The Introduction to FEMU
and
its newly related research

Presented by Chen Haodong

HDU – Center of Cloud computing

Outline

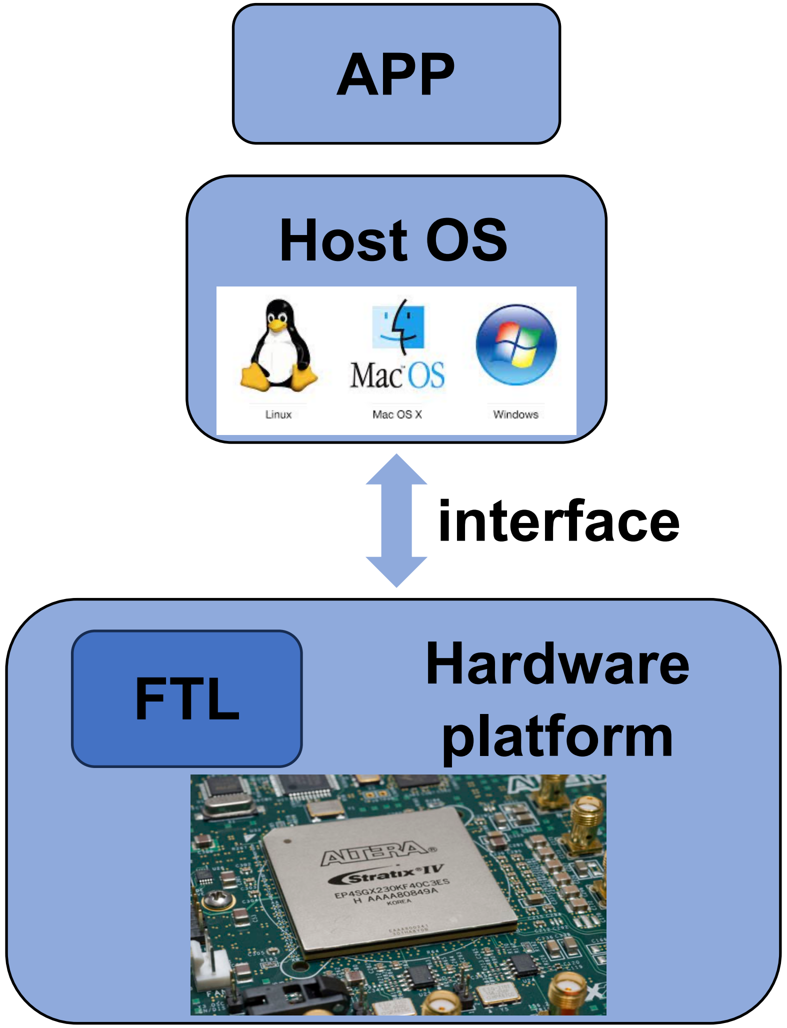
- 1. Backgrounds
- 2. Existing SSD platform
- 3. Case of FEMU
- 4. Newly related work in 2023
- 5. Future work

1. Backgrounds

SSD research mainly focus on 3 parts:

1. kernel changes
2. Interface changes
3. FTL changes

Typical full-stack research



2. Existing SSD platforms

Simulator

Emulator

Hardware platform

2. Existing SSD platforms

Simulator

DiskSim’s SSD;
FlashSim; SSDSim;

Emulator

Hardware platform

2. Existing SSD platforms

Simulator

DiskSim's SSD;
FlashSim; SSDSim;

Emulator

Hardware platform

simple

Time-saving

internal-SSD research

2. Existing SSD platforms

Simulator

DiskSim's SSD;
FlashSim; SSDSim;

- **simple**
- **Time-saving**
- **internal-SSD research**
- **kernel-level research**

Emulator

Hardware platform

2. Existing SSD platforms

Simulator

DiskSim’s SSD;
FlashSim; SSDSim;

Emulator

Hardware platform

OpenSSD; FPGA;
OpenChannel SSD;

- ☺ **simple**
- ☺ **Time-saving**
- ☺ **internal-SSD research**
- ☹ **kernel-level extension**

2. Existing SSD platforms

Simulator

DiskSim’s SSD;
FlashSim; SSDSim;

Emulator

Hardware platform

OpenSSD; FPGA;
OpenChannel SSD;

- ☺ **simple**
- ☺ **Time-saving**
- ☺ **internal-SSD research**
- ☹ **kernel-level extension**

- ☺ **Full-stack research**
- ☺ **Accurate**

2. Existing SSD platforms

Simulator

DiskSim's SSD;
FlashSim; SSDSim;

- simple
- Time-saving
- internal-SSD research
- kernel-level extension

Emulator

Hardware platform

OpenSSD; FPGA;
OpenChannel SSD;

- Full-stack research
- Accurate
- high costs
- complex to use

2. Existing SSD platforms

Simulator	Emulator	Hardware platform
DiskSim's SSD; FlashSim; SSDSim;	FlashEm; LightNVM QEMU; QEMU based VSSIM;	OpenSSD; FPGA; OpenChannel SSD;

Simulator

- ☺ **simple**
- ☺ **Time-saving**
- ☺ **internal-SSD research**
- ☹ **kernel-level extension**

Hardware platform

- ☺ **Full-stack research**
- ☺ **Accurate**
- ☹ **high costs**
- ☹ **complex to use**

2. Existing SSD platforms

Simulator

DiskSim's SSD;
FlashSim; SSDSim;

- simple
- Time-saving
- internal-SSD research
- kernel-level extension

Emulator

FlashEm;
LightNVM QEMU;
QEMU based VSSIM;

- Full-stack research
- Low costs

Hardware platform

OpenSSD; FPGA;
OpenChannel SSD;

- Full-stack research
- Accurate
- high costs
- complex to use

2. Existing SSD platforms

Simulator

DiskSim's SSD;
FlashSim; SSDSim;

- ☺ **simple**
- ☺ **Time-saving**
- ☺ **internal-SSD research**
- ☹ **kernel-level extension**

Emulator

FlashEm;
LightNVM QEMU;
QEMU based VSSIM;

- ☺ **Full-stack research**
- ☺ **Low costs**
- ☹ **Not scalability**
- ☹ **Not accurate**

Hardware platform

OpenSSD; FPGA;
OpenChannel SSD;

- ☺ **Full-stack research**
- ☺ **Accurate**
- ☹ **high costs**
- ☹ **complex to use**

3. The CASE of FEMU

FEMU is a software (QEMU-based) flash emulator for fostering future full-stack software/hardware SSD research.

3. The CASE of FEMU

FEMU is a software (QEMU-based) flash emulator for fostering future full-stack soft- ware/hardware SSD research.

Cheap

Open source,

download from:

https://github.com/vtess/FEMU

3. The CASE of FEMU

FEMU is a software (QEMU-based) flash emulator for fostering future full-stack soft- ware/hardware SSD research.

Cheap

(relatively) Accurate

Produce an advanced openChannel SSD Model

The error of resulting latencies drops to only 0.5-38%

3. The CASE of FEMU

FEMU is a software (QEMU-based) flash emulator for fostering future full-stack soft- ware/hardware SSD research.

- **Cheap**
- **(relatively) Accurate**
- **Scalable**

support 32 channels/chips

3. The CASE of FEMU

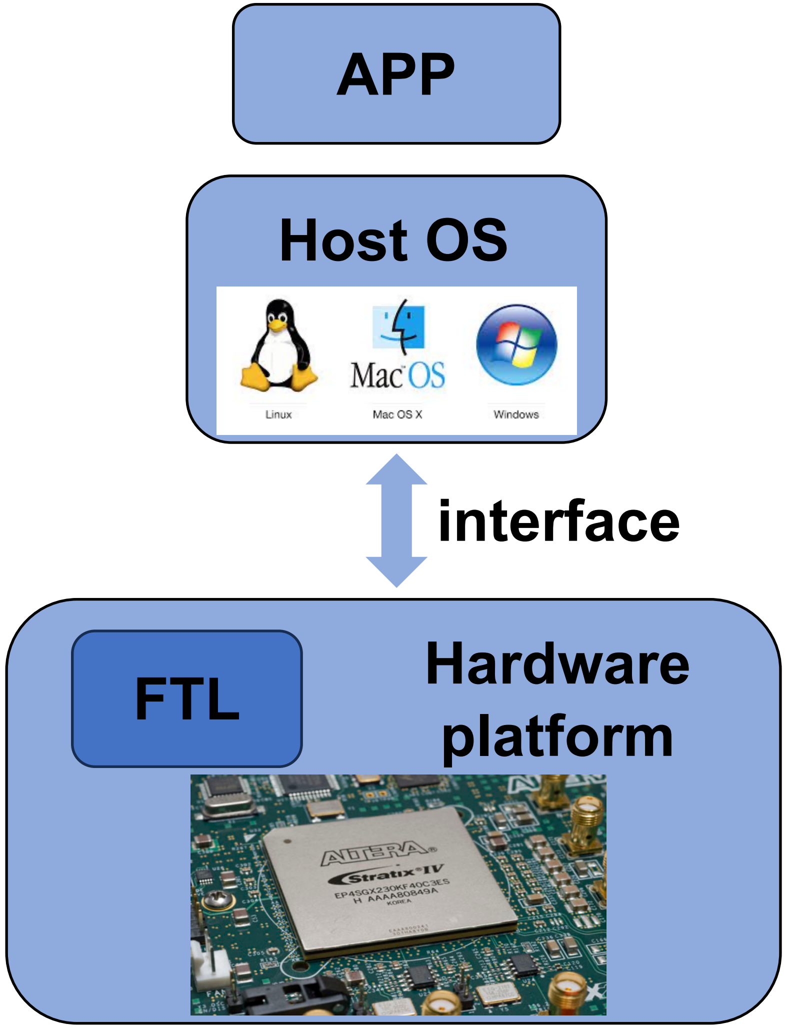
FEMU is a software (QEMU-based) flash emulator for fostering future full-stack soft- ware/hardware SSD research.

- Cheap
- (relatively) Accurate
- Scalable
- Extensible

- modifiable interface
- modifiable FTL
- modifiable SSD
- modifiable kernel

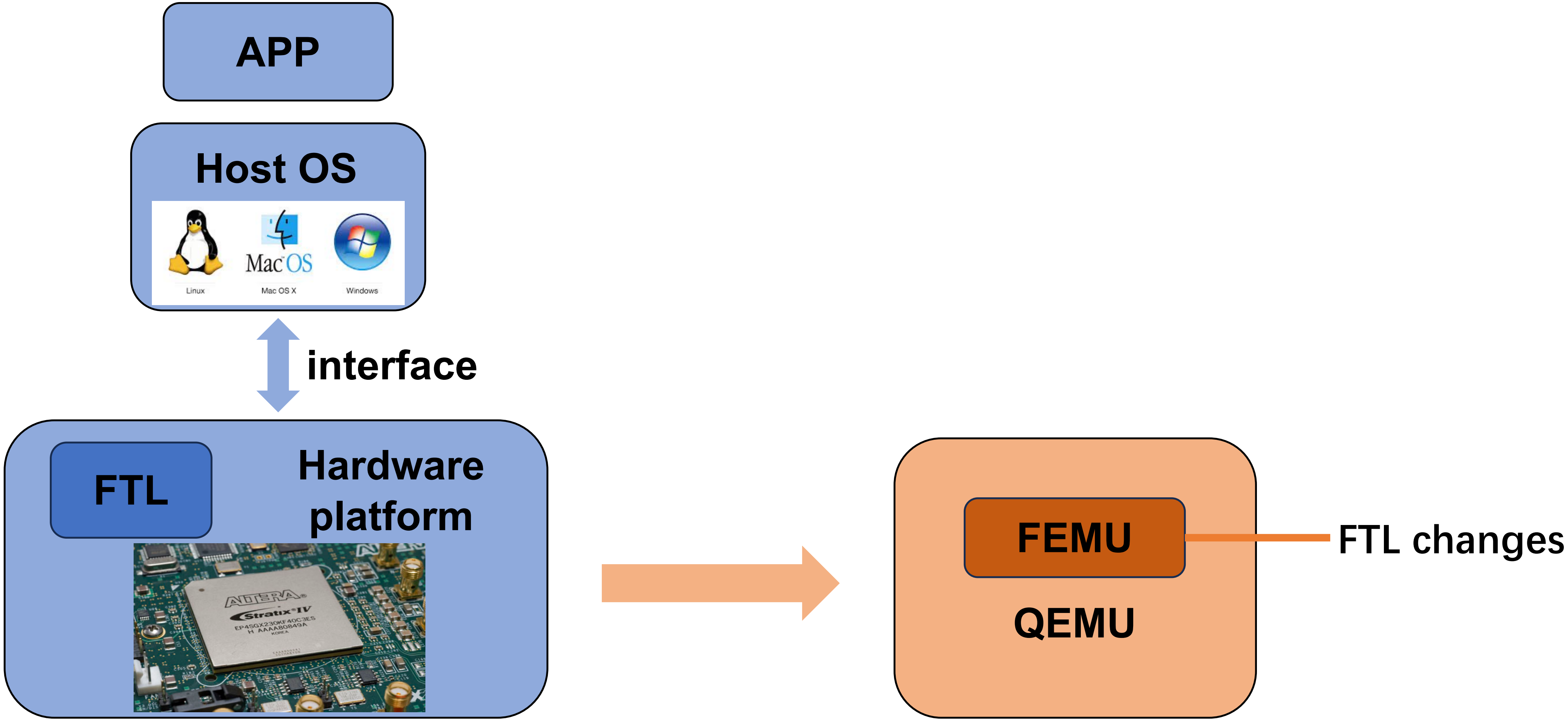
3. The CASE of FEMU

Typical full stack research .vs FEMU full stack research



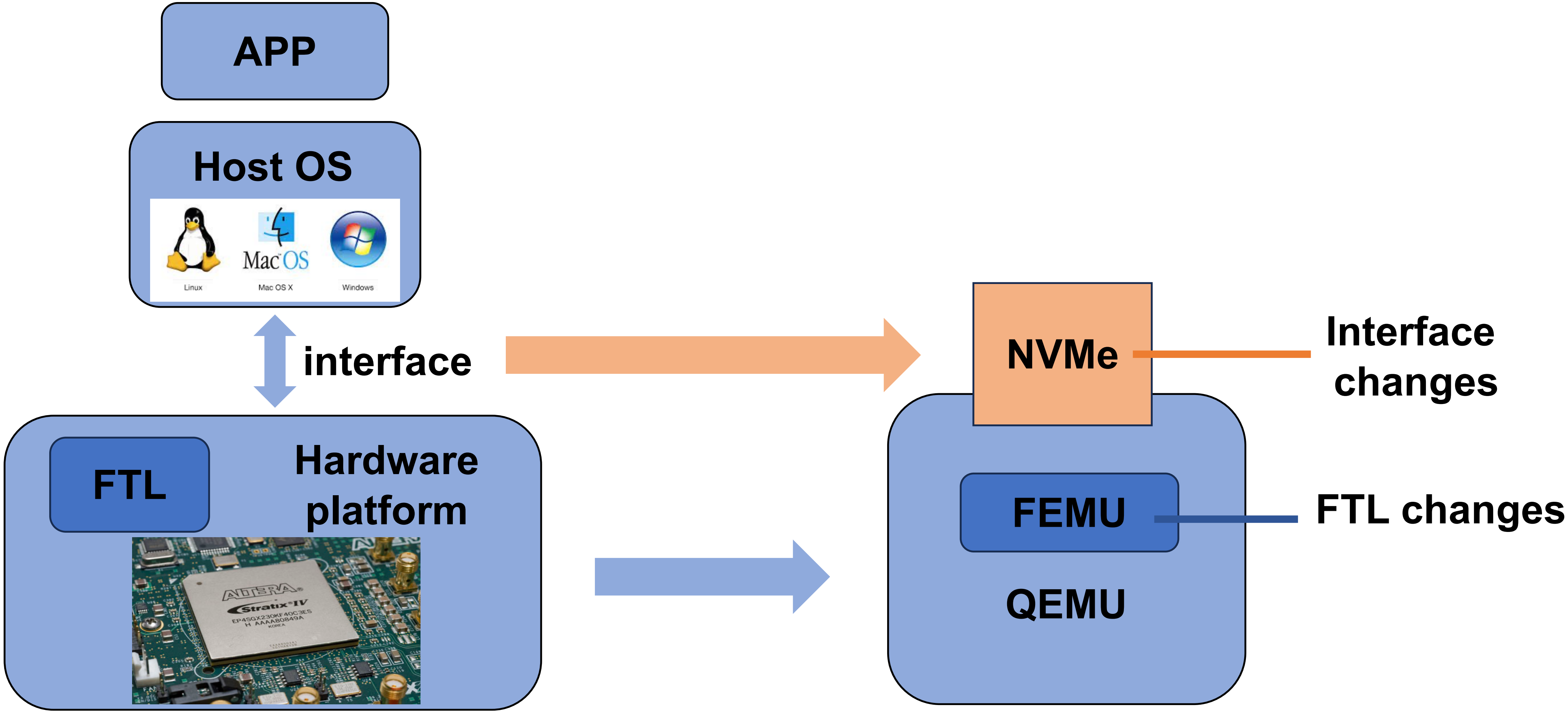
3. The CASE of FEMU

Typical full stack research .vs FEMU full stack research

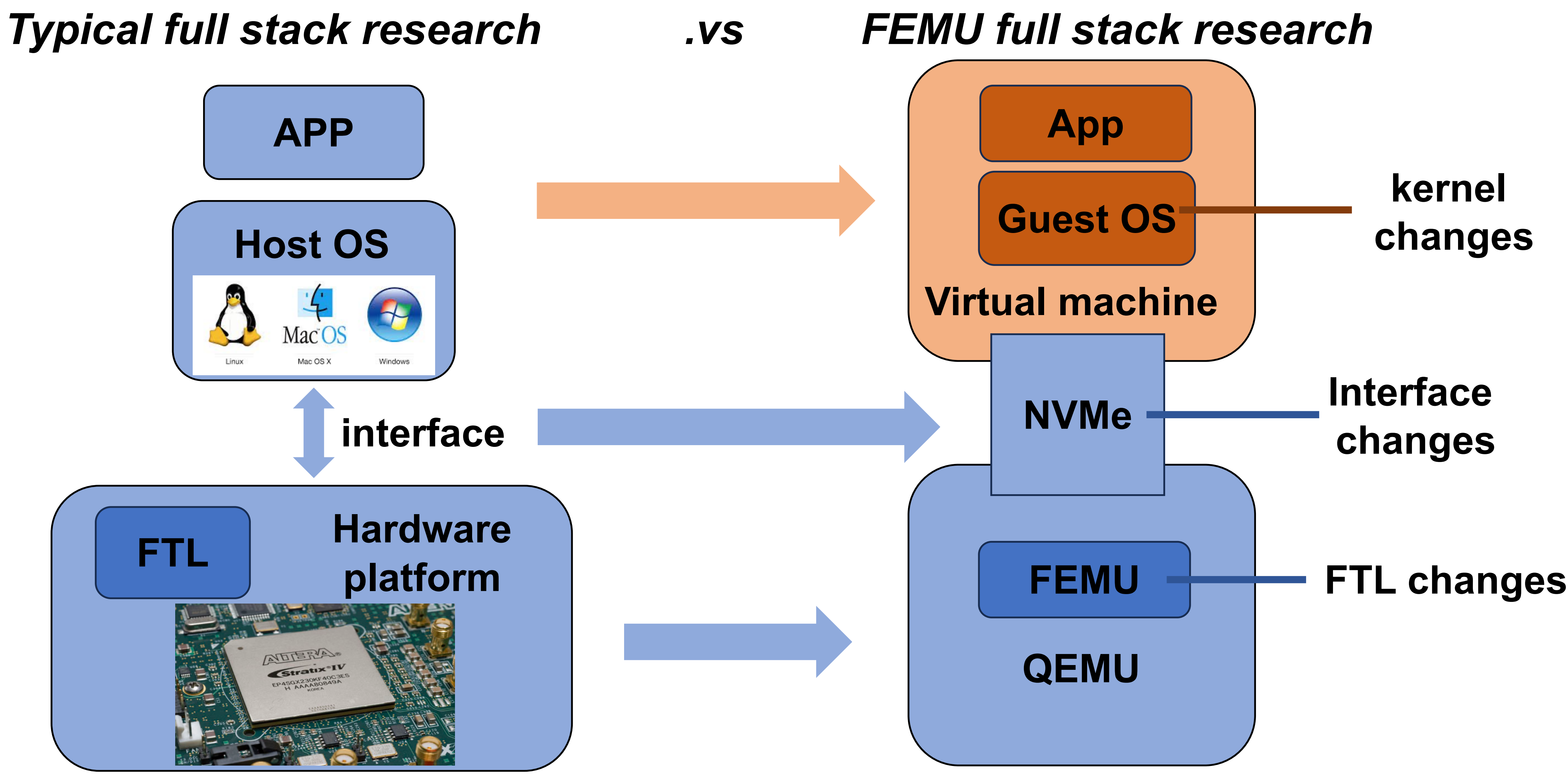


3. The CASE of FEMU

Typical full stack research .vs FEMU full stack research

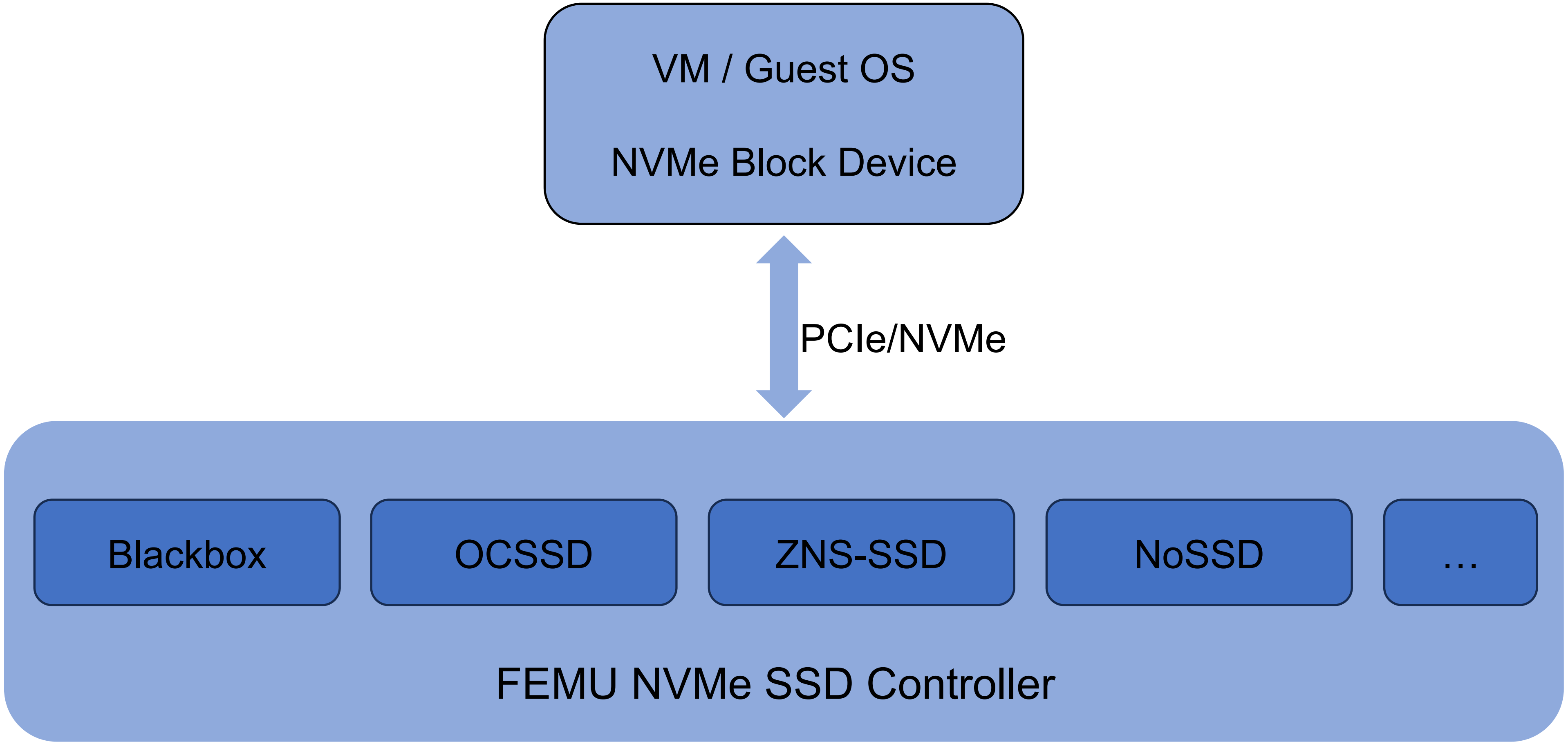


3. The CASE of FEMU



3. The CASE of FEMU

Project Description



4. Newly related work in 2023

*1. **Holistic and Opportunistic Scheduling of Background I/Os in Flash-based SSDs. (TC'23).** Yu Wang, You Zhou, Fei Wu, Yu Zhong, Jian Zhou, Zhonghai Lu, Shu Li, Zhenghong Wang, Changsheng Xie*

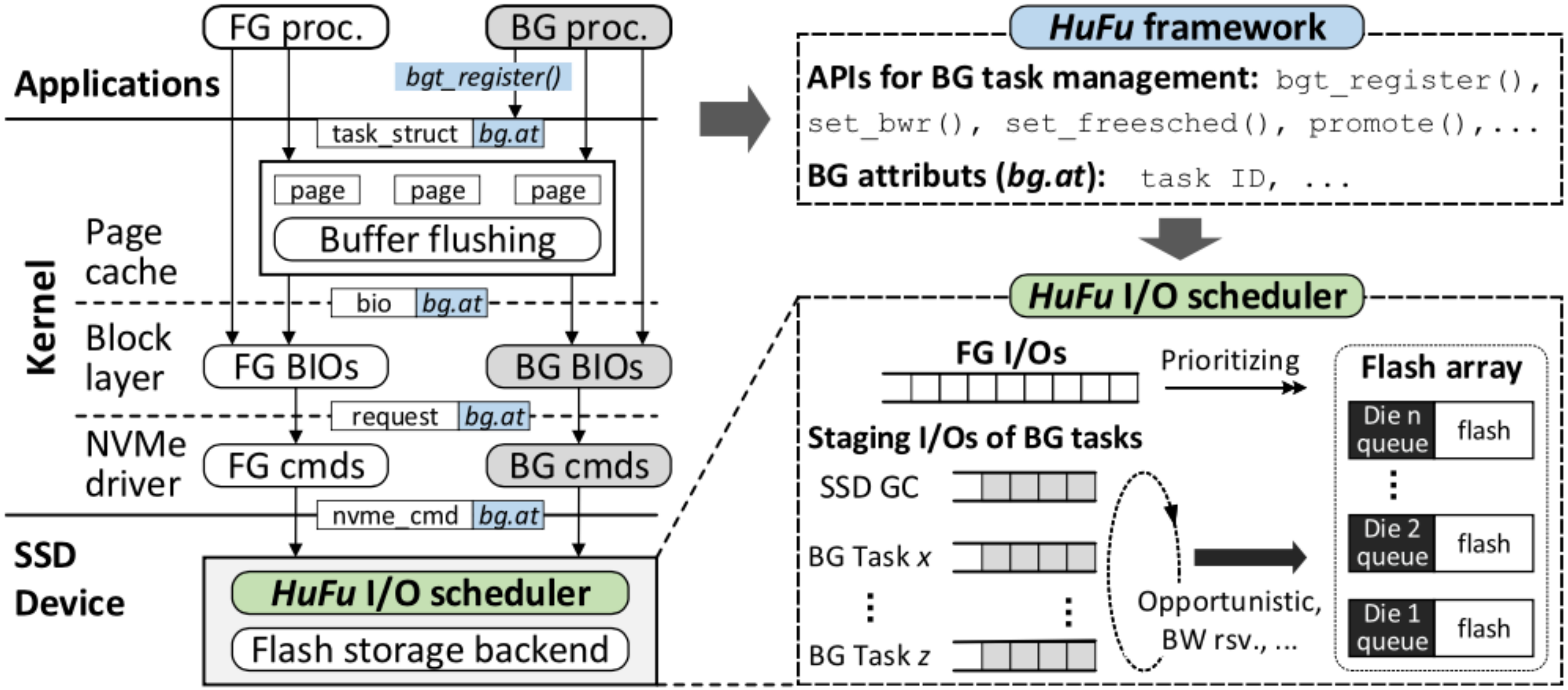


Fig. 3. Design overview of *HuFu*. The core components include a framework, which allows host software to offload I/O scheduling of BG tasks into the SSD, and an SSD-internal I/O scheduler, which prioritizes FG I/Os while processing system-wide BG I/Os opportunistically.

4. Newly related work in 2023

1. ***Holistic and Opportunistic Scheduling of Background I/Os in Flash-based SSDs. (TC'23).*** *Yu Wang, You Zhou, Fei Wu, Yu Zhong, Jian Zhou, Zhonghai Lu, Shu Li, Zhenghong Wang, Changsheng Xie*

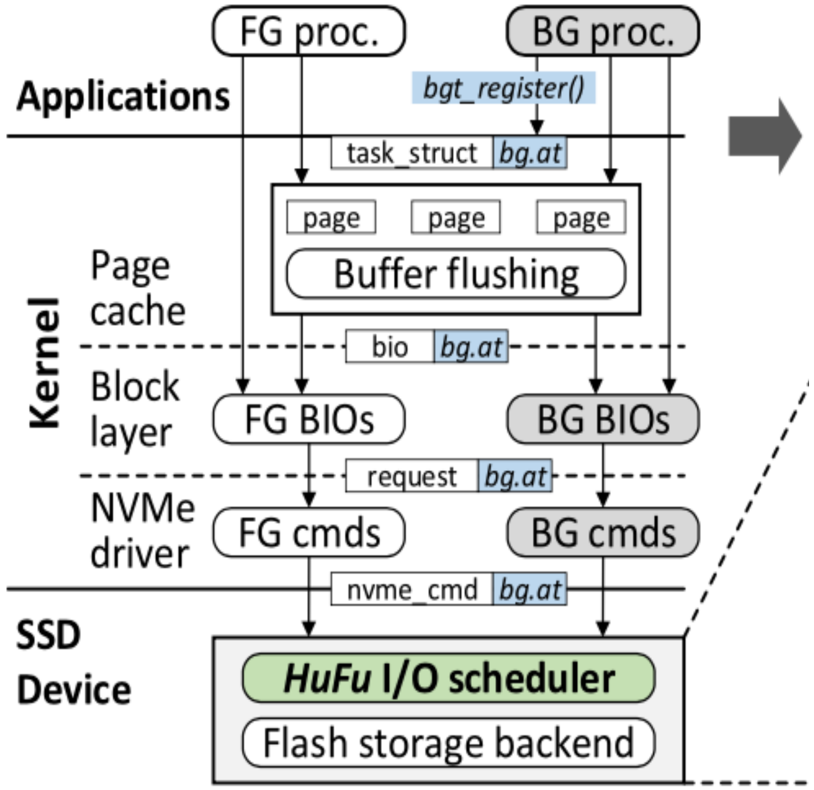


Fig. 3. Design overview of *HuFu*. The core components include a framework and an SSD-internal I/O scheduler, which prioritizes FG I/Os while processing s

FEMU

Kernel changes

FTL changes

4. Newly related work in 2023

*2. **NVMeVirt: A Versatile Software-defined Virtual NVMe Device. (FAST'23).** Sang-Hoon Kim, Jaehoon Shim, Seongyeop Jeong, Jin-Soo Kim*

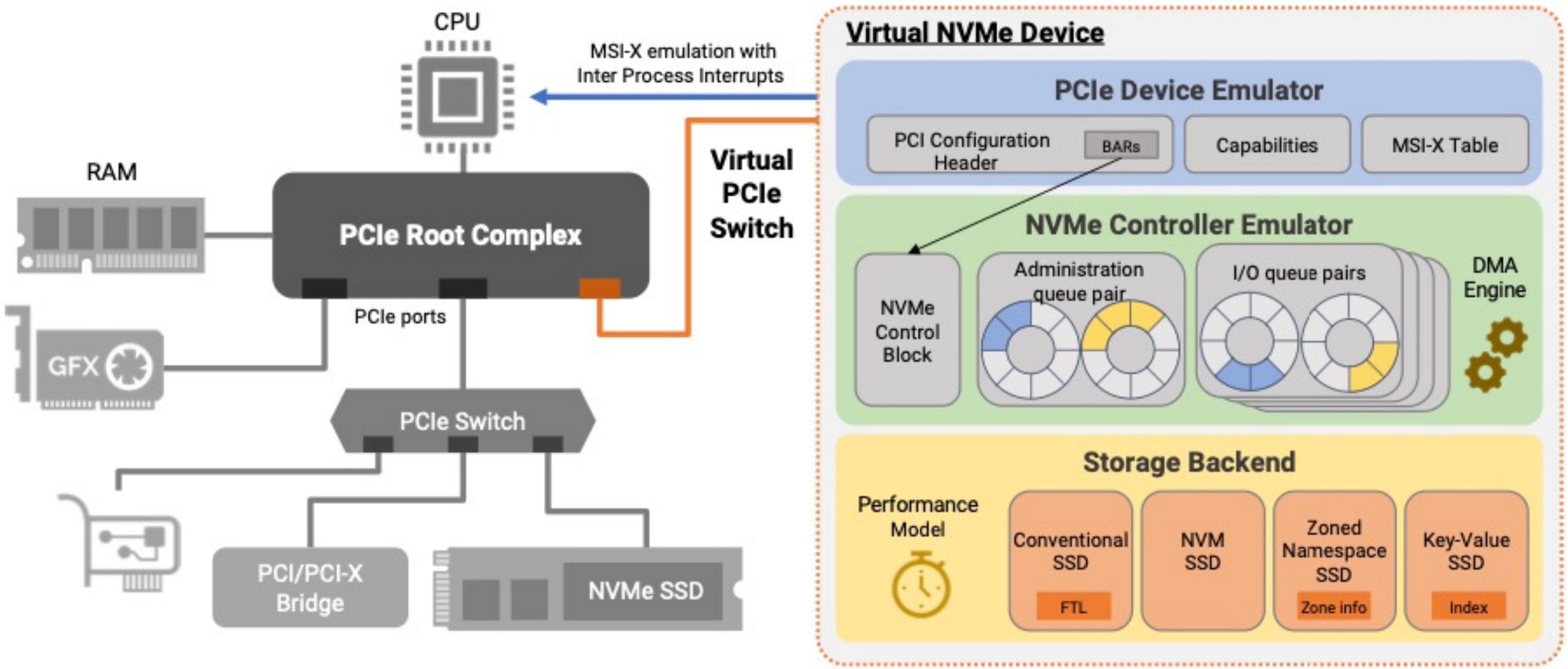


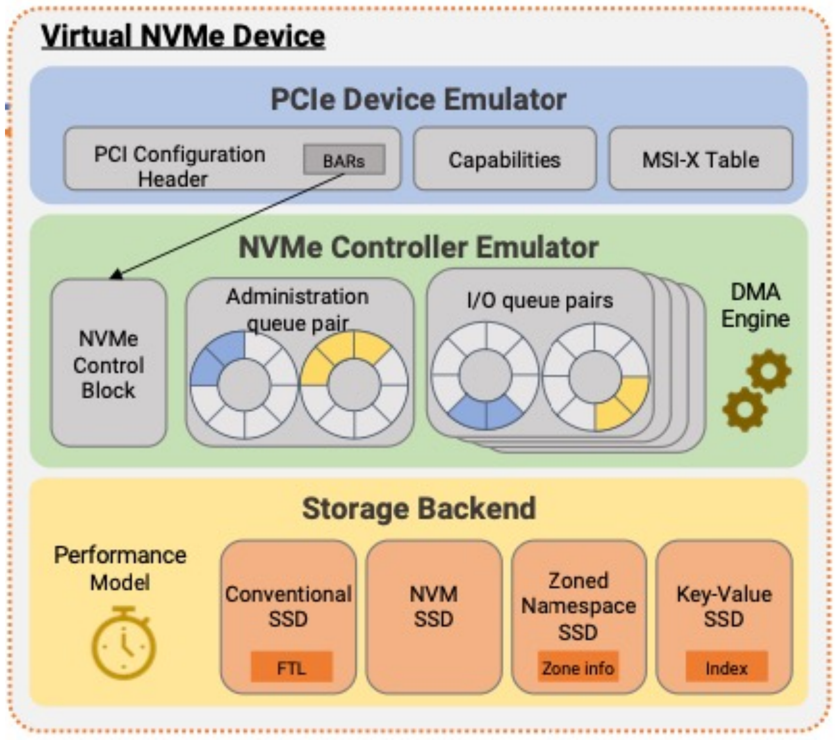
Figure 1: The overall architecture of NVMeVirt. NVMeVirt virtualizes a virtual NVMe device through the PCI bus and switch, so the device is seen as a native PCIe device from the host.

4. Newly related work in 2023

2. ***NVMeVirt: A Versatile Software-defined Virtual NVMe Device. (FAST'23). Sang-Hoon Kim**, Jaehoon Shim, Seongyeop Jeong, Jin-Soo Kim*

FEMU NVMe interface

.VS



4. Newly related work in 2023

3. ***DECC: Differential ECC for Read Performance Optimization on High-Density NAND Flash Memory.*** *(ASPDAC '23). Yunpeng Song, Yina Lv, Liang Shi*

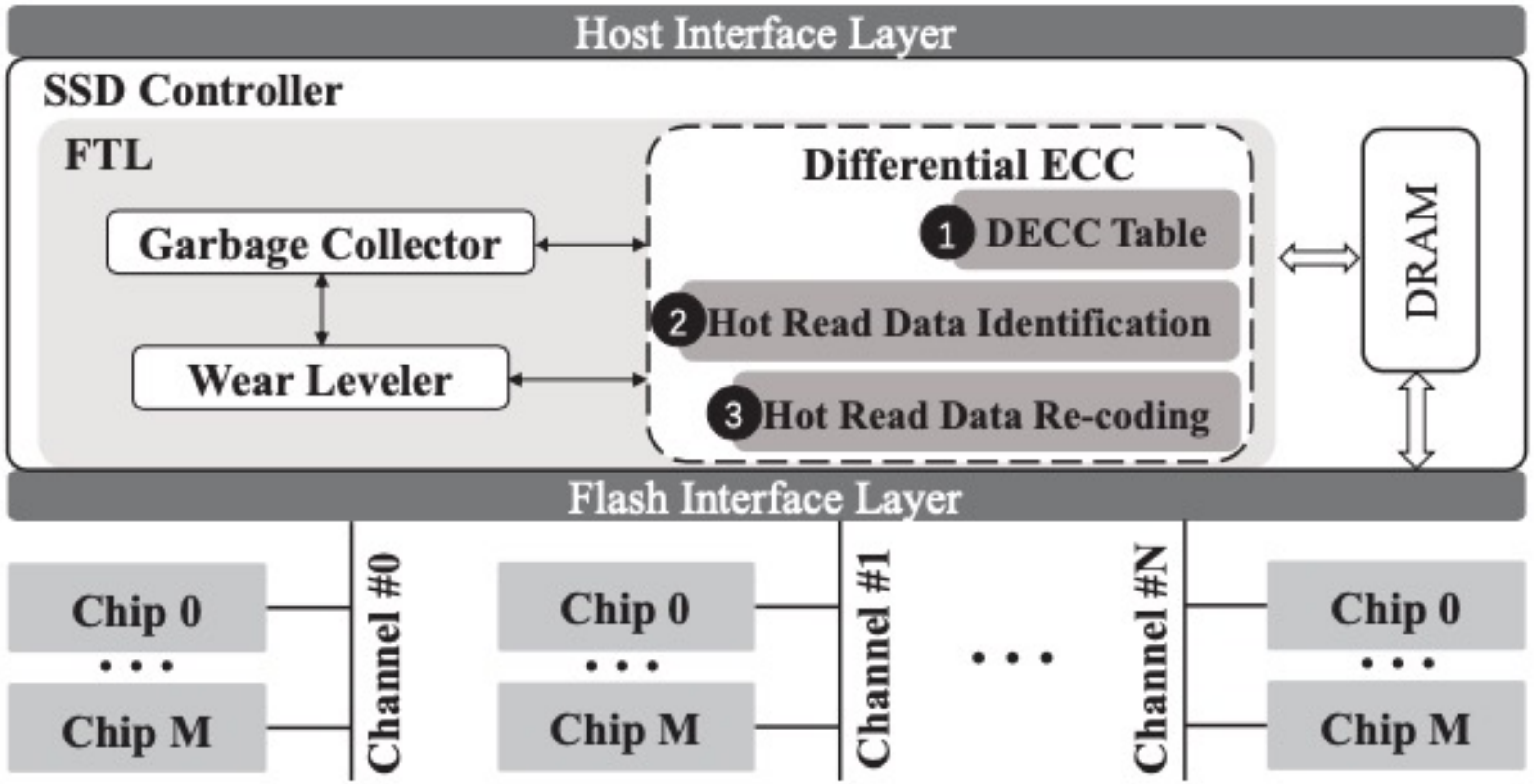


Figure 4: Overview of differential ECC.

4. Newly related work in 2023

*3. **DECC: Differential ECC for Read Performance Optimization on High-Density NAND Flash Memory.** (ASPDAC '23). Yunpeng Song, Yina Lv, Liang Shi*

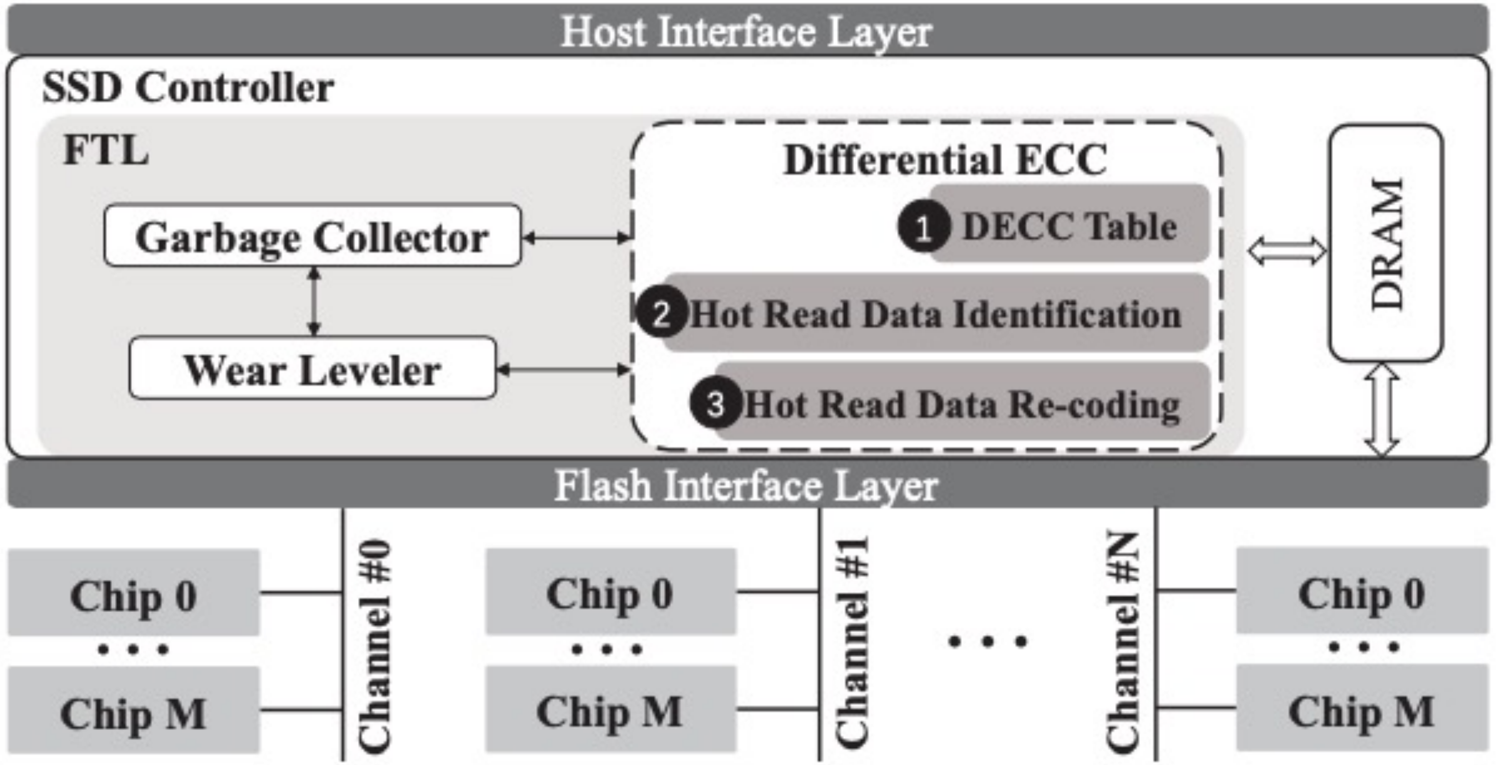


Figure 4: Overview of differential ECC.

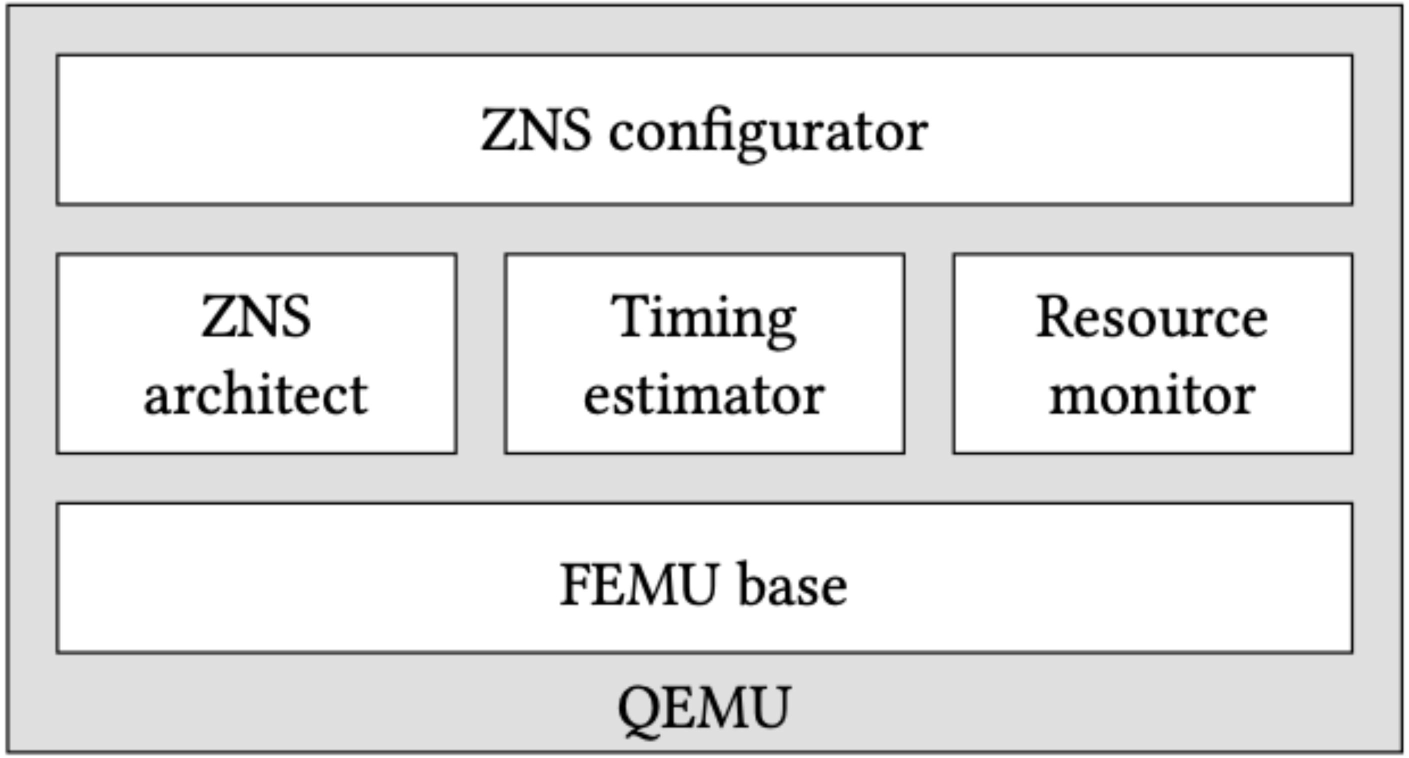
FEMU

Kernel changes

FTL changes

4. Newly related work in 2023

4. **ConfZNS : A Novel Emulator for Exploring Design Space of ZNS. (SYSTOR'23).** *Inho Song, Myounghoon Oh, Bryan S. Kim, Seehwan Yoo, Jaedong Lee, Jongmoo Choi*



An advance ZNS emulator based on FEMU

Figure 1: Key components of ConfZNS.

4. Newly related work in 2023

*5. **Optimizing Data Migration for Garbage Collection in ZNS SSDs. (DATE'23).** Zhenhua Tan, Linbo Long, Renping Liu, Congming Gao, Yi Jiang, Yan Liu*

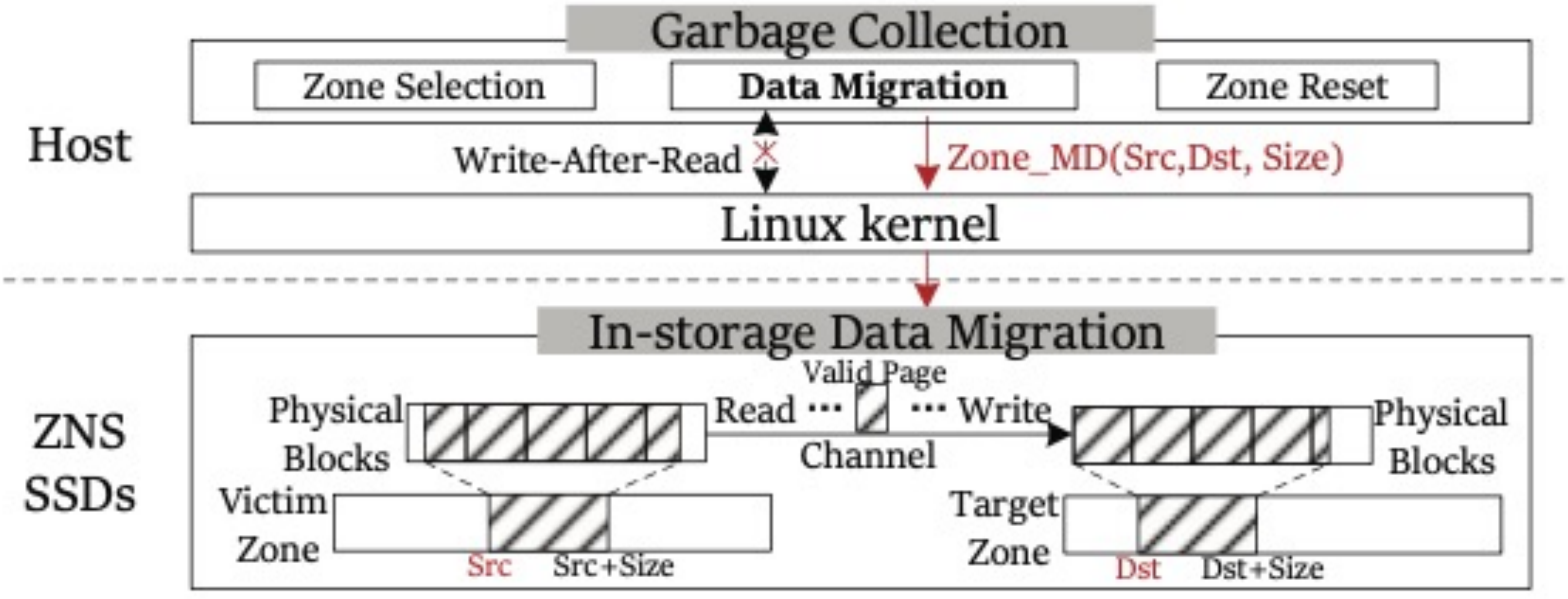


Fig. 1. The method of in-storage data migration.

4. Newly related work in 2023

*5. **Optimizing Data Migration for Garbage Collection in ZNS SSDs. (DATE'23).** Zhenhua Tan, Linbo Long, Renping Liu, Congming Gao, Yi Jiang, Yan Liu*

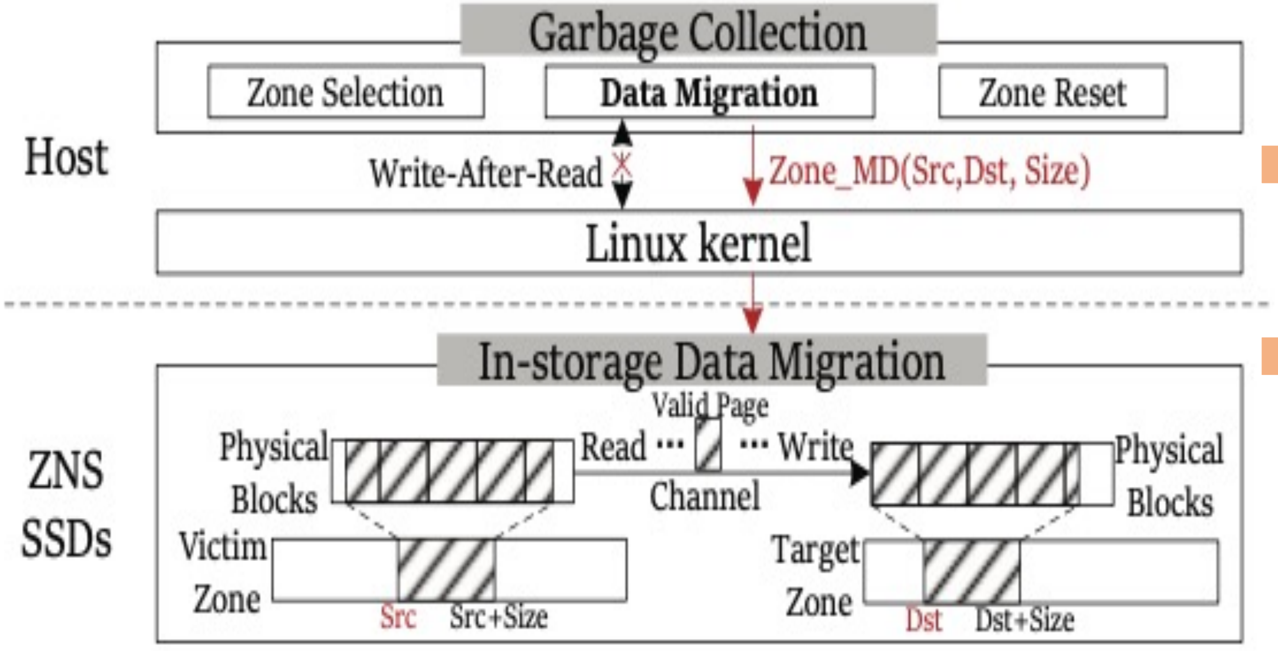


Fig. 1. The method of in-storage data migration.

FEMU

Kernel changes

Interface changes

4. Newly related work in 2023

6. ***Design of a High-Performance, High-Endurance Key-Value SSD for Large-Key Workloads**. (IEEE CAL'23).*
Chanyoung Park, Chun-Yi Liu, Kyungtae Kang, Mahmut Kandemir, Wonil Choi

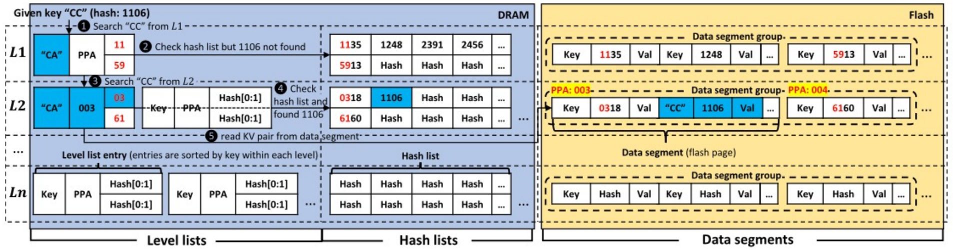


Fig. 4. Internal architecture of LK-SSD.

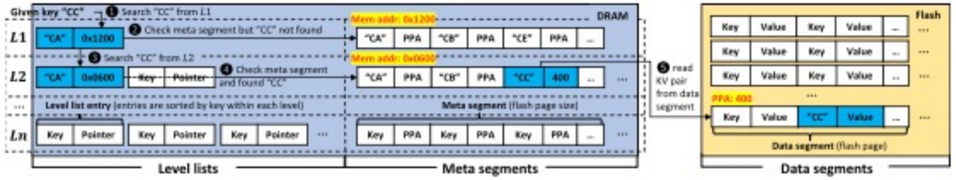


Fig. 2. Internal architecture of PinK.

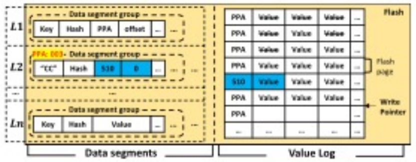


Fig. 5. Relocating a set of values to the value log in LK-SSD+.

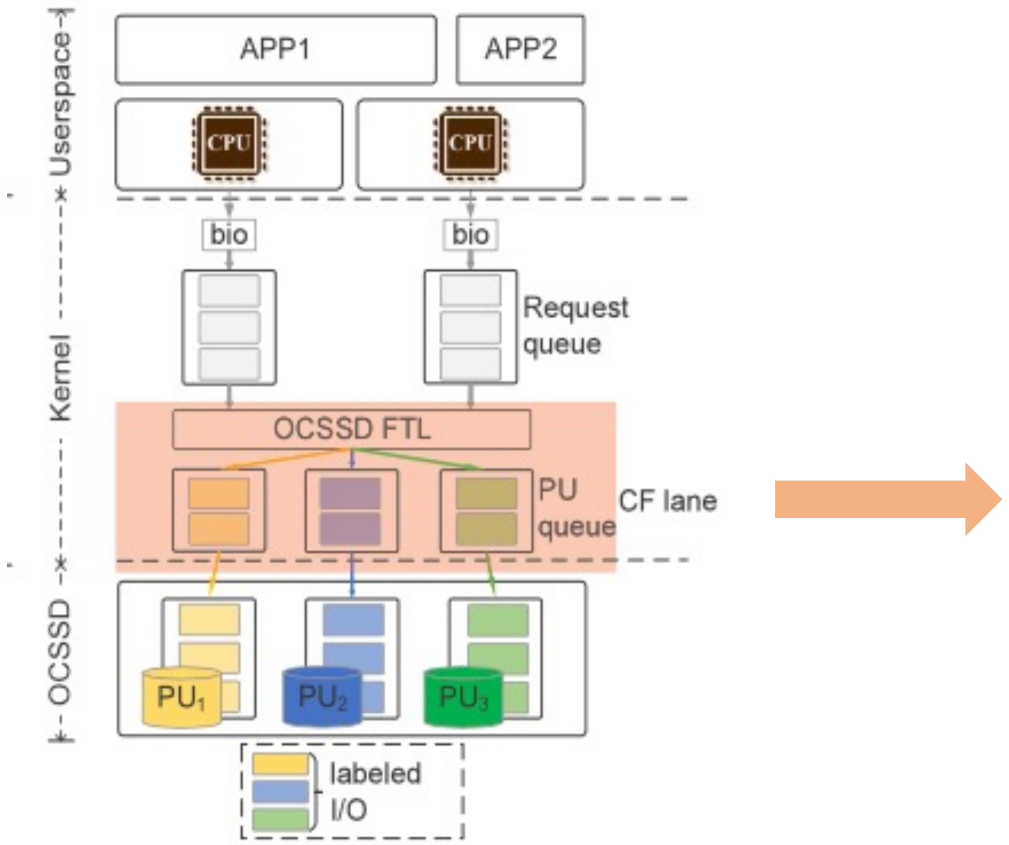
Emulate three architecture on

FEMU,

Compare their performance

4. Newly related work in 2023

7. ***CFIO: A conflict-free I/O mechanism to fully exploit internal parallelism for Open-Channel SSDs. (JSA'23).***
Jinbin Zhu, Liang Wang, Limin Xiao, Lei Liu, Guangjun Qi



(b) The optimized I/O stack with CF lane.

FEMU

FTL changes:

Optimized I/O stack with CF line based on original FEMU OC-SSD

4. Newly related work in 2023

*8**. An Efficient F2FS GC Scheme for Improving I/O Latency of Foreground Application. (ICCE'23).** Manjong Lee, Jonggyu Pa, Young Ik Eom*

IV. EVALUATION

A. Experimental Setup

Table 1. Experiment Setup

Host Spec.	CPU: Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz Memory: DDR4 64GB OS: Ubuntu 18.04.6 LTS
VM Spec.	Emulator: FEMU-v5.2 CPU: 4 SMP CPU Memory: 4GB OS: Ubuntu 20.04.4 LTS Kernel Version: Kernel v5.17 Storage: 12GB ZNS SSD

The detailed experiment setup is described in Table 1. To emulate ZNS storage, we use FEMU [15] with the default setting. To test various situations, we set various zone size from 16MB to 128MB. We use FIO [16] as the background application issuing write operations. The background application repeatedly creates/deletes 32KB ~ 64KB small files and 256KB ~ 1MB big files. For the foreground application, we use a program that issues 4KB write operations followed by fsync every 30ms.

Emulate different zone size on

FEMU,

Compare their performance

5. Future work

- 1. make FEMU running on my computer
- 2. Read the Friendly Source Code of FEMU
- 3. Find research direction

Thank you for listening

Presented by Chen Haodong