Future Storage Systems
A Dangerous Opportunity

Past, Present, Future

Rob Peglar
President
Advanced Computation and Storage LLC
rob@advanced-c-s.com
@peglarr

But First

GO BLUES!

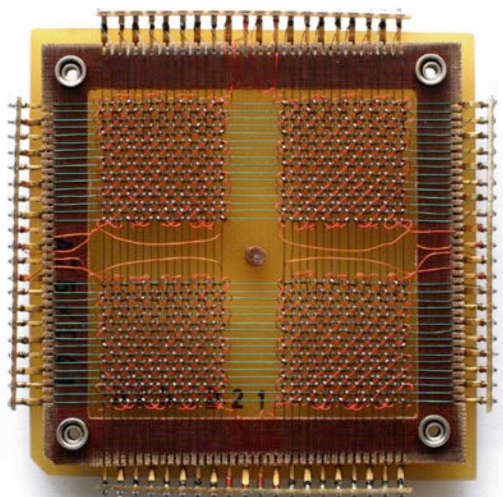
Wisdom


https://twitter.com/compscifact/status/602271417330225153
Twitter, Inc. [US]
Computer Science on Twitt...
New to Twitter?
Sign up
Search Twitter
Have an account? Log in
Computer Science
@CompSciFact
Follow
"The idea that people knew a thing or two in the '70s is strange to a lot of young programmers." -- Donald Knuth
RETWEETS
380
FAVORITES
336
5:34 PM - 23 May 2015
Thomas Irenaeus @peritutvivat · May 23
@CompSciFact You can say this about human "history" since today's post-modern arrogance dismisses historical knowledge as medieval.
1
4
Alec Clews @alecthegeek · May 23
6:28 AM
5/25/2015

The Micro Trend
The Start of the End of HDD

- The HDD has been with us since 1956
  - IBM RAMAC Model 305 (picture →)
  - 50 dual-side platters, 1,200 RPM, 100 Kb/sec
  - 5 million 6-bit characters (3MB)
- Today – the SATA HDD of 2019
  - 8 or 9 dual-side platters, 7,200 RPM, ~150 MB/sec
  - 14 trillion 8-bit characters (14TB) in 3.5” (w/HAMR, maybe 40TB)
  - Nearly 3 million X denser; 15,000 X faster (throughput)
  - Problem is only 6X faster rotation speed – which means latency
- With 3D QLC NAND technology we get 1 PB in 1U today
- Which means NAND solves the capacity/density problem
  - Throughput & latency problem was already solved
  - Continues to improve by leaps and bounds (e.g. NVMe, NVMe-oF)
- <u>HDD may be the “odd man out” in future storage systems</u>

The Distant Past: Persistent Memories in Distributed Architectures

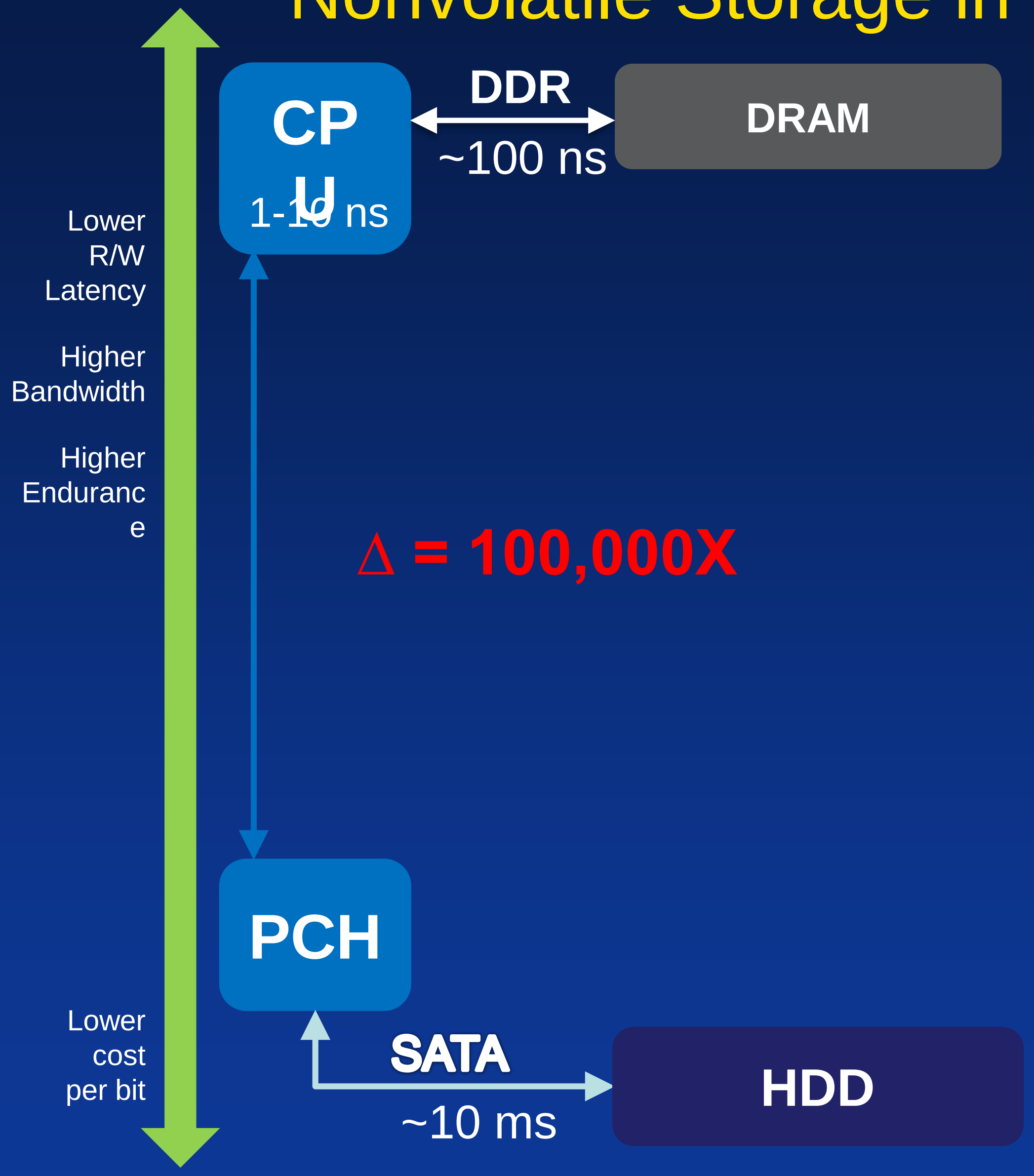


Courtesy Konstantin Lanzet

Courtesy CDC

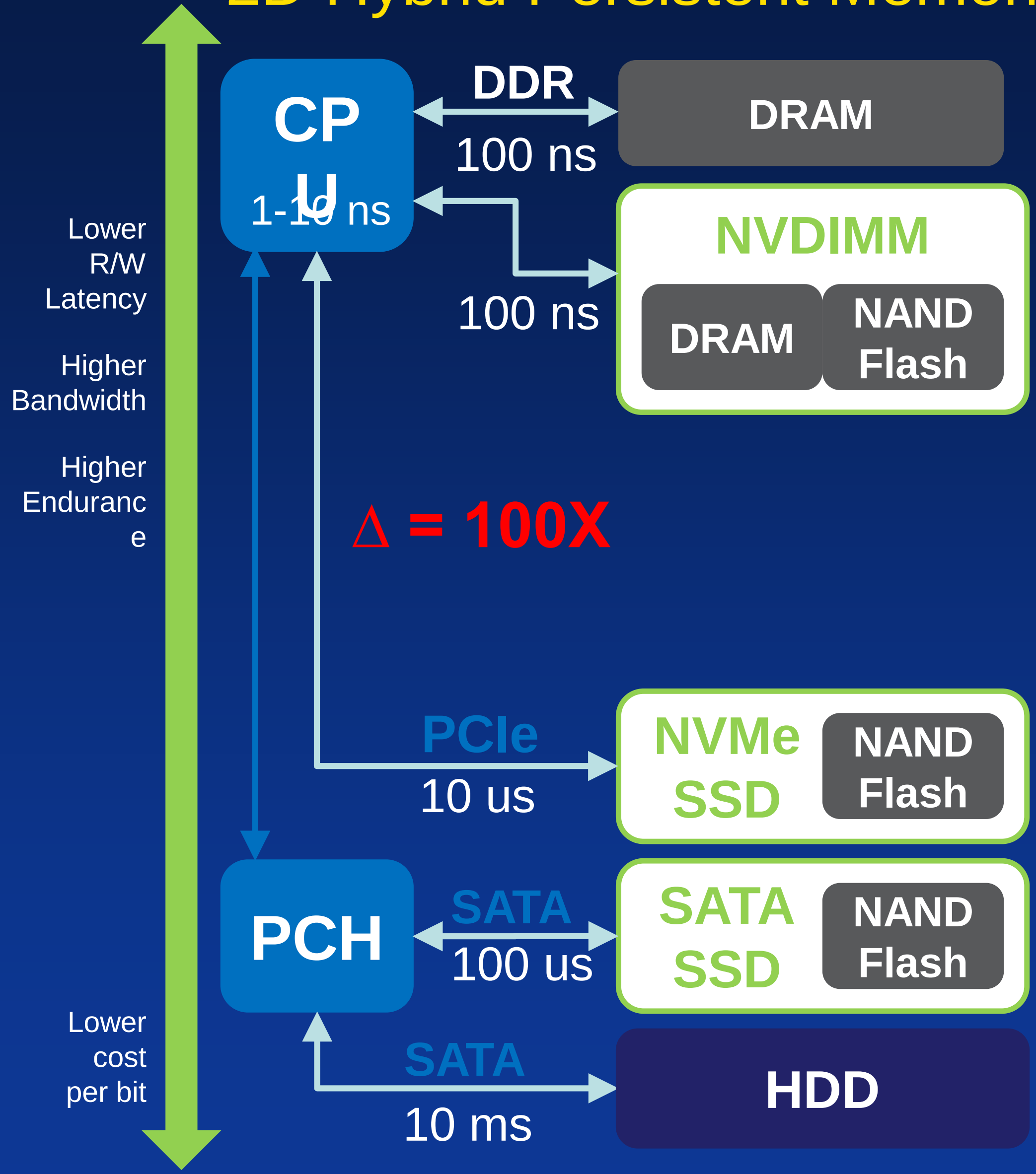
- Ferrite Core memory
- Module depicted holds 1,024 bits (32 x 32)
- Roughly a 25-year deployment lifetime (1955-1980)
- Machines like the CDC 6600 (depicted) used ferrite core as both local and shared memory
- CDC 7600 4-way distributed architecture – aka ‘multi-mainframe’
- Single-writer/multiple-reader concept enforced in hardware (memory controllers)

The Past:
Nonvolatile Storage in Server Architectures



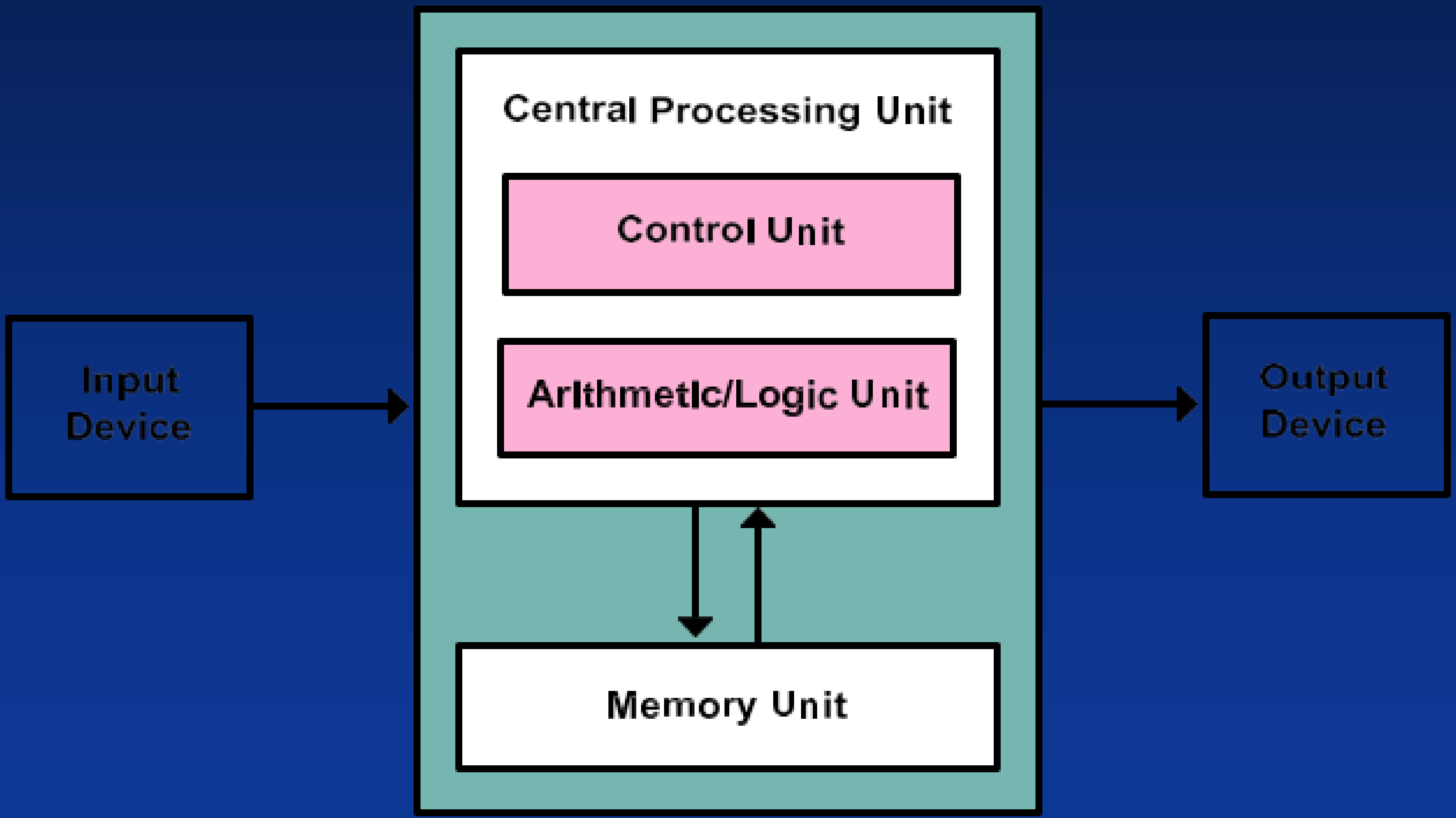
- For decades we’ve had two primary types of memories in computers: DRAM and Hard Disk Drive (HDD)
- DRAM was fast and volatile and HDDs were slower, but nonvolatile (aka persistent)
- Data moves from the HDD to DRAM over a bus where it is the fed to the processor
- The processor writes the result in DRAM and then it is stored back to disk to remain for future use
- HDD is 100,000 times slower than DRAM (!)

The Near Past:
2D Hybrid Persistent Memories in Server Architectures

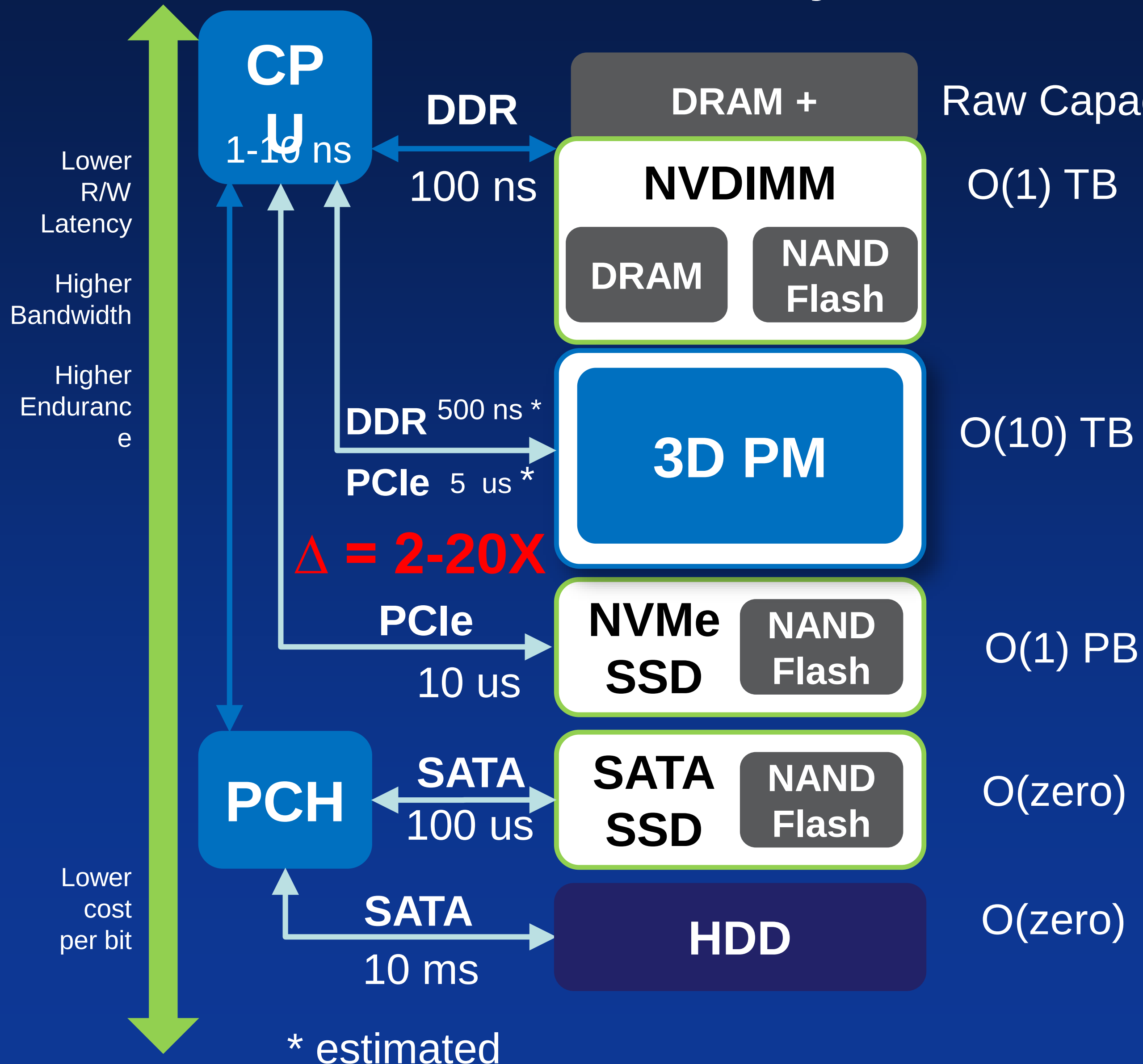


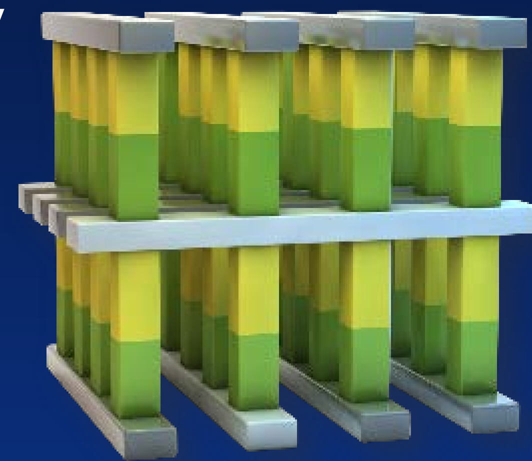
- System performance increased as the speed of both the interface and the memory accesses improved
- NAND Flash considerably improved the nonvolatile response time
- SATA and PCIe made further optimization to the storage interface
- NVDIMM provides super-capacitor-backed DRAM, operating at DRAM speeds and retains data when power is removed (-N, -P)

The Classic
Von Neumann Machine



The Present:
3D Persistent Memory in Server Architectures





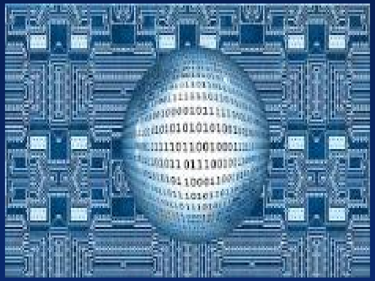
- PM technologies provide the benefit "in the middle"
- It is considerably lower latency than NAND Flash
- Performance can be realized on PCIe or DDR buses
- Lower cost per bit than DRAM while being considerably more dense

Persistent Memory (PM) Characteristics

- Byte addressable from programmer's point of view
- Provides Load/Store access
- Has Memory-like performance
- Supports DMA including RDMA
- Not prone to unexpected tail latencies associated with demand paging or page caching
- Extremely useful in distributed architectures
  - Much less time required to save state, hold locks, etc.
  - Reduces time spent in periods of mutex/critical sections

Persistent Memory Applications

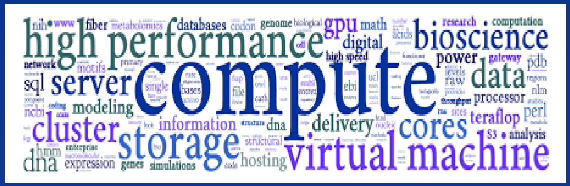
- Distributed Architectures: state persistence, elimination of volatile memory characteristics and pitfalls



- In Memory Database: Journaling, reduced recovery time, Ex-large tables

- Traditional Database: Log acceleration via write combining and caching

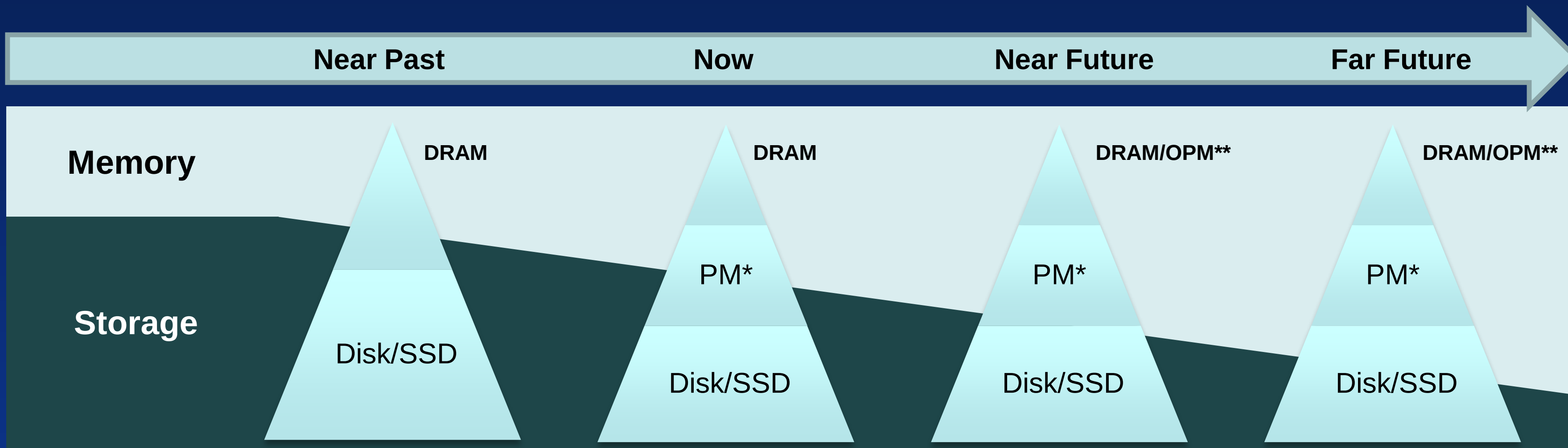
- Enterprise Storage: Tiering, caching, write buffering and meta data storage



- Virtualization: Higher VM consolidation with greater memory density

Memory & Storage Convergence

- Volatile and non-volatile technologies are continuing to converge



*PM = Persistent Memory

**OPM = On-Package Memory

New and Emerging Memory Technologies

- HMC
- HBM
- RRAM
- 3DXPoint™ Memory
- MRAM
- PCM
- Low Latency NAND
- Managed DRAM

Source: Gen-Z Consortium 2016

SNIA NVM Programming Model

- Version 1.2 approved by SNIA in June 2017
  - http://www.snia.org/tech_activities/standards/curr_standards/npm
- Expose new block and file features to applications
  - Atomicity capability and granularity
  - Thin provisioning management
- Use of memory mapped files for persistent memory
  - Existing abstraction that can act as a bridge
  - Limits the scope of application re-invention
  - Open source implementations available
- Programming Model, not API
  - Described in terms of attributes, actions and use cases
  - Implementations map actions and attributes to API's


ELECTRIC LIGHT DID NOT COME FROM THE CONTINUOUS
IMPROVEMENT OF CANDLES

Storage Systems - Weiji

危機

Traditional

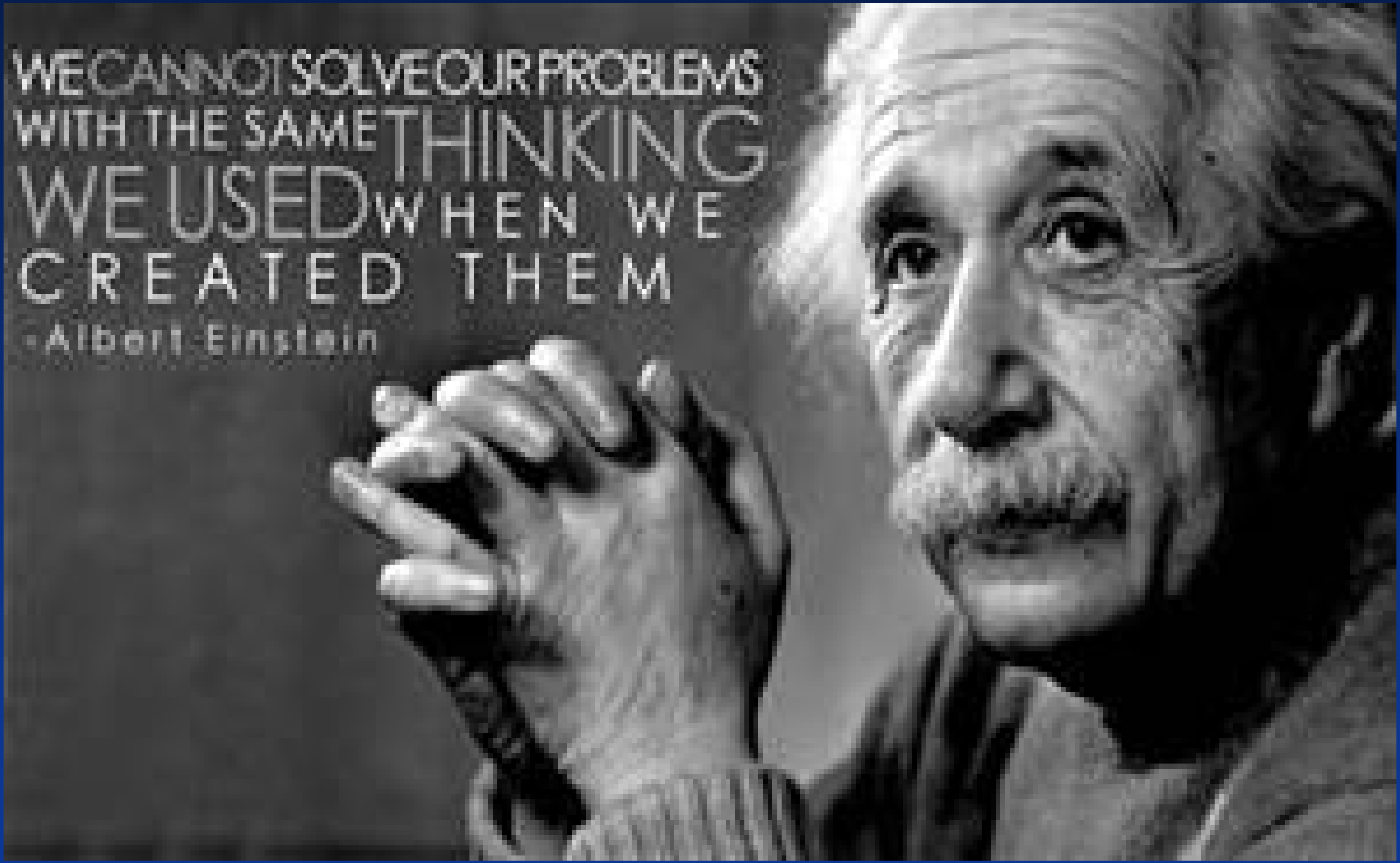
危机

Simplified

Popular Meaning:
“Dangerous Opportunity”

Accurate Meaning:
Crisis

Said in 1946


WE CANNOT SOLVE OUR PROBLEMS
WITH THE SAME THINKING
WE USED WHEN WE
CREATED THEM
-Albert Einstein

Yes we are At A Crisis in Storage Systems

- Hopefully this is not news to you all
- Question of the day – how could we (re-)design future storage systems?
  - in particular for HPC, but not solely for HPC?
- Answer – decompose it – <u>two roles</u>
  - First – rapidly pull/push data to/from memory as needed for jobs – “feed the beast”
  - Second – store (persist) gigantic datasets over the long term – “persist the bits”

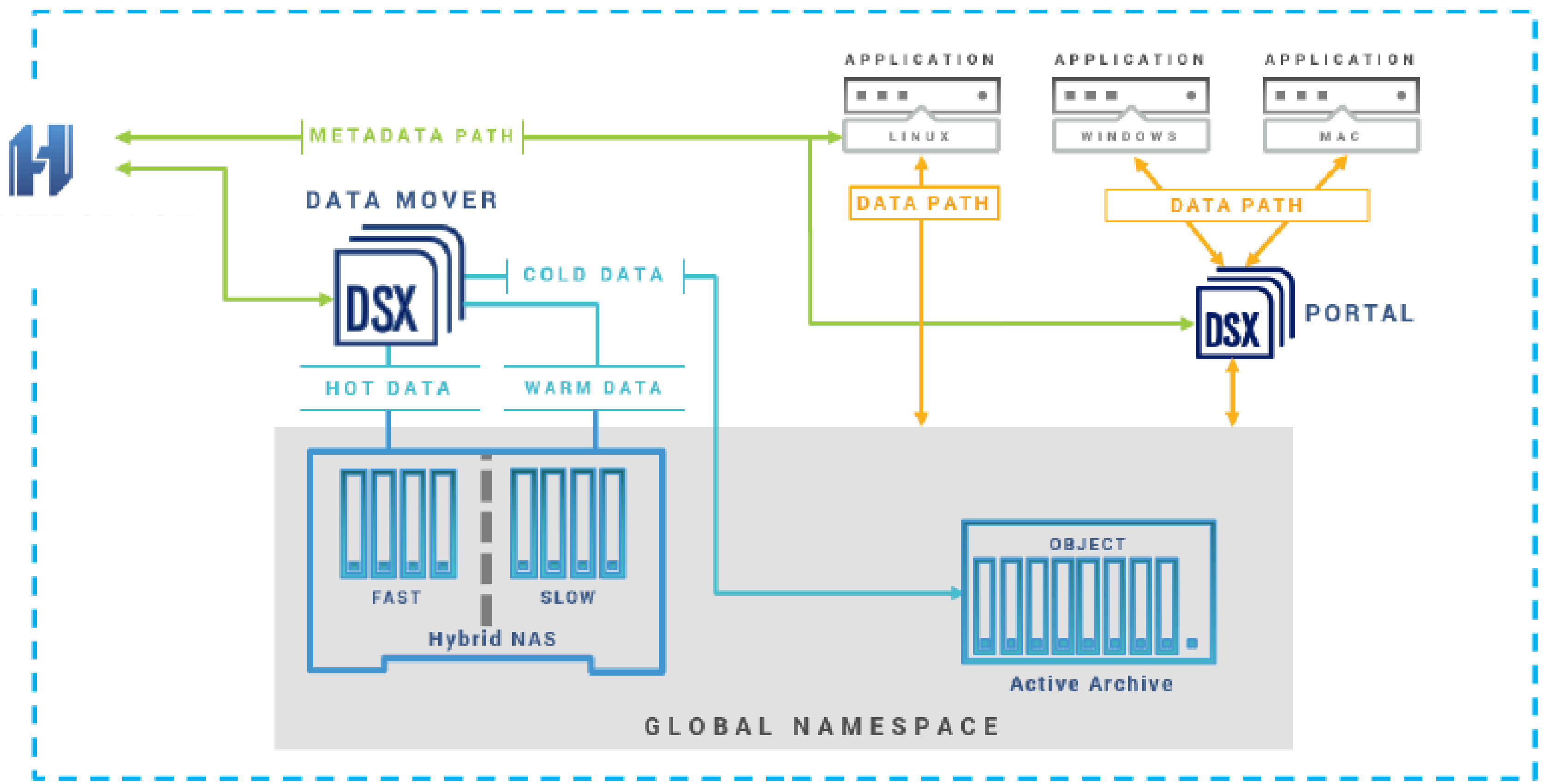
One System – Two Roles

- We must design radically different subsystems for those two roles
- But But But “more tiers, more tears”
- True – but you can’t have it both ways
  - or can you?
- The answer is <u>yes</u>
  - But not the way you might think

One Namespace to Rule Them All

- Future storage systems must have a *universal namespace (database)* for <u>all</u> files & objects
  - Yes, objects
- This means breaking <u>all</u> the metadata away from <u>all</u> the data
  - Think about how current filesystems work (yuck)
- User only interacts with the namespace
  - User sets objectives (intents) for data; system guarantees
  - Extremely rich metadata (tags, names, labels, etc.)
- User never directly moves data
  - No more cp, scp, cpio, ftp, tar, rcp, rsync, etc. (yay!)

Something Like This



Let’s do some Arithmetic

- Consider the lofty exaflop
  - 1,000,000,000,000,000,000 flop/sec
  - That’s a lotta flops
- A = B * C requires 3 memory locations
  - Let’s say 32-bit operands
- That’s 3*4 (bytes) = 12 bytes/flop
  - 12,000,000,000,000,000,000 bytes of memory (12 EB)
- That’s 2 loads and a store
  - That’s handy because it’s just about what one core can do today
  - Sad but true
- Goal – sustain that exaflop

Let’s do some Arithmetic

- Consider the lowly storage system
  - In conjunction with the lofty sustained exaflop
  - That’s a lotta data
- Must have at least 8 EB/sec burst read
  - To read operands into memory for said exaflop
- Must have at least 4 EB/sec burst write
  - To write results from memory for said exaflop
- All righty then

Cut to The Chase

- Future large storage systems should optimize for sequential I/O - only
  - Death to random I/O
- A future storage system looks like:
  - Node-local persistent memory
    - O(10) TB per node
    - Managed as memory (yup, memory)
    - Fastest/smallest area of persistence
    - Supports O(100) GB/sec transfers

Cut to The Chase

- A future storage system looks like:
  - <u>Node-local NAND-based block storage</u>
    - O(100) TB per node
    - Managed as storage (LBA, length)
    - Uses local NVMe transport (bus lanes)
    - Devices may contain compute capability
      - Computational-defined storage (SNIA)
  - Yes, node-local storage as part of the storage system. Get over it.
  - The all-external storage play is meh
    - You did say HPC, right?

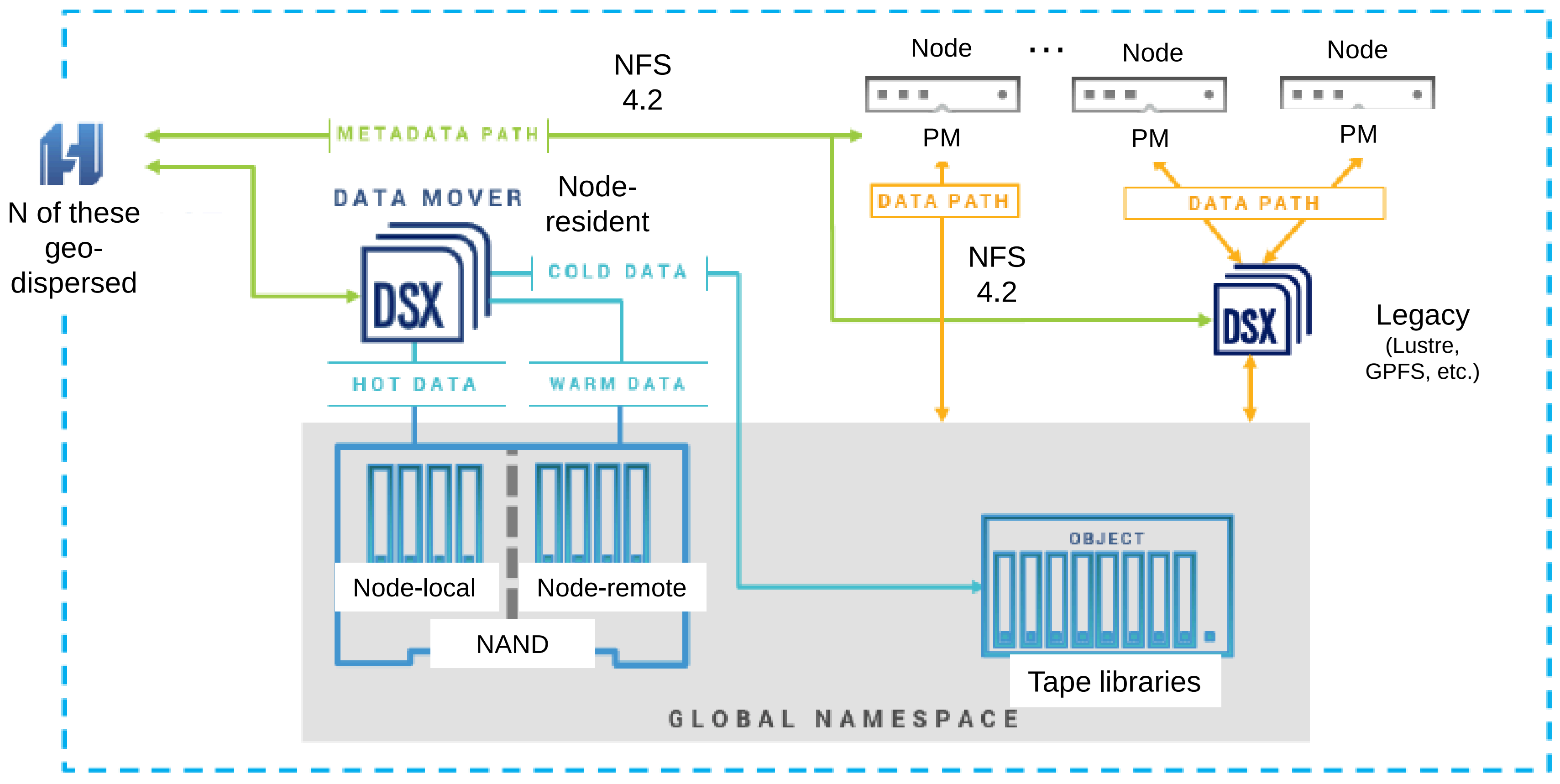
Cut to The Chase

- A future storage system looks like:
  - Node-remote NAND-based block storage
    - O(1) PB per node
    - Managed as storage (LBA, length)
    - Uses NVMe-oF transport (network)
    - Supports O(?) TB/sec transfers (see below)
  - Performance is fabric-dependent
    - Today – O(100) Gb/s Ethernet or IB
    - Tomorrow – O(1) Tb/s direct torus
    - Future – each block device is in torus (6D)

Cut to The Chase

- A future storage system looks like:
  - Node-remote BaFe tape storage
    - O(10) EB per system
    - Managed as object storage (metadata map)
    - Uses NVMe-oF transport (network)
    - Supports O(?) TB/sec transfers (see below)
    - Future – SrFe-based tape media
  - Performance is fabric-dependent
    - Today – O(100) MB/s per drive (e.g. 750)
    - Tomorrow – O(1) GB/s per drive

Something Like This



Future Storage Systems
A Dangerous Opportunity

Past, Present, Future

Rob Peglar
President
Advanced Computation and Storage LLC
rob@advanced-c-s.com

You did say HPC, right?

- Assume a socket does 500 GB/s
  - Memory bandwidth (to/from RDIMM-based DRAM)
  - HBM2 will be used too but as a smaller/faster memory tier
- Must have 12 EB/s overall flow
  - 8 EB/s ingress into memory, 4 EB/s egress from memory
  - So that’s 24 million socket flows
  - 24 million sockets is a lotta sockets
- Assuming 2,500 racks of fast storage
  - Each rack services ~10,000 sockets
  - Each rack must therefore provide 10,000*500 GB/s = 5 PB/sec
  - Using 40 GB/sec Ethernet that’s 125,000 links/rack
  - Whoops

You did say HPC, right?

- Long-term storage is (wait for it)
  - Tape
- Should be O(100) EB in total capacity
  - Very little of it would be in use at any one time
  - Specify objectives in metadata (namespace) to control residence

Conclusion

- Storage is not the problem
  - Network(s) are the problem
  - As usual – moving the bits is a near-death experience
- Direct Torus is the (near) future answer
  - Sound familiar? Consider compute design
  - Photonic transport(s)
- Stage One – systems using direct torus
  - Each rack services ~10,000 sockets
  - Each rack must therefore provide 10,000*500 GB/s = 5 PB/sec
  - Using 400 Gb/sec Ethernet that's 125,000 links/rack
  - Whoops – gotta have multiple 1 Tb/sec per NAND-based device and at least 4 1Tb/sec link per socket