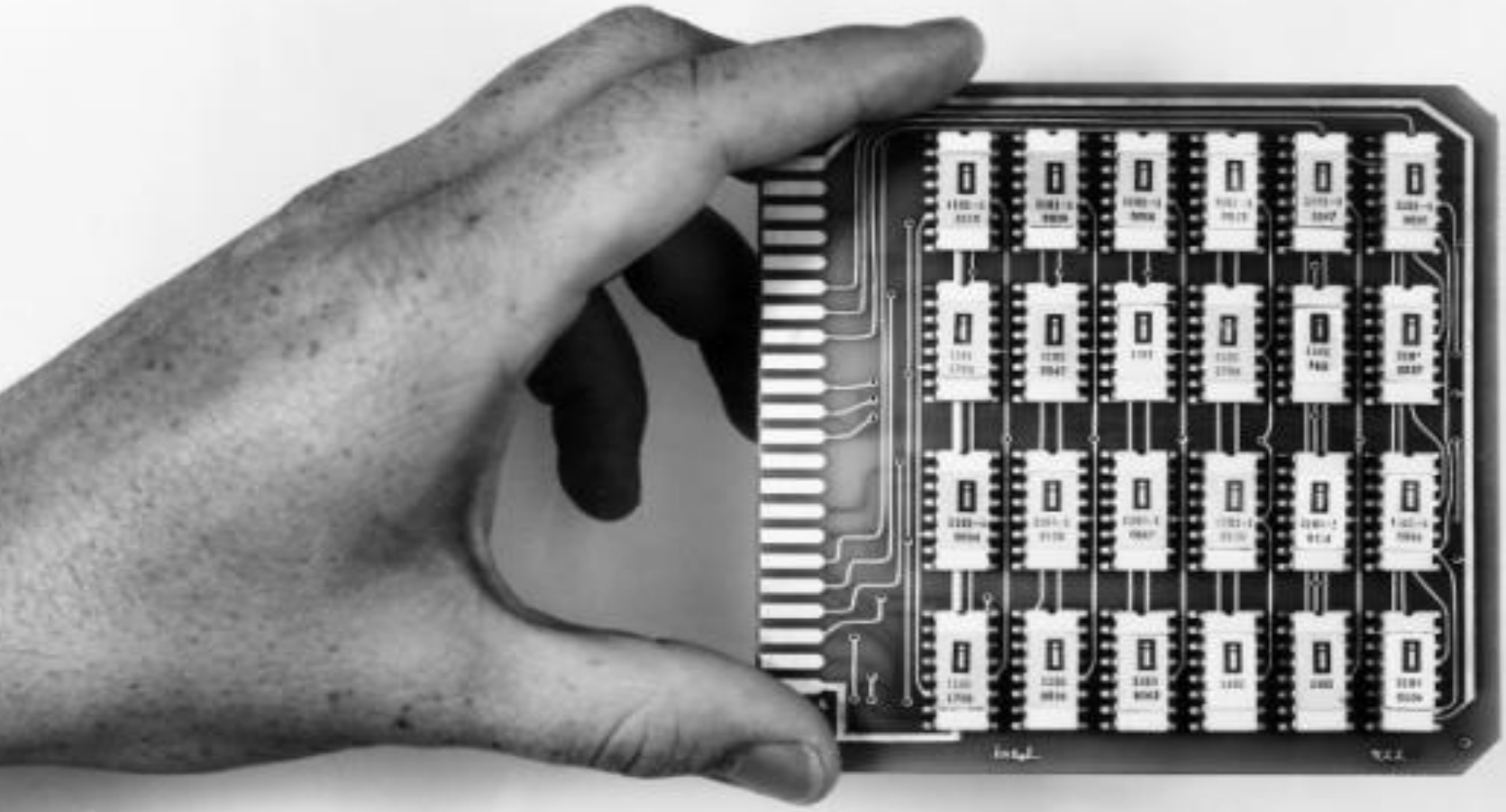
HOW WILL PERSISTENT MEMORY CHANGE SOFTWARE DESIGN

Maciej Maciejewski

HARDWARE



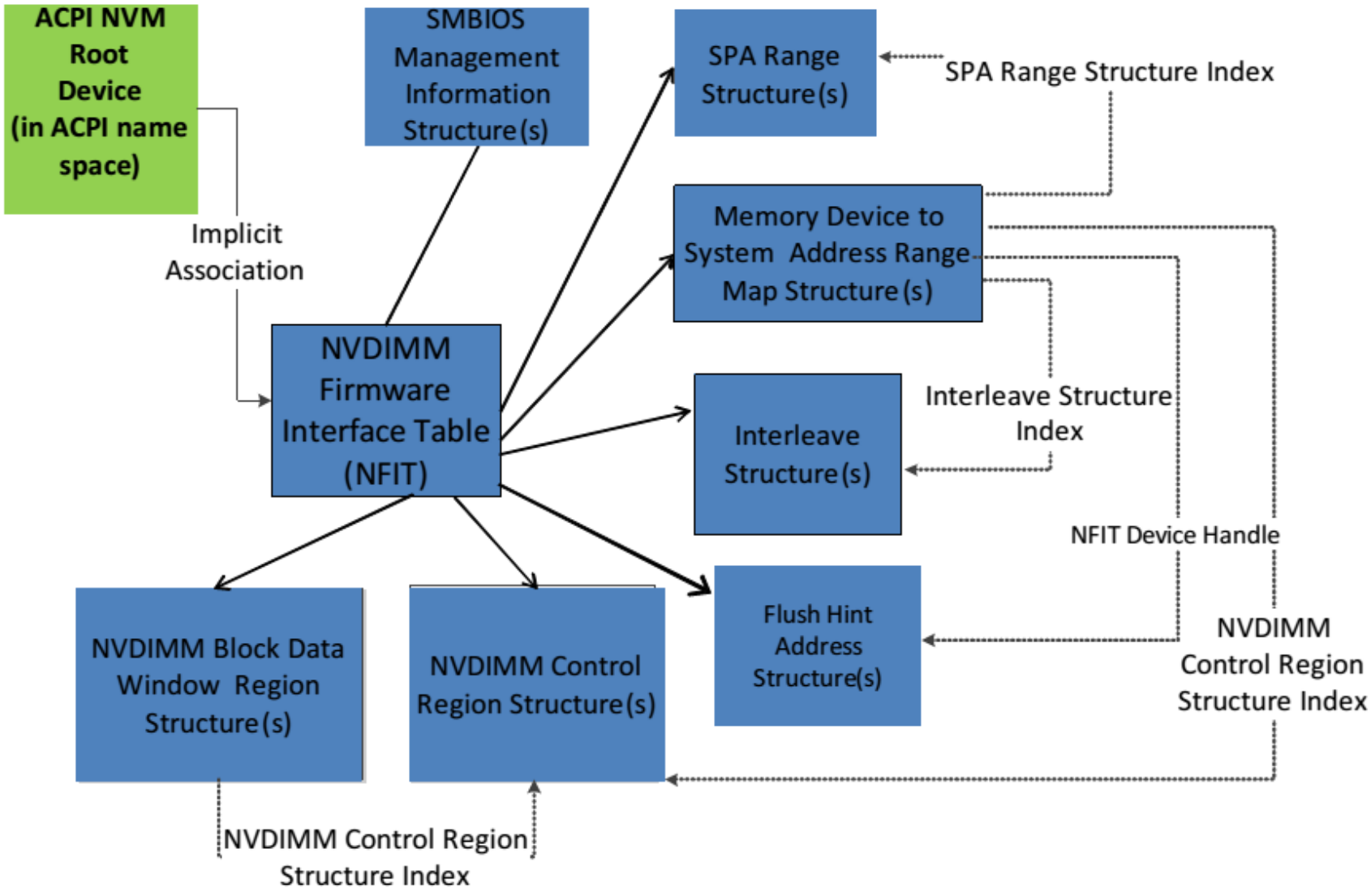
Persistent memory

How does it differ from DRAM

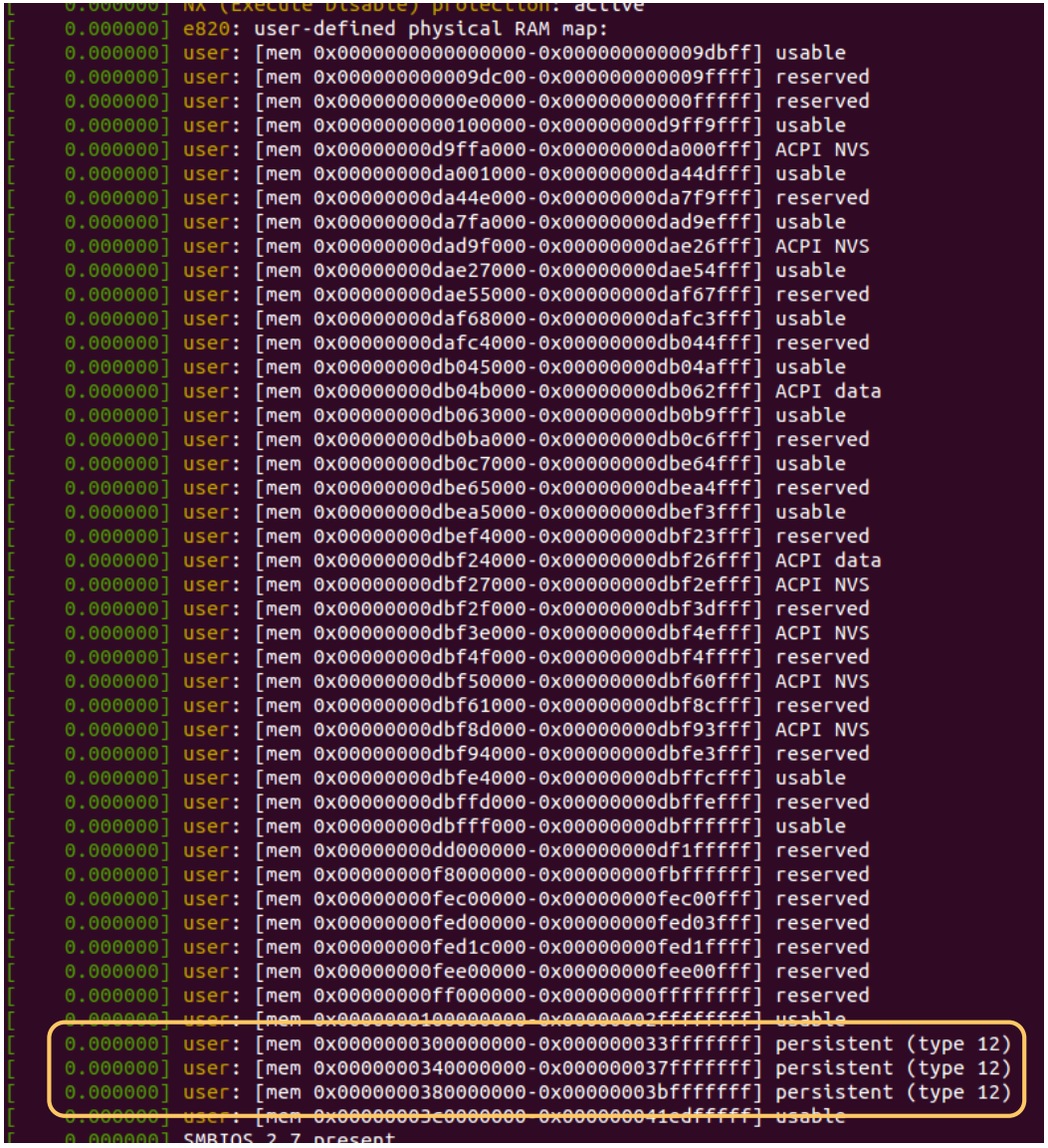
How fast is it?

OPERATING SYSTEM

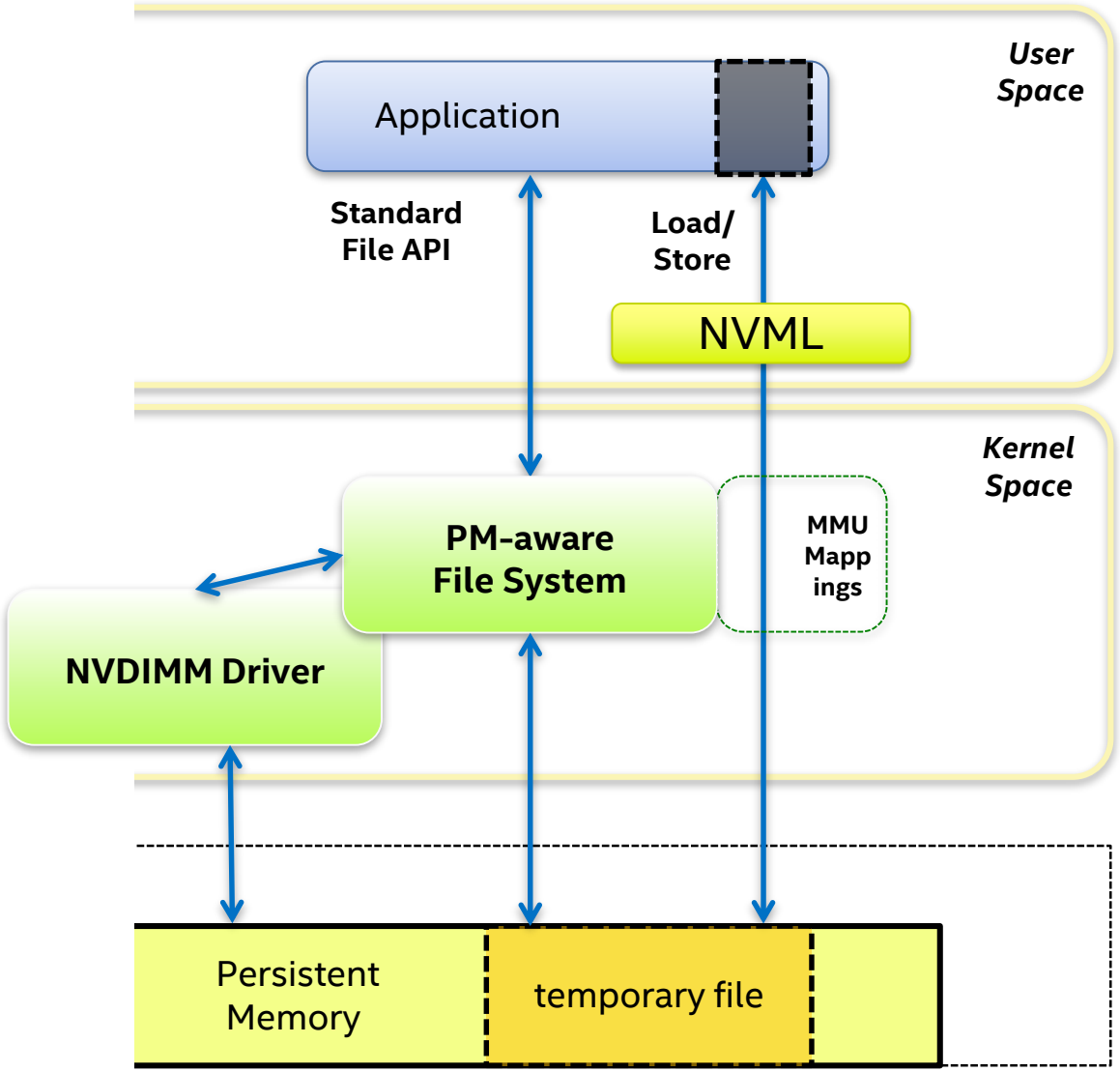
ACPI NFIT


ACPI NVM Root Device (in ACPI name space)
SMBIOS Management Information Structure(s)
SPA Range Structure(s)
SPA Range Structure Index
Implicit Association
Memory Device to System Address Range Map Structure(s)
NVDIMM Firmware Interface Table (NFIT)
Interleave Structure(s)
Interleave Structure Index
NFIT Device Handle
NVDIMM Block Data Window Region Structure(s)
NVDIMM Control Region Structure(s)
Flush Hint Address Structure(s)
NVDIMM Control Region Structure Index
NVDIMM Control Region Structure Index

E820


[0.000000] NX (Execute Disable) protection: active
[0.000000] e820: user-defined physical RAM map:
[0.000000] user: [mem 0x0000000000000000-0x000000000009dbff] usable
[0.000000] user: [mem 0x000000000009dc00-0x000000000009ffff] reserved
[0.000000] user: [mem 0x00000000000e0000-0x00000000000fffff] reserved
[0.000000] user: [mem 0x0000000000100000-0x00000000d9ff9fff] usable
[0.000000] user: [mem 0x00000000d9ffa000-0x00000000da000fff] ACPI NVS
[0.000000] user: [mem 0x00000000da001000-0x00000000da44dfff] usable
[0.000000] user: [mem 0x00000000da44e000-0x00000000da7f9fff] reserved
[0.000000] user: [mem 0x00000000da7fa000-0x00000000dad9efff] usable
[0.000000] user: [mem 0x00000000dad9f000-0x00000000dae26fff] ACPI NVS
[0.000000] user: [mem 0x00000000dae27000-0x00000000dae54fff] usable
[0.000000] user: [mem 0x00000000dae55000-0x00000000daf67fff] reserved
[0.000000] user: [mem 0x00000000daf68000-0x00000000dafc3fff] usable
[0.000000] user: [mem 0x00000000dafc4000-0x00000000db044fff] reserved
[0.000000] user: [mem 0x00000000db045000-0x00000000db04afff] usable
[0.000000] user: [mem 0x00000000db04b000-0x00000000db062fff] ACPI data
[0.000000] user: [mem 0x00000000db063000-0x00000000db0b9fff] usable
[0.000000] user: [mem 0x00000000db0ba000-0x00000000db0c6fff] reserved
[0.000000] user: [mem 0x00000000db0c7000-0x00000000dbe64fff] usable
[0.000000] user: [mem 0x00000000dbe65000-0x00000000dbea4fff] reserved
[0.000000] user: [mem 0x00000000dbea5000-0x00000000dbef3fff] usable
[0.000000] user: [mem 0x00000000dbef4000-0x00000000dbf23fff] reserved
[0.000000] user: [mem 0x00000000dbf24000-0x00000000dbf26fff] ACPI data
[0.000000] user: [mem 0x00000000dbf27000-0x00000000dbf2efff] ACPI NVS
[0.000000] user: [mem 0x00000000dbf2f000-0x00000000dbf3dfff] reserved
[0.000000] user: [mem 0x00000000dbf3e000-0x00000000dbf4efff] ACPI NVS
[0.000000] user: [mem 0x00000000dbf4f000-0x00000000dbf4ffff] reserved
[0.000000] user: [mem 0x00000000dbf50000-0x00000000dbf60fff] ACPI NVS
[0.000000] user: [mem 0x00000000dbf61000-0x00000000dbf8cfff] reserved
[0.000000] user: [mem 0x00000000dbf8d000-0x00000000dbf93fff] ACPI NVS
[0.000000] user: [mem 0x00000000dbf94000-0x00000000dbfe3fff] reserved
[0.000000] user: [mem 0x00000000dbfe4000-0x00000000dbffcfff] usable
[0.000000] user: [mem 0x00000000dbffd000-0x00000000dbffefff] reserved
[0.000000] user: [mem 0x00000000dbfff000-0x00000000dbffffff] usable
[0.000000] user: [mem 0x00000000dd000000-0x00000000df1fffff] reserved
[0.000000] user: [mem 0x00000000f8000000-0x00000000fbffffff] reserved
[0.000000] user: [mem 0x00000000fec00000-0x00000000fec00fff] reserved
[0.000000] user: [mem 0x00000000fed00000-0x00000000fed03fff] reserved
[0.000000] user: [mem 0x00000000fed1c000-0x00000000fed1ffff] reserved
[0.000000] user: [mem 0x00000000fee00000-0x00000000fee00fff] reserved
[0.000000] user: [mem 0x00000000ff000000-0x00000000ffffffff] reserved
[0.000000] user: [mem 0x0000000100000000-0x00000002ffffffff] usable
[0.000000] user: [mem 0x0000000300000000-0x000000033fffffff] persistent (type 12)
[0.000000] user: [mem 0x0000000340000000-0x000000037fffffff] persistent (type 12)
[0.000000] user: [mem 0x0000000380000000-0x00000003bfffffff] persistent (type 12)
[0.000000] user: [mem 0x00000003c0000000-0x00000041edfffff] usable
[0.000000] SMBIOS 2.7 present.

WHAT CAN WE DO WITH IT?


User Space
Application
Standard File API
Load/ Store
NVML
Kernel Space
PM-aware File System
MMU Mappings
NVDIMM Driver
Persistent Memory
temporary file

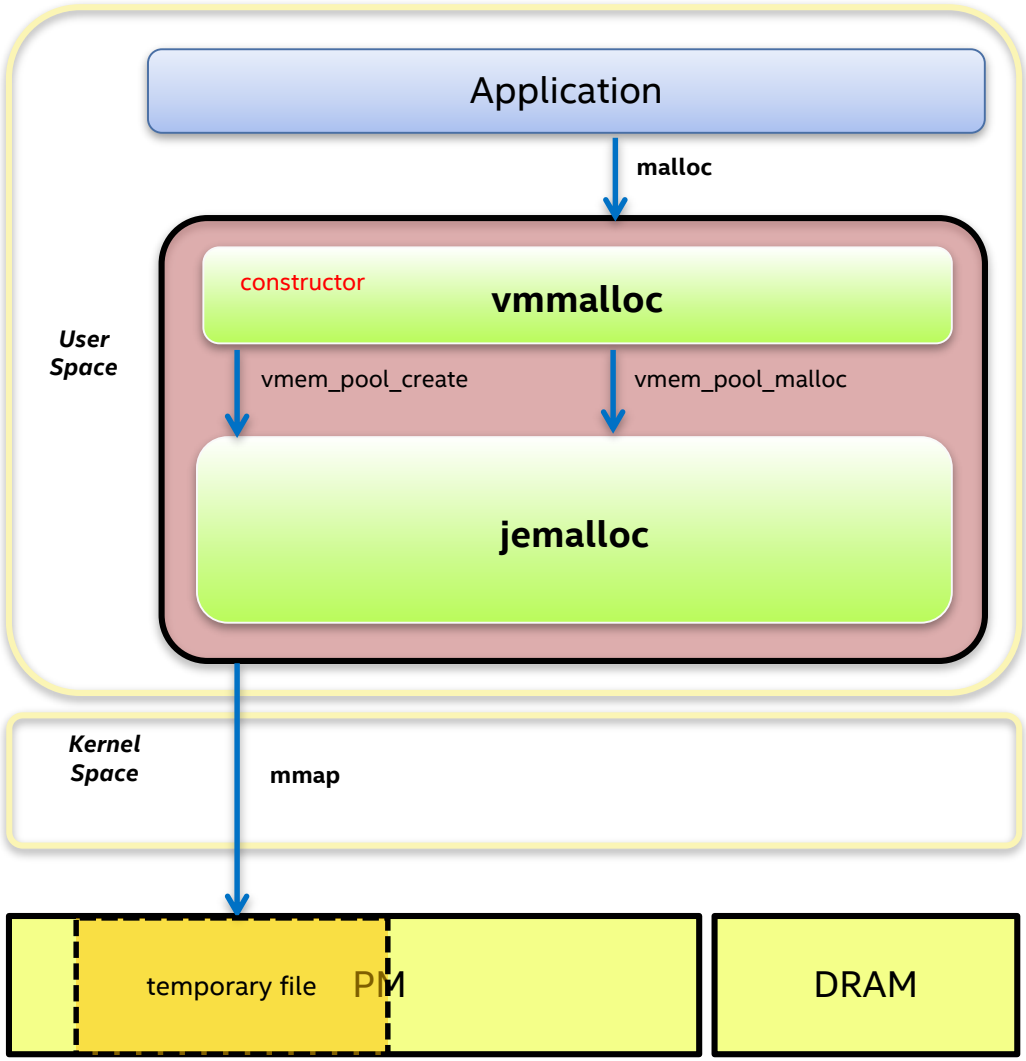
NVML

http://pmem.io

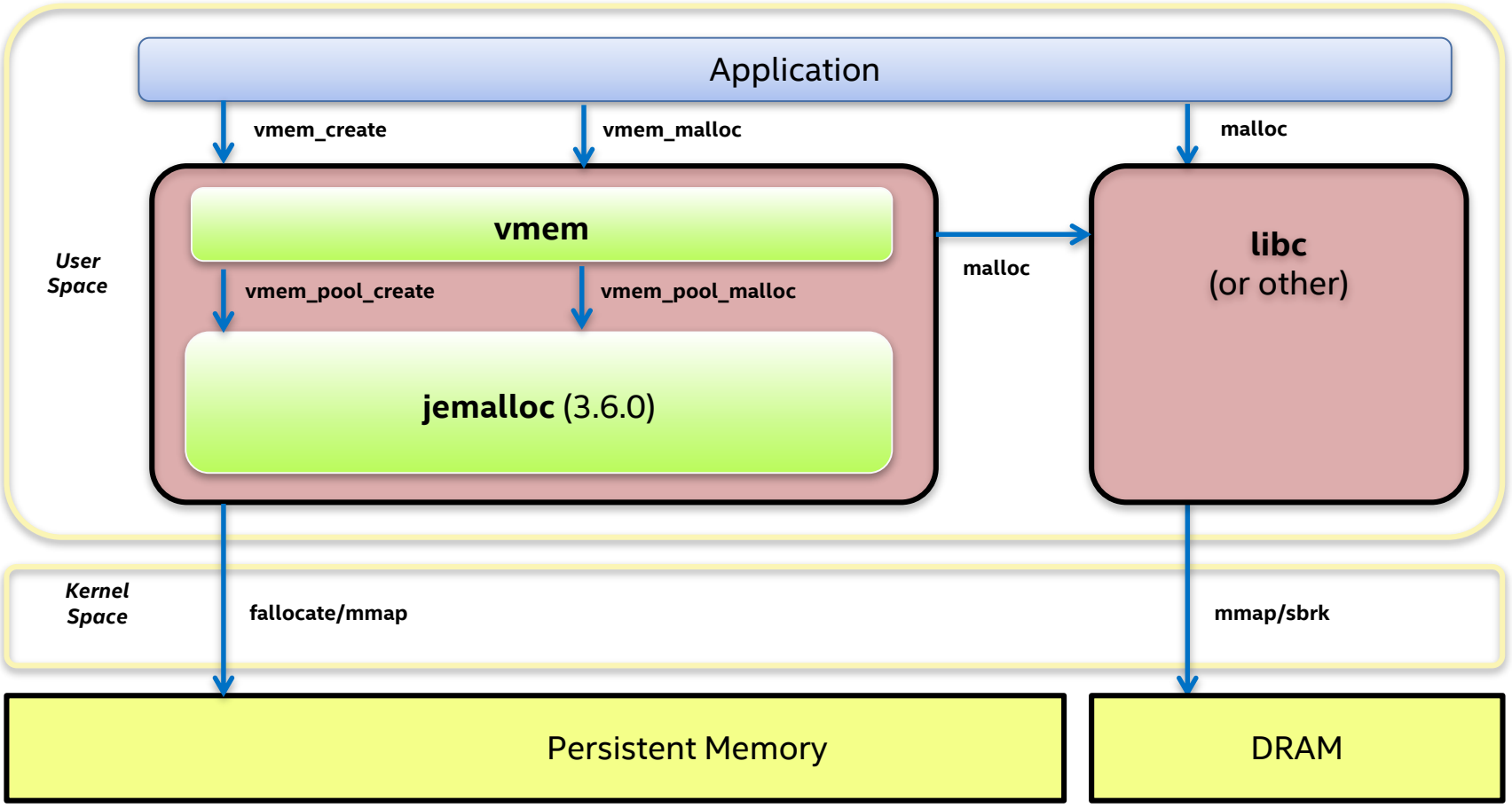
https://github.com/pmem/nvml/

Memory

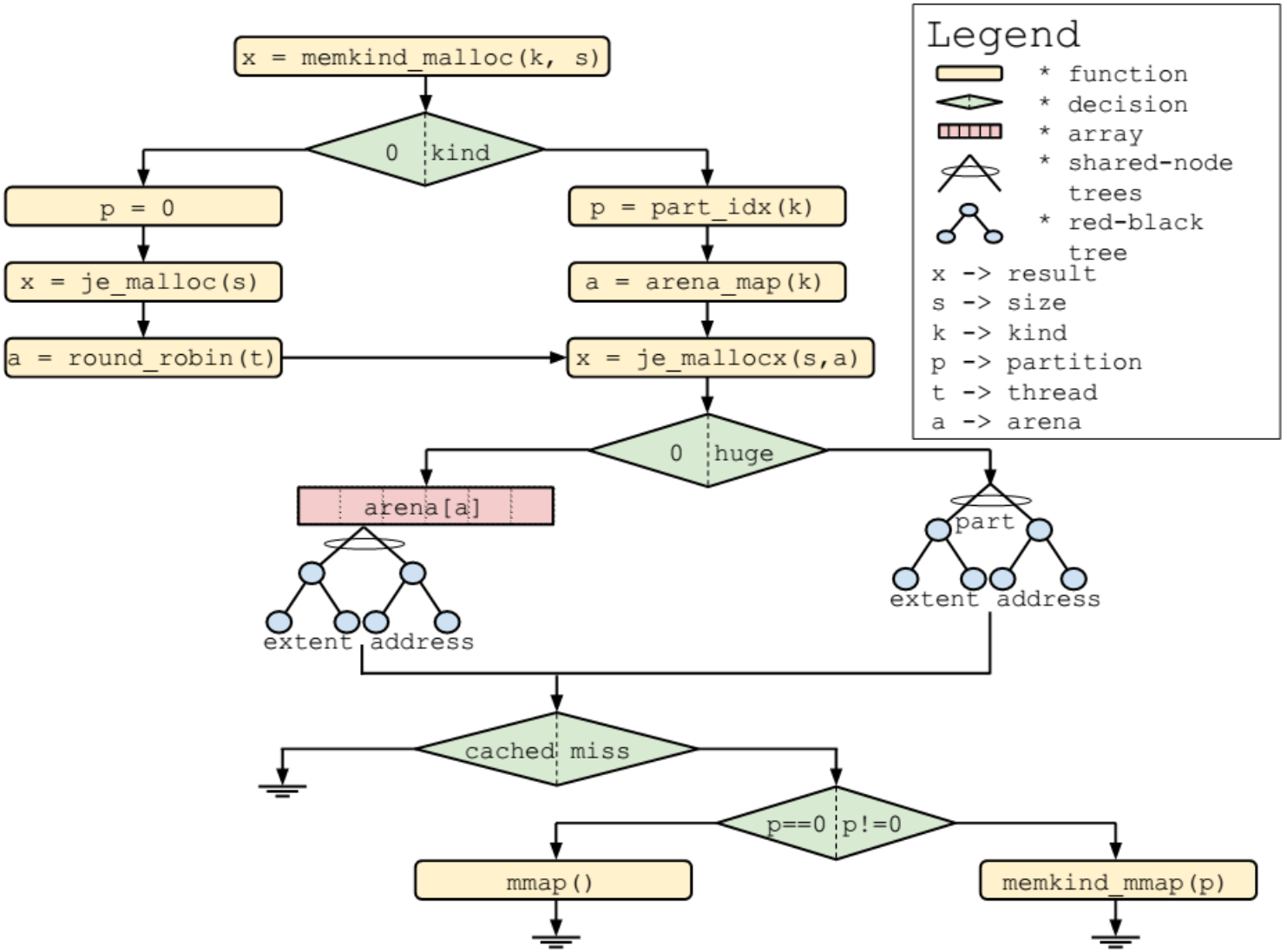
Memory - libvmmalloc



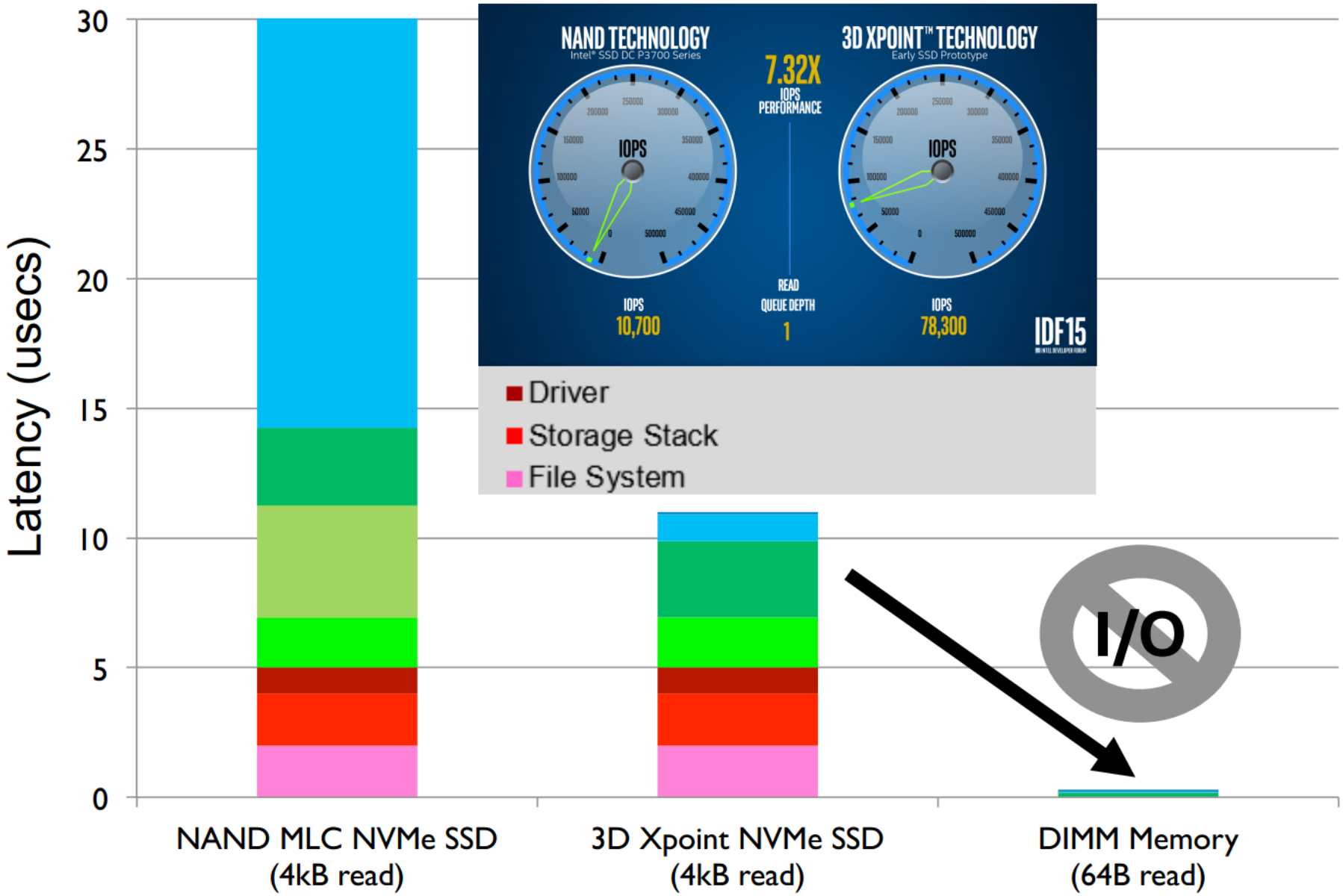
Memory - libvmem



Memory - memkind

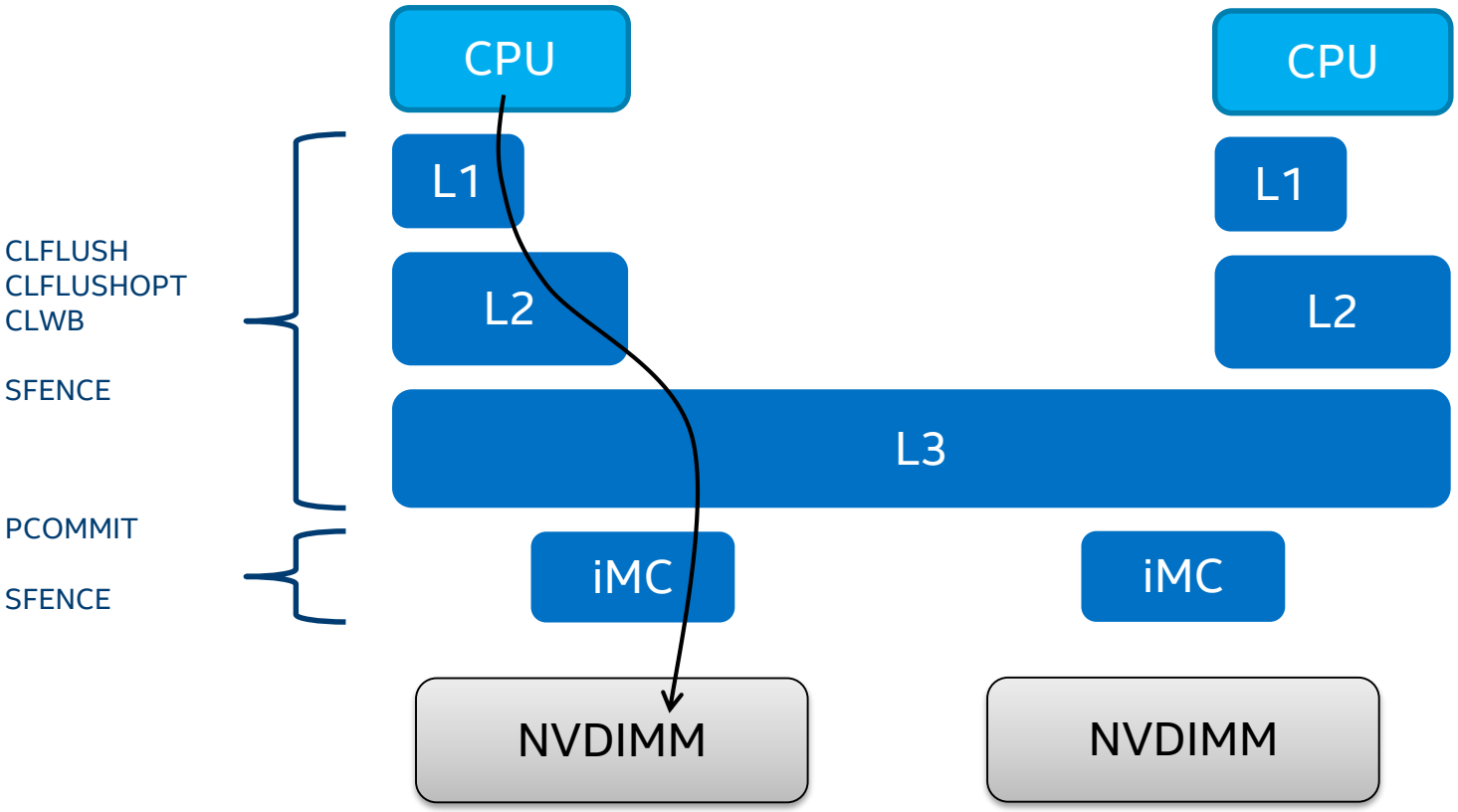


Block storage


Latency (usecs)
30
25
20
15
10
5
0
NAND TECHNOLOGY
Intel® SSD DC P3700 Series
3D XPOINT™ TECHNOLOGY
Early SSD Prototype
7.32X
IOPS
PERFORMANCE
IOPS
IOPS
10,700
READ
QUEUE DEPTH
1
IOPS
78,300
IDF15
Driver
Storage Stack
File System
I/O
NAND MLC NVMe SSD
(4kB read)
3D Xpoint NVMe SSD
(4kB read)
DIMM Memory
(64B read)

Byte persistency

Why persistency is an issue?



Flushing to Persistence

```
open(…);
mmap(…);
strcpy(pmem, „Hello");
msync(pmem, 6, MS_SYNC);
pmem_persist(pmem, 6);
strcpy(pmem, „Hello, World!");
pmem_persist(pmem, 14);
```

Crossing the 8-Byte Store

Result?

1. „\0\0\0\0\0\0\0\0\0...”
2. „Hello, W\0\0\0\0\0\0...”
3. „\0\0\0\0\0\0\0\0orld!\0”
4. „Hello, \0\0\0\0\0\0\0\0”
5. „Hello, World!\0”

nvml Persistent Libraries

libpmem – Basic persistency handling

libpmemblk – Block access to pmem

libpmemlog - Log file on pmem (append-mostly)

libpmemobj - Transactional Object Store on pmem

libpmemobj

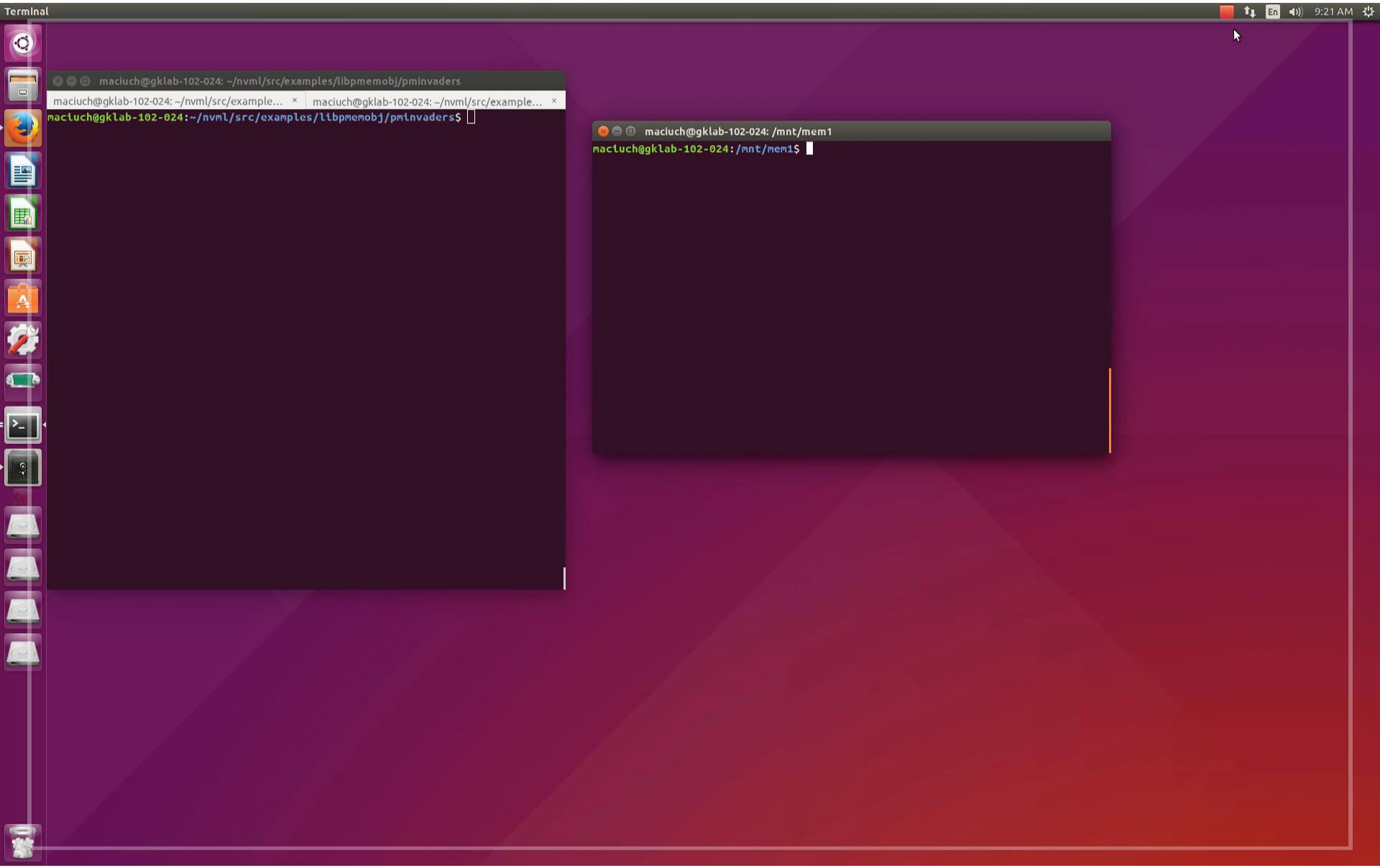
Object API

Transactional API

```
11  /* create a new collection */
12  TX_BEGIN(pop) {
13          tree_map_new(pop, &D_RW(root)->map);
14
15          /*
16           * We don't want an empty collection, so let's insert
17           * a few new entries in the same transaction.
18           */
19          TOID(struct store_item) n;
20
21          n = TX_NEW(struct store_item);
22          tree_map_insert(pop, D_RO(root)->map, 5, n.oid);
23
24          n = TX_NEW(struct store_item);
25          tree_map_insert(pop, D_RO(root)->map, 10, n.oid);
26  } TX_END
27
```

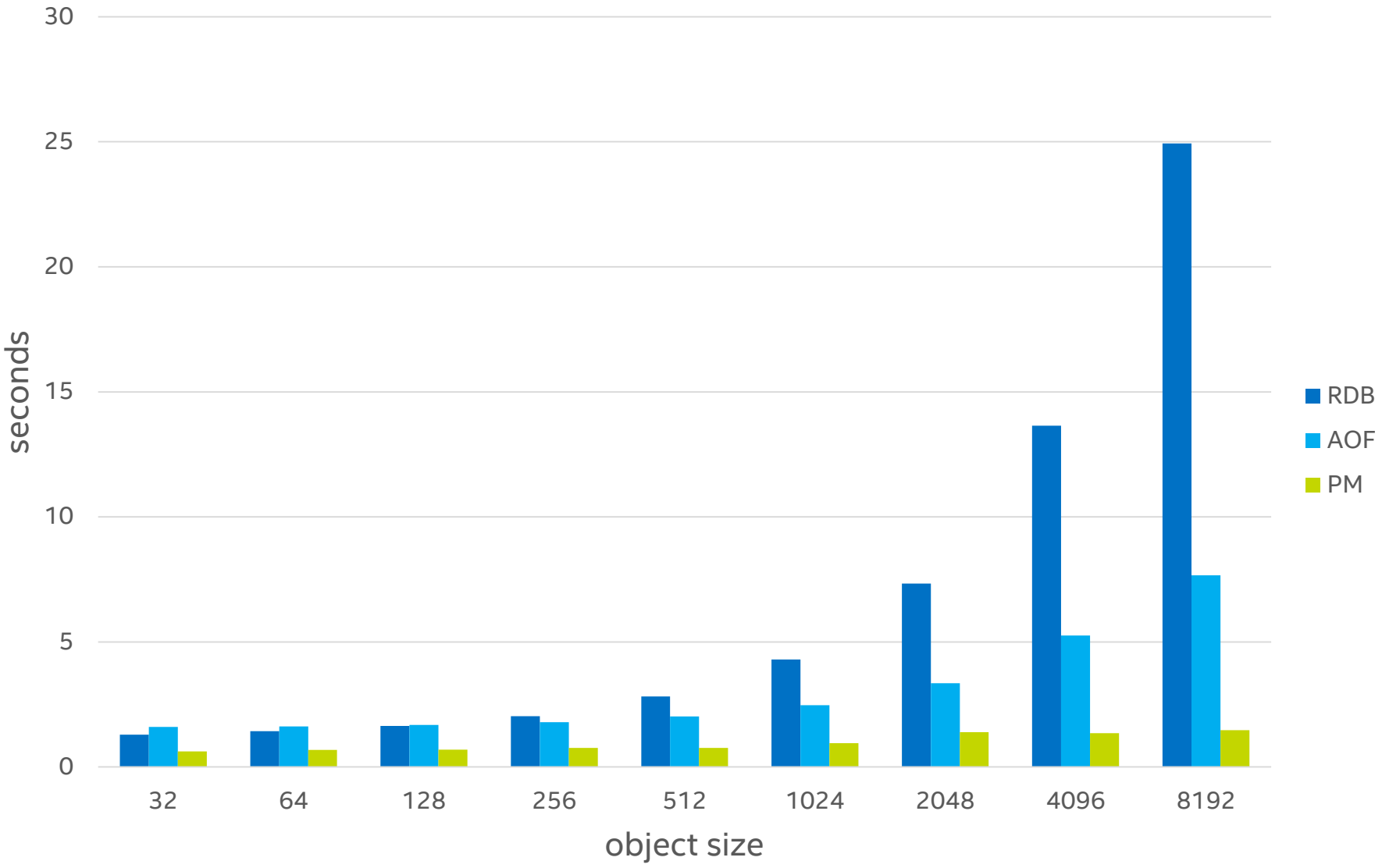
APPLICATIONS

pm_invaders

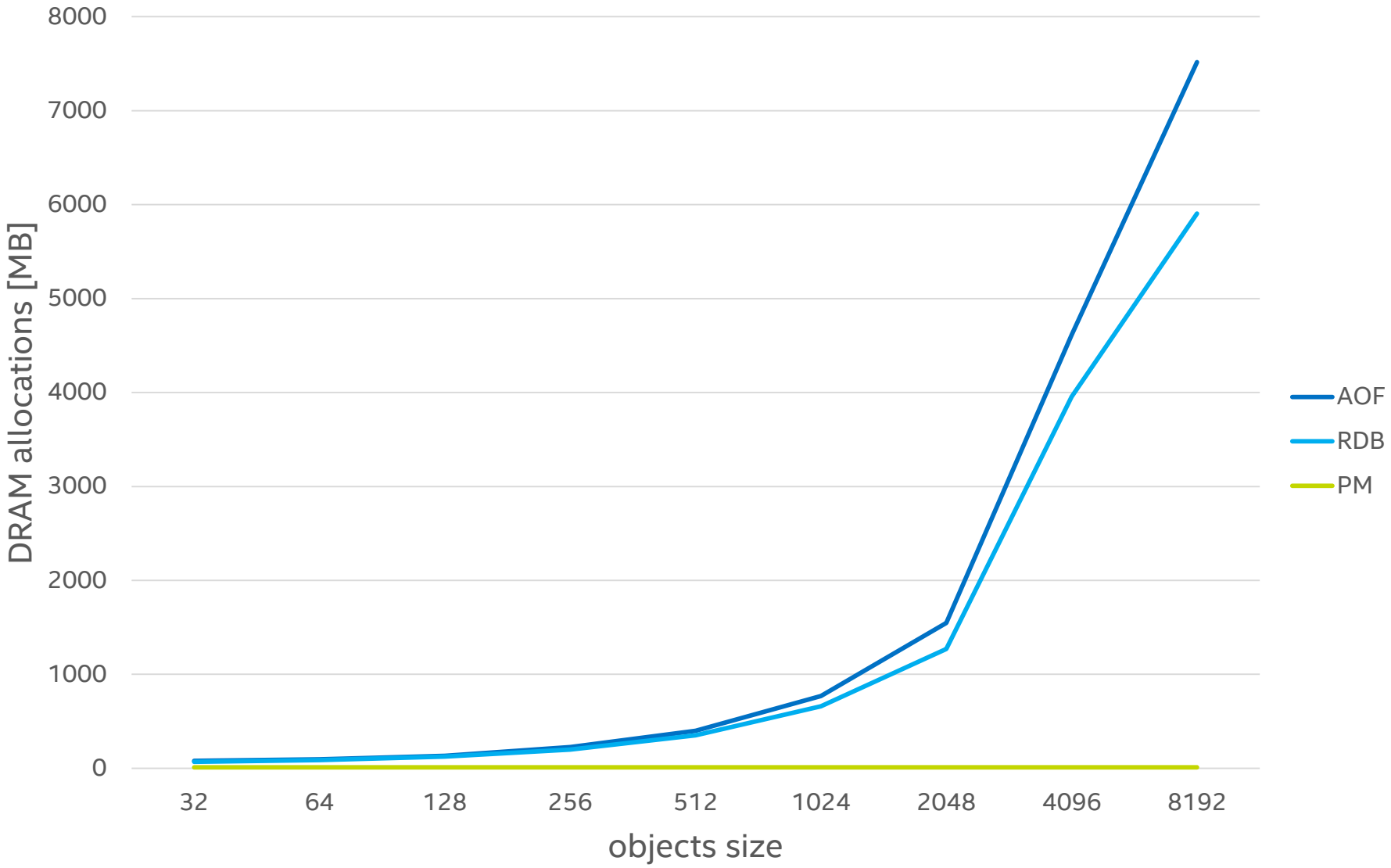

Terminal
9:21 AM
maciuch@gklab-102-024: ~/nvml/src/examples/libpmemobj/pminvaders
maciuch@gklab-102-024: ~/nvml/src/example...
maciuch@gklab-102-024: ~/nvml/src/example...
maciuch@gklab-102-024:~/nvml/src/examples/libpmemobj/pminvaders$
maciuch@gklab-102-024: /mnt/mem1
maciuch@gklab-102-024:/mnt/mem1$

Redis key/value store

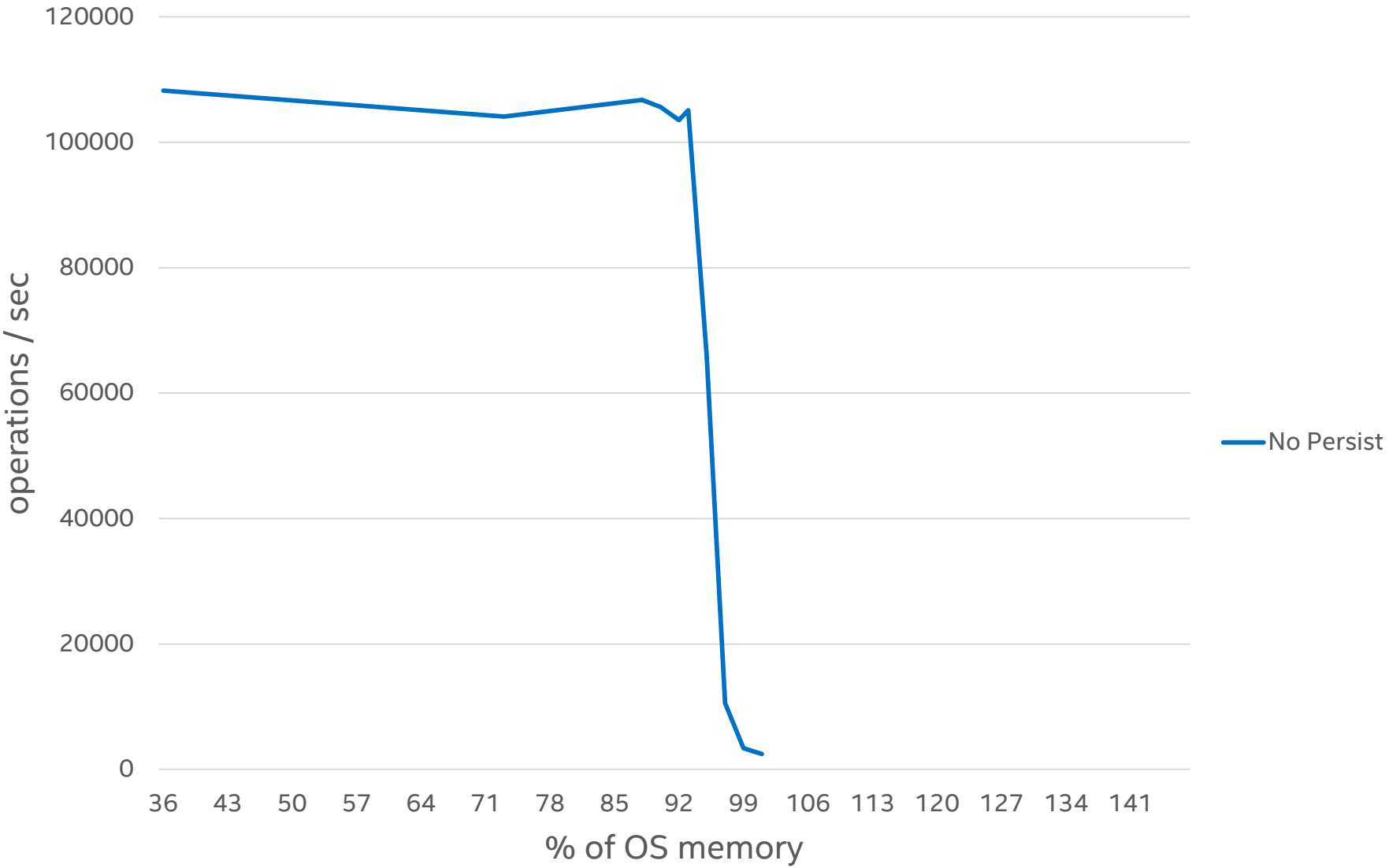
Startup time



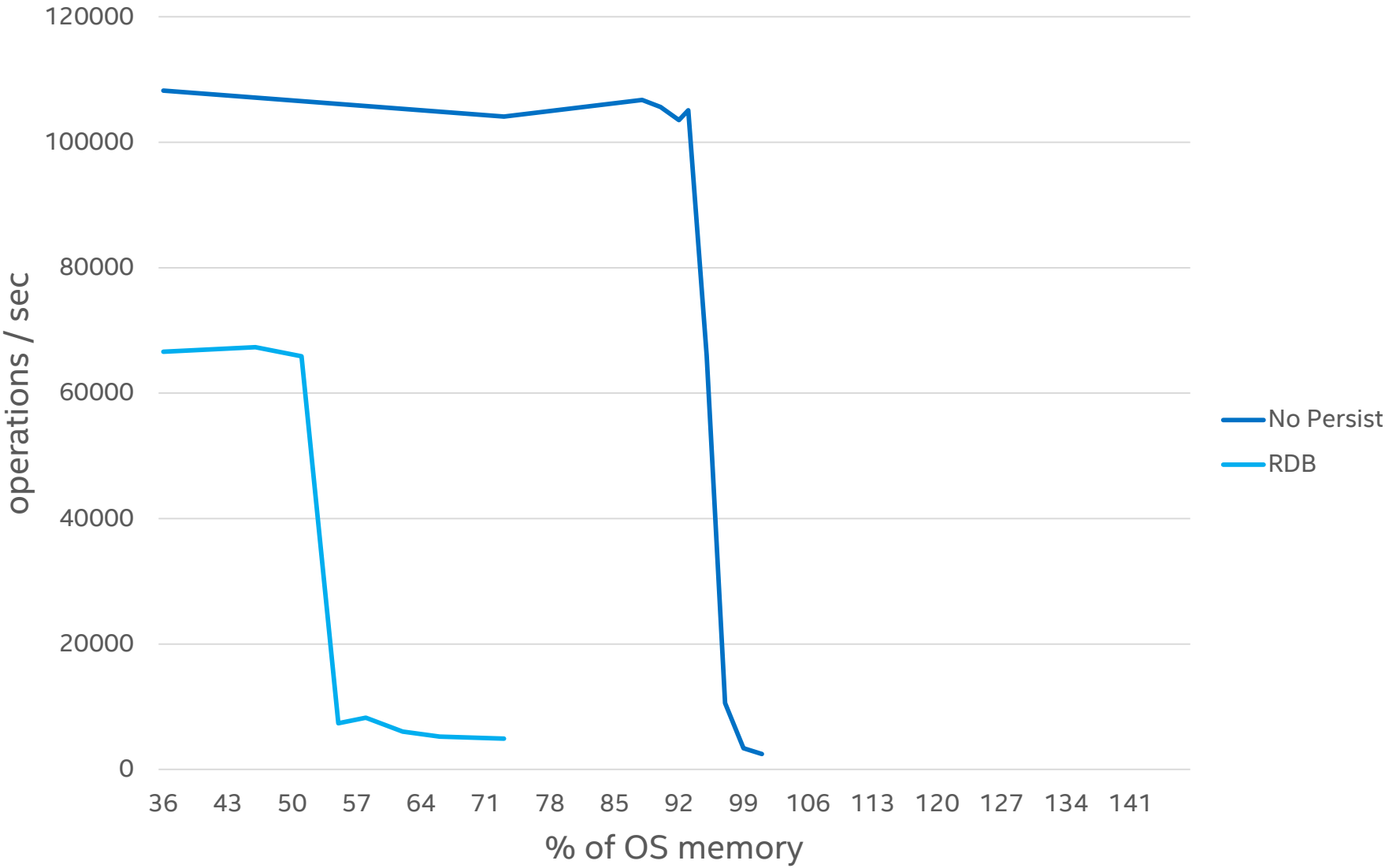
DRAM usage


8000
7000
6000
5000
4000
3000
2000
1000
0
DRAM allocations [MB]
32
64
128
256
512
1024
2048
4096
8192
objects size
AOF
RDB
PM

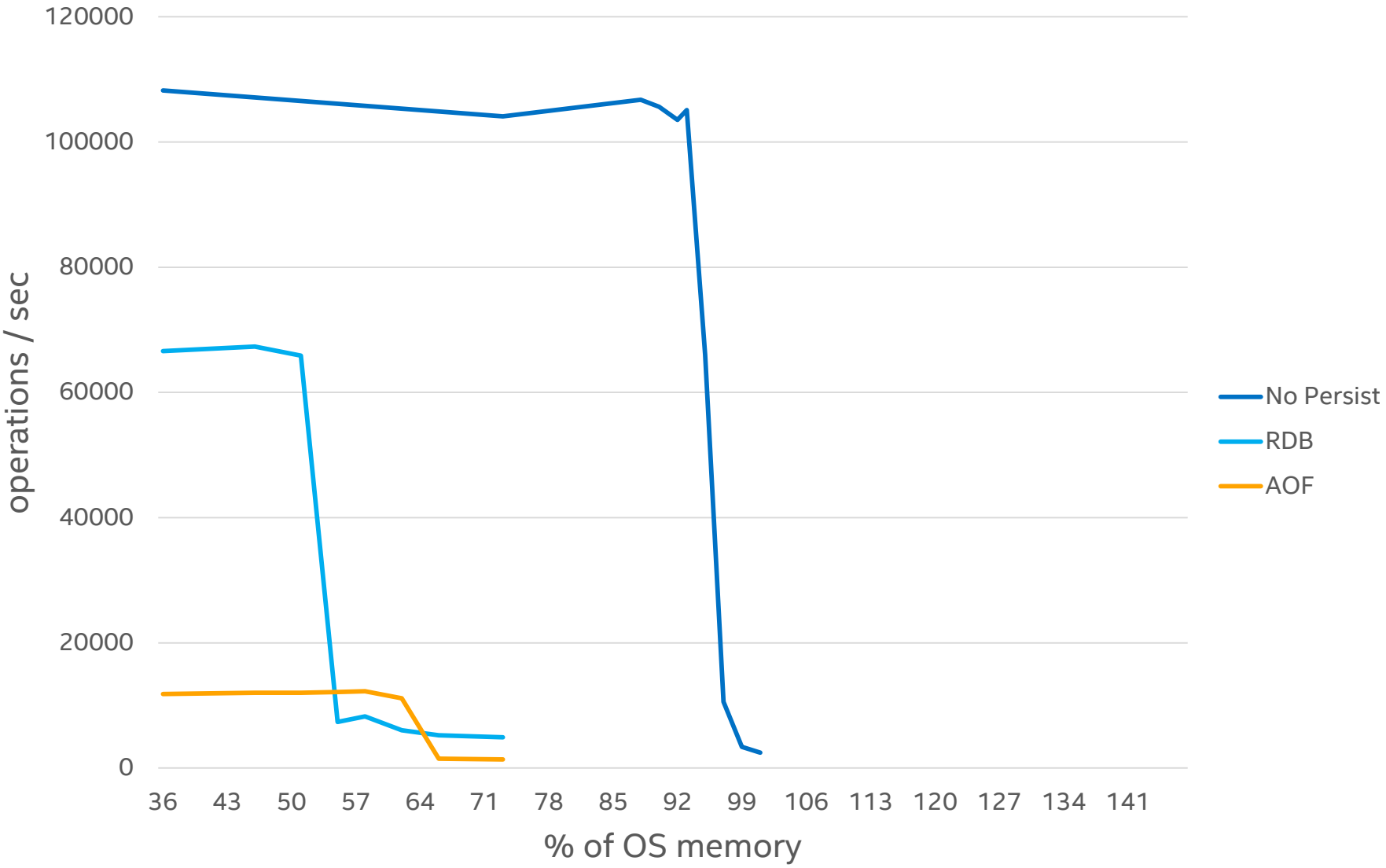
Running out of DRAM



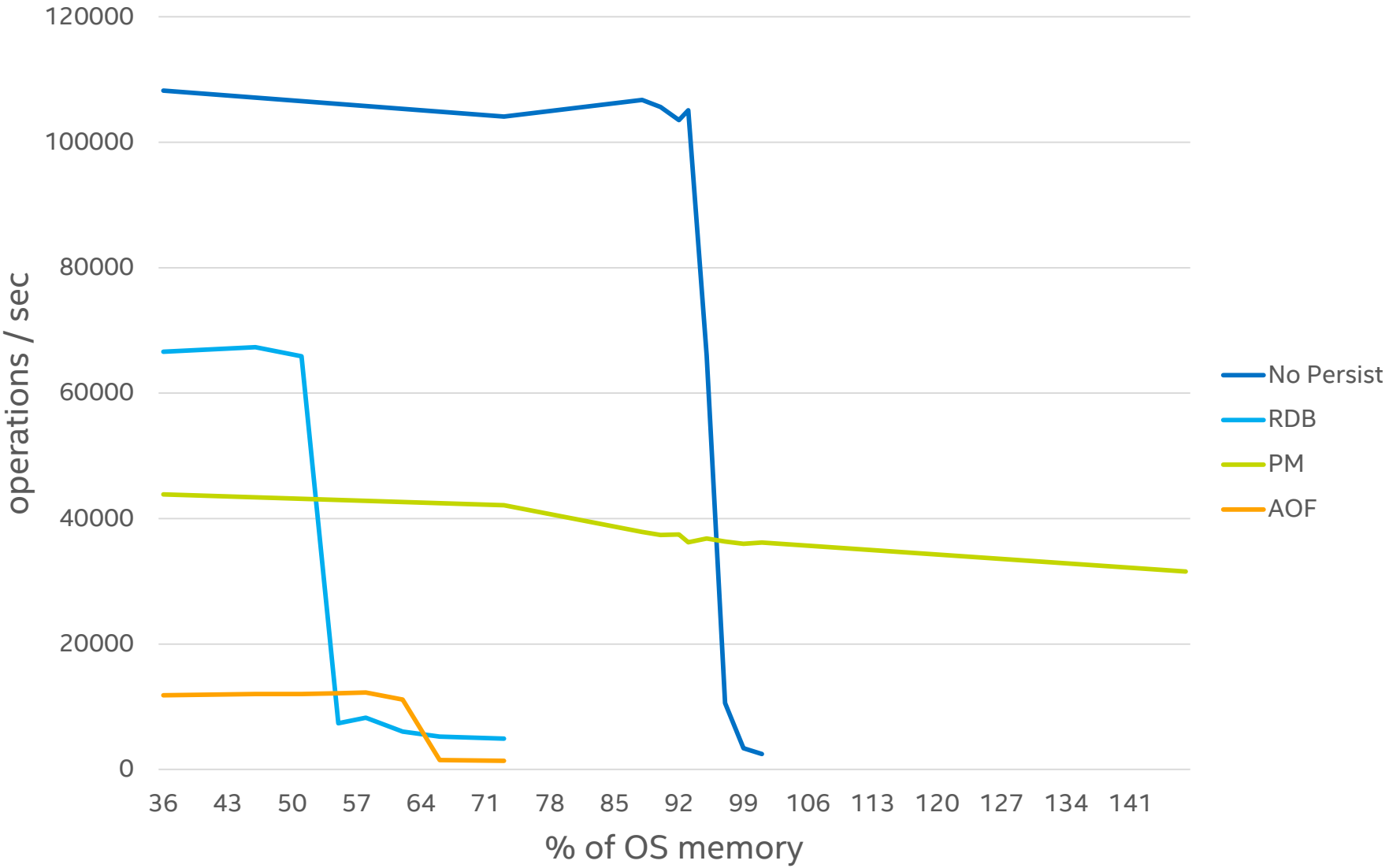
Running out of DRAM



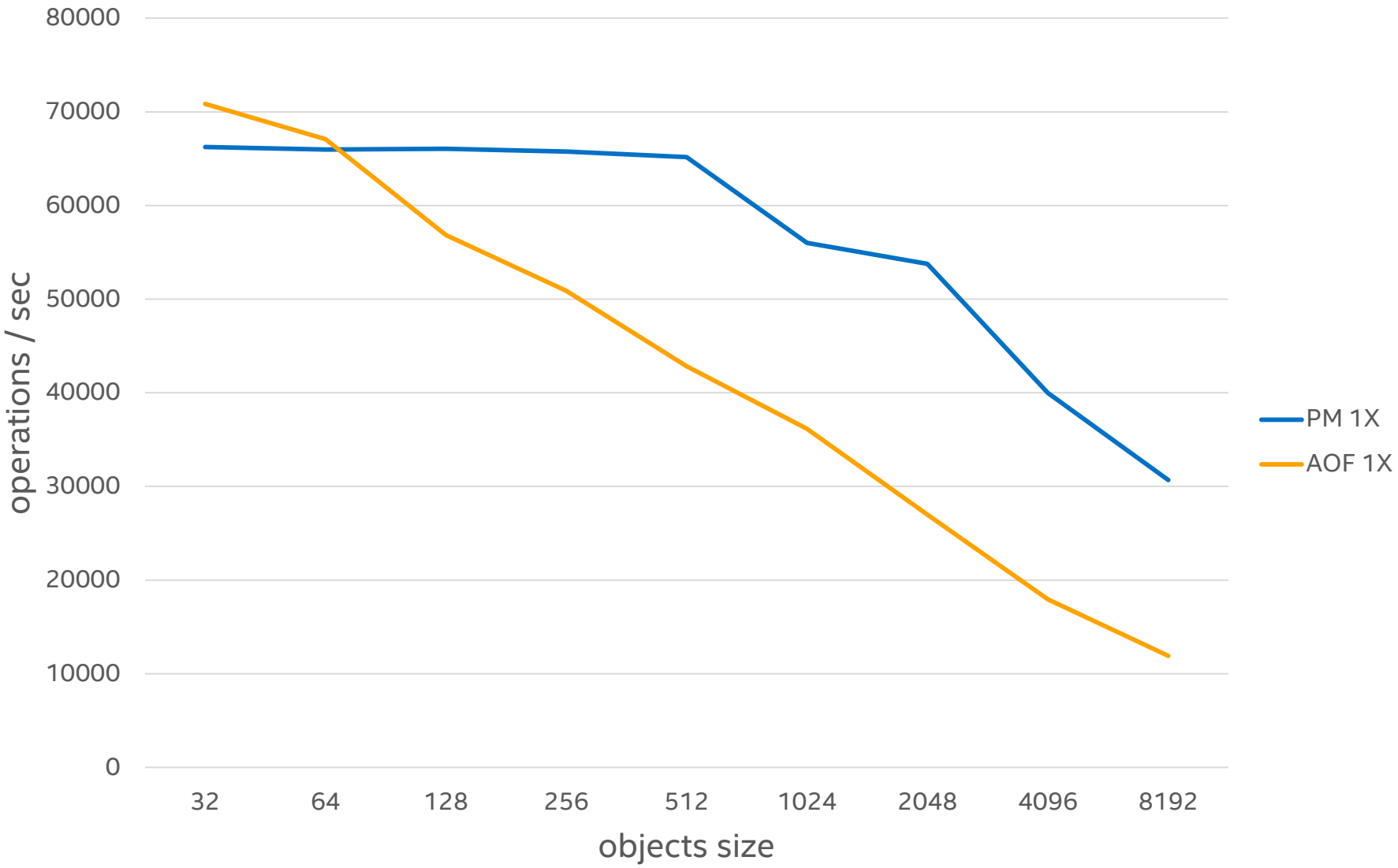
Running out of DRAM



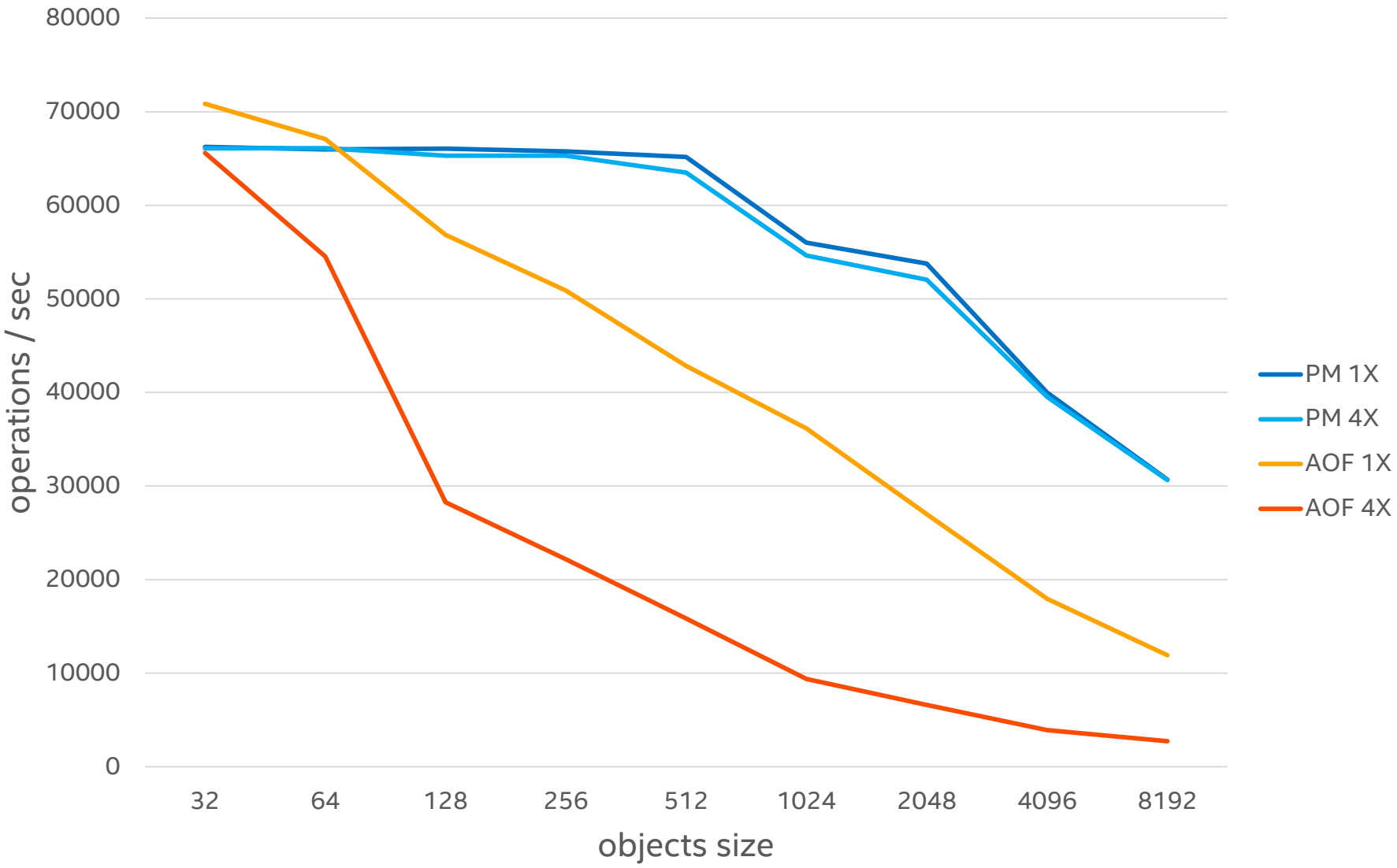
Running out of DRAM



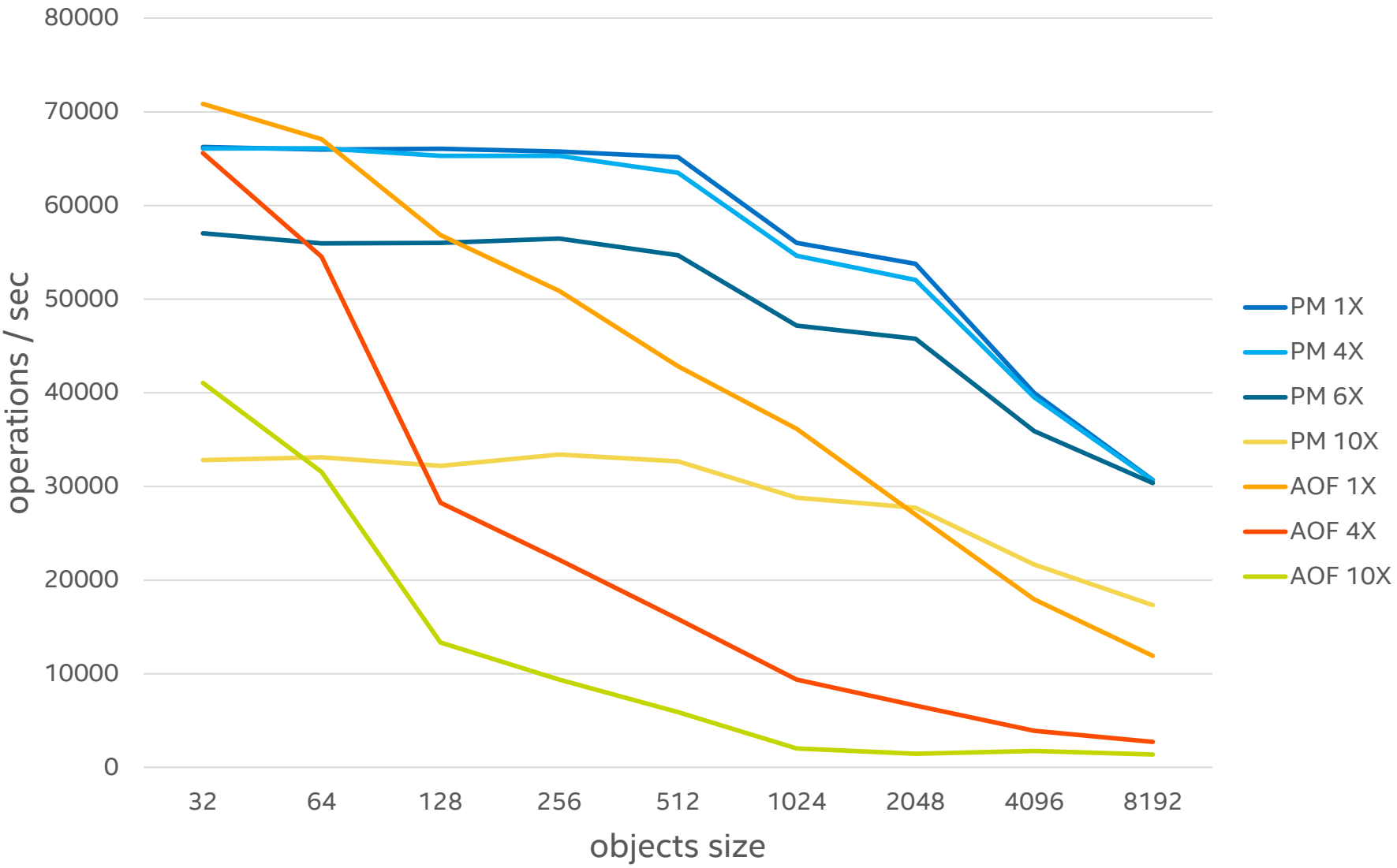
CPU usage


80000
70000
60000
50000
40000
30000
20000
10000
0
operations / sec
32
64
128
256
512
1024
2048
4096
8192
objects size
PM 1X
AOF 1X

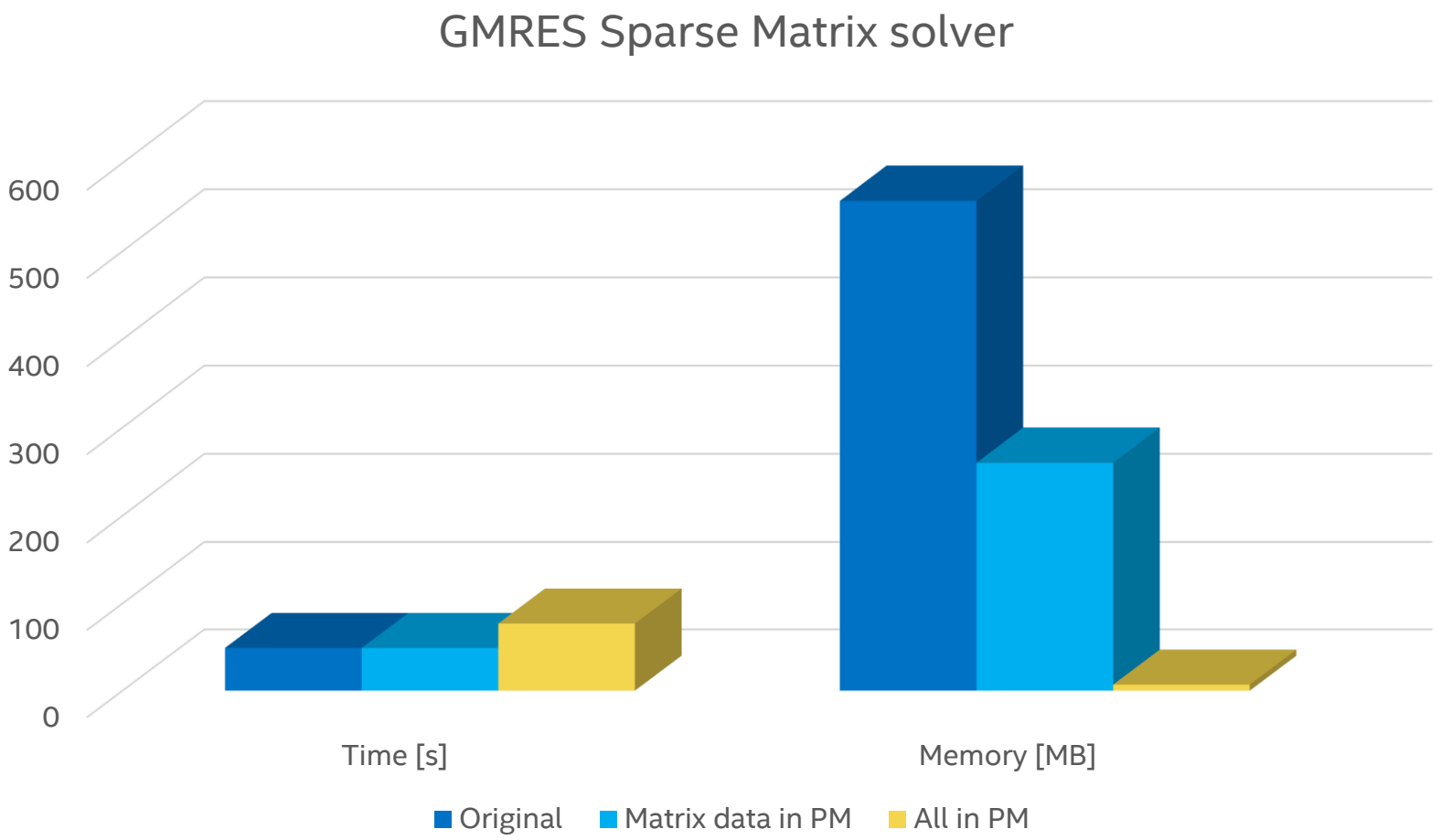
CPU usage


80000
70000
60000
50000
40000
30000
20000
10000
0
operations / sec
32
64
128
256
512
1024
2048
4096
8192
objects size
PM 1X
PM 4X
AOF 1X
AOF 4X

CPU usage


80000
70000
60000
50000
40000
30000
20000
10000
0
operations / sec
32
64
128
256
512
1024
2048
4096
8192
objects size
PM 1X
PM 4X
PM 6X
PM 10X
AOF 1X
AOF 4X
AOF 10X

Moving even more data



Modifying app engine

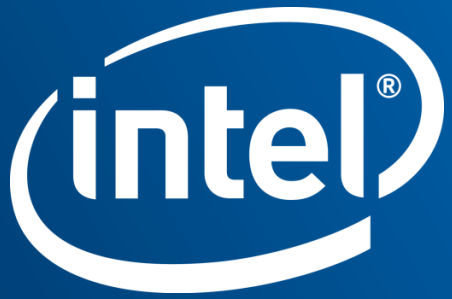
```
for (int i=0; i<NUMBER_OF_ITERATIONS; i++) {
        result = calculateThis(i, result);
}
```

```
i_pm = 0; //at first runtime of app only

...
...

for (i_pm; i_pm<NUMBER_OF_ITERATIONS; i_pm++) {
        TX_BEGIN(pool) {
                result_pm = calculateThis(i_pm, result_pm);
        } TX_END
}
i_pm = 0;
```

Possibilities

Choices

Even more fun …


intel®