TU Graz

SCIENCE
PASSION
TECHNOLOGY

An Analysis of HMB-based SSD Rowhammer

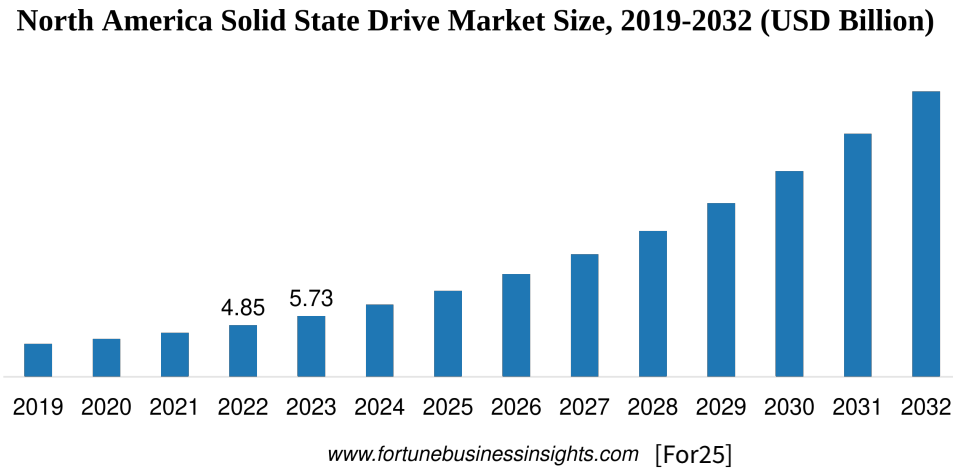
Jonas Juffinger

Graz University of Technology, Graz, Austria

uASC 2025

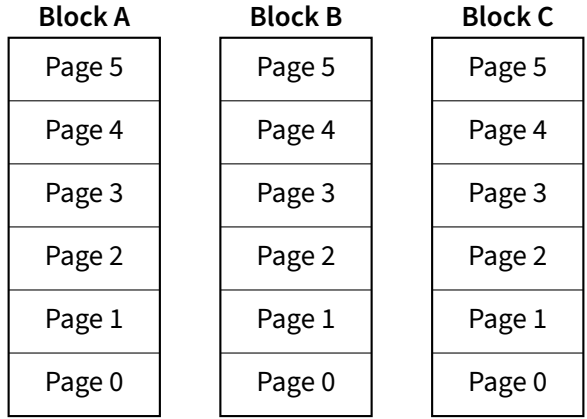
› isec.tugraz.at

Solid State Drives

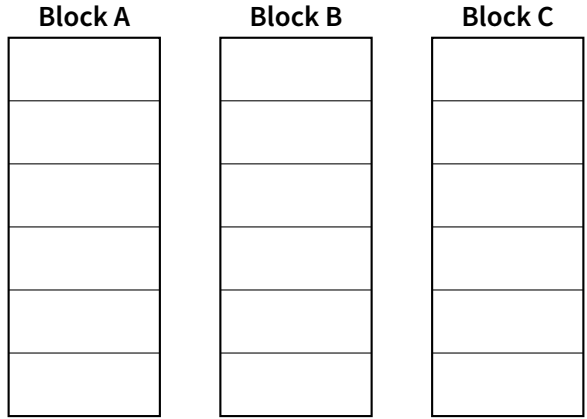


Background

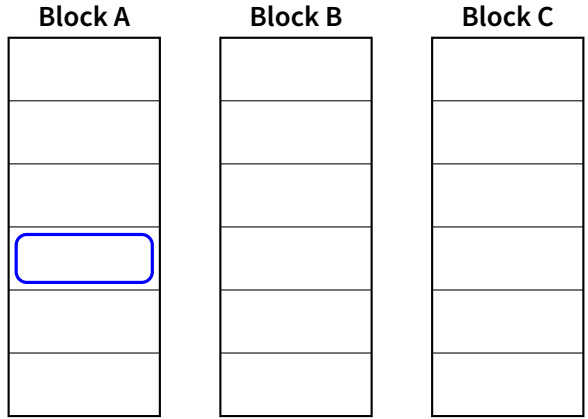
Flash Memory



Flash Operations: Read, Program, Erase

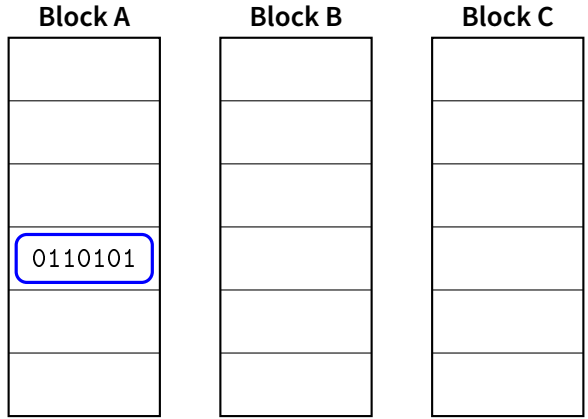


Flash Operations: Read, Program, Erase



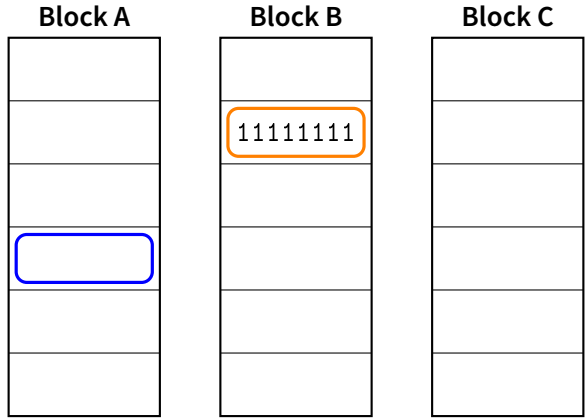
Read Block A, Page 2

Flash Operations: Read, Program, Erase



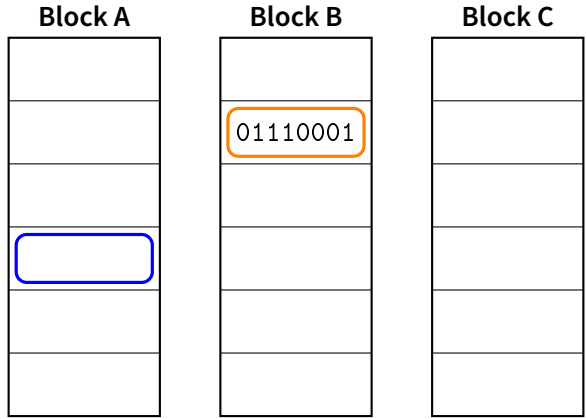
Read Block A, Page 2

Flash Operations: Read, Program, Erase



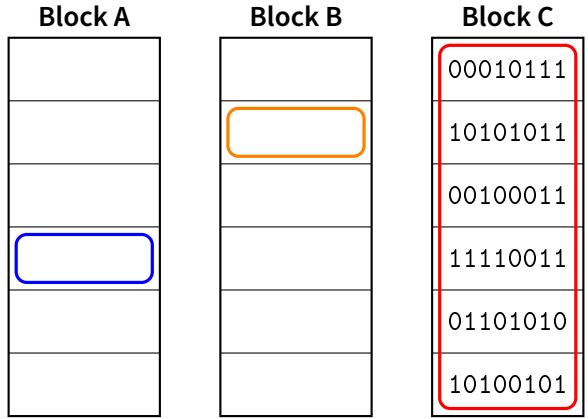
Program Block B, Page 4

Flash Operations: Read, Program, Erase



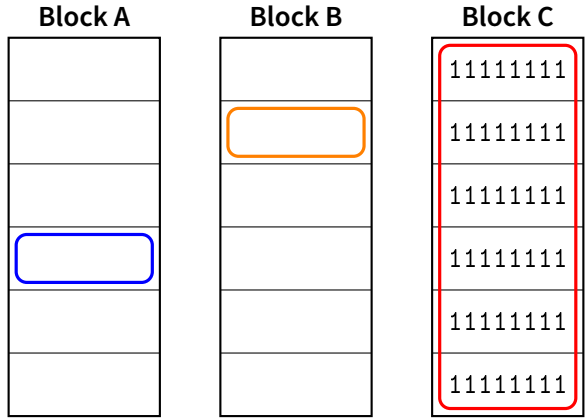
Program Block B, Page 4

Flash Operations: Read, Program, Erase



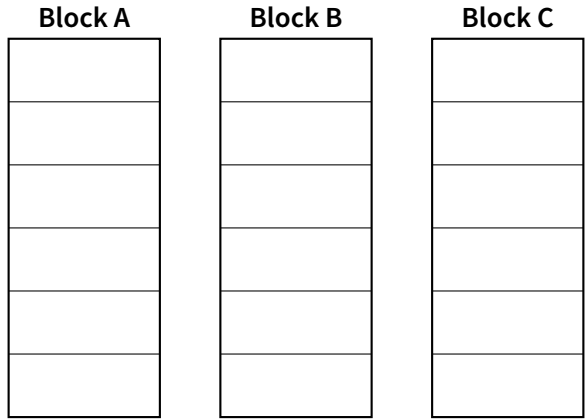
Erase Block C (all 6 pages)

Flash Operations: Read, Program, Erase

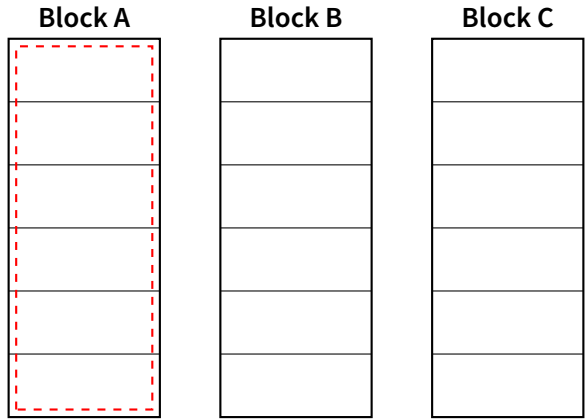


Erase Block C (all 6 pages)

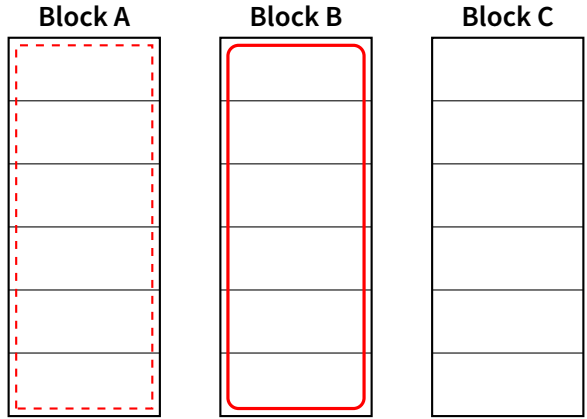
Wear-Leveling



Wear-Leveling

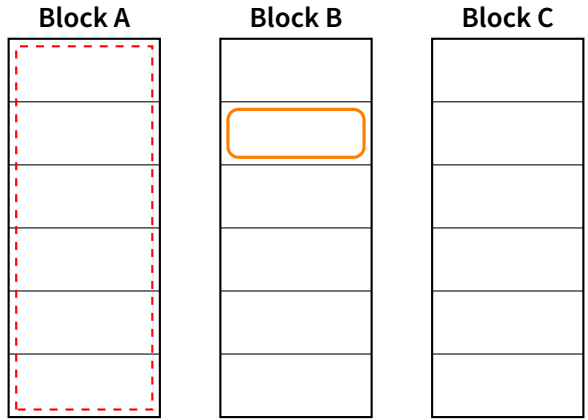


Wear-Leveling



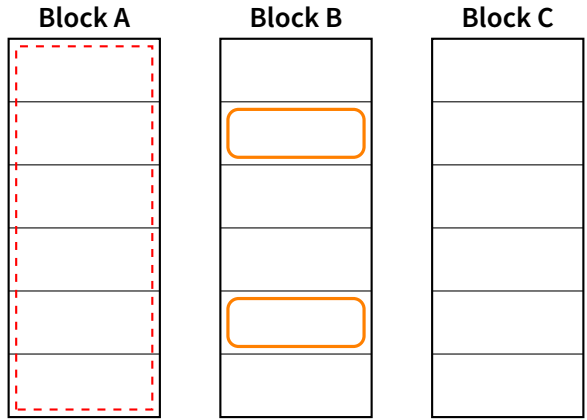
Erase Block B (all 6 pages)

Wear-Leveling



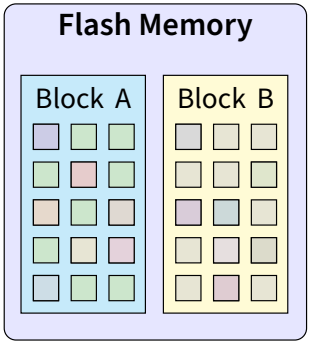
Program Block B, Page 4

Wear-Leveling



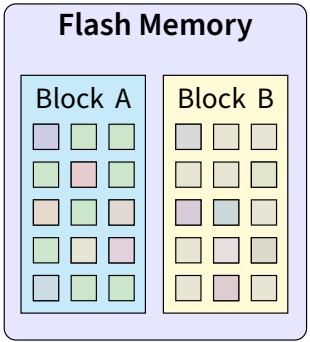
Program Block B, Page 2

Flash Translation Layer (FTL)

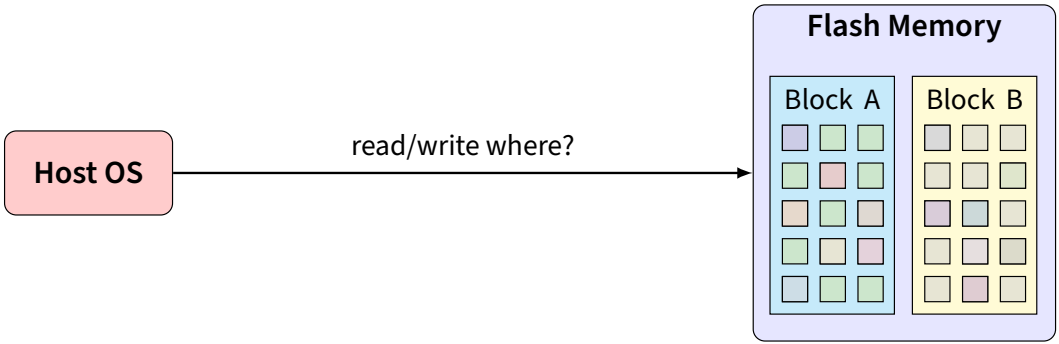


Flash Translation Layer (FTL)

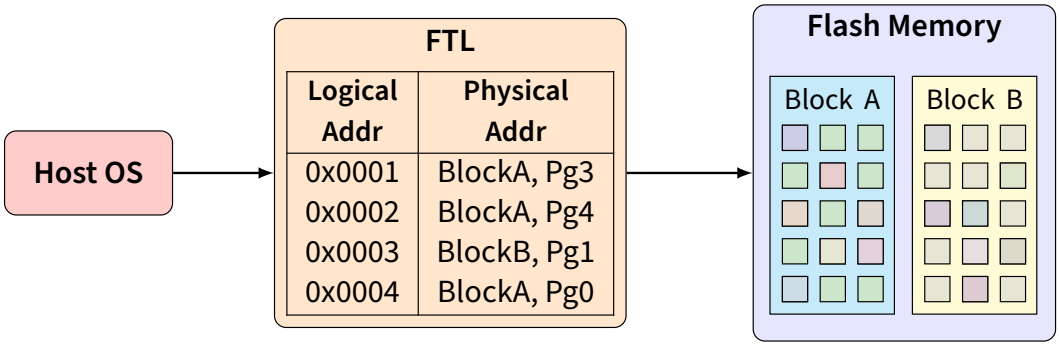
Host OS



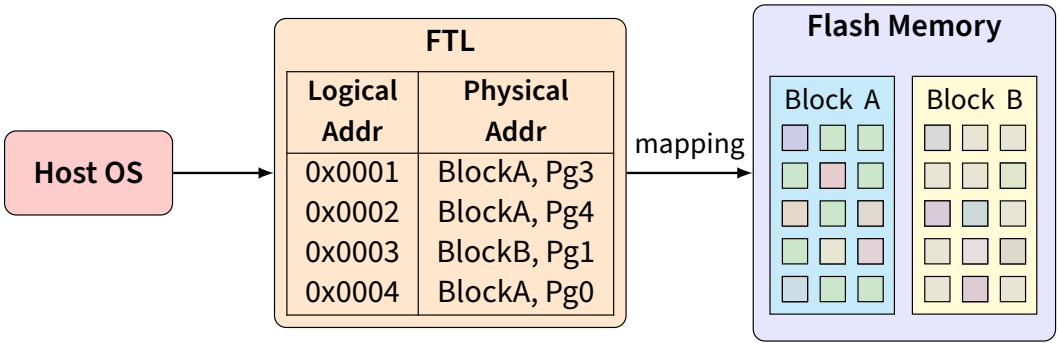
Flash Translation Layer (FTL)



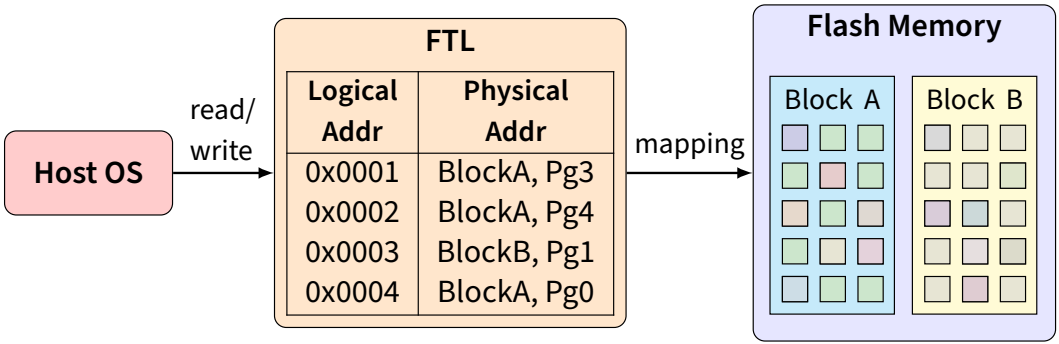
Flash Translation Layer (FTL)



Flash Translation Layer (FTL)



Flash Translation Layer (FTL)



Flash Translation Layer (FTL)

FTL

Logical Addr	Physical Addr
0x0001	BlockA, Pg3
0x0002	BlockA, Pg4
0x0003	BlockB, Pg1
0x0004	BlockA, Pg0

Flash Translation Layer (FTL)

FTL

Logical Addr	Physical Addr
0x0001	BlockA, Pg3
0x0002	BlockA, Pg4
0x0003	BlockB, Pg1
0x0004	BlockA, Pg0

- Required on each access

Flash Translation Layer (FTL)

FTL

Logical Addr	Physical Addr
0x0001	BlockA, Pg3
0x0002	BlockA, Pg4
0x0003	BlockB, Pg1
0x0004	BlockA, Pg0

- Required on each access
- Stored in flash memory

Flash Translation Layer (FTL)

FTL

Logical Addr	Physical Addr
0x0001	BlockA, Pg3
0x0002	BlockA, Pg4
0x0003	BlockB, Pg1
0x0004	BlockA, Pg0

- Required on each access
- Stored in flash memory
- Cache

FTL Caching

FTL

Logical Addr	Physical Addr
0x0001	BlockA, Pg3
0x0002	BlockA, Pg4
0x0003	BlockB, Pg1
0x0004	BlockA, Pg0

FTL Caching

FTL

Logical Addr	Physical Addr
0x0001	BlockA, Pg3
0x0002	BlockA, Pg4
0x0003	BlockB, Pg1
0x0004	BlockA, Pg0

- “Pro” SSDs

FTL Caching

FTL

Logical Addr	Physical Addr
0x0001	BlockA, Pg3
0x0002	BlockA, Pg4
0x0003	BlockB, Pg1
0x0004	BlockA, Pg0

- “Pro” SSDs
  - DRAM chip storing the FTL

FTL Caching

FTL

Logical Addr	Physical Addr
0x0001	BlockA, Pg3
0x0002	BlockA, Pg4
0x0003	BlockB, Pg1
0x0004	BlockA, Pg0

- "Pro" SSDs
  - DRAM chip storing the FTL
- Budget SSDs

FTL Caching

FTL

Logical Addr	Physical Addr
0x0001	BlockA, Pg3
0x0002	BlockA, Pg4
0x0003	BlockB, Pg1
0x0004	BlockA, Pg0

- "Pro" SSDs
  - DRAM chip storing the FTL
- Budget SSDs
  - Small cache in SSD controller

FTL Caching

FTL

Logical Addr	Physical Addr
0x0001	BlockA, Pg3
0x0002	BlockA, Pg4
0x0003	BlockB, Pg1
0x0004	BlockA, Pg0

- "Pro" SSDs
  - DRAM chip storing the FTL
- Mid-range SSDs
- Budget SSDs
  - Small cache in SSD controller

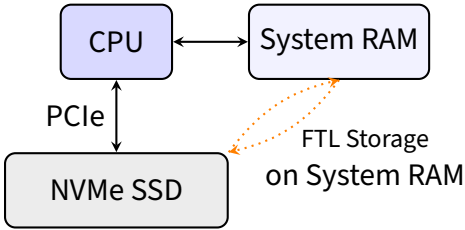
FTL Caching

FTL

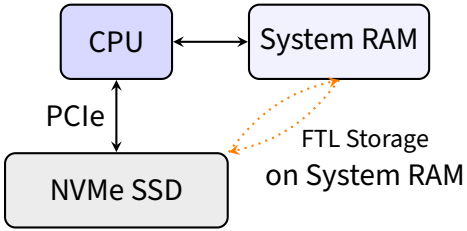
Logical Addr	Physical Addr
0x0001	BlockA, Pg3
0x0002	BlockA, Pg4
0x0003	BlockB, Pg1
0x0004	BlockA, Pg0

- "Pro" SSDs
  - DRAM chip storing the FTL
- Mid-range SSDs
  - Uses a host memory buffer
- Budget SSDs
  - Small cache in SSD controller

Host-Memory Buffer (HMB)

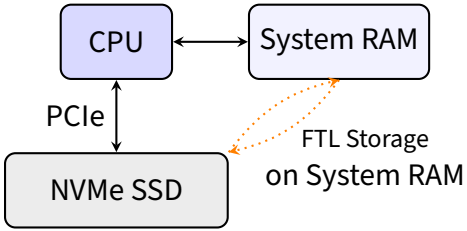


Host-Memory Buffer (HMB)



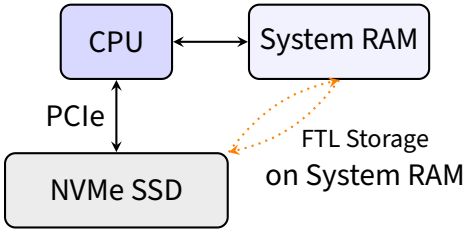
- Request memory from OS

Host-Memory Buffer (HMB)



- Request memory from OS
- Exclusive DMA access

Host-Memory Buffer (HMB)

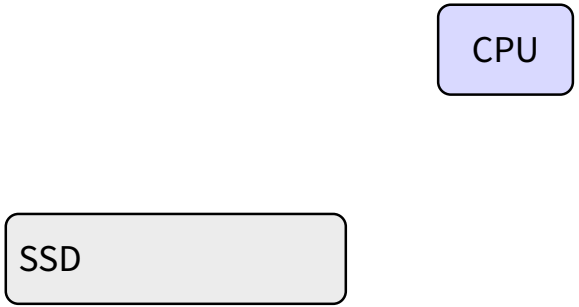


- Request memory from OS
- Exclusive DMA access
- Usage is undocumented

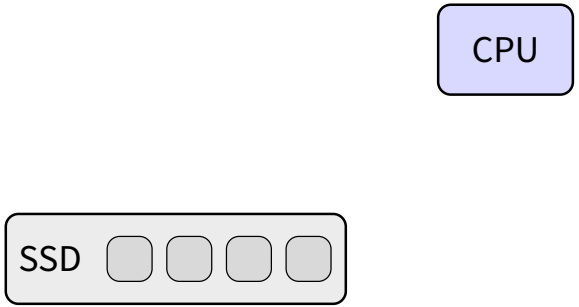
The SSD Uses the Main DRAM

What About Rowhammer?

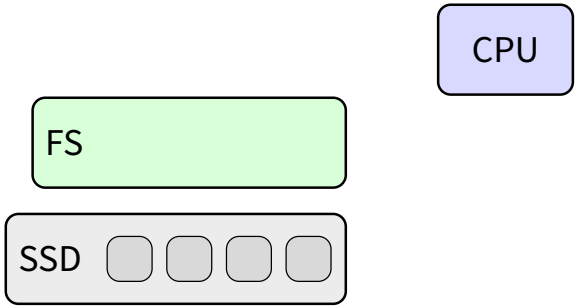
Attack – Based on work by Zhang et al. [Zha+21]



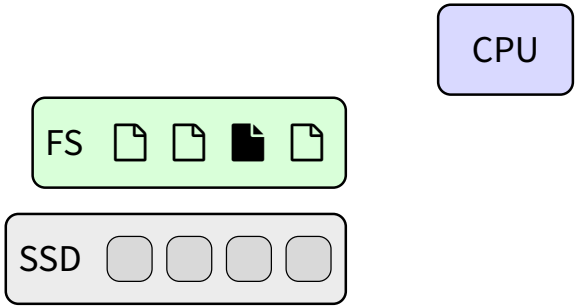
Attack – Based on work by Zhang et al. [Zha+21]



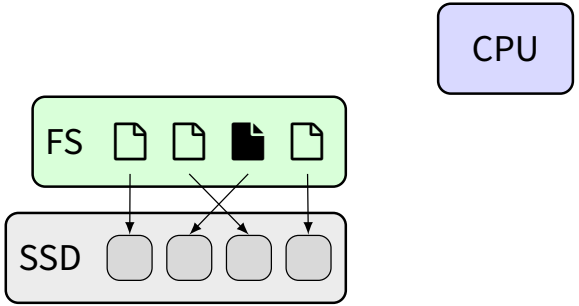
Attack – Based on work by Zhang et al. [Zha+21]



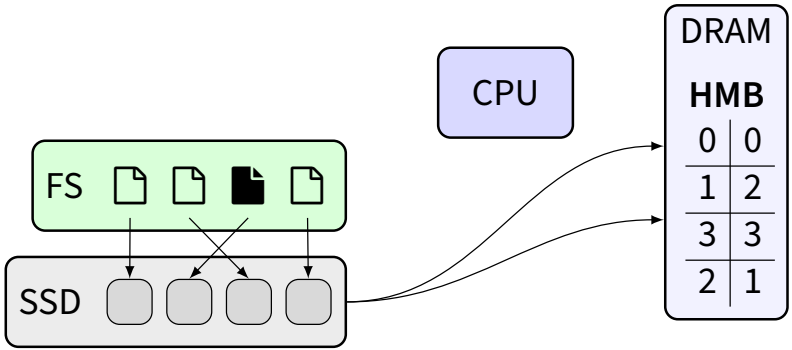
Attack – Based on work by Zhang et al. [Zha+21]



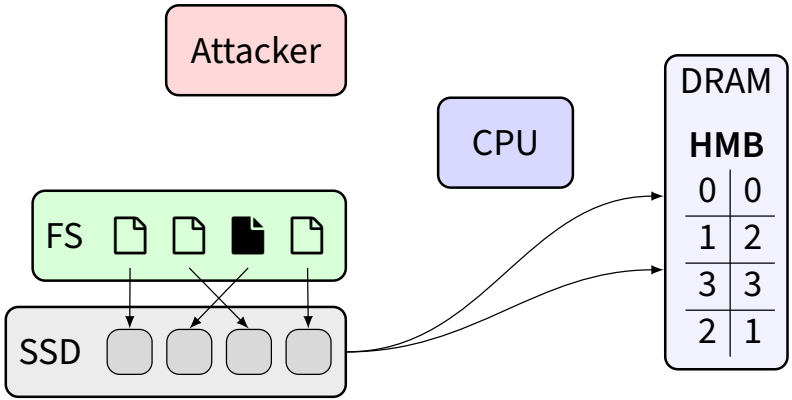
Attack – Based on work by Zhang et al. [Zha+21]



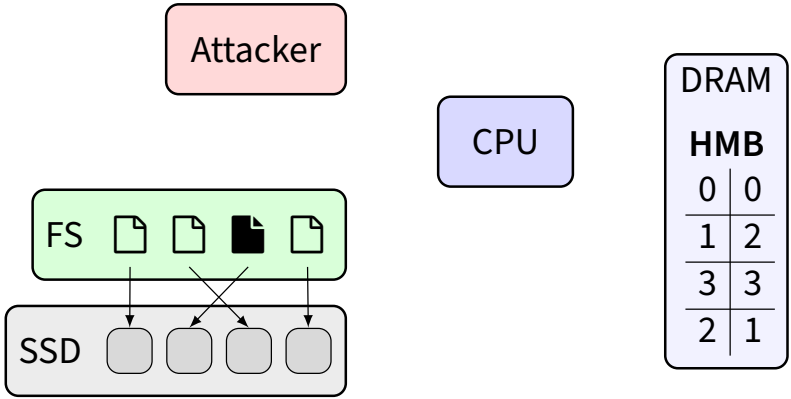
Attack – Based on work by Zhang et al. [Zha+21]



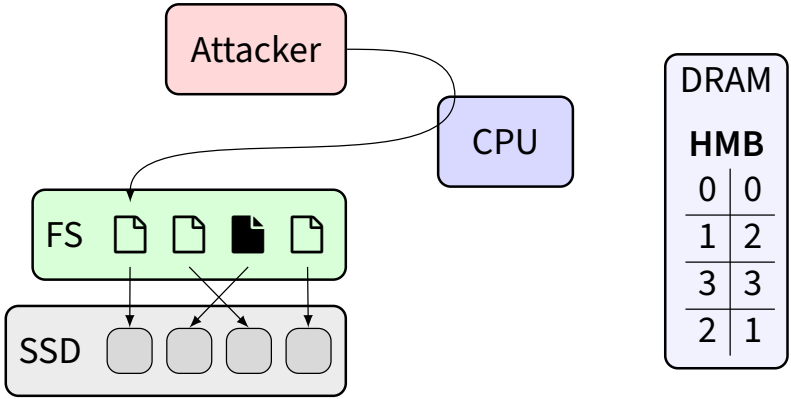
Attack – Based on work by Zhang et al. [Zha+21]



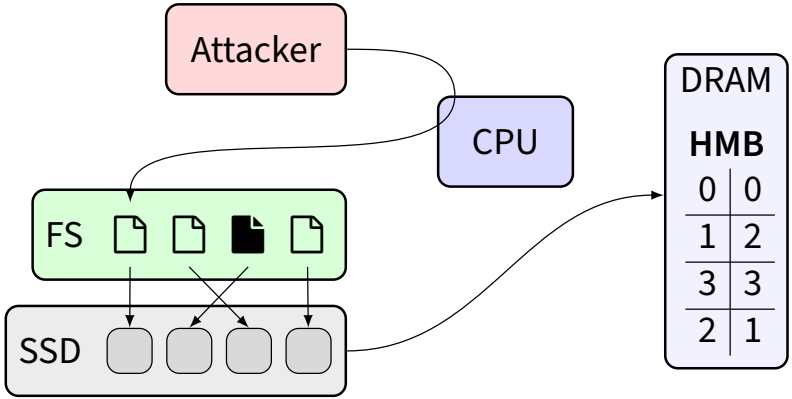
Attack – Based on work by Zhang et al. [Zha+21]



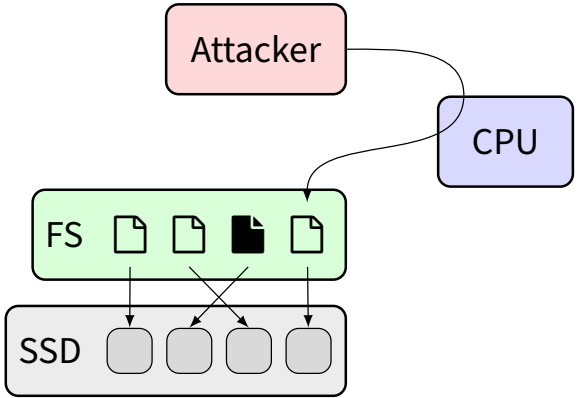
Attack – Based on work by Zhang et al. [Zha+21]



Attack – Based on work by Zhang et al. [Zha+21]



Attack – Based on work by Zhang et al. [Zha+21]

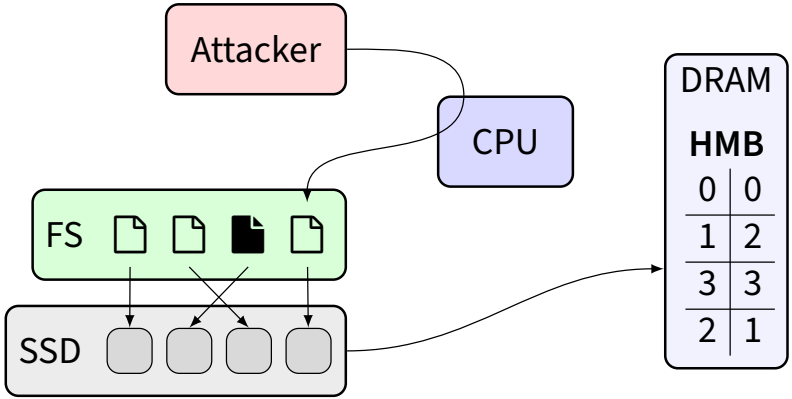


DRAM

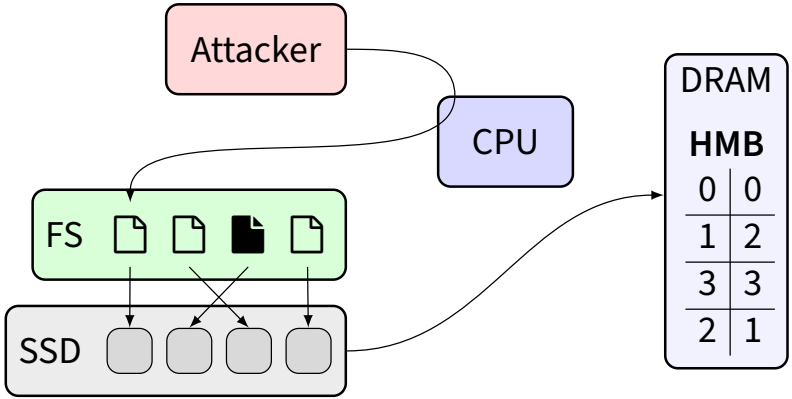
HMB

0	0
1	2
3	3
2	1

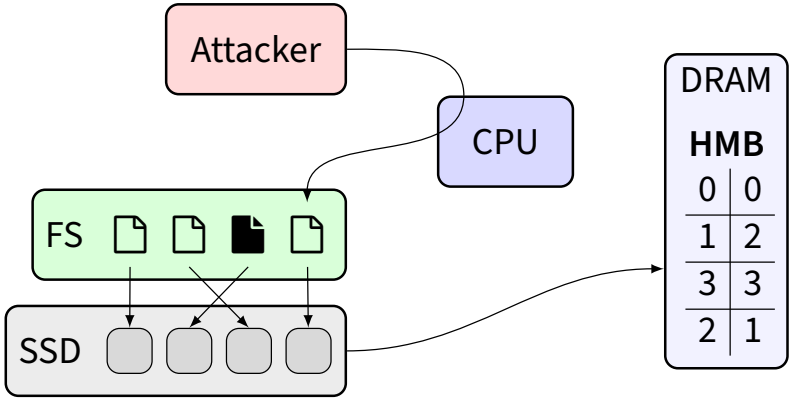
Attack – Based on work by Zhang et al. [Zha+21]



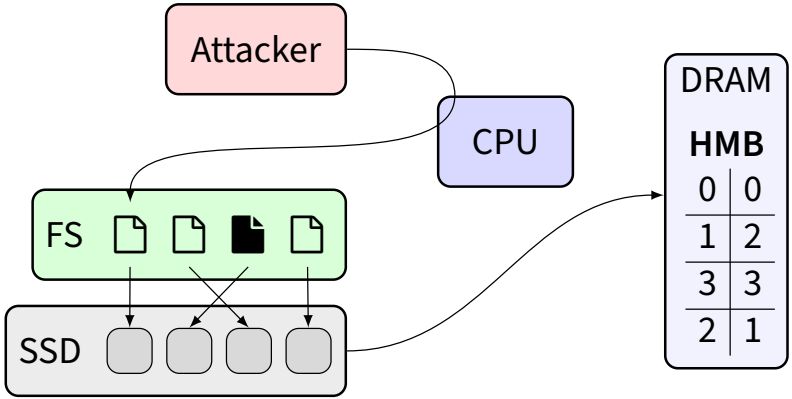
Attack – Based on work by Zhang et al. [Zha+21]



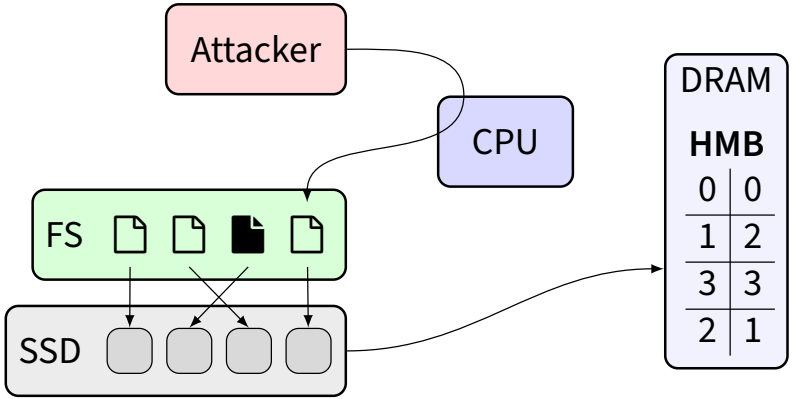
Attack – Based on work by Zhang et al. [Zha+21]



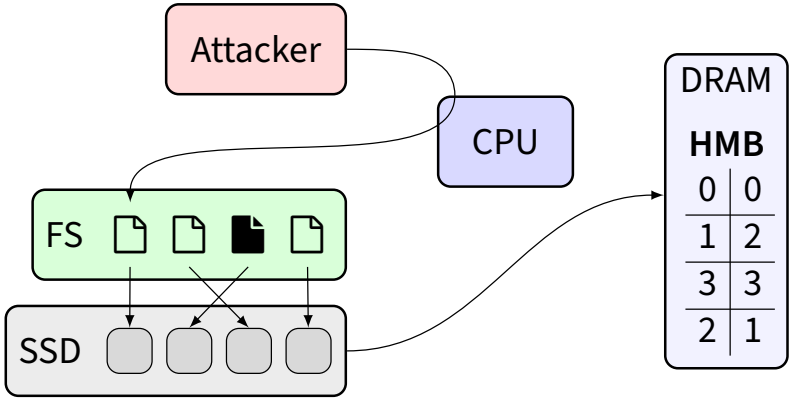
Attack – Based on work by Zhang et al. [Zha+21]



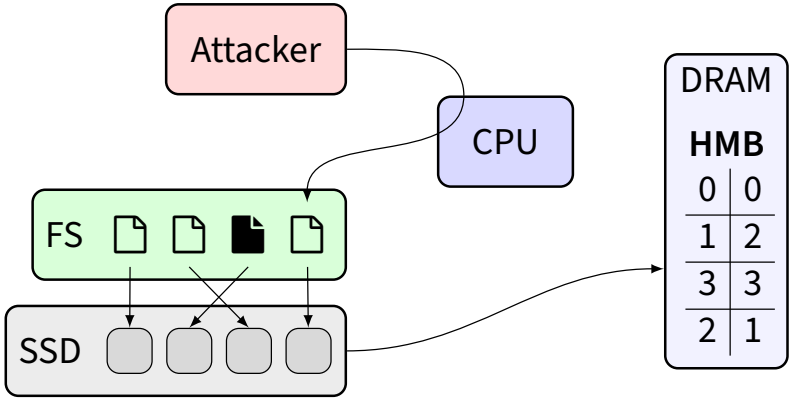
Attack – Based on work by Zhang et al. [Zha+21]



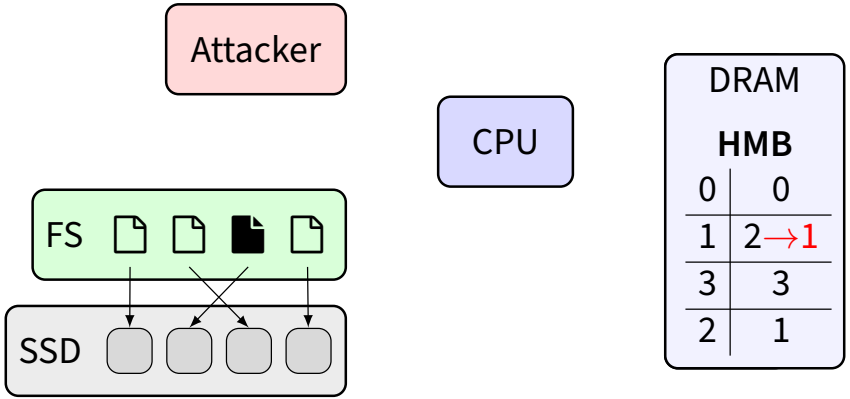
Attack – Based on work by Zhang et al. [Zha+21]



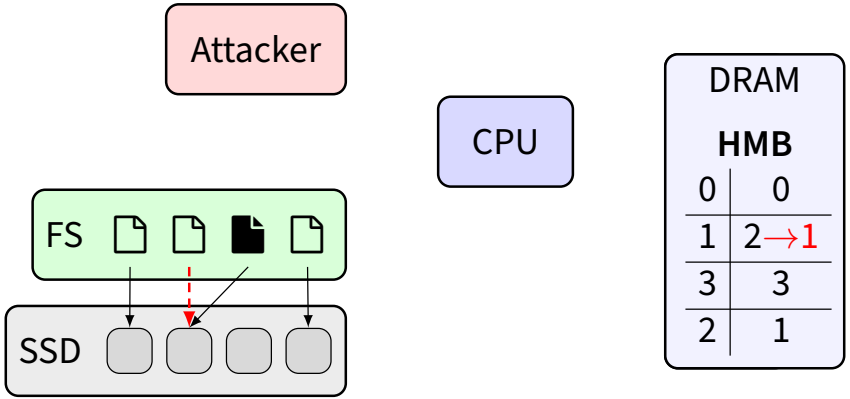
Attack – Based on work by Zhang et al. [Zha+21]



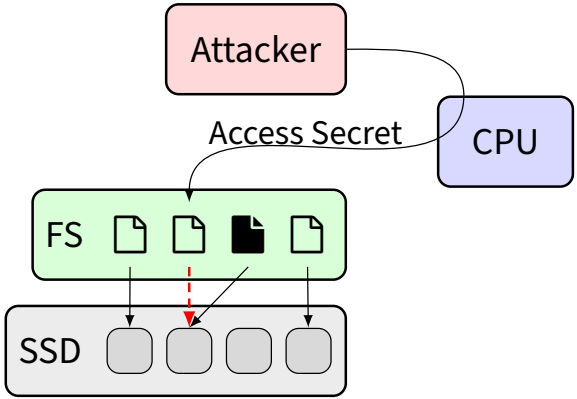
Attack – Based on work by Zhang et al. [Zha+21]



Attack – Based on work by Zhang et al. [Zha+21]



Attack – Based on work by Zhang et al. [Zha+21]



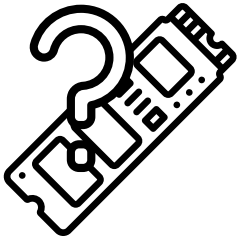
DRAM

HMB

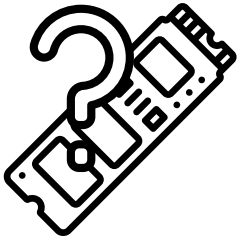
0	0
1	2→1
3	3
2	1

Research Questions

Research Questions

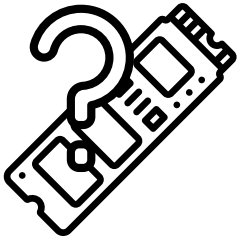


Research Questions



- What happens if bits flip in the HMB?

Research Questions



- What happens if bits flip in the HMB?
- Can we make the SSD hammer the HMB?

The SSDs used in our experiments

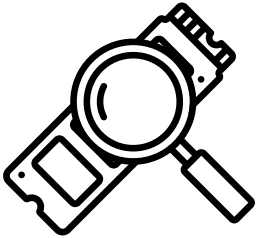
SSD	PCIe	Size	HMB Size	IOPS[1]
Samsung 990 EVO	4.0	2 TB	64 MB	680 000
Lexar NM790	4.0	2 TB	40 MB	1 000 000
Samsung 980	3.0	1 TB	64 MB	500 000

[1] Read IOPS with maximum queue depth as advertised by the manufacturer.

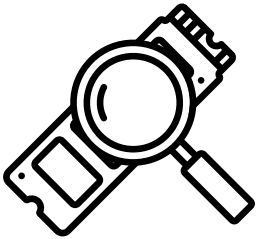
Tooling

Tooling

- IOMMU

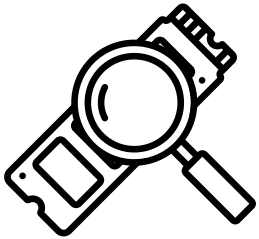


Tooling



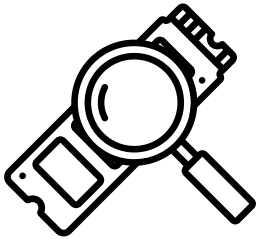
- IOMMU
  - Virtual memory but for DMA

Tooling



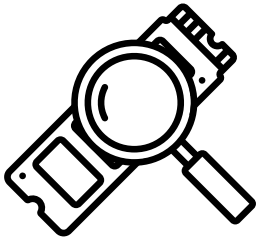
- IOMMU
  - Virtual memory but for DMA
  - Observe HMB accesses using the accessed bit

Tooling



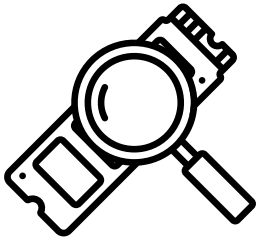
- IOMMU
  - Virtual memory but for DMA
  - Observe HMB accesses using the accessed bit
  - Only possible on AMD Zen 3

Tooling



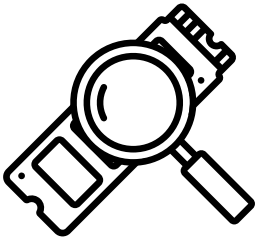
- IOMMU
  - Virtual memory but for DMA
  - Observe HMB accesses using the accessed bit
  - Only possible on AMD Zen 3
- Modified kernel

Tooling



- IOMMU
  - Virtual memory but for DMA
  - Observe HMB accesses using the accessed bit
  - Only possible on AMD Zen 3
- Modified kernel
  - Get physical address of the HMB

Tooling



- IOMMU
  - Virtual memory but for DMA
  - Observe HMB accesses using the accessed bit
  - Only possible on AMD Zen 3
- Modified kernel
  - Get physical address of the HMB
  - Get IOMMU page tables

Physical HMB Addresses


```
# cat /sys/block/nvme0n1/device/hmb_addresses
0 cfc00000 400000
1 cf800000 400000
2 cf400000 400000
3 cf000000 400000
...
12 ccc00000 400000
13 cc800000 400000
14 cc400000 400000
15 cc000000 400000
```

Listing: `/sys/block/nvmeXn1/device/hmb_addresses`

HMB IOMMU Page Tables


```
# cat /sys/kernel/debug/iommu/amd/iommu00/0000:01:00.0/page_tables
...
0xcc000000| 0x6000000101e7b421 0x6000000101e9f221 0x7000000113000021
0xcc001000| 0x6000000101e7b421 0x6000000101e9f221 0x7000000113001021
0xcc002000| 0x6000000101e7b421 0x6000000101e9f221 0x7000000113002021
0xcc003000| 0x6000000101e7b421 0x6000000101e9f221 0x7000000113003021
0xcc004000| 0x6000000101e7b421 0x6000000101e9f221 0x7000000113004021
...
```

Listing: `/sys/kernel/debug/iommu/amd/iommu00/0000:01:00.0/page_tables`

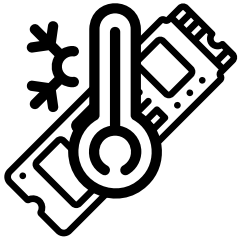
HMB Content


```
3f73000: 30ae30d030ae30c 30ae30f030ae30e 30aa3100036c500 30aa312030aa311
3f73020: 30ae310030aa313 30ae312030ae311  1eeae9030ae313 30aa315030aa314
3f73040: 30aa317030aa316 30ae315030ae314 30ae317030ae316 50aa30c001e5677
3f73080: 50aa30e050aa30d 50ae30c050aa30f 50ae30e050ae30d 3d1ee8b050ae30f
3f73080: 50aa311050aa310 50aa313050aa312 50ae311050ae310 50ae313050ae312
3f730a0: 50aa31403f9c162 50aa316050aa315 50ae314050aa317 50ae316050ae315
3f730c0: 3f97dfc050ae317 70aa30d070aa30c 70aa30f070aa30e 70ae30d070ae30c
3f730e0: 70ae30f070ae30e 70aa310001ef7f2 70aa312070aa311 70ae310070aa313
```

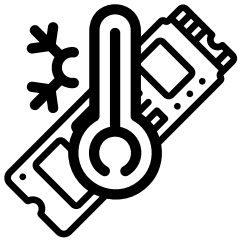
Listing: Content at HMB offset 0x3f73000 after reading 100000 random pages from the SSD.

Bit Flips in The HMB

SSD Freeze

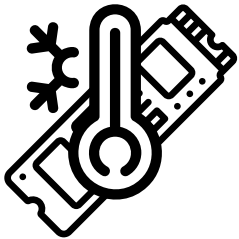


SSD Freeze



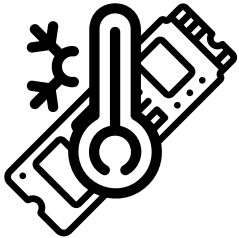
- Flipping any bit in the HMB freezes the SSD

SSD Freeze



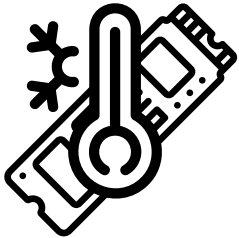
- Flipping any bit in the HMB freezes the SSD
- Reboot not enough

SSD Freeze



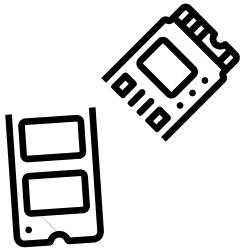
- Flipping any bit in the HMB freezes the SSD
- Reboot not enough
- **Full power cycle** required

SSD Freeze

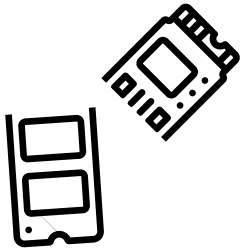


- Flipping any bit in the HMB freezes the SSD
- Reboot not enough
- **Full power cycle** required
- Can cause data corruption

Broken SSD

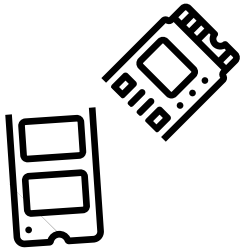


Broken SSD



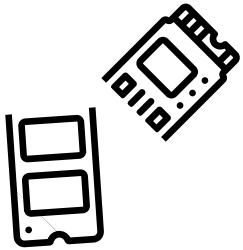
- Broke one Samsung 980

Broken SSD



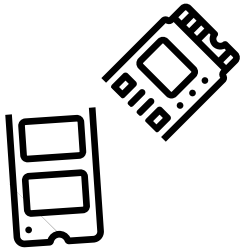
- Broke one Samsung 980
- Combination of bit flips and block discard

Broken SSD



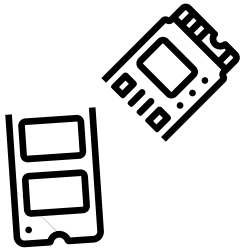
- Broke one Samsung 980
- Combination of bit flips and block discard
- Breaking hardware **purely** from software

Broken SSD



- Broke one Samsung 980
- Combination of bit flips and block discard
- Breaking hardware **purely** from software
- Did not try to reproduce a second time

Broken SSD

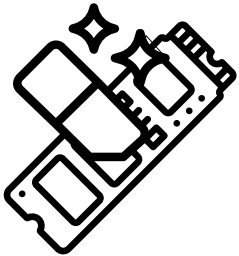


- Broke one Samsung 980
- Combination of bit flips and block discard
- Breaking hardware **purely** from software
- Did not try to reproduce a second time

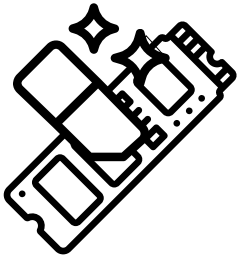
```
[30.920] nvme2: Device not ready; aborting initialisation, CSTS=0x0
[30.920] nvme2: Removing after probe failure status: -19
```

Listing: Kernel log from trying to initialize the SSD

“Unwriting Data”

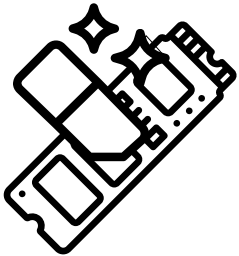


“Unwriting Data”



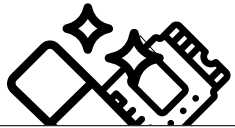
- Wrote to one SSD page

“Unwriting Data”



- Wrote to one SSD page
- Observed HMB change

“Unwriting Data”

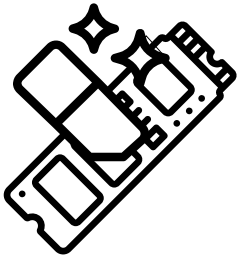


- Wrote to one SSD page
- Observed HMB change

```
0x228f500:  0xb4691cd680300010  0x39ad3a4735a548e6
            0xb4691cd686300000  0x39ad3a4735a548e6
0x3a111e0:  0x08de003a1613149e  0x1da9144607d70994
            0x08de003a1613149e  0x1da91446 0bc40994
```

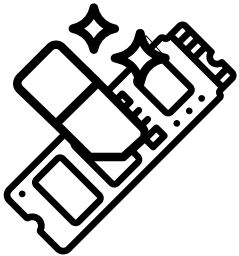
Listing: Changes in the HMB after writing to one SSD page.

“Unwriting Data”



- Wrote to one SSD page
- Observed HMB change
- Reverted HMB change

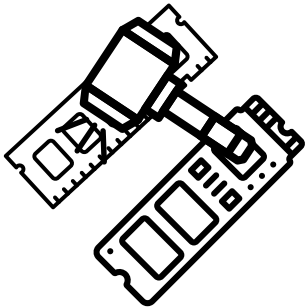
“Unwriting Data”



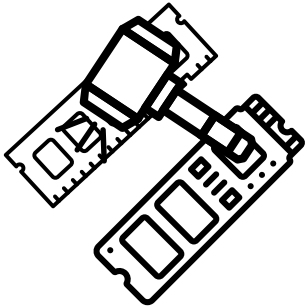
- Wrote to one SSD page
- Observed HMB change
- Reverted HMB change
- **Read old data**

HMB Rowhammer

HMB Rowhammer

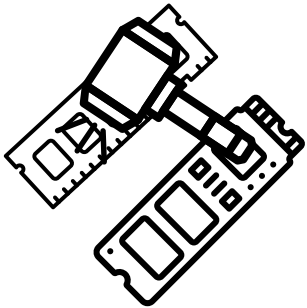


HMB Rowhammer



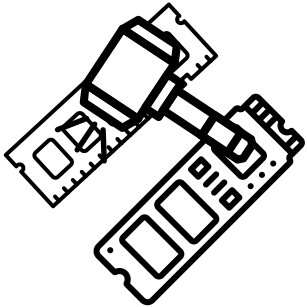
- Generate **high frequency** HMB accesses

HMB Rowhammer



- Generate **high frequency** HMB accesses
- Combine SSD with software accesses

HMB Rowhammer

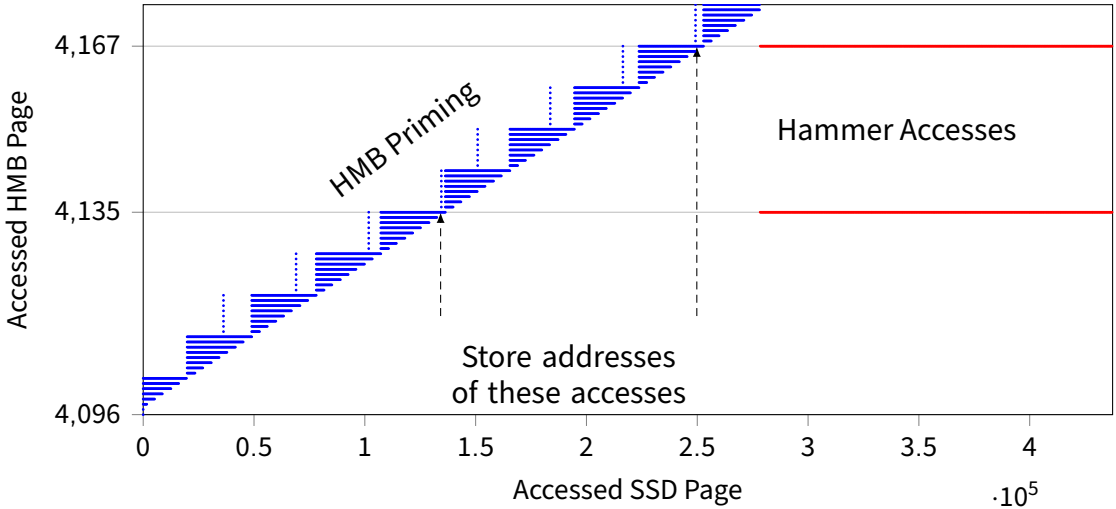


- Generate **high frequency** HMB accesses
- Combine SSD with software accesses
- Calculate hammer effectiveness

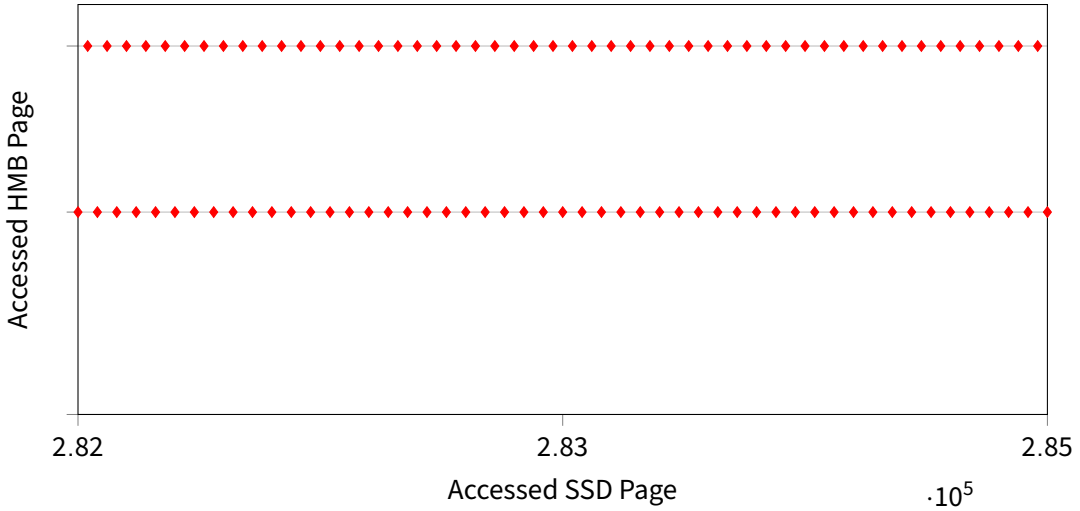
HMB Rowhammer

Generating High Frequency Hammer Accesses

Samsung 990 EVO



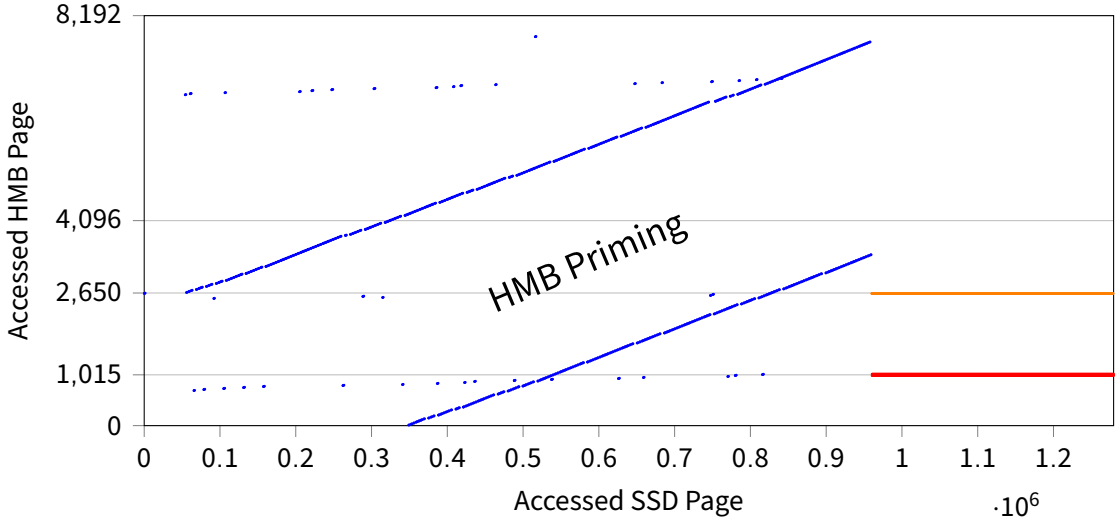
Samsung 990 EVO – Hammer Accesses



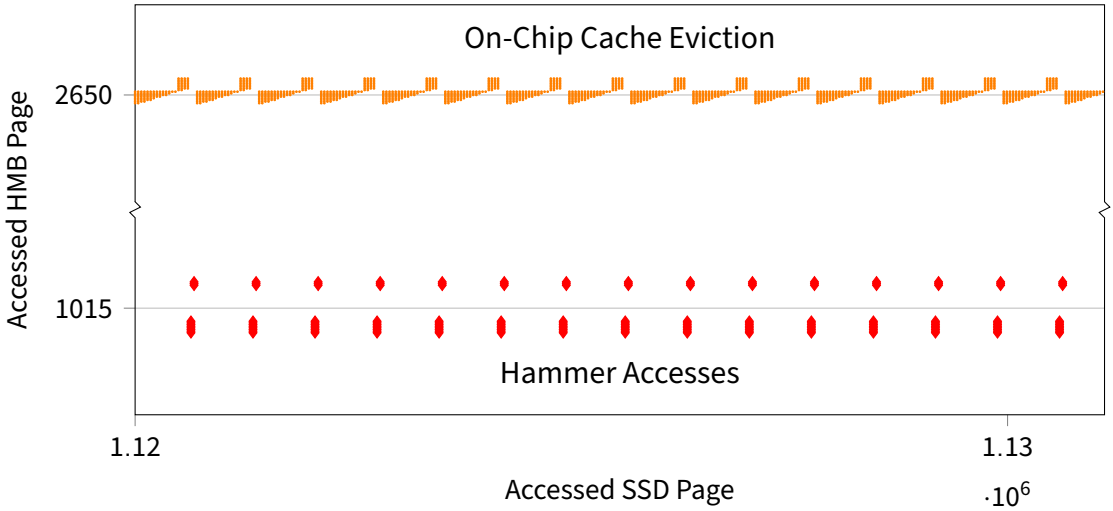
Samsung 990 EVO – Hammer Frequency

$$\frac{135\,000\ \text{IOPS} \cdot 64\ \text{ms}}{31\ \text{Dummy Accesses}} = 280\ \text{Hammer Accesses}$$

Lexar NM790


8,192
4,096
2,650
1,015
0
Accessed HMB Page
HMB Priming
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
1.1
1.2
Accessed SSD Page
·10^6

Lexar NM790 – Hammer Accesses

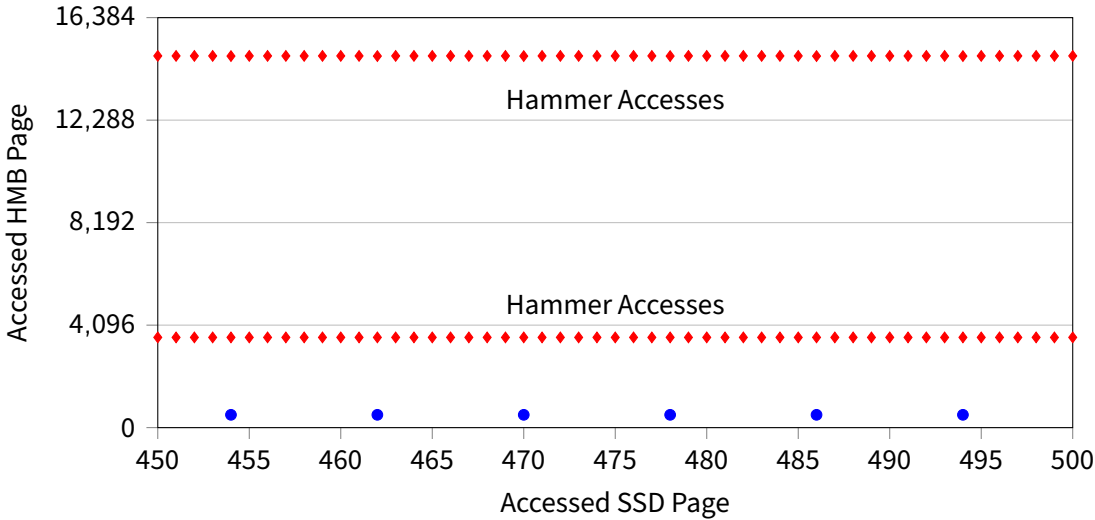


Lexar NM790 – Hammer Frequency

$$\frac{112000\,\text{IOPS} \cdot 64\,\text{ms}}{10\,\text{Eviction Accesses}} = 717\,\text{Hammer Accesses}$$

Samsung 980 – Hammer Accesses

Samsung 980 – Hammer Accesses



Samsung 980 – Hammer Accesses

$$PageB = PageA + 37\,200\,\text{kB}$$

Accessed SSD Page	Accessed HMB Pages		
$PageA + 0 \cdot 1\,200\,\text{kB}$	$512 - 519,$	$3606 - 3607,$	$14848 - 14855$
$PageB + 0 \cdot 1\,200\,\text{kB}$		$3605 - 3607,$	$14854 - 14855$
$PageA + 1 \cdot 1\,200\,\text{kB}$		$3606 - 3607,$	$14854 - 14855$
$PageB + 1 \cdot 1\,200\,\text{kB}$		$3605 - 3607,$	$14854 - 14855$
	$\dots$		
$PageA + 4 \cdot 1\,200\,\text{kB}$	$512 - 519,$	$3606 - 3607,$	$14848 - 14855$
$PageB + 4 \cdot 1\,200\,\text{kB}$		$3605 - 3607,$	$14854 - 14855$
	$\dots$		
$PageA + 15 \cdot 1\,200\,\text{kB}$		$3606 - 3607,$	$14854 - 14855$
$PageB + 15 \cdot 1\,200\,\text{kB}$		$3605 - 3607,$	$14854 - 14855$

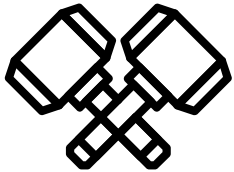
Samsung 980 – Hammer Frequency

$$78000\,\text{IOPS} \cdot 64\,\text{ms} = 4992\,\text{Hammer Accesses}$$

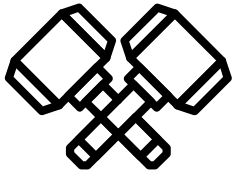
HMB Rowhammer

Combined Rowhammer

Combined Rowhammer

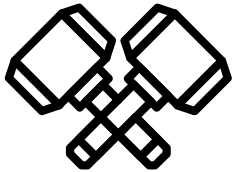


Combined Rowhammer



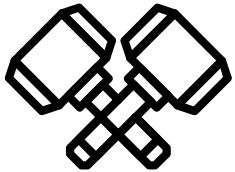
- SSD alone cannot hammer fast enough

Combined Rowhammer



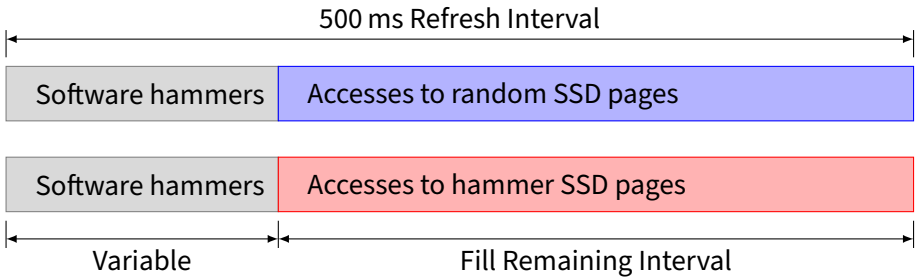
- SSD alone cannot hammer fast enough
- Combine with hammers from software

Combined Rowhammer

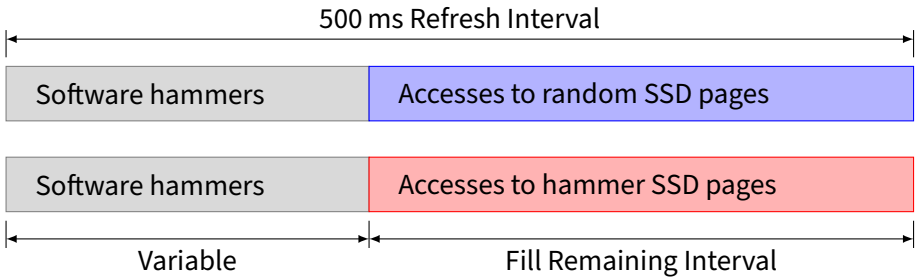


- SSD alone cannot hammer fast enough
- Combine with hammers from software
- Increased refresh interval to 500 ms

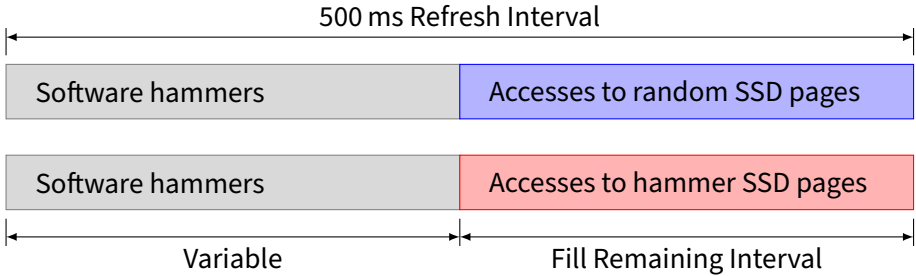
Combined Rowhammer



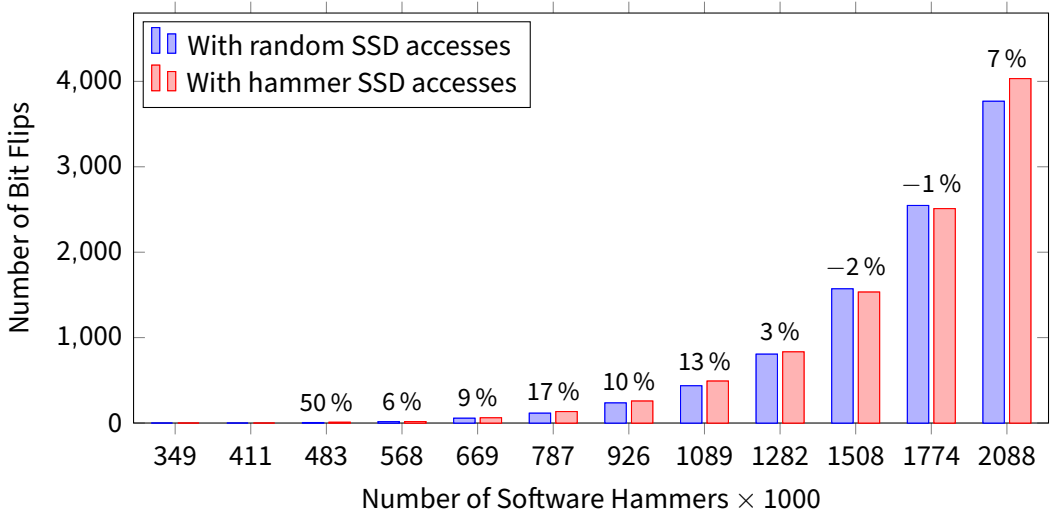
Combined Rowhammer



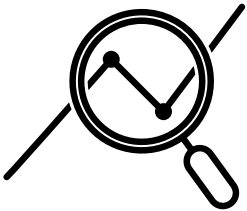
Combined Rowhammer



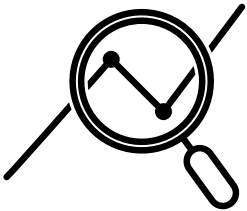
Combined Rowhammer – Results



Combined Rowhammer – Results

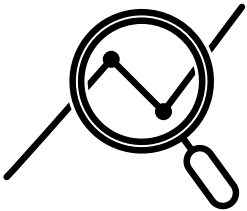


Combined Rowhammer – Results



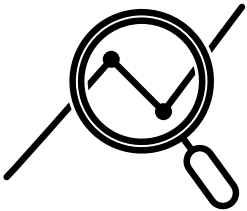
- 9 555 bit flips with random SSD accesses

Combined Rowhammer – Results



- 9 555 bit flips with random SSD accesses
- 9 877 bit flips with hammer SSD accesses

Combined Rowhammer – Results



- 9 555 bit flips with random SSD accesses
- 9 877 bit flips with hammer SSD accesses
- +3.4 %

Combined Rowhammer – Results

- 1000 more software accesses $\rightarrow$ +1.8 bit flips

Combined Rowhammer – Results

- 1000 more software accesses $\rightarrow$ +1.8 bit flips
- 78 000 IOPS for 320 ms cause 24 960 HMB accesses.

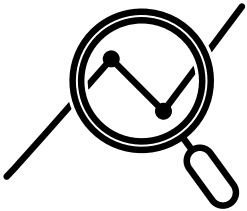
Combined Rowhammer – Results

- 1000 more software accesses $\rightarrow$ +1.8 bit flips
- 78 000 IOPS for 320 ms cause 24 960 HMB accesses.
- Expect $\frac{24960}{1000} \cdot 1.8 = 45$ more bit flips

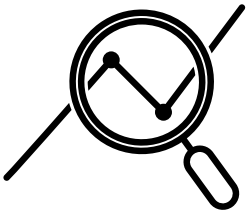
Combined Rowhammer – Results

- 1000 more software accesses $\rightarrow$ +1.8 bit flips
- 78 000 IOPS for 320 ms cause 24 960 HMB accesses.
- Expect $\frac{24960}{1000} \cdot 1.8 = 45$ more bit flips
- Average absolute increase: 40 bit flips

Conclusion

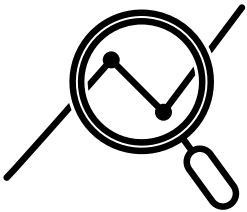


Conclusion



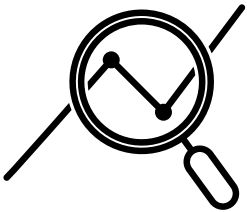
- No privilege escalation from HMB bit flips

Conclusion



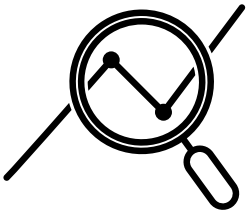
- No privilege escalation from HMB bit flips
- “Only” denial of service

Conclusion



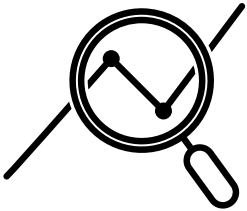
- No privilege escalation from HMB bit flips
- “Only” denial of service
- ... requiring a **full power cycle**

Conclusion

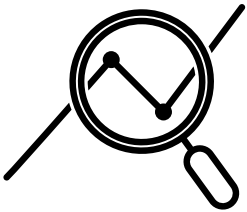


- No privilege escalation from HMB bit flips
- “Only” denial of service
- ... requiring a **full power cycle**
- Worst case: broken SSD

Conclusion

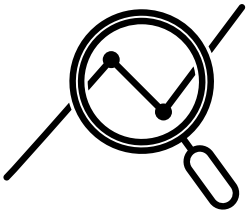


Conclusion



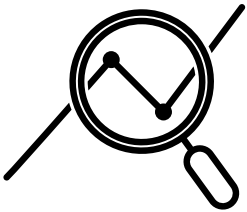
- SSD accessing the HMB does hammer

Conclusion



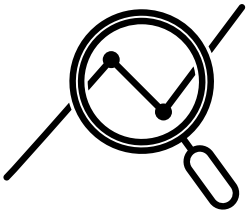
- SSD accessing the HMB does hammer
- But way **too slow**

Conclusion



- SSD accessing the HMB does hammer
- But way **too slow**
- Including state of the art TRR

Conclusion



- SSD accessing the HMB does hammer
- But way **too slow**
- Including state of the art TRR
- ... bit flips are impossible

Acknowledgments

This research was made possible by generous funding from:

This research is supported in part by the European Research Council (ERC project FSSec 101076409), and the Austrian Science Fund (FWF SFB project SPyCoDe 10.55776/F85). Additional funding was provided by generous gifts from Red Hat and Google. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding parties.

TU Graz
SCIENCE
PASSION
TECHNOLOGY

An Analysis of HMB-based SSD Rowhammer

Jonas Juffinger

Graz University of Technology, Graz, Austria

uASC 2025

› isec.tugraz.at

Bibliography I

[For25] Fortune Business Insights. **Solid State Drive Market Size, Share & Industry Trends Analysis**. 2025. URL: https://www.fortunebusinessinsights.com/solid-state-drive-market-109839.

[Zha+21] T. Zhang, B. Pismenny, D. E. Porter, D. Tsafrir, and A. Zuck. **Rowhammering Storage Devices**. ACM Workshop on Hot Topics in Storage and File Systems. 2021.