



Experimenting with SSDs:

A reproducibility perspective

Philippe Bonnet

phbo@itu.dk

Joint work with Pinar Tözün, Simon Lund, Andreas Blanke, Magnus Krøyer, Luc Bouganim, Björn Jónsson and Alberto Lerner

DASYA

Data-Intensive Systems and Applications



Björn Thor Jónsson
Pinar Tözün
Sebastian Büttrich

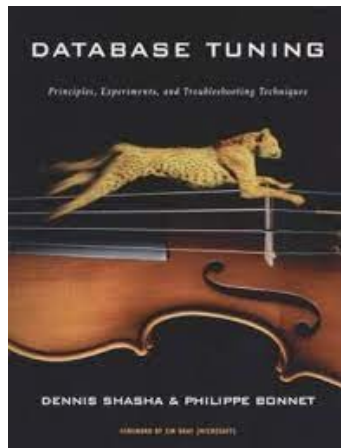


dasya.itu.dk

We are recruiting
postdoc and faculty!



SensIT: Sensor Information Technology



Alibaba Open Channel Ecosystem



As Alibaba's strategic partner on Open Channel SSDs, Intel has worked with Alibaba extensively since 2017 to co-develop and co-validate this innovative solution. Alibaba's strength as a leading cloud service provider combined with Intel's strength as the leading memory and storage innovator puts us in a position to deliver the industry's 1st Open Channel SSD product.

- Alibaba is collaborating with major vendors in industry to build an ecosystem for Open Channel SSD
- Share development & debug resources
- Reduce time & complexity for SSD qualification in Alibaba
- Massive deployment in 2019



Very Large Data Bases



© Alibaba Group 2018

www.alibaba.com

Reproducibility and Replicability

ACM Emerging Interest Group

Fostering a diverse and inclusive community around the issues of reproducibility and replicability of computational research.

<https://reproducibility.acm.org/>

Conference Working Group (C-WG)

The Conference WG is charged with carrying out the strategic planning and outreach required to establish a standalone annual EIG conference. The WG will work with the larger EIG and its leadership as well as ACM to achieve this goal. The WG will produce a written report at least annually documenting its activities.

C-WG Leader: [Ivo Jimenez](#)

Practices Working Group (P-WG)

The Practices WG is charged with shepherding a robust discussion on (best/better) practices for reproducibility and generating recommendations for the community that advance reproducibility. It is anticipated that P-WG liaises with other SIGs, in particular with those members who are active in reproducibility in their own field (whether in research, standards development, or other areas). The WG will produce a written report at least annually documenting its activities.

P-WG Leaders: [Limor Peer](#) & [Daniel Oberski](#) & [Vicky Rampin](#)

=> Towards ACM SIGREPRO

Evolution of p-recs

P-RECS'21

4th International Workshop on Practical Reproducible Evaluation of Systems

June 21st, 2021

(Online event, co-located with [HPDC'21](#))

Check out keynotes:

- Lisa Yan - Learning Networking by Reproducing Research Results
<https://p-recs.github.io/2020/keynote>
- Tanu Malik - Artifact Description/Artifact Evaluation
<https://sc21.supercomputing.org/submit/reproducibility-initiative/>

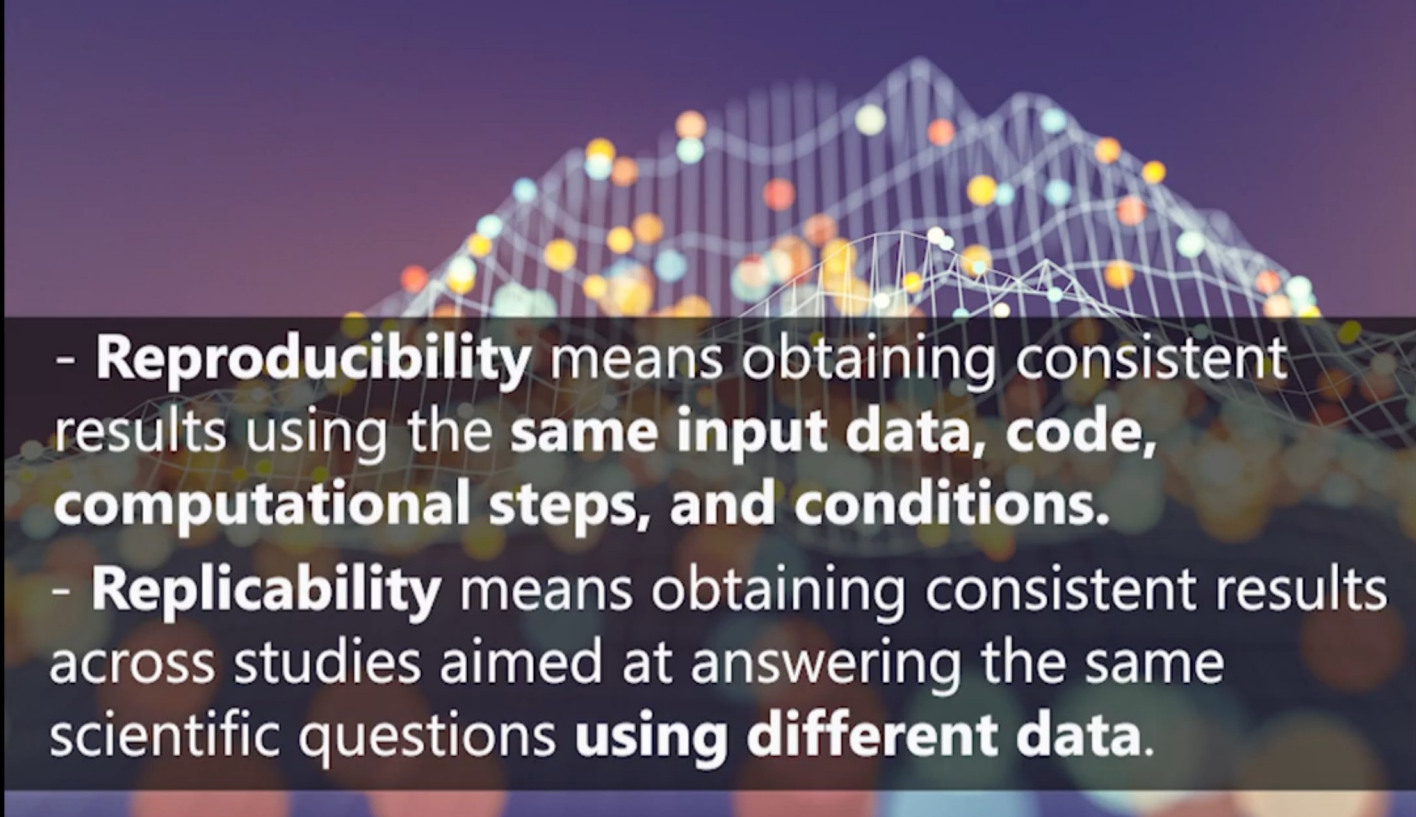
Engaging the SIGREPRO community

<https://reproducibility.acm.org/blog>

PRINCIPLES	March 25	We will explicate ACM principles with respect to reproducibility.
SOLUTIONS	April 22	The current state of solutions and tools that support reproducibility.
TRAINING	May 20	How and where is scientific reproducibility taught?
PUBLISHING	June 24	Journals' and conferences' approaches to computational reproducibility.
PRESERVATION	July 29	Reproducibility in the long term requires curation and preservation.

Definitions (2019)

<https://www.nap.edu/catalog/25303/reproducibility-and-replicability-in-science>

- 
- **Reproducibility** means obtaining consistent results using the **same input data, code, computational steps, and conditions.**
 - **Replicability** means obtaining consistent results across studies aimed at answering the same scientific questions **using different data.**

Outline

- **The problem: Experimenting with SSDs**
- SSD internals
- SSD characterization challenges
 - Black box approach
 - Experimental framework
 - Challenges
 - White box approach
 - Simulation vs. Hardware-Based Prototyping
 - Challenges
- Best Practices

NVMe SSD models

NVMe SSD models ↕	Grenoble ↕	Lille ↕	Luxembourg ↕	Lyon ↕	Nancy ↕	Nantes ↕	Rennes ↕	Sophia ↕	NVMe SSDs total ↕
Dell Express Flash NVMe PM1725 1.6TB AIC	8								8
SAMSUNG MZ1LB1T9HALS-00007				10					10
Samsung PM1735	4								4
Sites total	12			10					22

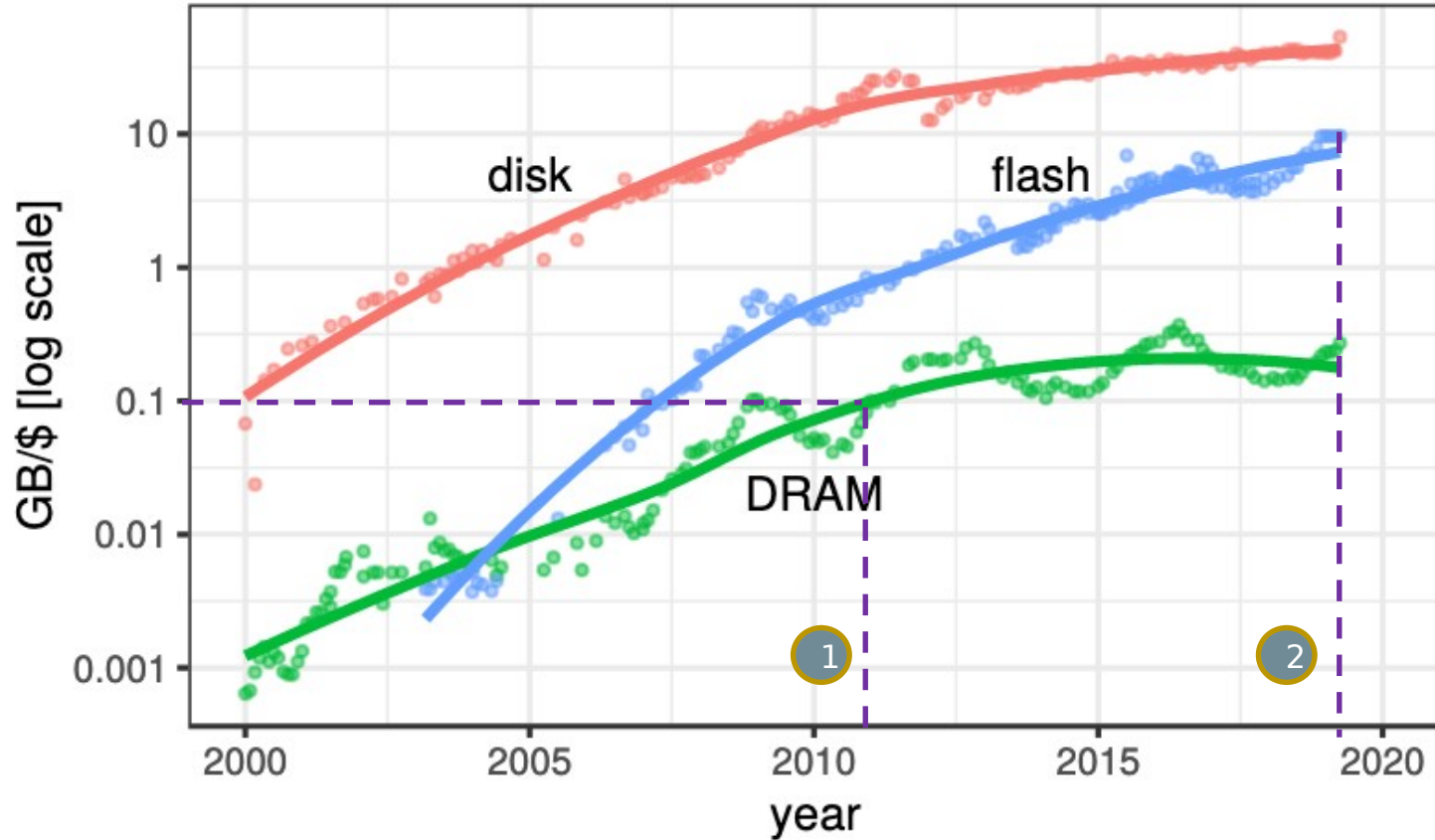
Nodes with several disks

Site ↕	Cluster ↕	Number of nodes ↕	Main disk ↕	Additional HDDs ↕	Additional SSDs ↕
Grenoble	dahu	32	SSD 240 GB	1 (4.0 TB)	1 (480 GB)
Grenoble	drac	12	HDD 1.0 TB	1 (1.0 TB)	0
Grenoble	troll	4	SSD 480 GB	0	1 (1.6 TB)
Grenoble	yeti	4	SSD 480 GB	3 (2.0 TB*, 2.0 TB*, 2.0 TB*)	2 (1.6 TB, 1.6 TB)
Lille	chetemi-10	1	HDD 600 GB	1 (300 GB)	0
Lille	chetemi-[1-9,11-15]	14	HDD 300 GB	1 (300 GB)	0
Lille	chiclet	8	SSD 480 GB	2 (4.0 TB*, 4.0 TB*)	0
Lille	chifflet	8	SSD 400 GB	2 (4.0 TB*, 4.0 TB*)	1 (400 GB*)
Lille	chiffnot	8	SSD 480 GB	4 (4.0 TB*, 4.0 TB*, 4.0 TB*, 4.0 TB*)	1 (480 GB*)
Lyon	gemin	2	SSD 480 GB	0	4 (1.92 TB*, 1.92 TB*, 1.92 TB*, 1.92 TB*)
Lyon	hercule	4	HDD 2.0 TB	2 (2.0 TB, 2.0 TB)	0
Lyon	neowise	10	SSD 1.92 TB	0	1 (1.92 TB)
Lyon	pyxis	4	SSD 250 GB	0	1 (250 GB)
Nancy	graouilly	16	HDD 600 GB	1 (600 GB)	0
Nancy	graphite	4	SSD 300 GB	0	1 (300 GB)
Nancy	grappe	16	SSD 480 GB	1 (8.0 TB*)	0
Nancy	grele	14	HDD 299 GB	1 (299 GB)	0
Nancy	grimoire	8	HDD 600 GB	4 (600 GB*, 600 GB*, 600 GB*, 600 GB*)	1 (200 GB*)
Nancy	grisou	51	HDD 600 GB	1 (600 GB)	0
Nancy	gros	124	SSD 480 GB	0	1 (960 GB*)
Nancy	grouille	2	SSD 1.92 TB	0	1 (960 GB*)
Rennes	paranoia	8	HDD 600 GB	4 (600 GB, 600 GB, 600 GB, 600 GB)	0
Rennes	parasilo	27	HDD 600 GB	4 (600 GB*, 600 GB*, 600 GB*, 600 GB*)	1 (200 GB*)
Rennes	paravance	72	HDD 600 GB	1 (600 GB)	0

*: disk is reservable

Fundamental Trend

V.Leis: <http://cidrdb.org/cidr2020/papers/p16-haas-cidr20.pdf>



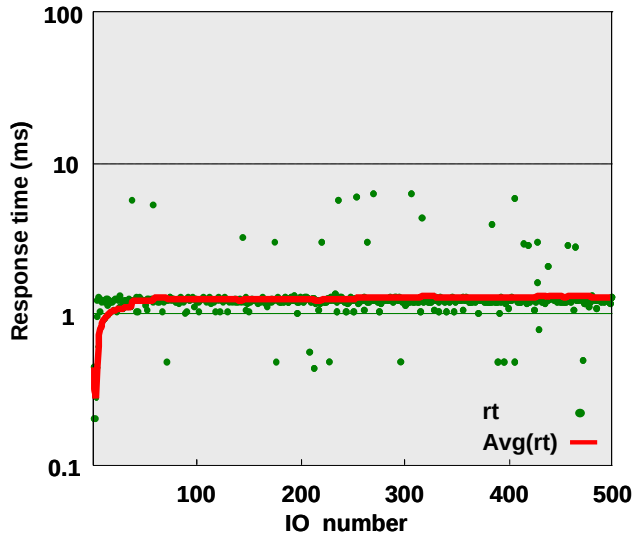
①

Memory is what disks used to be

②

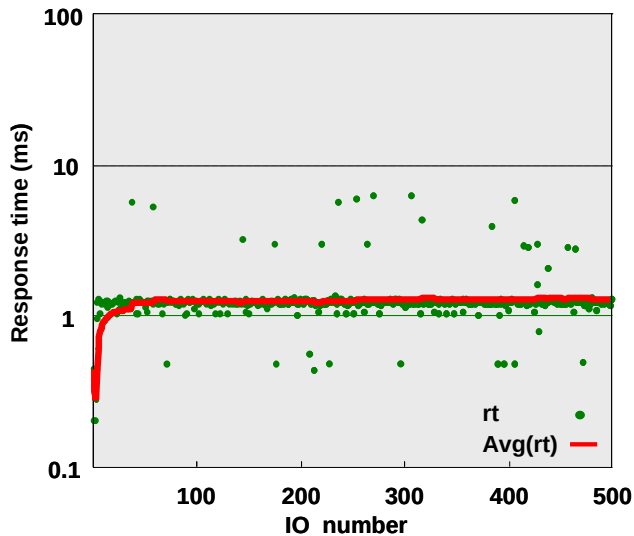
Flash is 20X cheaper than RAM

- Measuring Samsung SSD RW performance
 - Out-of-the-box ...

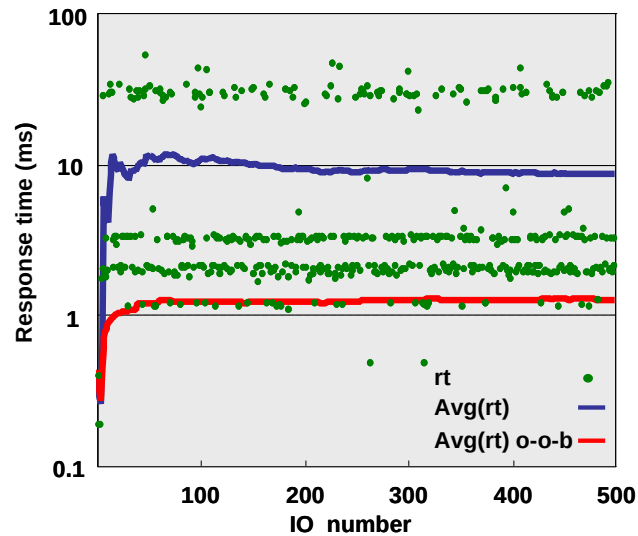


*Random Writes – Samsung SSD
Out of the box*

- Measuring Samsung SSD RW performance
 - Out-of-the-box ... and after filling the device!!! (similar behavior on Intel SSD)



Random Writes – Samsung SSD
Out of the box

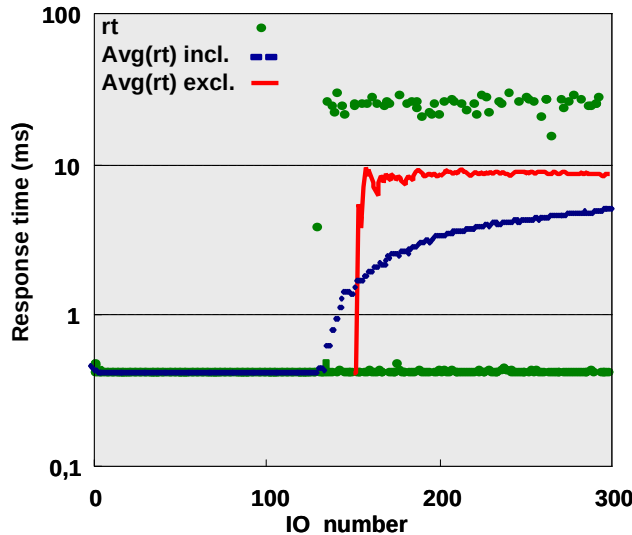


Random Writes – Samsung SSD
After filling the device

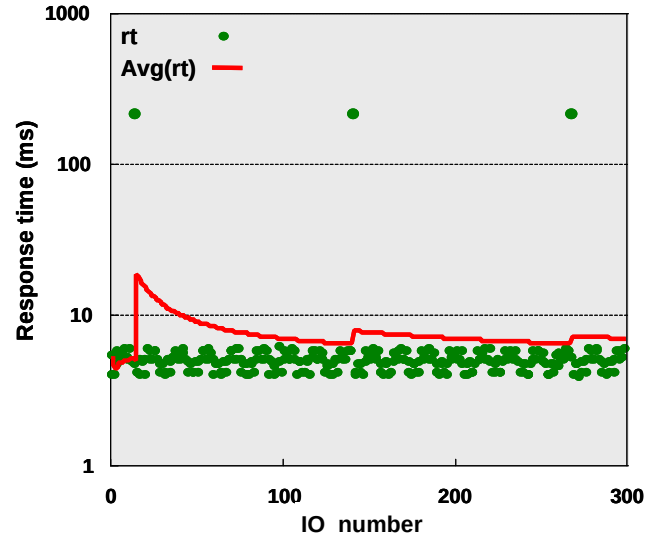
Methodology (2): Startup and running phases

CIDR 2009

- When do we reach a steady state? How long to run each test?



Startup and running phases for the Mtron SSD (RW)



Running phase for the Kingston DTI flash Drive (SW)

What is a meaningful SSD experiment?

- Experimental conditions are crucial for the reproducibility of SSD experiments
 - For which state of a given SSD is a result valid?
 - Experimental conditions are often ignored from reproducibility frameworks focusing on availability of code, data and experiment workflows.
- Reproducibility as a means to define a meaningful experiment, i.e., results with a well-defined scope:
 - Does a result depend on the duration of the experiment?
 - Is a result valid across states for a given SSD?
 - Is a result valid across SSDs?

Outline: Experimenting with SSDs

- The problem
- **SSD internals**
- SSD characterization challenges
 - Black box approach
 - Experimental framework
 - Challenges
 - White box approach
 - Simulation vs. Hardware-Based Prototyping
 - Challenges
- Best Practices

Flash cell technology (read, program, erase)

- TLC/QLC for archival, SLC/MLC for performance
- Limited lifetime for entire blocks (when a cell wear out, the entire block is marked as failed).

NAND Layout and structure

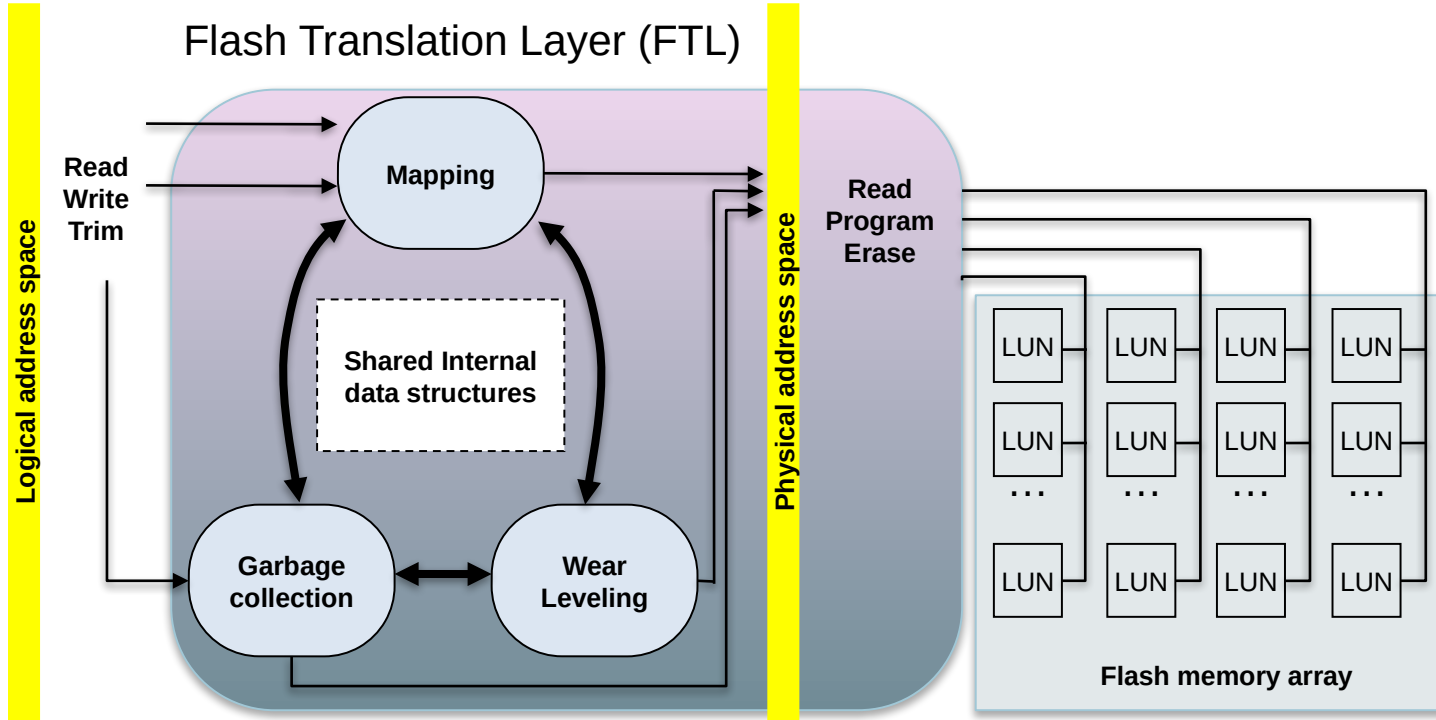
- Block is the smallest erase granularity

Program Disturb

- Page is the smallest program granularity ($\frac{1}{4}$ for SLC)
- Pages must be programmed sequentially within a block
- Use of ECC is mandatory ✉ ECC unit is the smallest read unit (generally 1 or $\frac{1}{4}$ page)

SSDs as block devices

SIGMOD 2011



Simple FTL

One free flash block opened per LUN.

Writes are buffered.

Logical-to-Physical mapping (L2P) is round robin.

When buffers contains full blocks for all LUNs, these are flushed to NAND

=> maximal channel utilization and LUN parallelism for writes

No update in place

=> Updates invalidate entries in L2P tables

=> Need for garbage collection to reclaim blocks with valid and invalid pages, otherwise disks fills up with invalid pages

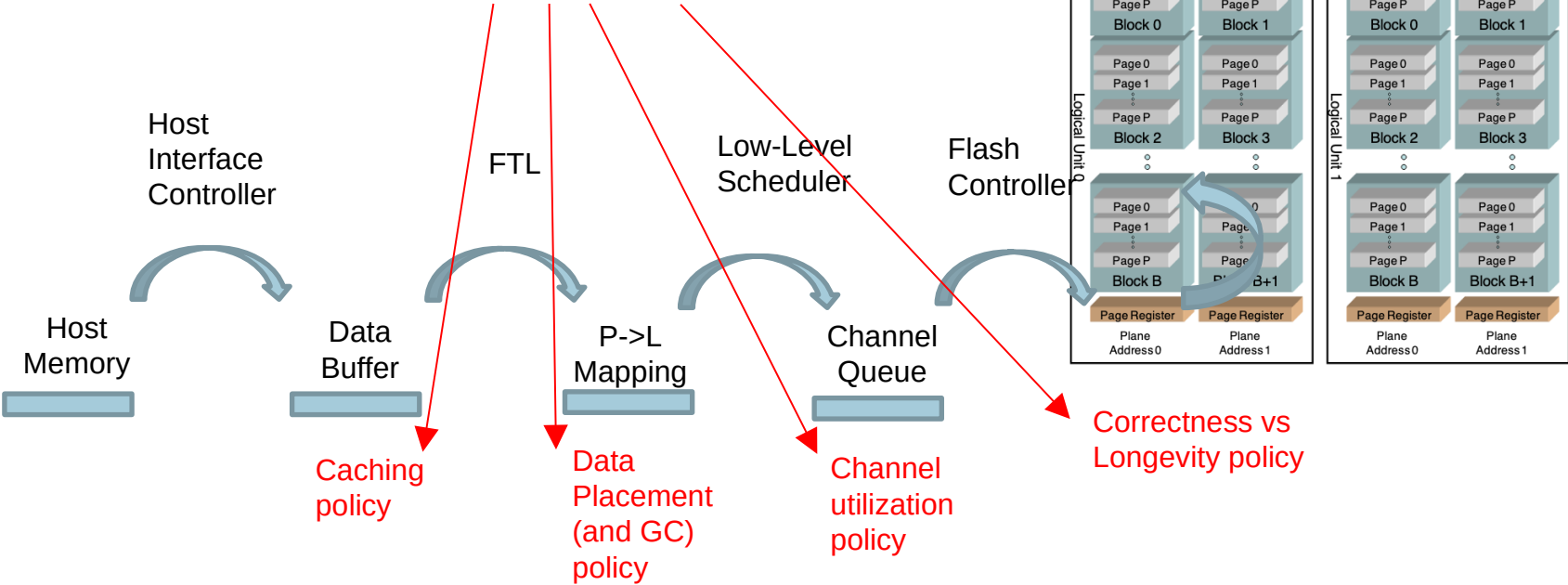
=> Need for overprovisioning (= physical space >> logical space)

- Large number of free flash blocks to avoid blocking writes to reclaim free flash blocks

Decisions, Decisions, Decisions

SIGMOD 2021

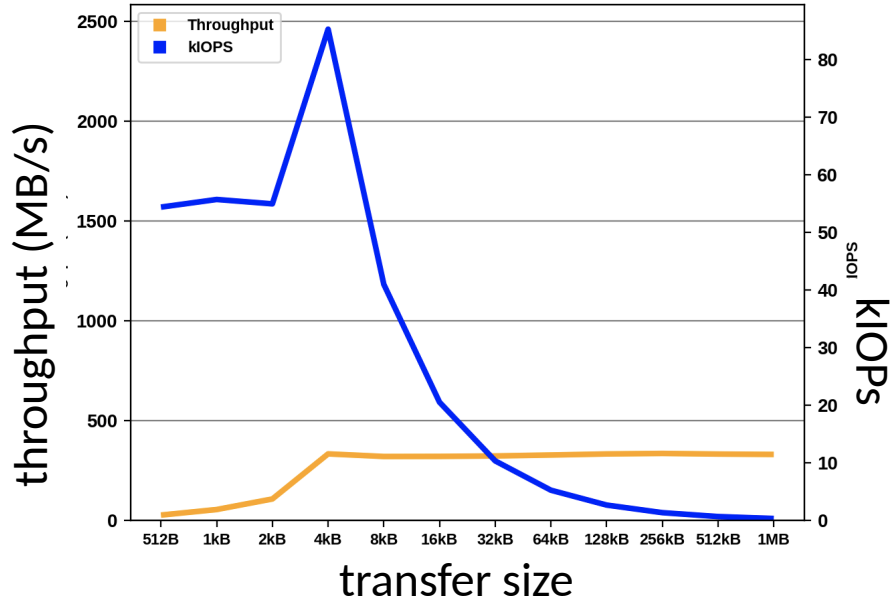
The decisions can be more or less effective, depending on the workload.



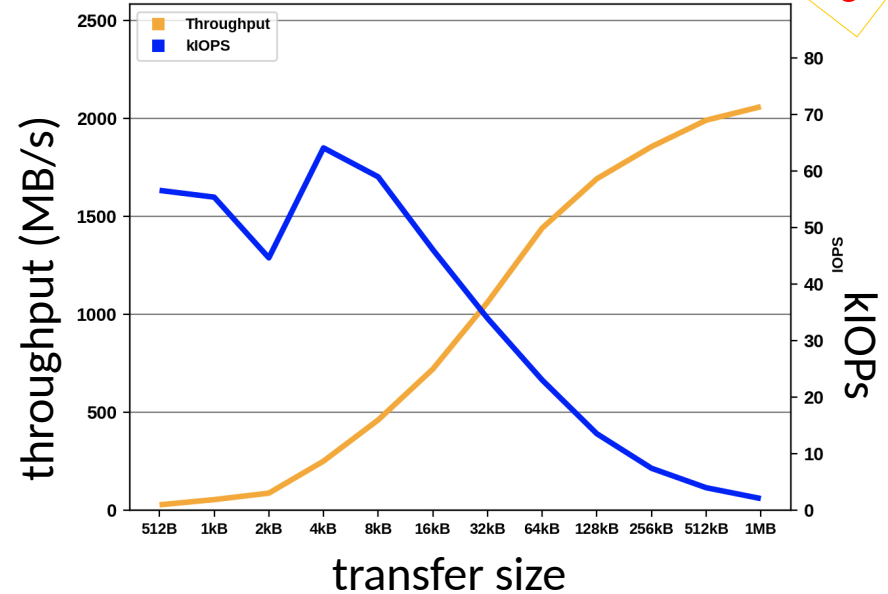
random writes- source: [AnandTech](https://www.anandtech.com/show/11444/samsung-z-nand-ssd-performance) 2019

HTTPS 2019

Samsung SSD with Z-NAND



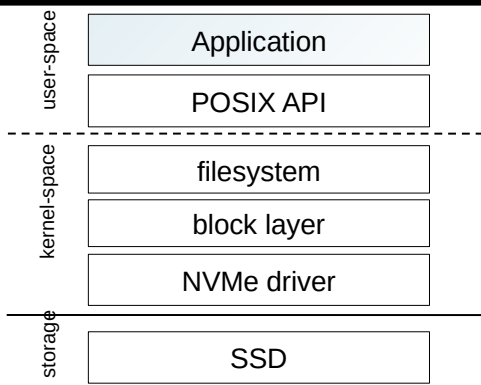
Intel Optane



**No intrinsic performance characteristics
for SSDs equipped with a generic FTL**

Linux I/O Frameworks

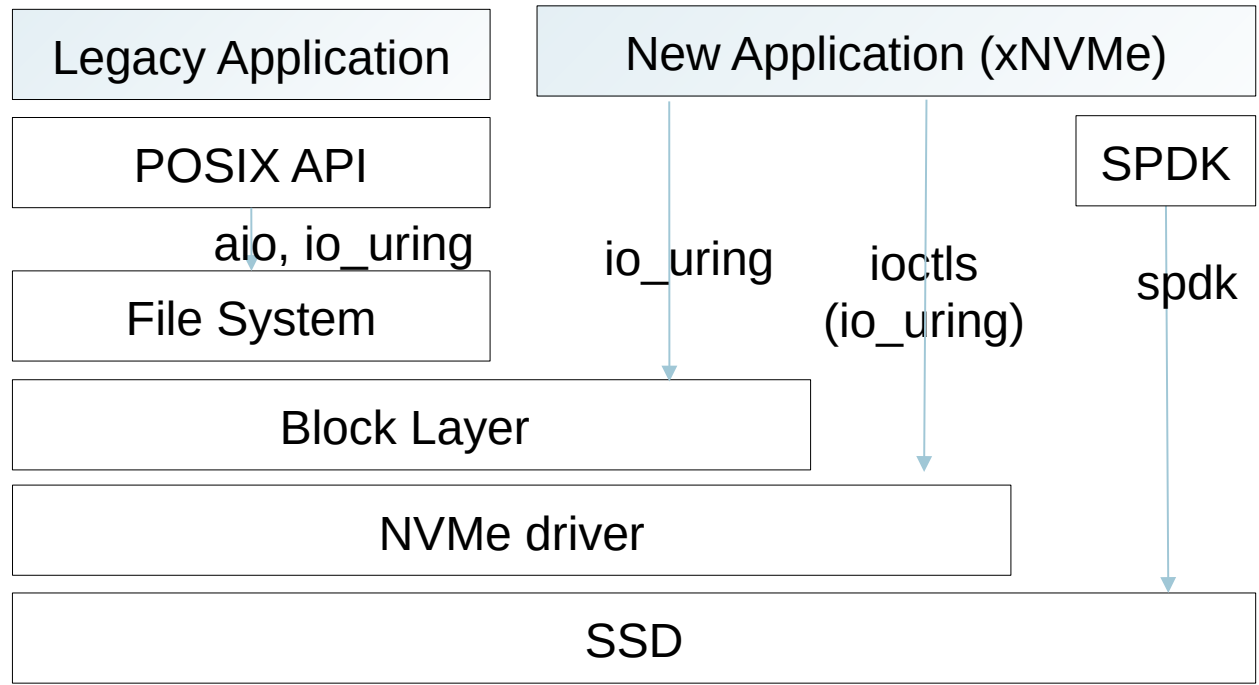
SIGMOD 2021



File system extensions for ZNS
In F2FS, XFS

mlq-blk for NVMe since 2013

No support for KV in NVMe driver yet



SSD Internals Take-Aways

1. Complex firmware needed to handle parallelism, error correction and flash idiosyncracies
2. There is nothing intrinsic about SSD performance. It depends on hidden design decisions.
3. SSDs are not a uniform class of devices.
=> “SSD-optimized systems” is meaningless label
4. Software complexity and diversity within and above SSD

Outline: Experimenting with SSDs

- The problem
- SSD internals
- **SSD characterization challenges**
 - Black box approach
 - Experimental framework
 - Challenges
 - White box approach
 - Simulation vs. Hardware-Based Prototyping
 - Challenges
- Best Practices

Black Box SSD Characterization

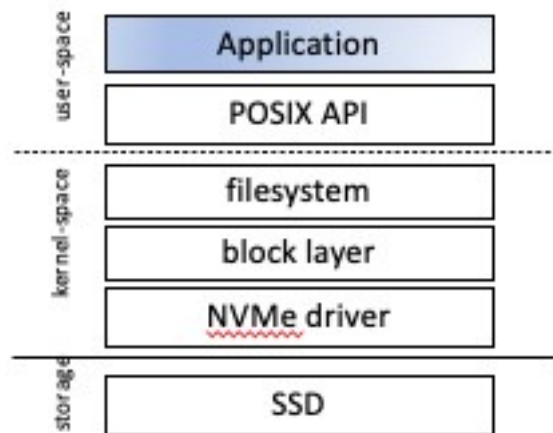
State is not observable; Internals are not documented:

- We want to understand the behavior of a specific SSD

How does it react to various workloads?

How does this behavior evolve in time and space?

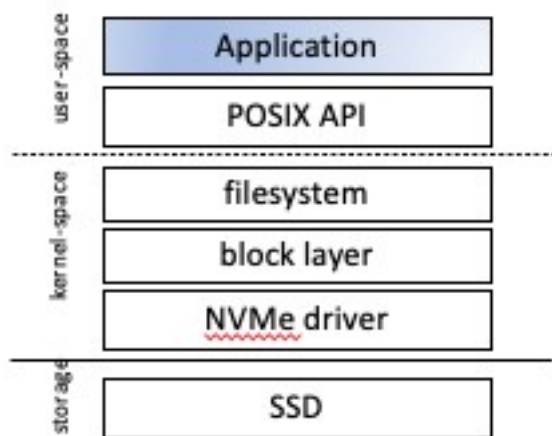
Experimental Framework #1: System



Key questions:

- What layers are part of the system?
- How to fix default system parameters for these layers?
- How to define a reference initial state?
- How to submit workload?

Experimental Framework #1: System

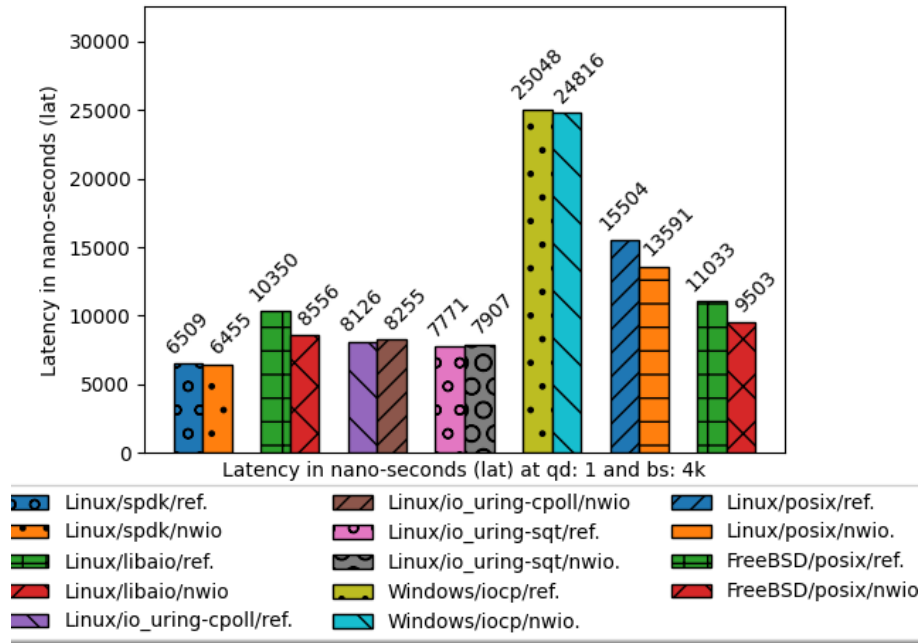


Example pitfalls:

- General claim based on experiments that reflect behavior of a given SSD
- Performance-optimized application on top of legacy file system
- SATA SSD for performance evaluation
- No well-defined initial state

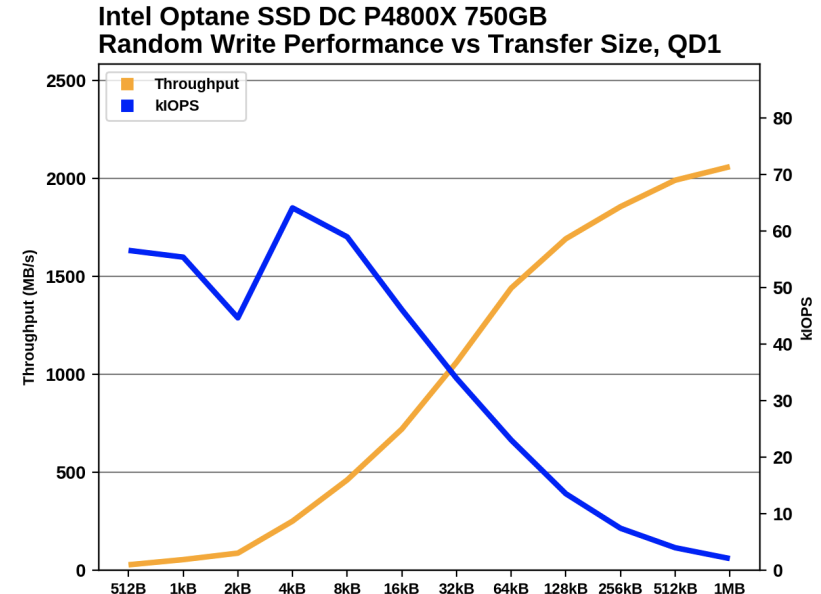
Experimental Framework #2: Metrics

1. Latency for each IO (nsec)



2. Number of IO completed per unit of time (kIOPS)

3. Throughput (MB/sec)



Experimental Framework #3: Workload

SIGMOD 2021

	Tx Log	Checkpoint	...
Sequential or random access	Seq	Seq	
Read/write/mixed operations	Write (but for recovery)	Write (but for recovery)	
Queue depth (# of outstanding operations)	1	1 or more per core	
Size of an operation	Page	Multiple Pages	
Other relevant characteristics			
Circularity	yes	no	
Latency Sensitiveness	yes	no	
Bursty-ness	yes	no	
Priority	high (user perceived)	depends on recovery goals	

Checkpoint is much more parallel than and can overpower Tx Log if proper scheduling is not in place.

Experimental Framework #4: Experiments

1. Design

- Start-up vs. running phase
- Steady state? How long to run an experiment?

2. Analysis

- Time series of latency
 - What kind of statistics to use?
 - Mean and variance only if time series is stationary
 - Alternatives: multimodal distribution, trends & seasonality

Black box SSD Characterization Challenges

- Designing experiments
 - Default system parameters, workload.
- Running experiments
 - Steady state, stationarity test.
- Analyzing experiments
 - Managing and analyzing time series composed of millions of latency measurements.

Outline: Experimenting with SSDs

- The problem
- SSD internals
- **SSD characterization challenges**
 - Black box approach
 - Experimental framework
 - Challenges
 - White box approach
 - Simulation vs. Hardware-Based Prototyping
 - Challenges
- Best Practices

White Box SSD Characterization

SSD state is observable; Internals are documented

- We want to understand the behavior of a specific SSD

How does it react to various workloads?

How does this behavior evolve in time and space?

- We want to explore design space

How do choice of internal policy impact performance?

How does choice impact various workloads?



MINIMAL FTL: TAKE THE FTL OUT OF THE EQUATION!

Pros

Maximal performance for

- SR, RR, SW
- Semi-Random Writes

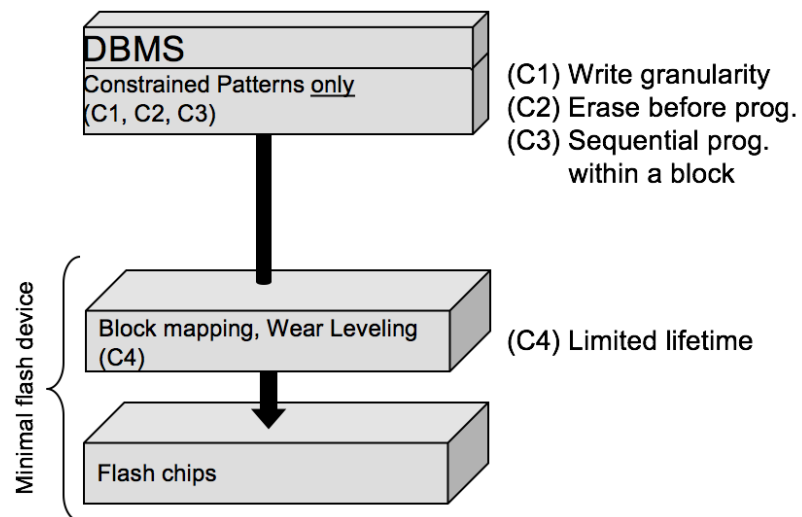
Maximal control for the DBMS

Cons

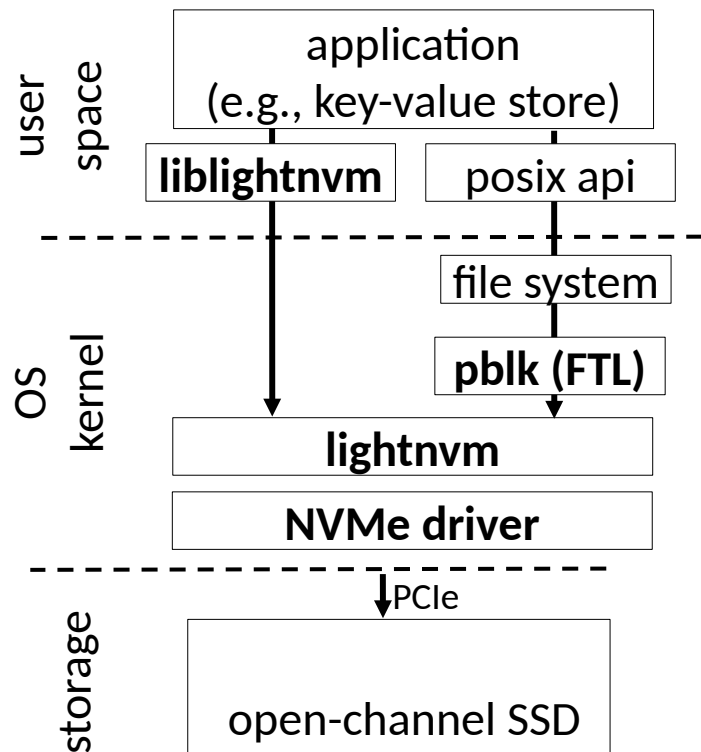
All complexity is handled by the DBMS

All IOs must follow C1-C3

- The whole DBMS must be rewritten
- The flash device is dedicated



Lightnvm



FAST 2017

NVMe driver

- detection of OCSSD
- implements PPA (physical page address) interface

lightNVM subsystem

- uses NVMe protocol to handle address translations over OCSSD
- makes it possible for applications to leverage OCSSD

high level I/O interface

- **pblk** – block device via full-fledge FTL
- **liblightnvm** – access to core SSD management from user-space

Open-Channel SSDs: Grey Box

External interface exposes

- Device geometry
- Block metadata (including bad blocks)

External interface hides

- Channel utilization
- Error-correction
- Caching policy at block level across channels

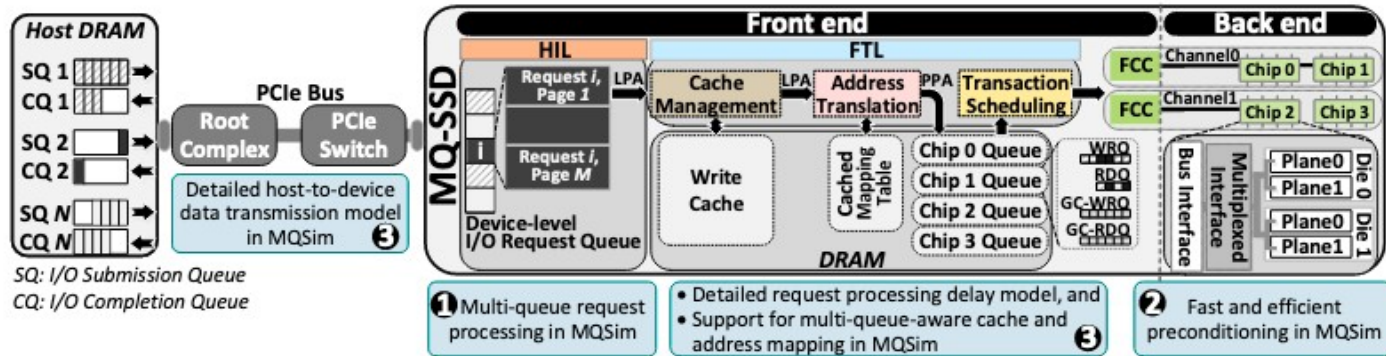
Example of an SSD Simulator

SIGMOD 2021

MQSIM can be used **stand-alone or inside a system-wide simulator** such as Gem5.

Allows access to three (out of four) internal interfaces we discussed.

Can be used to implement new NVMe directives or command sets.



Source: Tavakkol 2018

Simulation Trade-offs

SIGMOD 2021

Pros

100% software.

Easy learning curve to use certain simulators.

The sky is the limit as to what new hardware can be modeled.

As long as it reflects a reasonable datasheet of such hardware (which is not always available).

Cons

Can't capture aspects that are not modeled:

non-determinism in general, e.g., flash errors, flash aging.

Slow execution:

Gen5 is equivalent to a **machine running at a few MHz.**

Hardware-Based Prototyping Platforms

SIGMOD 2021

OpenSSD Family of Devices

3rd generation of devices using **actual Flash memory.**

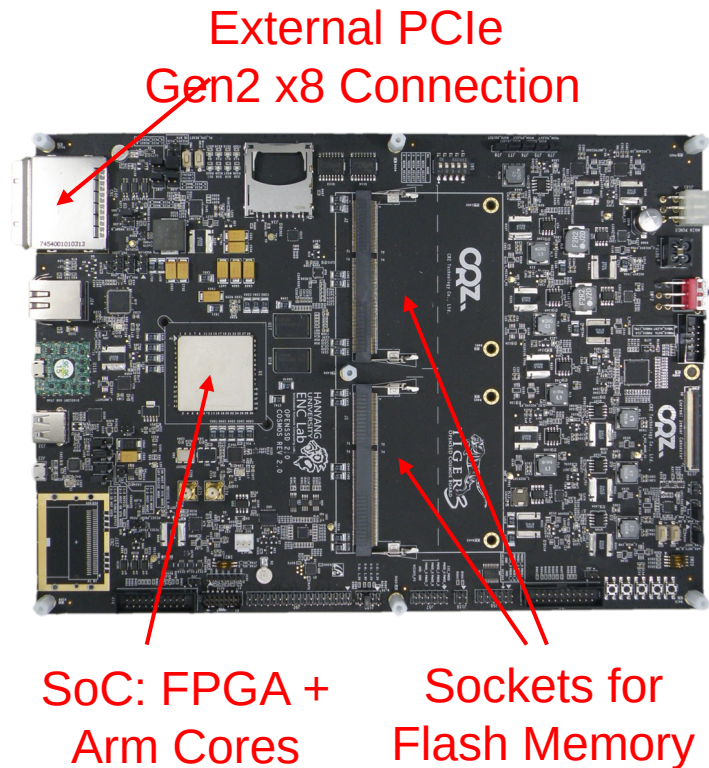
Full-fledged SSD, 100% compatible with Linux/Windows NVMe drive

Large functionality set implemented as firmware (C coding) running on ARM cores

Daisy Family

Informal 4th generation of OpenSSDs

Commercial spin-off backed by CRZ in Korea



Prototyping Trade-Offs

SIGMOD 2021

Pros

Real flash with real (at times idiosyncratic) behavior.

Many examples of programmable devices.

Growing community.

Cons

Software-based features can add latency.

Not many options of Flash and channel designs.

Changing features close to the Flash require a steep learning curve.

White box SSD Characterization Challenges

Black box challenges

+

Representing SSD state and its evolution in time

- Ideally, at IO-level granularity

Outline: Experimenting with SSDs

- The problem
- SSD internals
- SSD characterization challenges
 - Black box approach
 - Experimental framework
 - Challenges
 - White box approach
 - Simulation vs. Hardware-Based Prototyping
 - Challenges
- **Best Practices**

System Parameters

Hardware	Model
Cpu	Intel Core i5-9400 2.90GHz
Memory	Corsair 2x 16GB DDR4 3200Mhz CL18
Board	MSI MPG Z390I GAMING EDGE AC
SSD	Intel Optane Memory M10 Series (MEMPEK1J016GAL)
Software	Model
FreeBSD	Version 12.1
Linux	Debian Bullseye / Kernel 5.14
Windows 10	Version 21H1
fio	Version 3.27
gcc	Version 10.2.1
clang	Version 12.0.1
SPDK	Version 21.04
nwio	Version x.y.z

How to capture/report experimental conditions?

SSD Preconditioning

Experimental conditions

	Firmware state	NAND flash state
Blackbox approach	Unknown	Unknown
Whitebox approach	L2P table	Flash block metadata

Blackbox heuristic: SSD preconditioning as a means to force Logical-to-Physical (L2P) table in a “well-defined” state, assuming simple FTL:

- Writing disk several times over with sequential writes
 - => Each flash block contains contiguous logical addresses
 - => Locality in logical space corresponds to locality in physical space
 - => Free blocks are reclaimed faster (by GC) than they are used (by sequential writes)
 - => GC does not kick in immediately
- Writing disk several times over with random writes
 - => Minimize number of free blocks => GC kicks in immediately

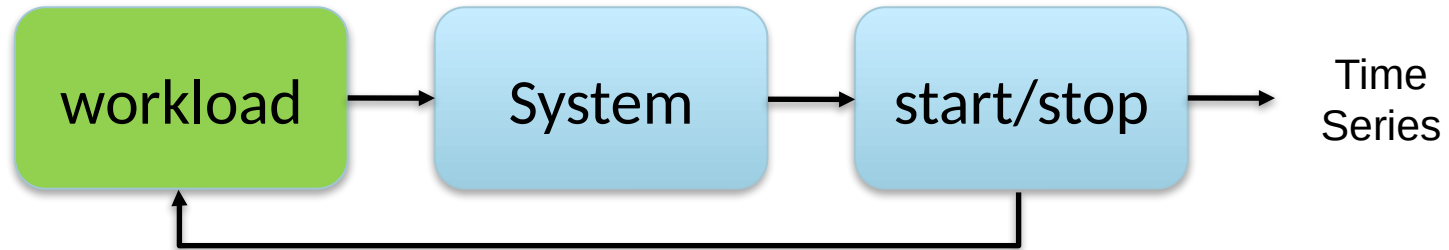
Steady state and Stationarity Testing

When to start / stop experiment?

Need for adaptive control

strict stationarity test =>

mean and variance constant for a fixed time window



fio, developed by Jens Axboe, has become the standard tool for submitting I/O workloads

- Diversity of I/O frameworks on Linux, Windows, FreeBSD
- Configuration files for flexible/reproducible experiments
 - Basic, mixed, parallel I/O patterns
- Result report for each run with statistics

=> Overview

- Log with latency per I/O (end-to-end, submission, completion)

=> In-depth analysis

⇒ **Ideal for reproducibility**

⇒ **Integral part of CI**

fio overhead

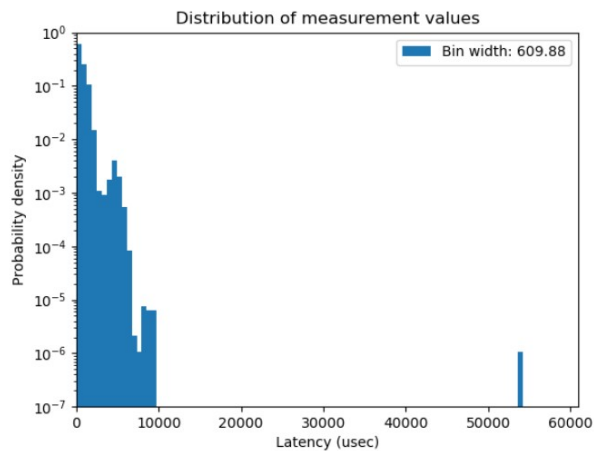
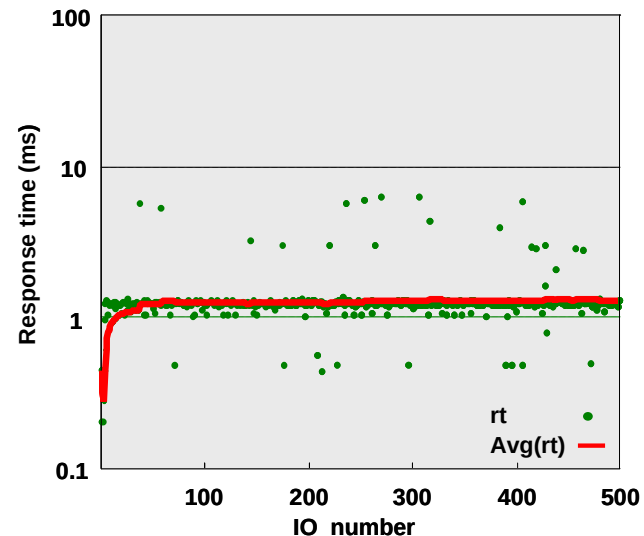
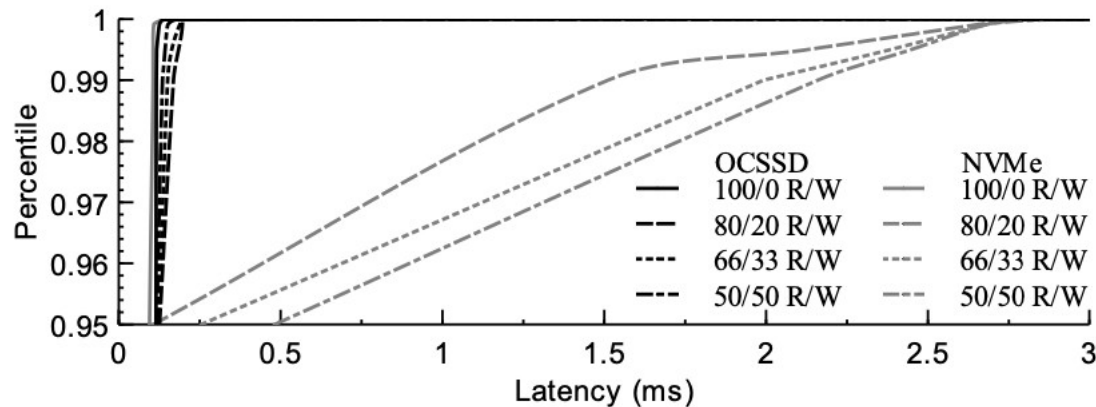
Configuration with nil-backend (no actual I/O).

Queue depth=1, block size=4KB, random writes

Latency:

- Minimum: 8 nsec
- Maximum: **17916** nsec (~I/O latency on Optane SSD)
- Average: 36 nsec
- Standard deviation: 48 nsec

Reporting latency results



Mean latency is most often meaningless.
The characteristics of the I/O latency distribution are often more important than temporal patterns.

⇒ **Be mindful of your choice of statistical representation**

⇒ Be mindful of covariance across runs

Conclusions

Experimenting with SSD is challenging, because **experimental conditions are crucial** – yet they cannot be directly controlled (without whitebox approach).

Reporting experimental results with SSD requires **appropriate statistical representation**.

Reproducibility as a framework to reason about **meaningful experiments**.