DASYA
www.dasya.dk

DAPHNE
https://daphne-eu.github.io/

IT UNIVERSITY OF COPENHAGEN
www.itu.dk

data-intensive systems in the microsecond era

Pınar Tözün

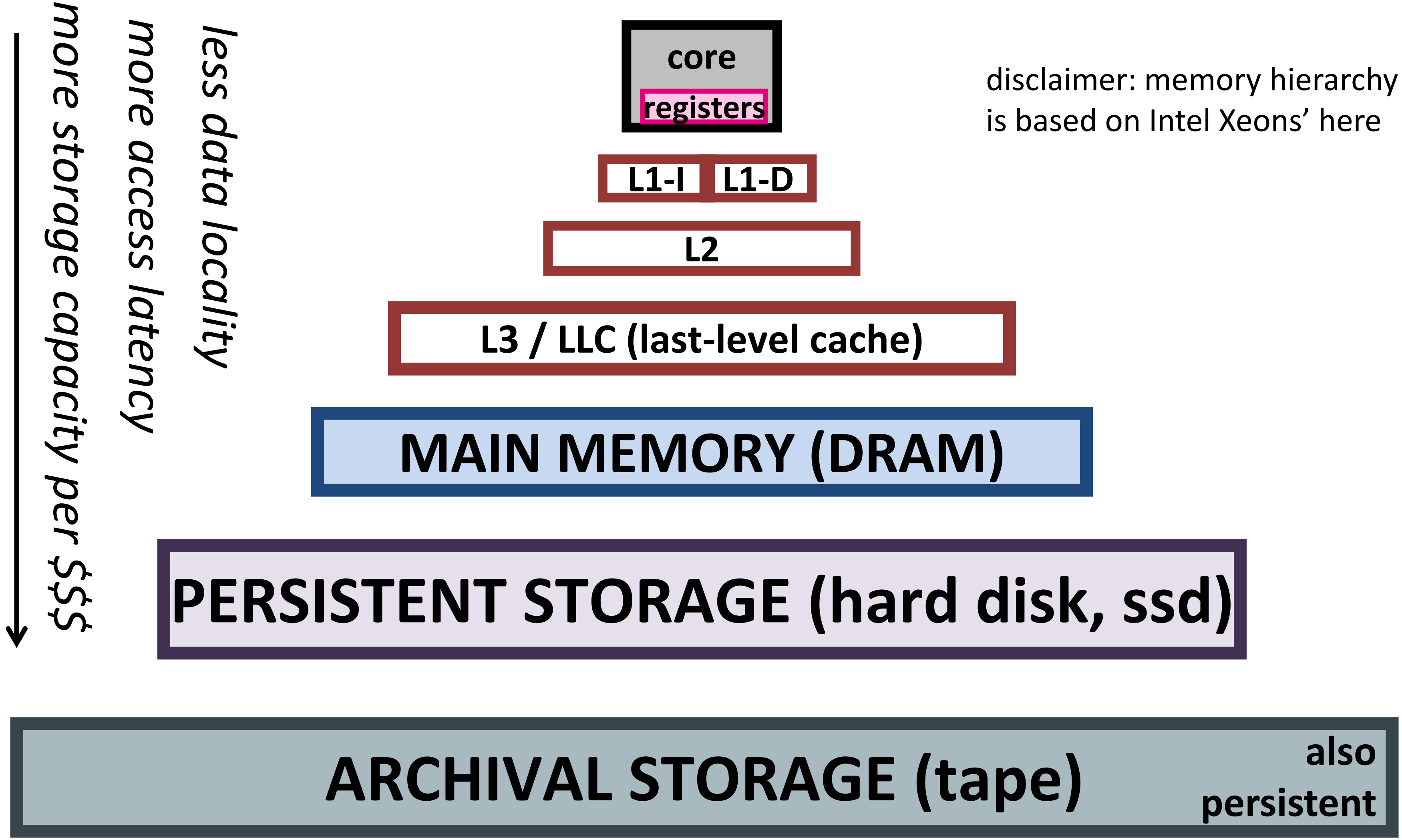
pito@itu.dk, www.pinartozun.com

work done in collaboration
with Philippe Bonnet @ ITU

Dutch Seminar on Data Systems Design

16/07/2021

(typical) storage hierarchy



more storage capacity per $$$

more access latency

less data locality

core

registers

L1-I

L1-D

L2

L3 / LLC (last-level cache)

MAIN MEMORY (DRAM)

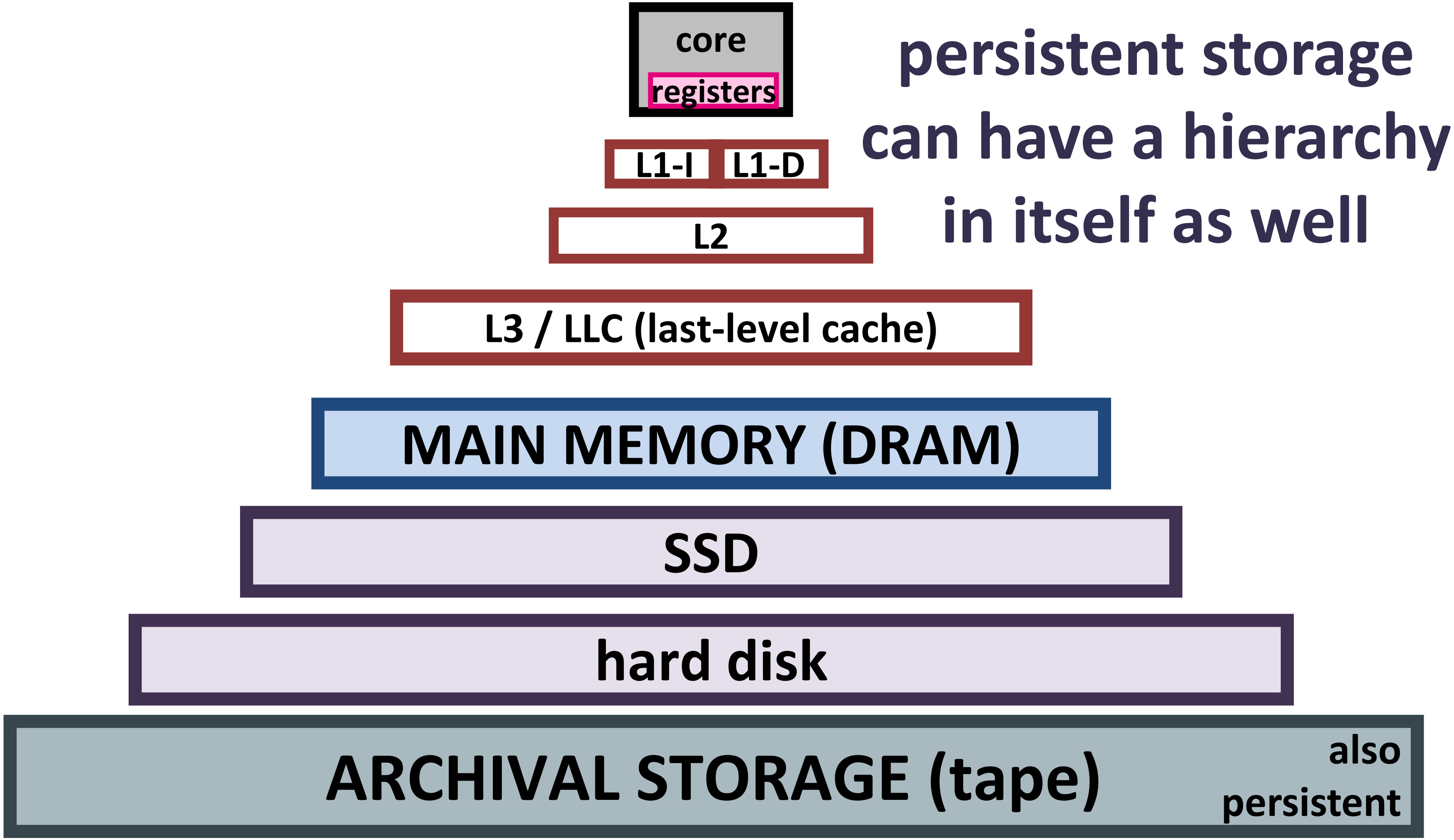
PERSISTENT STORAGE (hard disk, ssd)

ARCHIVAL STORAGE (tape)

also persistent

disclaimer: memory hierarchy is based on Intel Xeons’ here

(typical) storage hierarchy



core

registers

L1-I

L1-D

L2

L3 / LLC (last-level cache)

MAIN MEMORY (DRAM)

SSD

hard disk

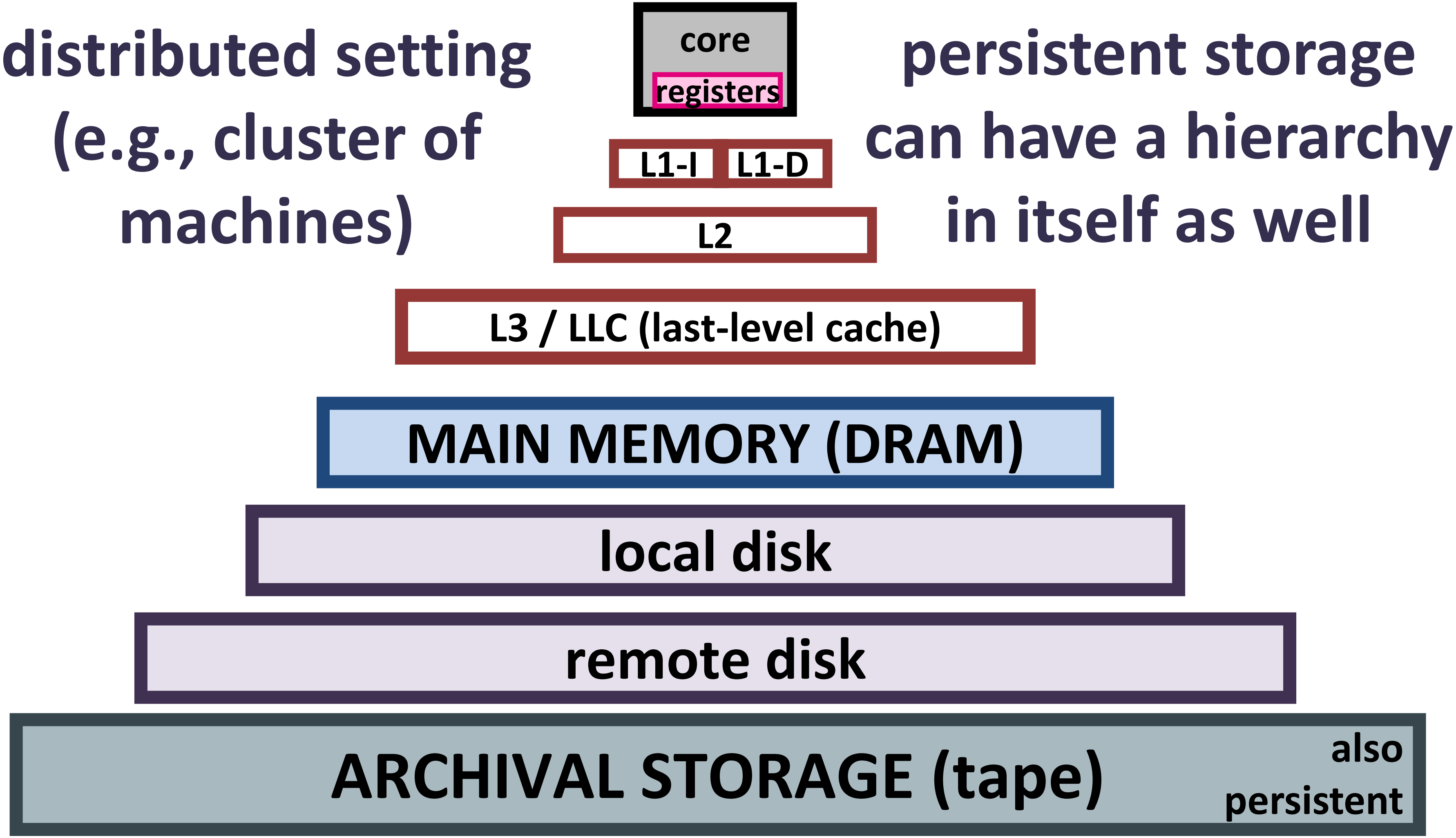
ARCHIVAL STORAGE (tape)

also persistent

persistent storage can have a hierarchy in itself as well

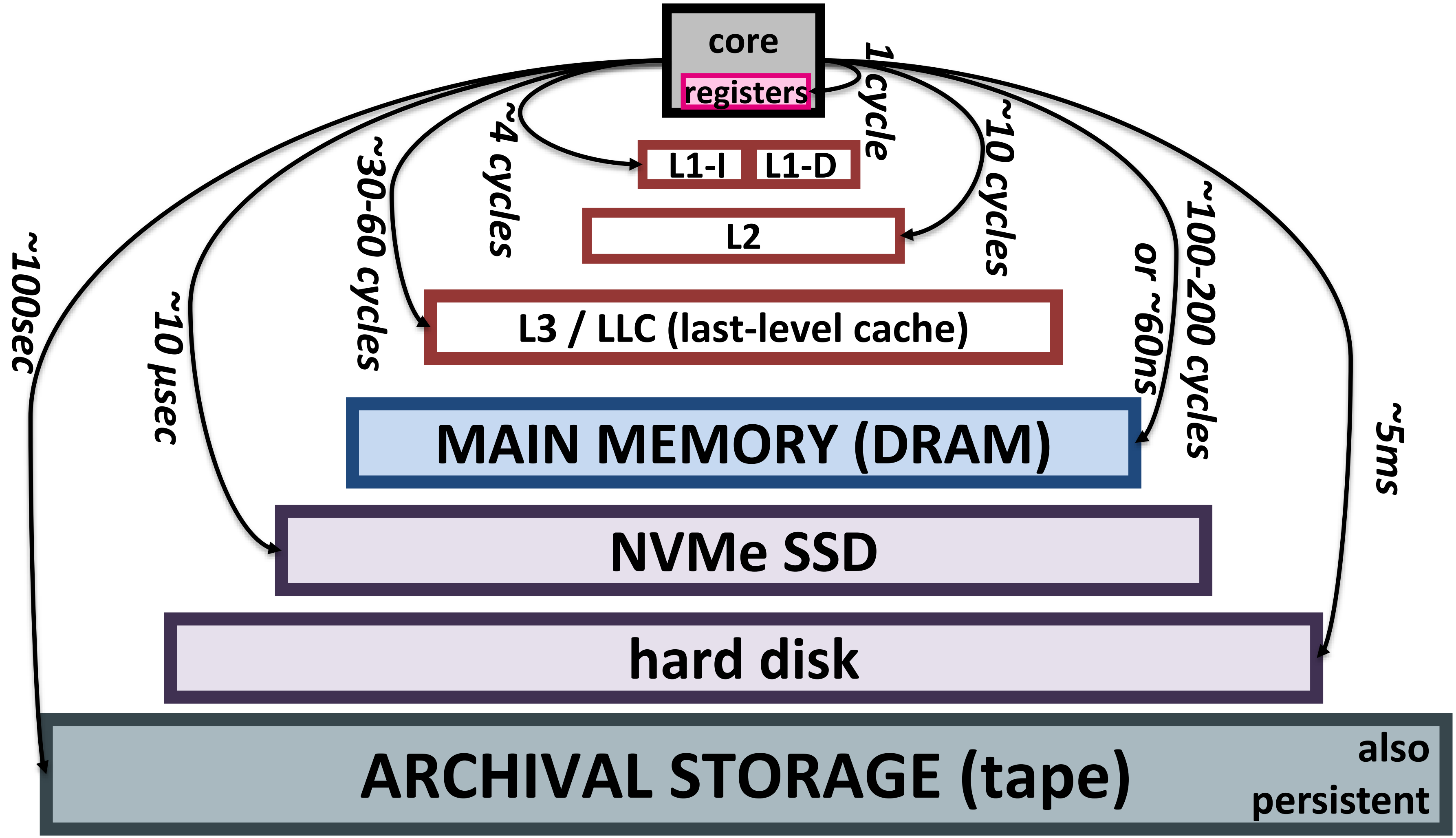
(typical) storage hierarchy

distributed setting (e.g., cluster of machines)



persistent storage can have a hierarchy in itself as well

latency to fetch data



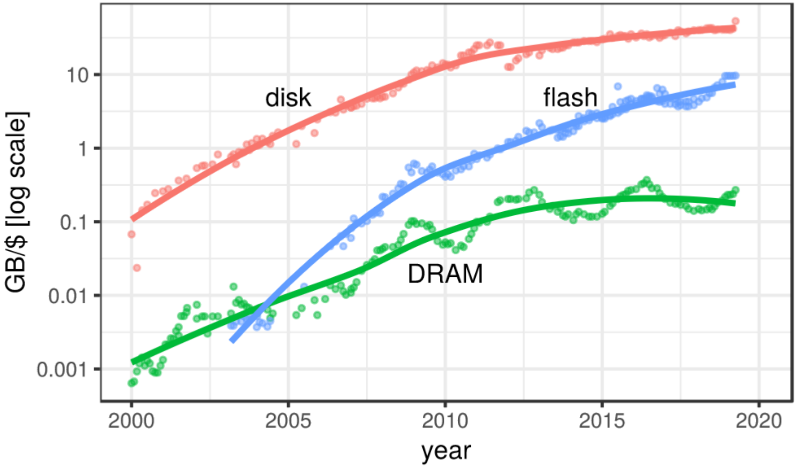
why focus on SSDs in this talk?

➔ except for SSDs, each layer stayed almost stable the last decade in terms of latency

- improvements on SSD internals
- from SAS/SATA to PCIe
- linux block IO improvements e.g., multiqueue

➔ improved price/capacity

source: Haas et al., CIDR 2020



➔led to several SSD-optimized data systems

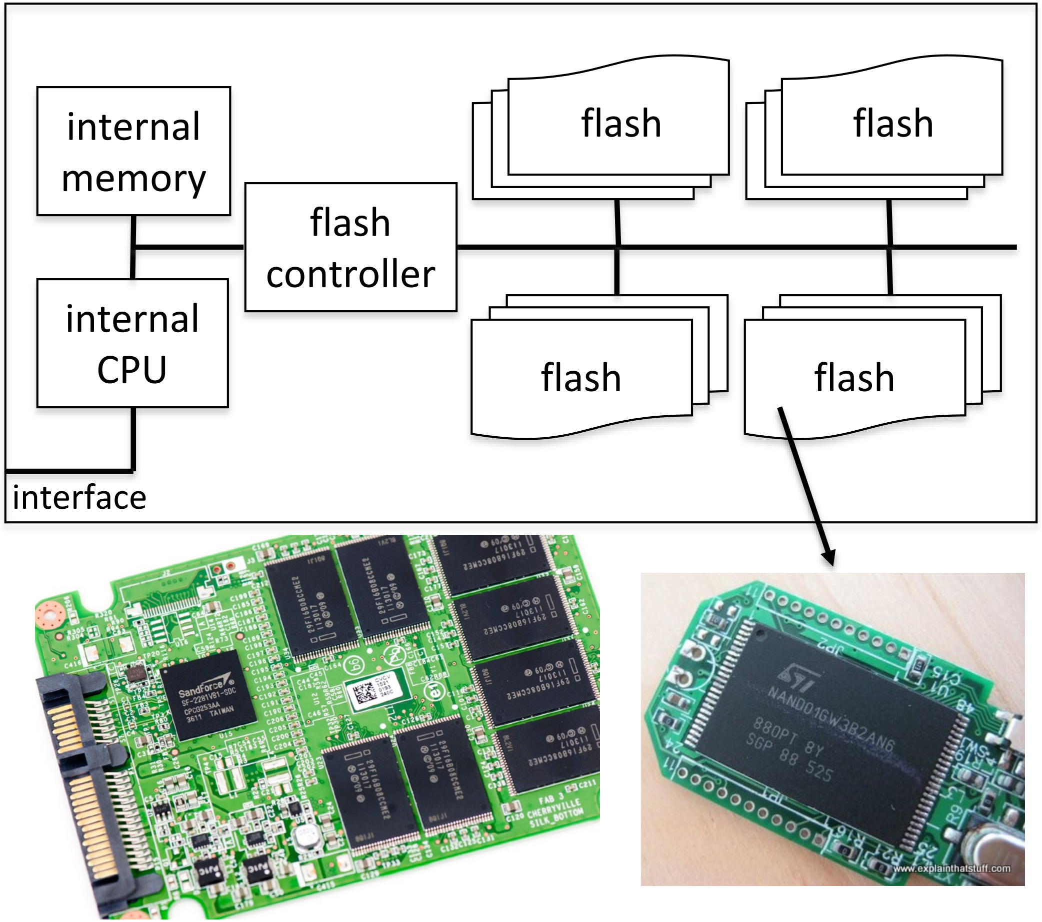
- RocksDB, BwTree, LeanStore, Umbra …

increasing shift from pure in-memory-optimized to SSD-optimized data systems!

agenda

- SSD internals & state of affairs today
- emerging SSD & computational storage landscape

solid-state disk (SSD)



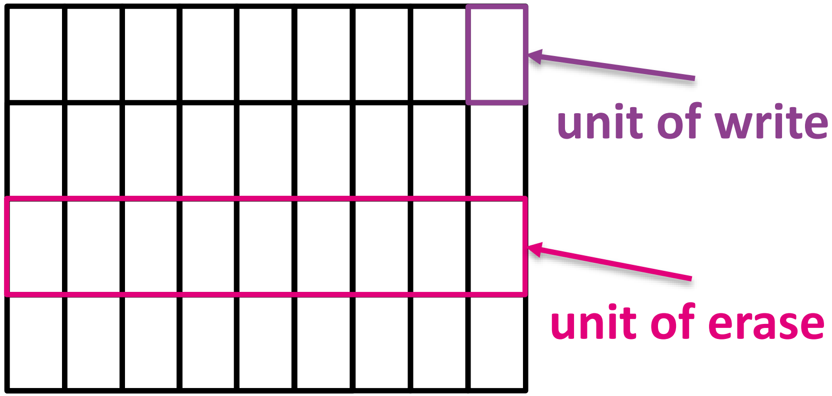
interconnected flash chips

hard disk compatible API

compared to hard disks

- efficient random access
- internal parallelism

flash chips



flash translation layer (FTL) hides the internal complexities of flash chips from end-users

but knowing them can lead to smarter software design

cannot override a unit before erasing it first

garbage collection – for not used blocks so we can rewrite them

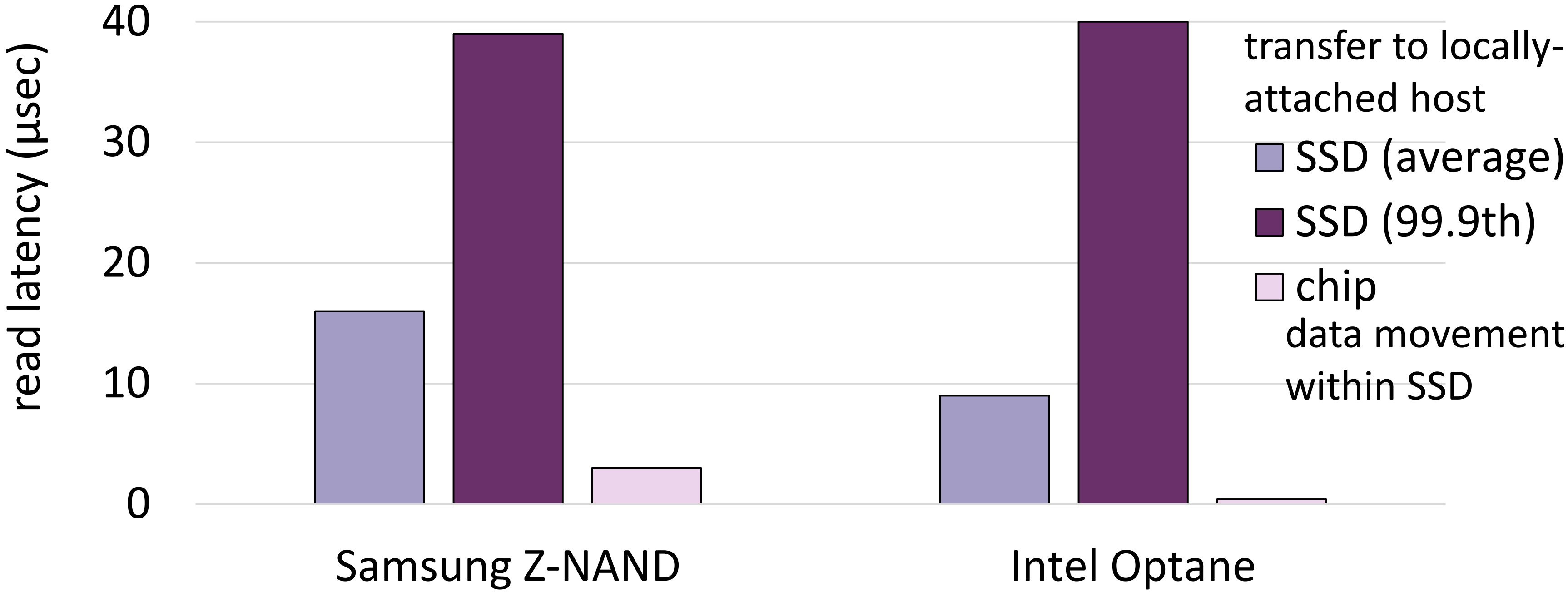
write amplification = data physically written / data logically written >= 1
writing data might cause rewrites & garbage collection

wear leveling – some cells/blocks die over time

unpredictable read/write latencies

SSDs in the μsec era

4K random read using fio - sources: [1, 2, 3]

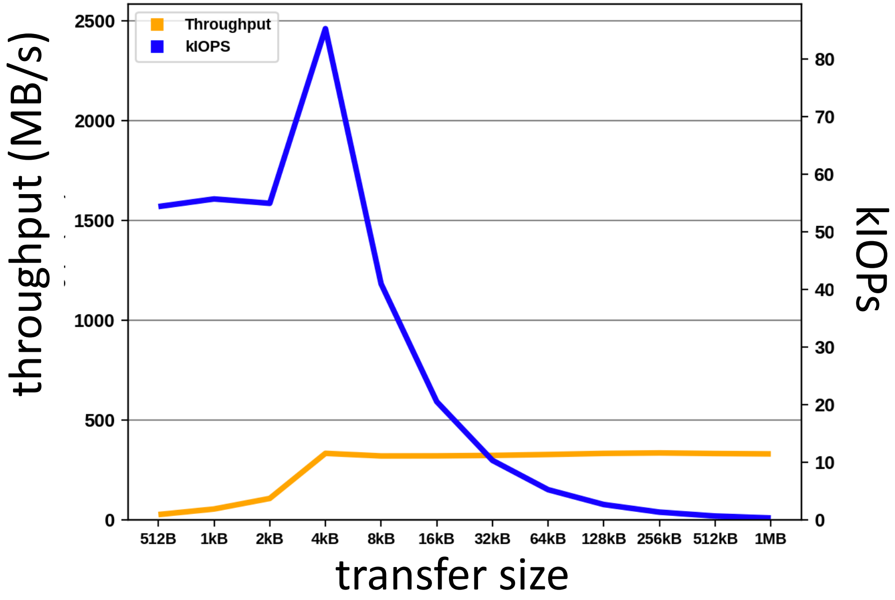


SSDs equipped with Z-NAND & Optane deliver at best 5x & 20x the read latency of the underlying storage chip, respectively.

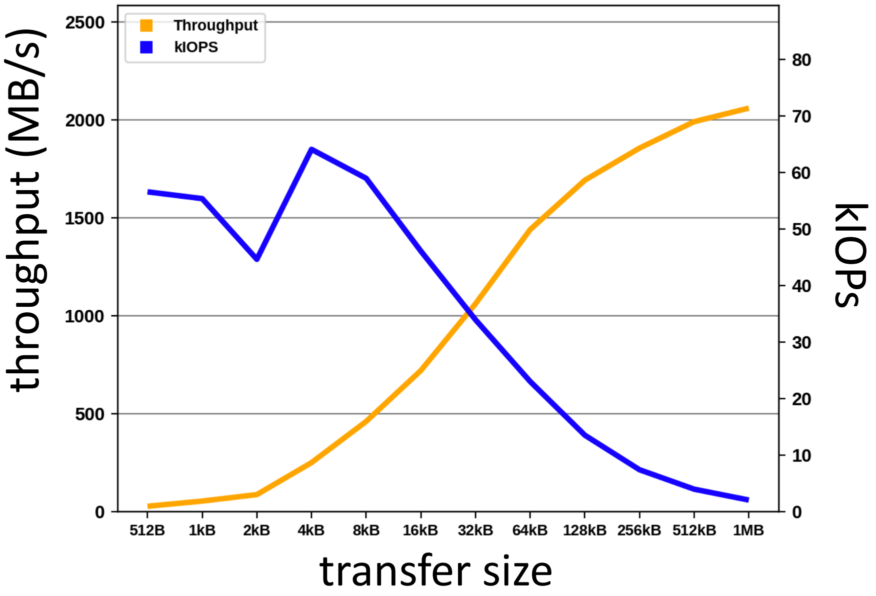
FTLs in the µsec era ..

random writes- source: AnandTech

Samsung SSD with Z-NAND

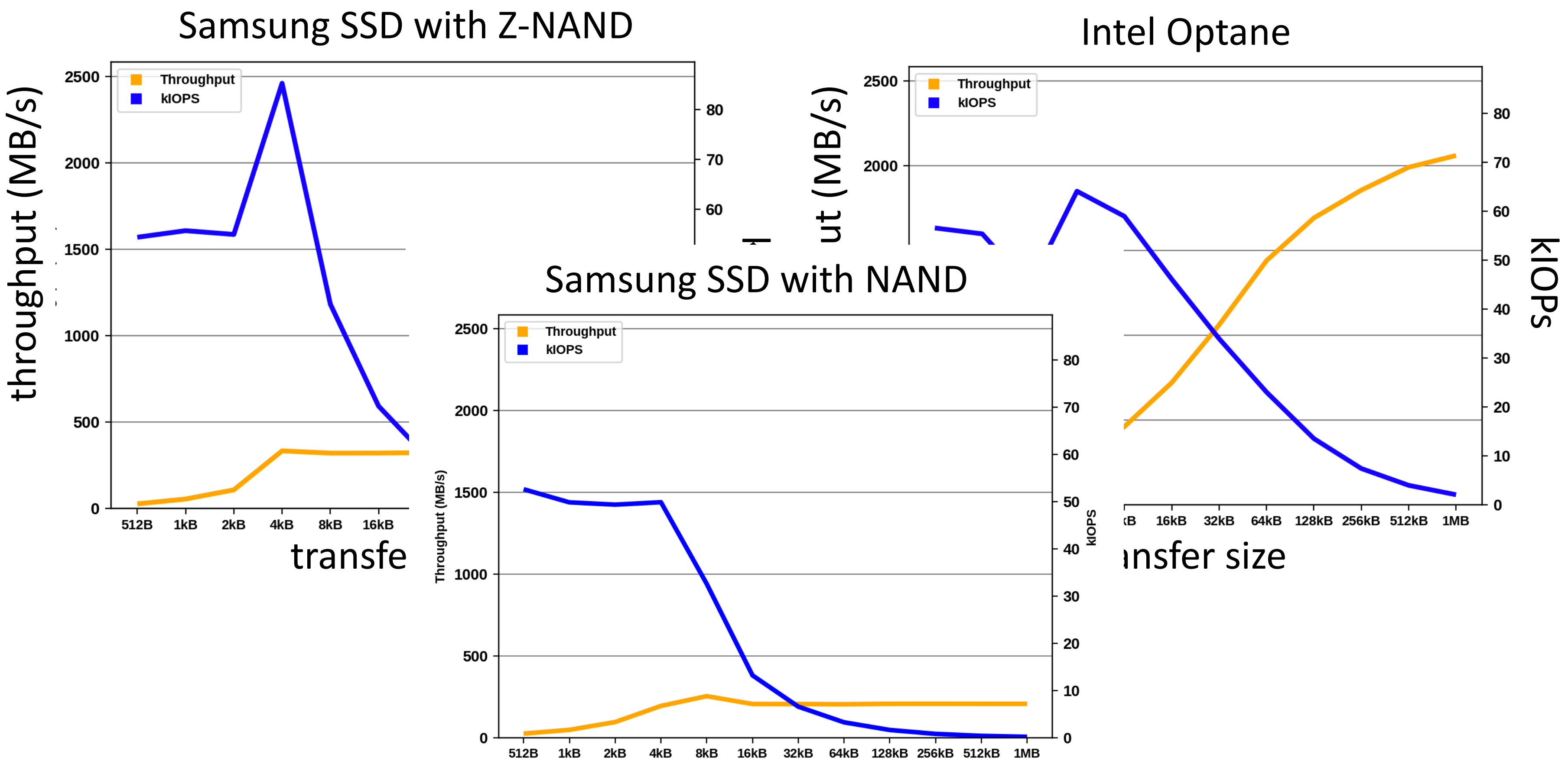


Intel Optane



FTLs in the µsec era ..

random writes- source: AnandTech



Samsung SSD with Z-NAND

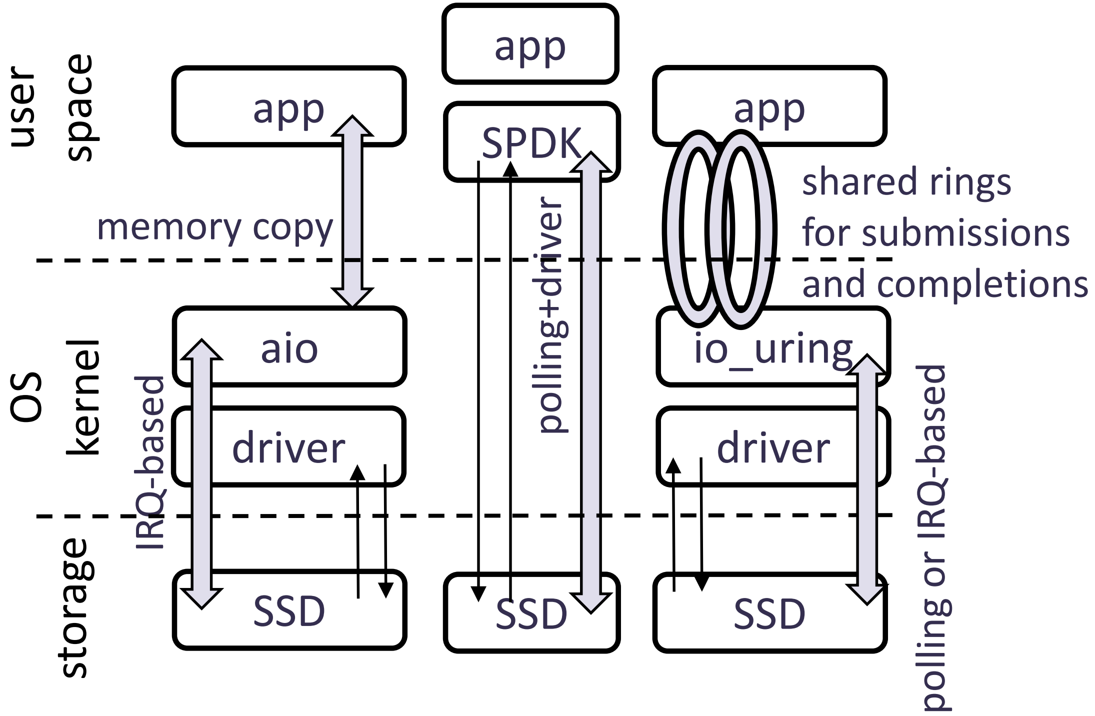
Intel Optane

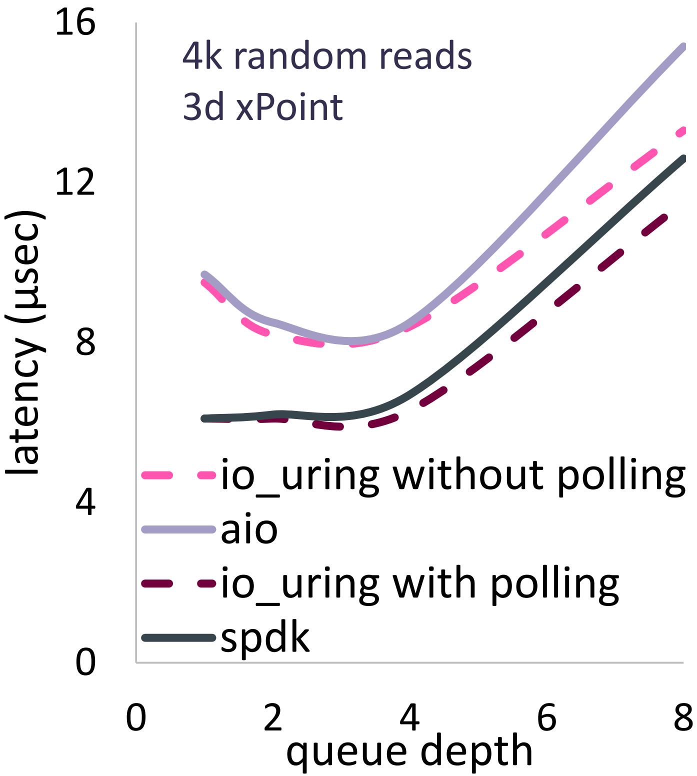
Samsung SSD with NAND

... have drastic impact on throughput!

linux IOs in the µsec era

sources: Faster IO through io_uring & Efficient I/O with io_uring & J.Axboe





separation of control & data plane in linux now

zero copy & minimized synchronization overhead

the benefits of fast storage wasted by
- data movement overheads
(from device to host & across network)
- black-box generic flash-translation layers
- multitude of software layers

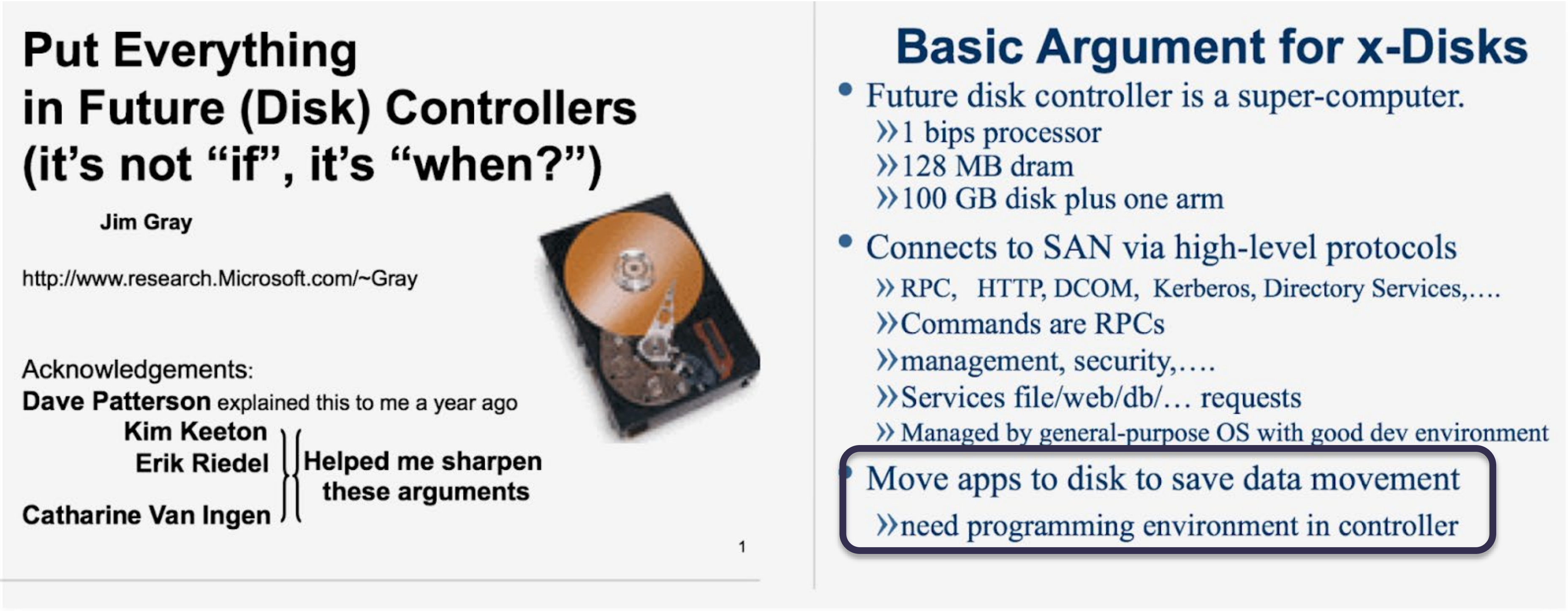
how do we prevent these?

agenda

- SSD internals & state of affairs today
- emerging SSD & computational storage landscape

computational storage

back when I was a kid



Put Everything in Future (Disk) Controllers (it's not "if", it's "when?")

Jim Gray

http://www.research.Microsoft.com/~Gray

Acknowledgements:
Dave Patterson explained this to me a year ago
Kim Keeton
Erik Riedel
Catharine Van Ingen
Helped me sharpen these arguments

1

Basic Argument for x-Disks

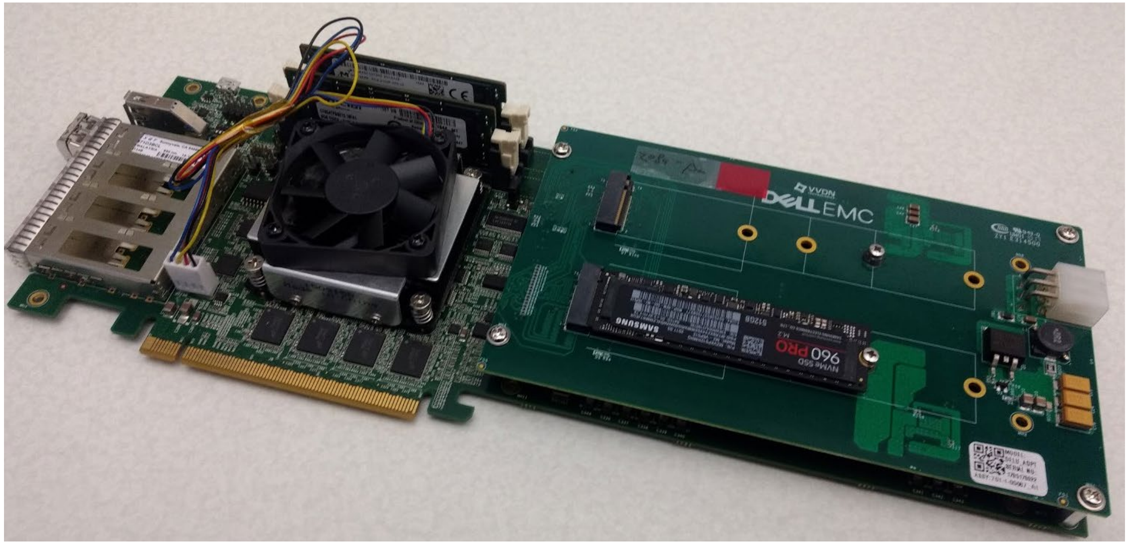
- Future disk controller is a super-computer.
  - 1 bips processor
  - 128 MB dram
  - 100 GB disk plus one arm
- Connects to SAN via high-level protocols
  - RPC, HTTP, DCOM, Kerberos, Directory Services,….
  - Commands are RPCs
  - management, security,….
  - Services file/web/db/… requests
  - Managed by general-purpose OS with good dev environment
- Move apps to disk to save data movement
  - need programming environment in controller

Jim Gray, NASD Talk, 6/8/98
http://jimgray.azurewebsites.net/jimgraytalks.htm

= computation on the IO path

computational storage

back when I joined ITU



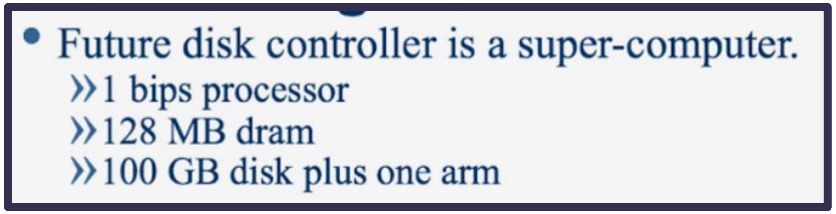
8-core ARMv8 processor

32GB DRAM

2TB+ of NVM via M.2 slots

4x 10Gb Ethernet

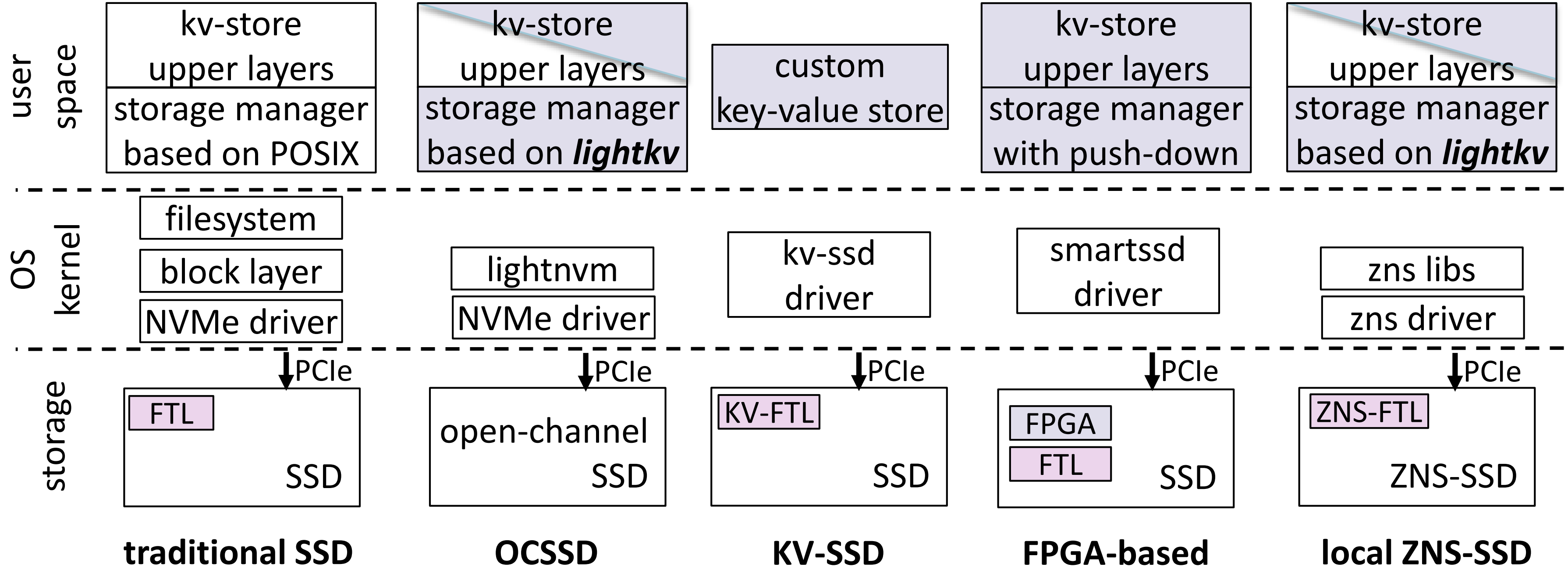
Dragon Fire Card (DFC)
https://github.com/DFC-OpenSource/



- Future disk controller is a super-computer.
  - »1 bips processor
  - »128 MB dram
  - »100 GB disk plus one arm

SSD landscape – local

kv-store needs to change when you start app-specific storage management & pushing functionality down!



black-box

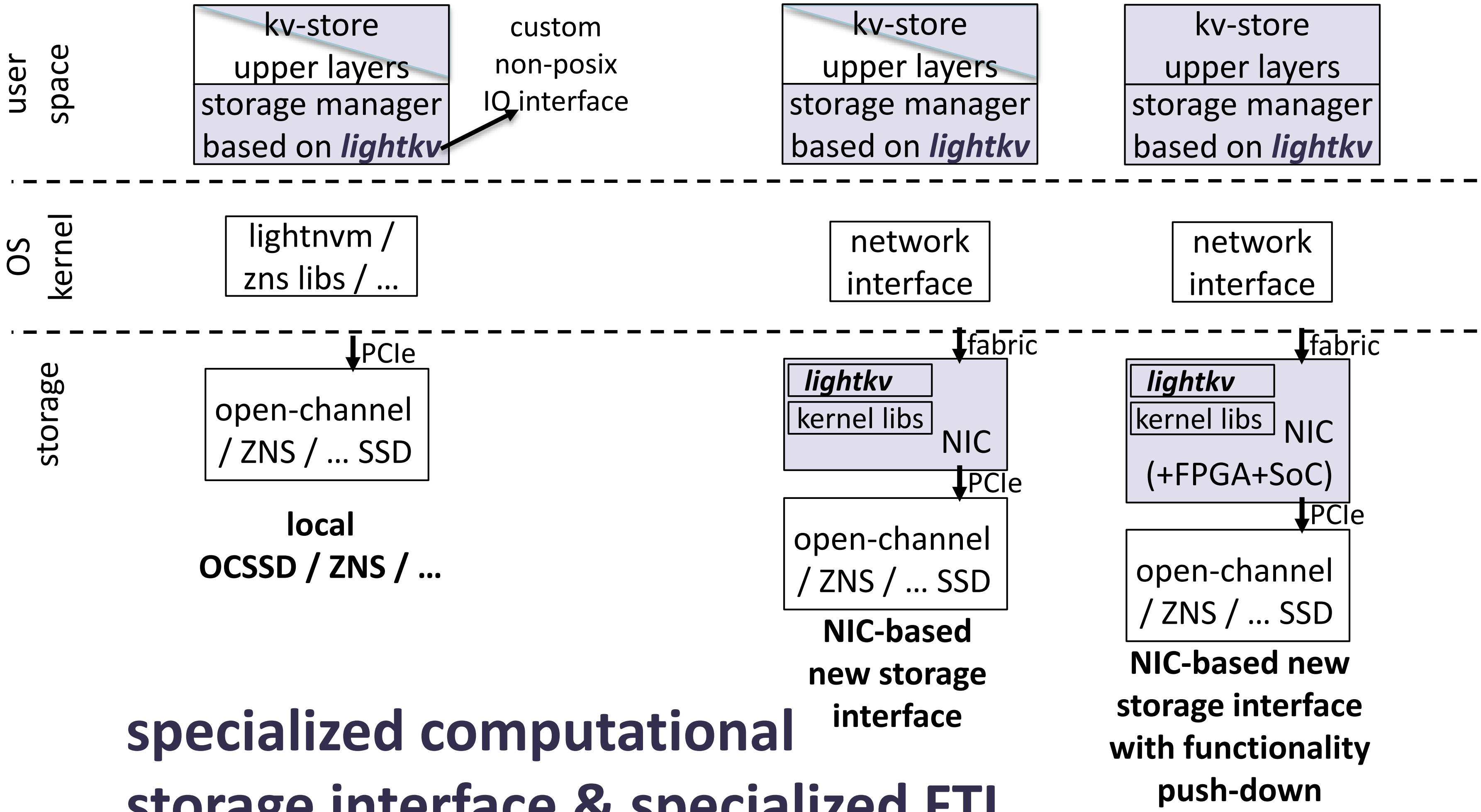
app-specific, no comp. storage

static custom

generic FTL

similar to OCSSD, less customizable

SSD landscape – disaggregated



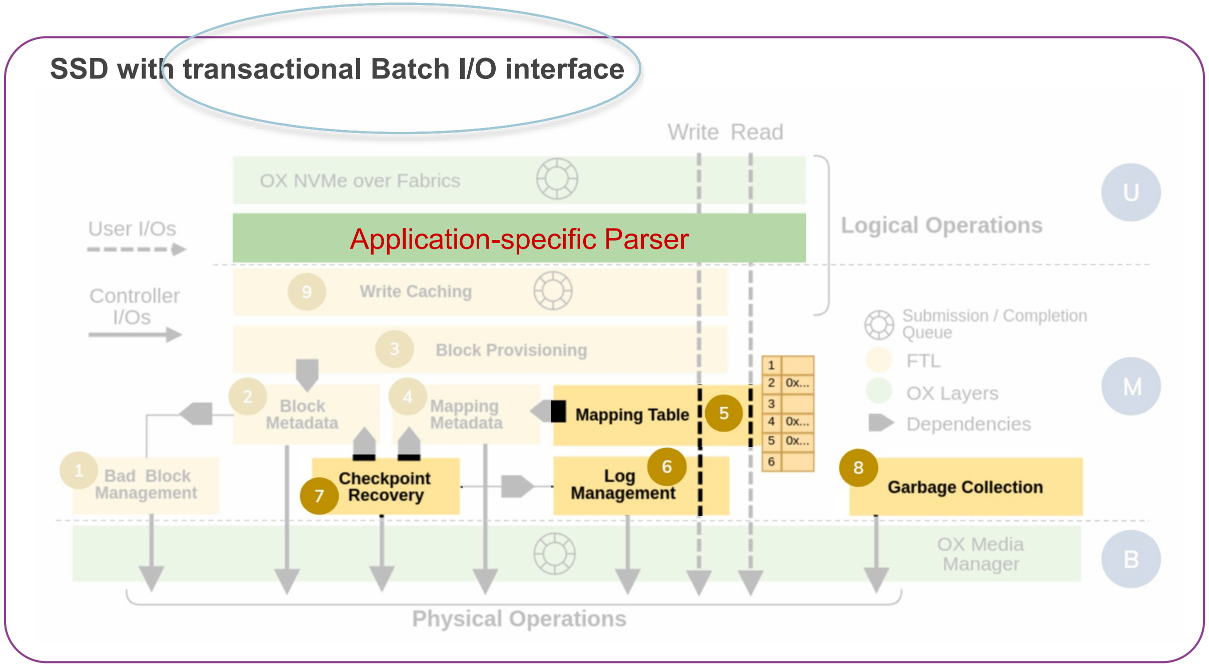
specialized computational storage interface & specialized FTL

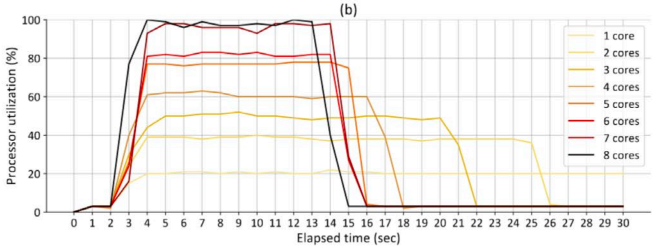
programming SSDs

The VLDB Journal 2021

	Host-Based LS	SSD-Based LS
Host	LPID → SSDAdr	LPID
SSD	SSDAdr → FlashAdr	LPID $\xrightarrow{\text{MAP}}$ FlashAdr

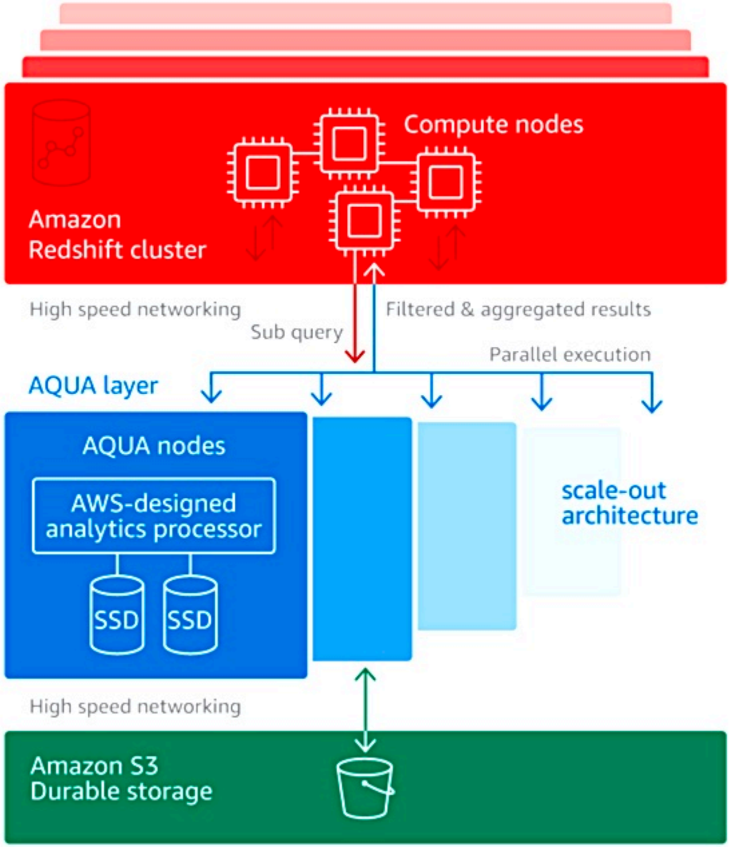
- Philippe Bonnet and Ivan Picoli in collaboration with MSR
- programming a storage controller using OX framework on an OCSSD





BwTree-specific FTL!

AWS AQUA

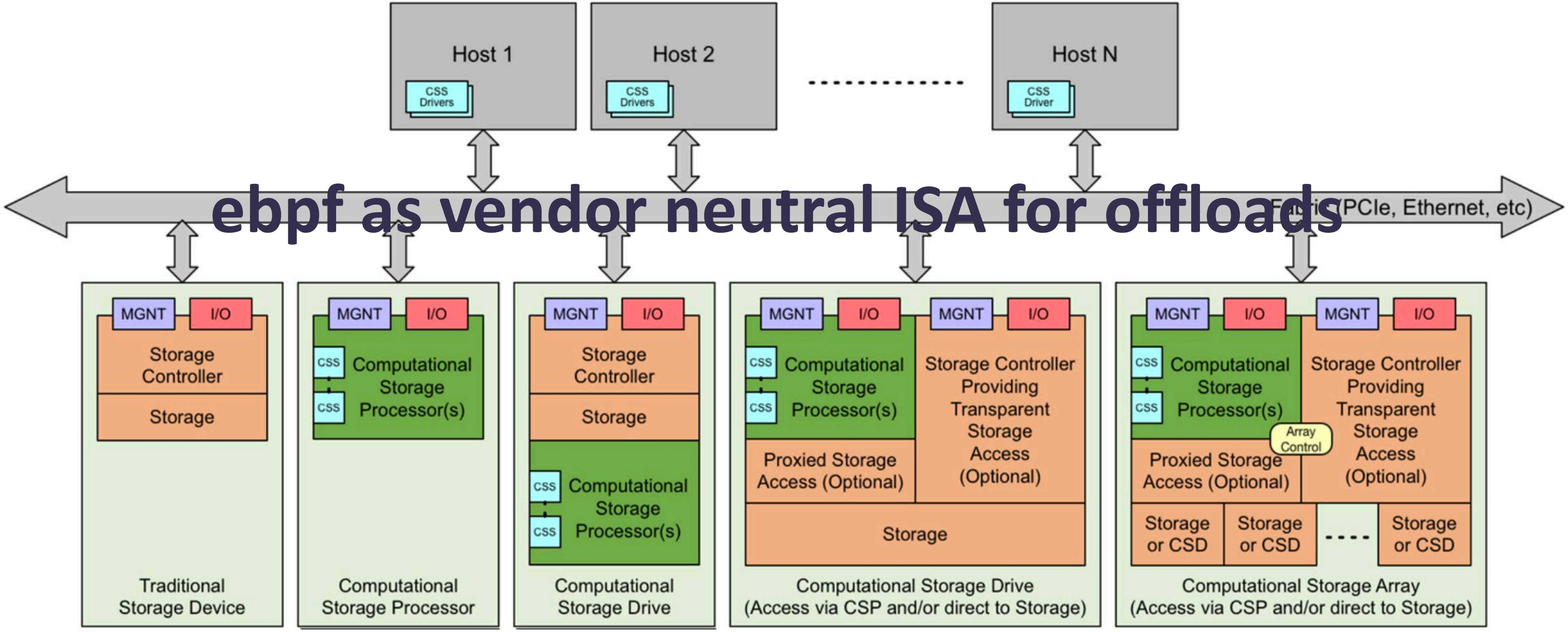


Advanced Query Accelerator

- near-data processing from AWS (also called computational storage)
- announced in 2019
  (see video if interested)
- they are using SSDs and FPGAs at the AQUA layer
- goal: to reduce network traffic by reducing data movement

envisioned architectures

SNIA. Computational Storage Architecture and Programming Model. V0.5, Rev 1. Aug 2020


Host 1
CSS Drivers
Host 2
CSS Drivers
Host N
CSS Driver
Fabric (PCIe, Ethernet, etc)
MGNT
I/O
Storage Controller
Storage
Traditional Storage Device
MGNT
I/O
CSS
CSS
Computational Storage Processor(s)
Computational Storage Processor
MGNT
I/O
Storage Controller
Storage
CSS
CSS
Computational Storage Processor(s)
Computational Storage Drive
MGNT
I/O
CSS
CSS
Computational Storage Processor(s)
Proxied Storage Access (Optional)
MGNT
I/O
Storage Controller Providing Transparent Storage Access (Optional)
Storage
Computational Storage Drive (Access via CSP and/or direct to Storage)
MGNT
I/O
CSS
CSS
Computational Storage Processor(s)
Array Control
Proxied Storage Access (Optional)
MGNT
I/O
Storage Controller Providing Transparent Storage Access (Optional)
Storage or CSD
Storage or CSD
Storage or CSD
Computational Storage Array (Access via CSP and/or direct to Storage)

ebpf as vendor neutral ISA for offloads

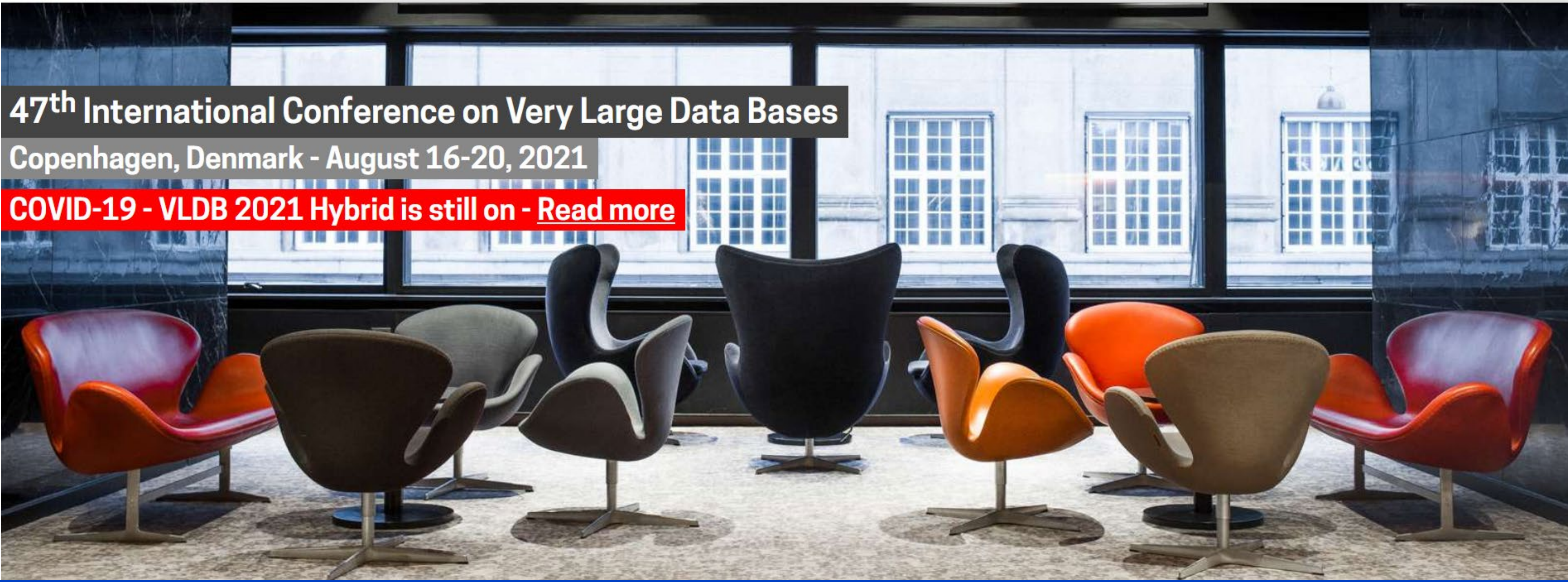
being standardized in NVMe (expected in 2022)

conclusion

- data management community increasingly shifts from pure in-memory optimized to SSD-optimized
- NVMe SSDs aren’t a uniform class of devices
- expanding range of standardized storage interfaces (block, ZNS, KV, OCSSD)
  ➔ **the storage interface is a design choice**
- computational storage enables the definition of even more specialized storage interfaces

need for co-design of storage engine – FTL – SSD

VLDB 2021 looking for student volunteers, contact us if interested!



47th International Conference on Very Large Data Bases

Copenhagen, Denmark - August 16-20, 2021

COVID-19 - VLDB 2021 Hybrid is still on - Read more

- ➢ Get to attend the top international data management conference!
- ➢ Get insight into inner workings of a conference
- ➢ Contribute as virtual or on-site volunteer

You can help with

- Registration desk support
- Microphone duty for on-site discussions
- Registering participants in conference app
- Check program artefacts (videos, posters,…) –2-4 weeks prior to conference

Check out vldb.org/2021

Contact volunteer chair Ira Assent: ira@cs.au.dk