Flexible Data Placement
Open Source Ecosystem

Adam Manzanares, Director, Samsung
Joel Granados, Software Engineer, Samsung
Arun George, Associate Technical Director, Samsung Semiconductor India Research

www.storagedeveloper.org
FDP Ecosystem

- Libraries
  - CacheLib
  - RocksDB

- Supporting Projects
  - FIO
  - NVMe CLI
  - QEMU
  - xNVMe
  - IO Passthru
Libraries

Cachelib - RocksDB
Libraries | Cachelib

What is Cachelib? – Motivation for FDP – On boarding FDP – Status?
Local cache leveraging DRAM and SSD (Navy)

Navy = engine for small and large items

- Large Items – (BlockCache 1KB..16MB)
  - Sequential write (Good for SSDs)
  - IO pattern $\rightarrow$ WAF=\~1

- Small Items – (BigHash <1KB)
  - Random write (Bad for SSDs)
  - IO pattern $\rightarrow$ High WAF

- LBA range for each engine type
Libraries | Cachelib | Motivation for FDP

- **Problem:**
  - Large item mix with small item
  - Blocks have no particular order
  - Small items update/invalidate faster
  - Invalid blocks peppered all over
  - GC works harder to create valid Rus
  - WAF increases

- **Expectation**
  - Segregate small and large items
  - Facilitates GC
  - Brings WAF down
- Uses `io_uring_cmd`
- Speaks to NVMe driver through char device (Ex: `/dev/ng01`)
- One Placement identifier (PID) per engine type
- Add FDP PID to write functions
- Add `io_uring_cmd` infrastructure
- New FDP Device type
- WAF for 80% and 70% utilization
- WAF =~ 1 when FDP enabled
  - Even for high utilization (80%)
- Throughput maintained
- PR:
  https://github.com/facebook/CacheLib/pull/247
- Future
  - Further segregation of Large Items
  - Generalize FDP library?
Libraries / RocksDB

What is RocksDB? – Motivation for FDP – On boarding FDP – Status?
A key/value Storage Engine
- It is a Log Structure Data Base
  - It has an in memory memtable
  - Which is flushed to leveled SSTables
- Executes Regular compaction
- Time to Live increases downwards
- Storage can be abstracted as a plugin
Problem

- Levels are all mixed
- Blocks have no particular order
- Lower levels update/invalidate faster
- Invalid blocks are peppered all over
- GC works harder to create valid RUs
- WAF increases

Expectation

- Segregation of all levels
- Easier GC as entire RUs organically invalidated
- Bring WAF down
- Uses `io_uring_cmd` through xNVMe
- New RockDB environment plugin
- New Writer classes to forward PIDs
- Deallocation on every SST deletion
- We use RocksDB `::WriteLifeTimeHint`:
  - `WLTH_{NOT_SET,NONE}` → Placement ID0
  - `WLTH_SHORT` → Placement ID1
  - `WLTH_MEDIUM` → Placement ID2
  - `WLTH_LONG` → Placement ID3
  - `WLTH_EXTREME` → Placement ID4
- WAF = ~1 when FDP enabled
  - Even for high utilization (~80%)
- Experimental
- Testbed for FDP and other Data Placement approaches
Supporting Projects

FIO – NVMe-CLI – QEMU – xNVMe – IO Passthru
Projects / FIO

Motivation for FDP – On boarding FDP – Status?
Projects | FIO | Motivation for FDP

- Easy FDP “hello world” JOB
- Clarify hypothesis
- Test performance
  - Compare FDP with no FDP
  - Run FDP with your favorite JOB
- Test out different paths to the device
  - SPDK (bypass kernel) –engine=SPDK
  - io_uring passtrhu (bypass block layer) – engine=io_uring_cmd
  - xNVMe (ioctl, io_uring, SPDK) -- engine=xnvme
Projects | FIO | On Boarding FDP

- Read available Placement Identifiers (PID) from device
- Attach a Placement identifier to the outgoing write
- Assign PIDs to FIO JOBs
- Control PID selection within a JOB
  - Random
  - Round Robin
Projects | FIO | Status

- Available since 3.33 (Nov 2022)
- Features
  - (De)Activate: --fdp=1
  - Select PIDs: --fdp_pli=[OFFSET_LIST]
  - Selection method: --fdp_pli_select=[TYPE]
- Example Job
  - Write-heavy
  - random write
  - io_uring_cmd

```bash
#FDP.fio.job
[global]
filename=/dev/ng0n1
ioengine=io_uring_cmd
cmd_type=nvme
iodepth=32
bs=4K
fdp=1
time_based=1
runtime=1000

[write-heavy]
rw=randrw
rwmixwrite=90
fdp_pli=0,1,2,3
offset=0%
size=30%
```
Projects / NVMe-CLI

Motivation for FDP – On boarding FDP – Status?
Projects | NVMe-CLI | Motivation for FDP

- An easy way to talk FDP
  - NVMe Compliant
- A way to enable FDP
- A way to configure FDP
- Ask about the state of FDP
Projects | NVMe-CLI | On Boarding FDP

- Log helpers
  - Statistics
  - Events
  - Configurations
  - Reclaim Unit Handel Usage

- Additional helpers
  - Fdp-status
  - Fdp-update
  - fdp-set-events

- Add IO mgmt send/receive
  - Receive Active Time remaining
  - Receive available Writes
  - Reset Reclaim Units if they have been written
Projects | NVMe-CLI | Status

- Available upstream since v2.3 (Jan 2023)
- Here is how to activate FDP

# 1. Validate the FDP capability. 19th bit on.
```
nvme id-ctrl /dev/nvme0 | grep -i cattrt.
```

# 2. Delete NSs in the endurance group
```
nvme delete-ns /dev/nvme0 -n 1
```

# 3. Get log page command to print configs
```
nvme fdp configs /dev/nvme0 -e 1 -H
```

# 4. Enable FDP with config 0
# 0x1D -> Flexible Data Placement
# -c sends Index 0 and FDPE =1
```
nvme set-feature /dev/nvme0 -f 0x1D -c 1 -s
```

# This should print out that fdp is enabled
```
nvme get-feature /dev/nvme0 -f 0x1D -H
```

# Create an ns
```
NSZE=$(nvme id-ctrl /dev/nvme0 | grep -i tnvmcap \
   | sed "s/"/"g" | awk '{print $3/4096}')
nvme create-ns /dev/nvme0 -b 4096 --nsze=$NSZE \ 
   --ncap=$NSZE -p 0,1,2,3 -n 4
```

# Attach nvme namespace to controller
```
nvme attach-ns /dev/nvme0 --namespace-id=1 \ 
   --controllers=0x7
```

# Directives. Fifth bit set
```
nvme id-ctrl /dev/nvme0 | grep oacs
Projects | QEMU

Motivation for FDP – On boarding FDP – Status?
Projects | Qemu | Motivation

- Emulate an FDP device
- Development and simple testing without HW
- Identify how to onboard FDP without HW
- Debug FDP implementations
  - Tracing
  - Using a debugger
- Get inspired by QEMUs implementation
Projects | Qemu | On Boarding FDP

- IO management send/receive
- Support for directives (used by FDP)
- FDP Logs
  - FDP configurations
  - FDP RUH usage
  - FDP Stats
  - FDP Events
- FDP is always enabled
- Not Persistent, don’t reboot Qemu!
Projects | Qemu | Status

- Upstream. Available since v8.0
- New device arguments:
  - Enabled (--fdp=true/false)
  - Number of Reclaim Unit Handles (--fdp.nruh=#)
  - Number of Reclaim Groups (--fdp.nrg=#)
  - Reclaim Unit Size (--fdp.runs=#)
- Not in QEMU:
  - Timers (How long an RU is writable)
  - Not persistent
  - Enablement

```bash
-device virtio-net-pci,netdev=net0 -device virtio-rng-pci
-drive id=boot,file=./base.qcow2,format=qcow2,if=virtio,discard=unmap,media=disk,read-only=no -s
-device pcie-root-port,id=pcie_root_port0.chassis=1.slot=0
-device nvme-subsys,id=subsys0,fdp=true,fdp.nruh=8,fdp.nrg=32,fdp.runs=40960
-device nvme,id=ctrl0,serial=deadbeef,bus=pcie_root_port0,subsys=subsys0
-drive id=nvm-1,file=./nvm-1.img,format=raw,if=none,discard=unmap,media=disk,read-only=no
-device nvme-ns,id=nvm-1,drive=nvm-1,bus=ctrl0,nsid=1,logical_block_size=4096,physical_block_size=4096
```
Projects

xNVMe – IO Passthru
Projects | xNVMe

- FDP for library users
- FDP on multiple paths
- On Boarding
  - Add IO mgmt send/receive
  - FDP statistics
  - FDP Events
  - FDP Configurations
  - RUH Usage
  - FDP testing
- Status
  - Upstream since v0.7 (June 16)
Projects | IO Passthru

- Get hints down to device
- Two paths
  - Sync ioctl
  - Async io_uring_cmd
- Both paths currently available
- Related patchests
  - Char device v5.13
  - io_uring_cmd v5.19
Conclusions

Summary
Conclusion | We mentioned...

- **Libraries**
  - CacheLib
  - RocksDB
- **Supporting Projects**
  - FIO
  - NVMe CLI
  - QEMU
  - XNVMMe
  - IO Passthrue
Questions?
Please take a moment to rate this session.

Your feedback is important to us.