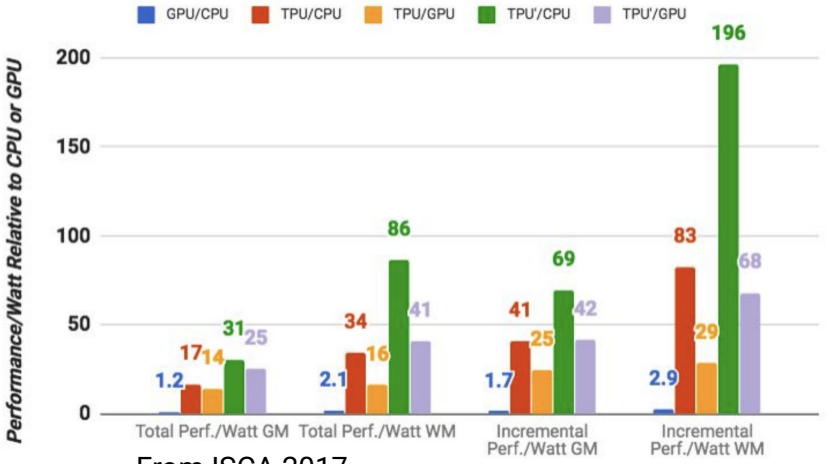


- Systolic arrays first proposed in *1970's*
  - But largely forgotten in mainstream computer architecture
  - Only multipliers, adders, and flops
- Wiring by abutment within array
  - Saves wire power
  - Avoids memory accesses
  - No complex control logic

Late 2013

- TPUv1 project started
  - TPU = Tensor Processing Unit, an example of a DSA
  - DSA = Domain-specific architecture
  - Tensor = multidimensional array
- Provided >10X better perf/TCO than contemporary alternatives
  - perf/TCO = end-to-end performance / total cost of ownership (including power over lifetime)
  - Simple to deploy PCIe card
  - But it only accelerated inference





From ISCA 2017

Late 2014

- TPUv1 was being fabbed
- We realized training capability was the limiting factor to producing models
- People thought a training chip would be too complicated to build

Late 2014

- TPUv1 was being fabbed
- We realized training capability was the limiting factor to producing models
- People thought a training chip would be too complicated to build
- So we decided to build a training chip plus a supercomputer 😀

Basic Plan for TPUv2

- Don’t invent anything more than necessary
  - Required to meet aggressive schedule
- Codesign from compiler down to chip physical design
- Start from a typical vector CPU architecture and add matrix operations
  - Similar to how the Cray-1 extended previous scalar machines with vector operations *in 1975*
  - Advantage: start with an architecture model with a compiler and add stuff
  - Leverage long-known compiler techniques for matrices in HPC (e.g., blocking, loop unrolling)
  - Use 8-operation VLIW architecture baseline since compiler schedules multiple ops/cycle
  - 8 instructions per cycle is a super-beefy scalar core!

Cray-1 Architecture

Circa 1975

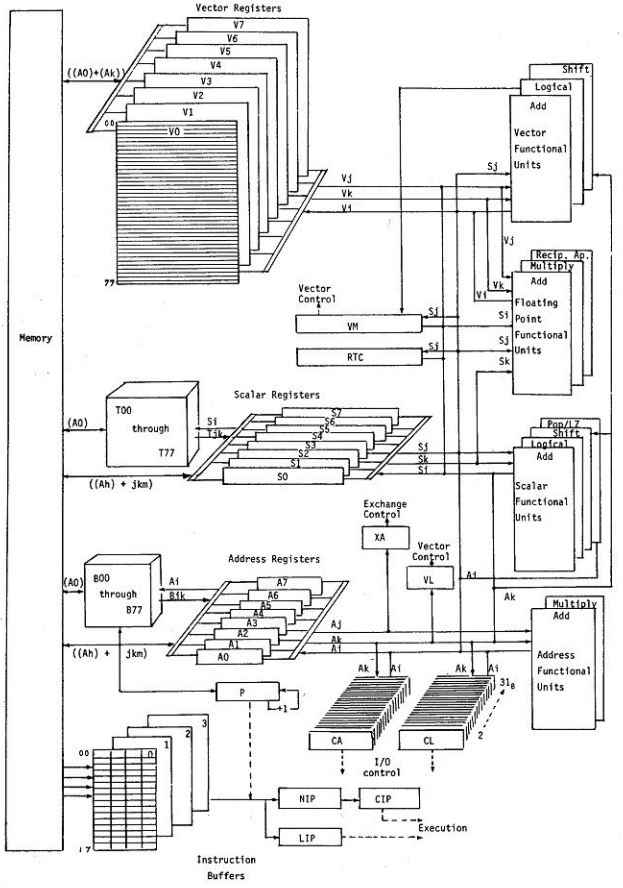


Figure 3-1. Computation section

This part looks like previous CDC6600 and CDC7600 machines

Cray-1 Architecture

Circa 1975

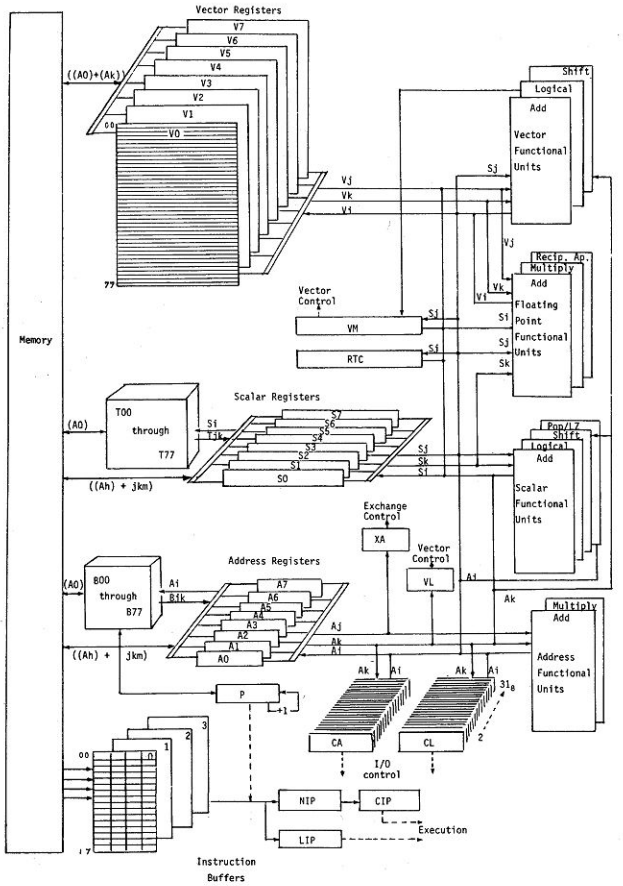


Figure 3-1. Computation section

Cray-1 added vector hardware in a consistent manner

This part looks like previous CDC6600 and CDC7600 machines

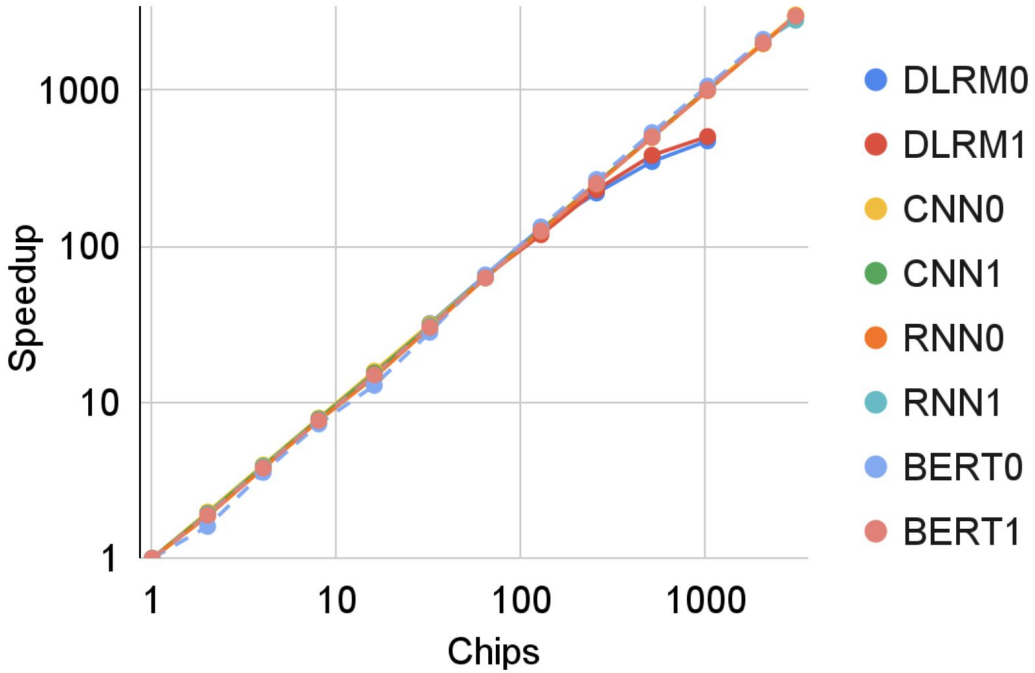
Basic Plan for TPUv2 (Part 2)

Key Insight #2

- Connect TPUs with **shared memory** distributed using high-bandwidth torus (ICI)
  - Similar to the Cray T3D's 3D torus (*circa 1993*), but simpler
  - Leverages the intrinsic multidimensional array structure of tensor math mapped to interconnect
- ICI is 50X faster and 10X cheaper than Ethernet
  - No layers of protocol stacks
  - Many connections are PCB traces or short cheap copper cables
- **ICI is the second key TPU feature (after systolic arrays)**

Scalability on Real Workloads Using High-BW Torus

- Due to **shared memory** using **extreme interconnects** at unprecedented scales (many ExaFLOPS):
  - 99% scaling efficiency on 75% of workloads to 3K TPUv4



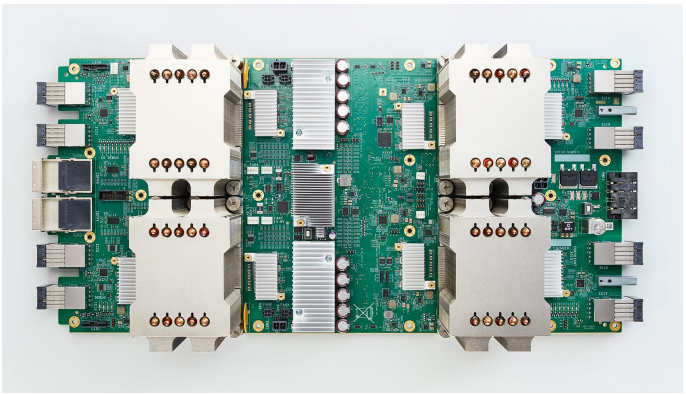
Basic Plan for TPUv2 (Part 3)

Key Insight #3

- Training was currently being done on CPUs and GPUs using FP32
  - Google's SW stored FP32 values in 16 high-order bits to save storage space
  - Conversion from FP32 to 16 bits was performed by simple truncation (ouch!)
  - Preserved dynamic range while reducing precision
  - This datatype was called BF16 (Brain Float 16) in the SW
- Existing 16-bit FP formats (IEEE FP16) didn't have enough dynamic range
- We realized we could supply BF16 inputs to multipliers, keep all product bits (i.e., FP24), and perform FP32 accumulation and get identical results as current SW
  - This saved multiplier HW
  - And we rounded to nearest even on conversion, giving better results
- But most importantly, it maintained SW compatibility with CPUs and GPUs
  - Models could train on CPUs, GPUs, and TPUs and all get the same results
- **Hence BF16 is the 3rd key TPU feature**

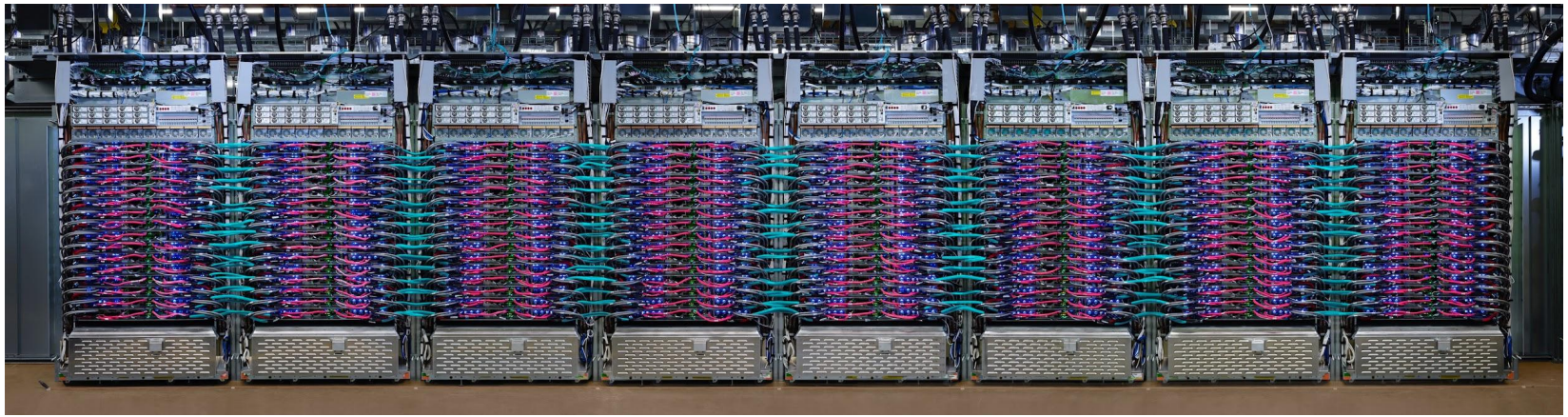
TPUv2

- 256 chips connected in a 2D torus
  - Narrow TPU trays, 4 per rack shelf
  - ICI ran 64 SerDes at 40Gb/s (“only” 2.56Tb/s per chip)
- Air cooled due to lower power consumption and faster time-to-market
- Servers were in separate racks



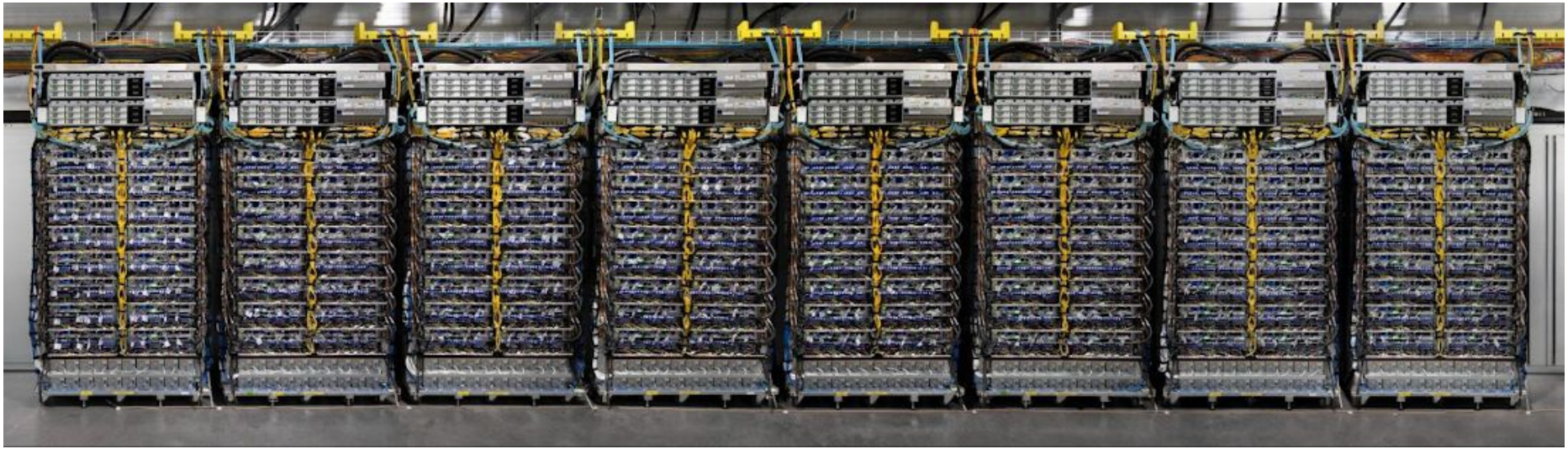
TPUv3

- Most of the team was working on TPUv2 bringup
- Hence only limited logic changes to TPUv2 were possible in TPUv3, but:
  - Optimized chip physical design to make room for 2nd MXU per core
  - Larger scale (4X chips) in 2X racks with 2X rack power supplies per rack
  - First TPU with water cooling
  - Optical cables for wrap-around torus links
  - 2X capacity per HBM
- 10X TPUv2 performance



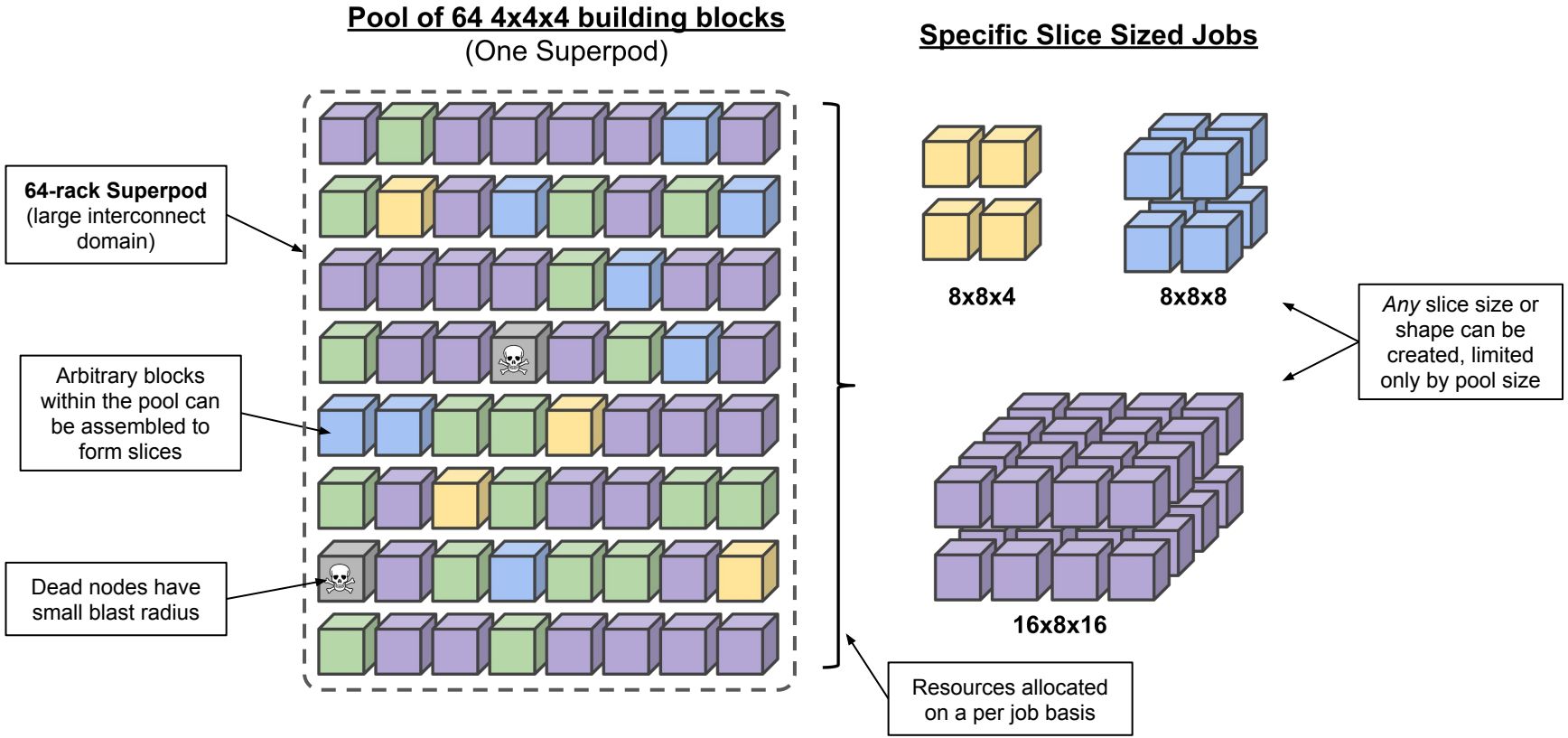
TPU v4p

- Superpods of 64 racks of water-cooled compute (8 shown in photo)
  - Provides over 1 ExaFLOP (10X TPUv3 performance)
- Many superpods are connected via datacenter networking into bigger clusters



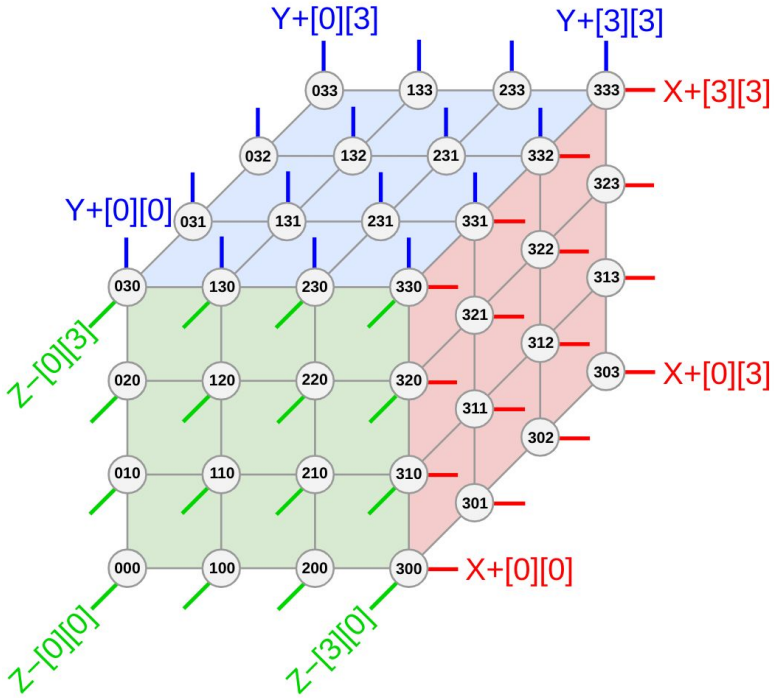
What Is a TPU Superpod?

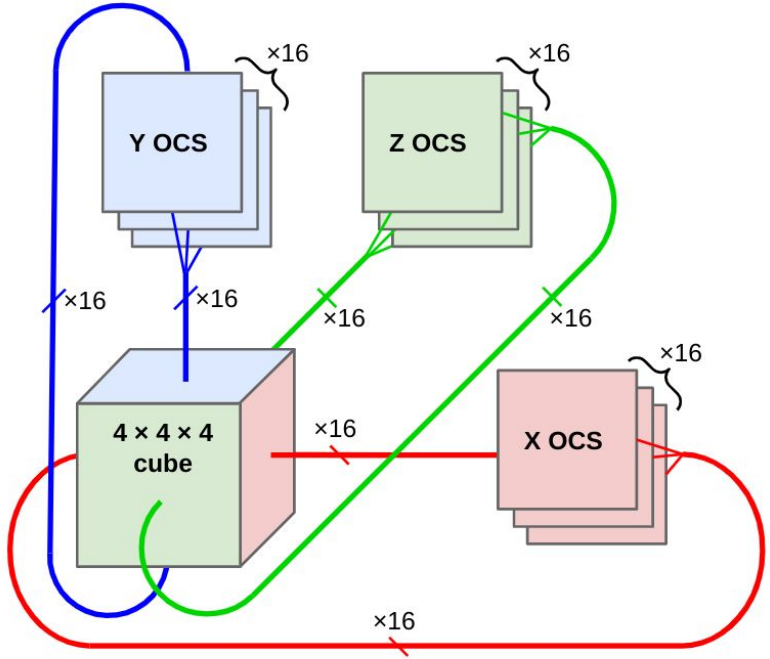
A large pool of building blocks that can be connected on a per-job basis to form larger slices.



Racks Are Connected With Optical Circuit Switches (OCS)

- Different ranks of OCS connect different dimensions and indices





V5p Superpods (8960 chips each, 4+ ExaFLOP)

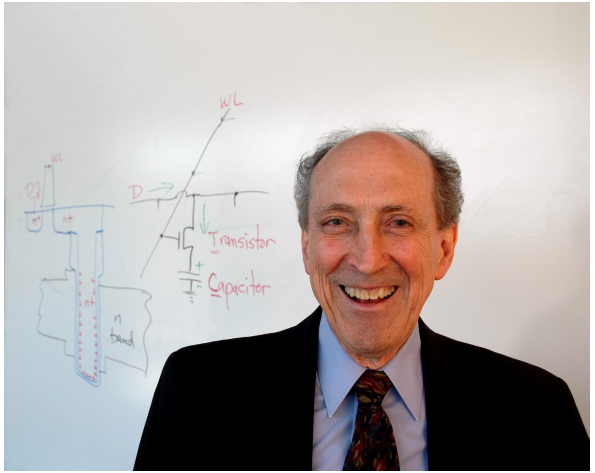
https://www.youtube.com/watch?v=hszd5UqnfLk&t=3s

Google I/O 2024 Keynote

The Small

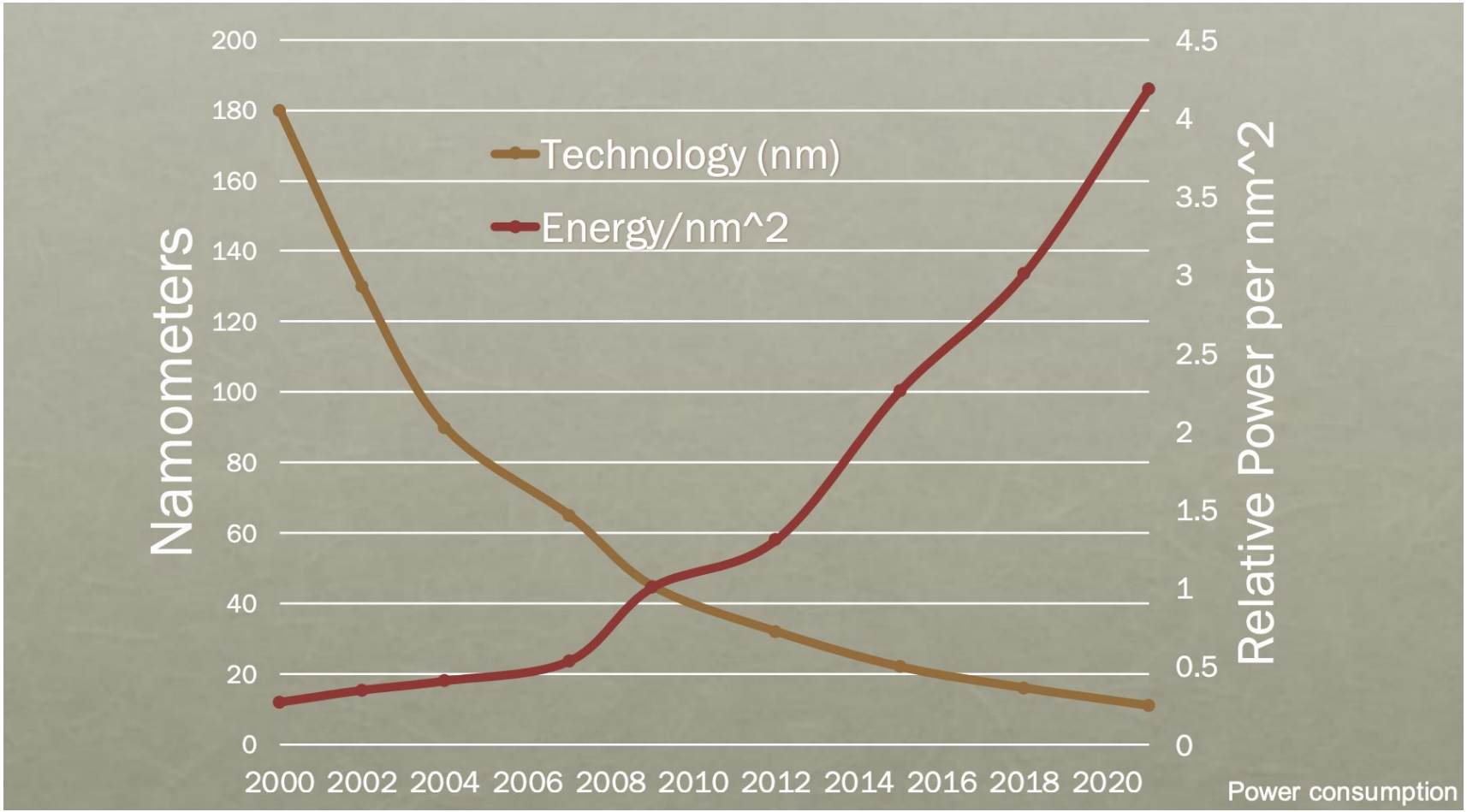
The End of Dennard Scaling

- Dennard Scaling:
  - Shrinking process lithography gives you more transistors
  - Scale voltage down
  - Power per area at same frequency constant
- This scaling ended around around 2004
- Hence chip **power density** will increase every year from now to the end of lithographic scaling
  - This is not the same world as 15 years ago
  - We need to “Think Different”



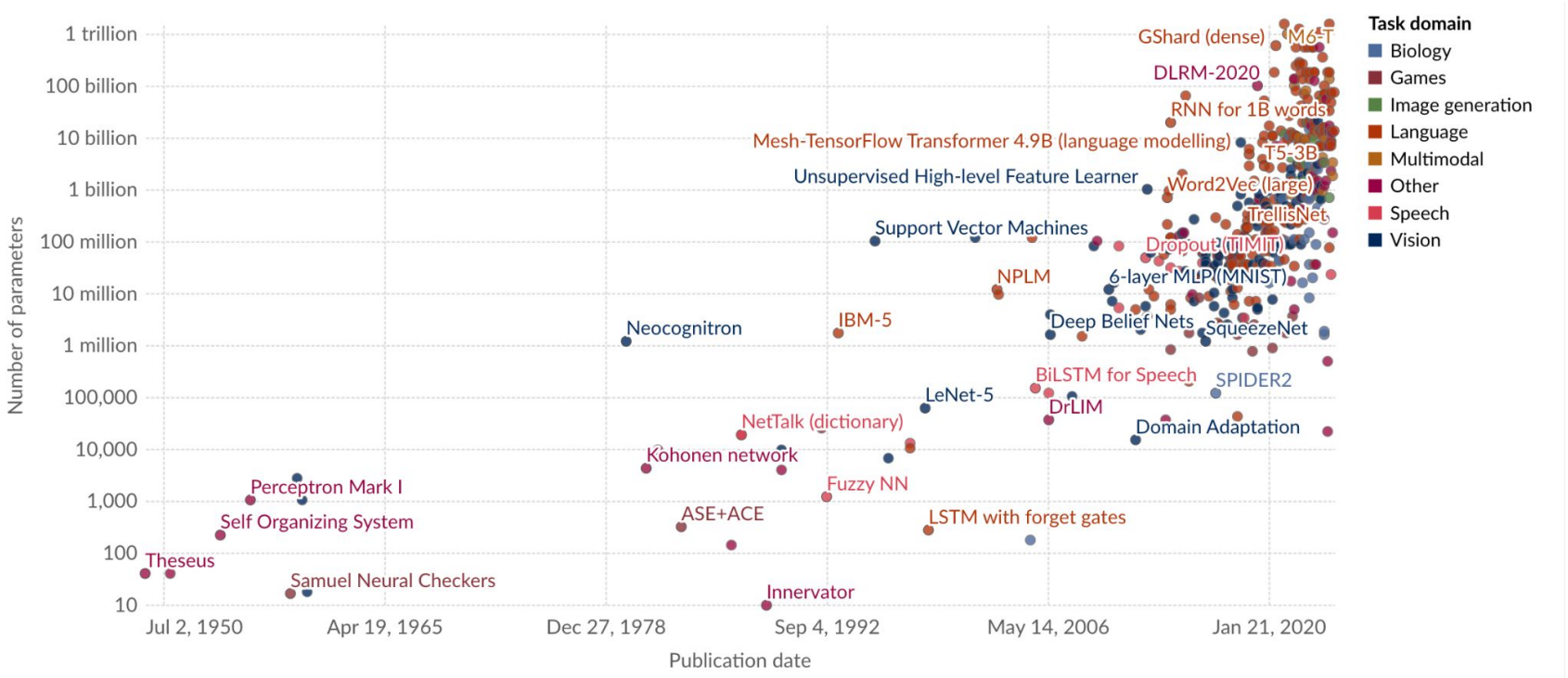
Robert Dennard, member NAE
P.S. He also invented the 1T1C DRAM

- Reference: John Hennessy’s Turing Award talk: “The end of Dennard scaling is an equally important problem as the end of Moore’s Law, but doesn't receive as much attention”



From John Hennessy's Plenary talk at DARPA 2018 ERI Summit

Current Era: Model Growth Accelerates to 10X/Year!



Source: Epoch (2024) – with minor processing by Our World in Data

Reduced Precision Formats

- Important to reduce the cost of serving large LLMs:
  - Memory footprint
  - Compute
  - Energy / power consumption
- But must preserve accuracy compared to larger formats
  - One mistake in a billion can make the news
- Automation is crucial to enable adoption
  - We can change the model to make it easier to quantize without loss of accuracy
  - Our ASPLOS paper: Hyperscale Hardware Optimized Neural Architecture Search

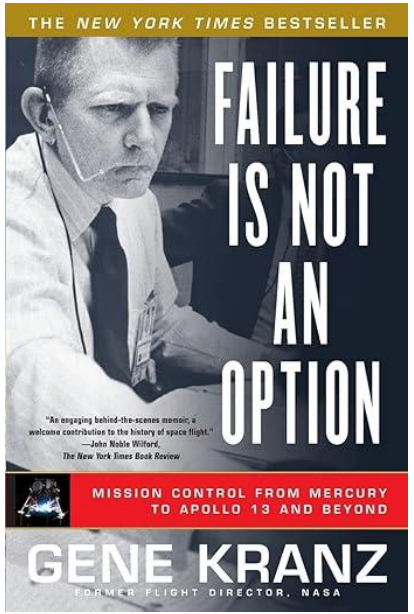
The Not Right at All

RAS: Reliability, Availability, and Serviceability

- Received a lot of attention in the past from finance and supercomputing
  - Finance: Debit of $1M vs. $1 is a big real-time problem
  - Less of a problem with large-scale scientific computations
- Now a significant issue with ML training
  - Similar to large-scale supercomputer calculations, only worse due to even larger scale
- Real-time reliability not required
  - Recovery from checkpoints OK
  - But errors must be detected
  - And overhead from restart can become burdensome if too frequent

Silent Data Corruption (SDC)

- **Compared to silent data corruption, failure is a good option**
  - We need to know we had an error so we can restart from a checkpoint
- But they can be expensive to detect
  - Tandem Nonstop fully duplicates computation and storage
- Algorithm Based Fault Tolerance
  - Opportunities for further research
  - A range of useful techniques would be helpful



Conclusions

Summary

- Research in AI is advancing at a tremendous pace!
- But we are limited by the compute to run these tools
- Research is needed in:
  - Reliable computation totalling from yotta 10^{24} to ronna 10^{27} ops
    - Provided by zetascale systems running for months
  - Advances in composite models (mixture of experts and beyond)
  - Higher interconnect speeds (including bisection for all-to-all)
  - Use SOTA software engineering for modeling and simulation, including but not limited to:
    - Code reviews
    - Regression testing
    - Include regular reviews by architects
    - Making sure models at different levels of abstraction meet at their boundaries
  - And many others

Q & A