SPDK PROGRAMMING FRAMEWORK AND NVME-OF OPTIMIZATION

Ziye Yang

Senior Software Engineer

Network Platforms Group, Intel Corporation

Agenda

- SPDK programming framework
- Accelerated NVMe-oF via SPDK
- Conclusion

Agenda

- SPDK programming framework
- Accelerated NVMe-oF via SPDK
- Conclusion

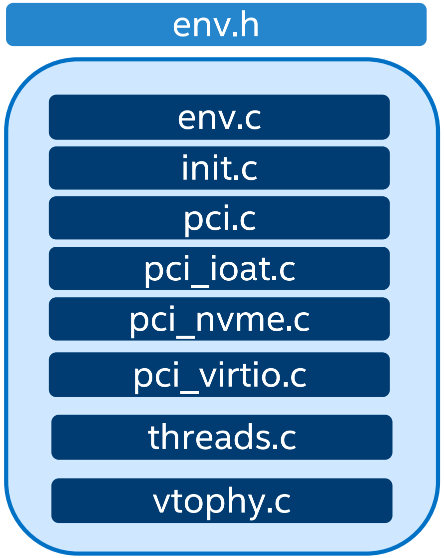
SPDK ENVIRONMENT ABSTRACTION

WHY AN ENVIRONMENT ABSTRACTION?

FLEXIBILITY FOR USER

ENVIRONMENT ABSTRACTION

- Memory allocation (pinned for DMA) and address translation
- PCI enumeration and resource mapping
- Thread startup (pinned to cores)
- Lock-free ring and memory pool data structures



ENVIRONMENT ABSTRACTION

Configurable:

```
./configure --with-env=...
```

Interface defined in `spdk/env.h`

Default implementation uses **DPDK** (`lib/env_dpdk`)

env.h

- env.c
- init.c
- pci.c
- pci_ioat.c
- pci_nvme.c
- pci_virtio.c
- threads.c
- vtophy.c

FLEXIBILITY: DECOUPLING AND DPDK ENHANCEMENTS

APPLICATION FRAMEWORK

HOW DO WE COMBINE SPDK COMPONENTS?

THE SPDK APP FRAMEWORK PROVIDES THE GLUE

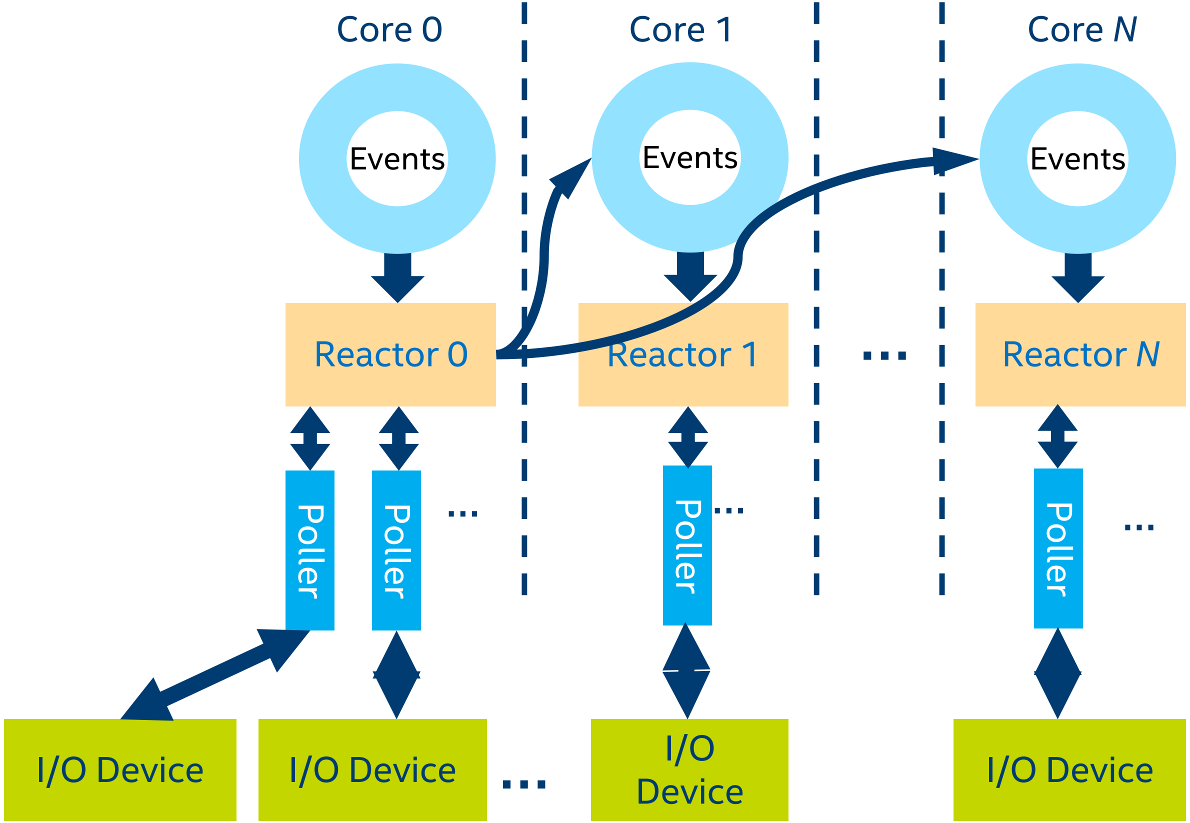
APP FRAMEWORK COMPONENTS

REACTOR

POLLER

EVENT

I/O CHANNEL


Core 0
Core 1
Core N
Events
Events
Events
Reactor 0
Reactor 1
...
Reactor N
Poller
Poller
...
Poller
...
Poller
...
I/O Device
I/O Device
...
I/O Device
I/O Device

POLLER

Essentially a “task” running on a reactor

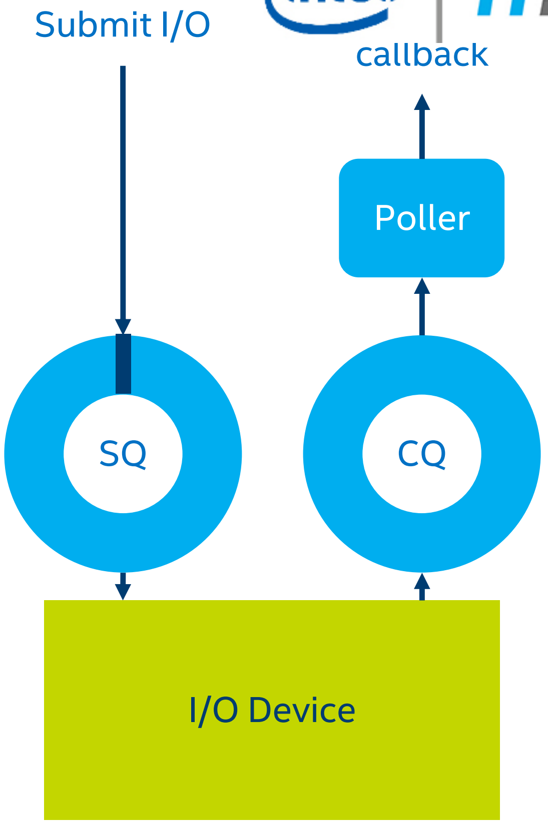
Primarily checks hardware for async events

Can run periodically on a timer

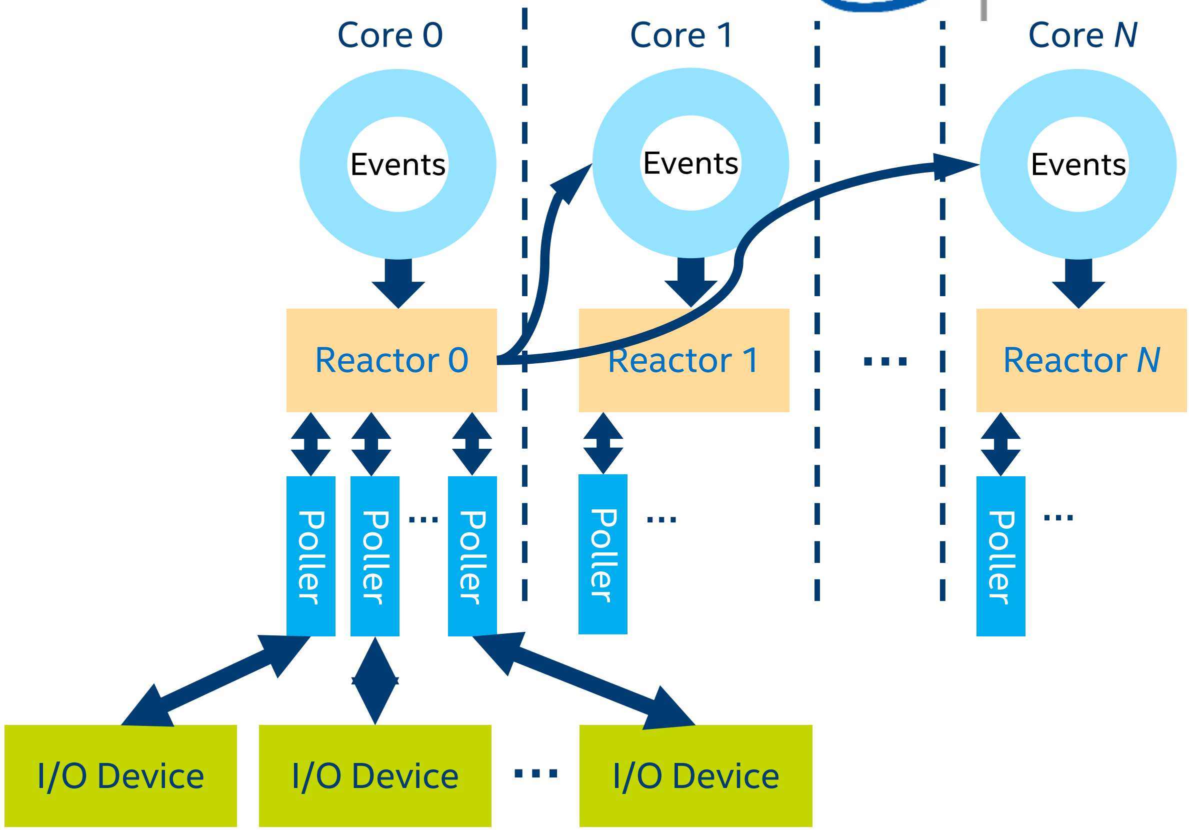
Example: poll completion queue

Callback runs to completion on reactor thread

Completion handler may send an event



EVENT


Core 0
Core 1
Core N
Events
Events
Events
Reactor 0
Reactor 1
···
Reactor N
Poller
Poller
···
Poller
Poller
···
Poller
···
I/O Device
I/O Device
···
I/O Device

EVENT

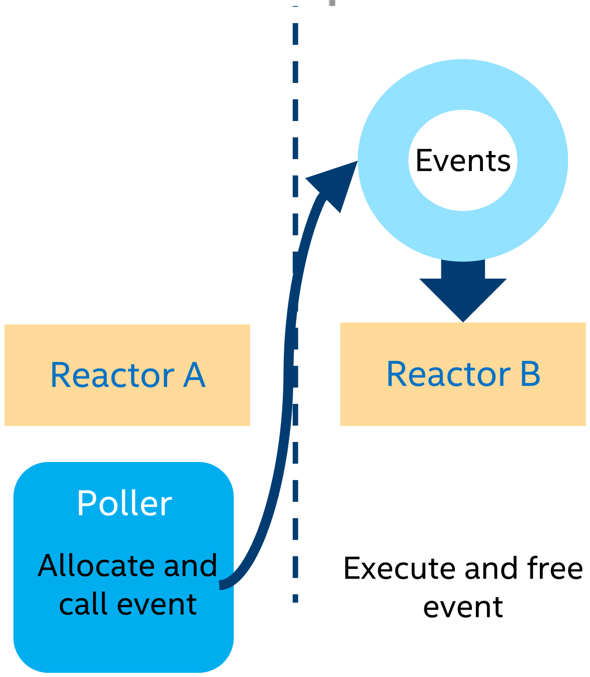
Cross-thread communication

Function pointer + arguments

One-shot message passed between reactors

Multi-producer/single-consumer ring

Runs to completion on reactor thread



I/O CHANNEL

Abstracts hardware I/O queues

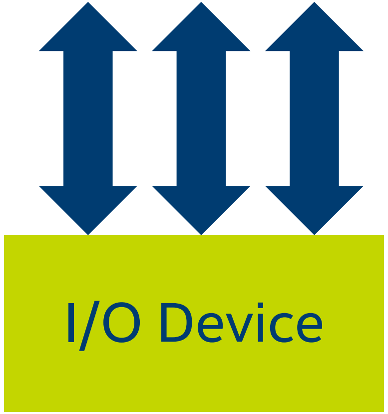
Register I/O devices

Create I/O channel per thread/device combination

Provides hooks for driver resource allocation

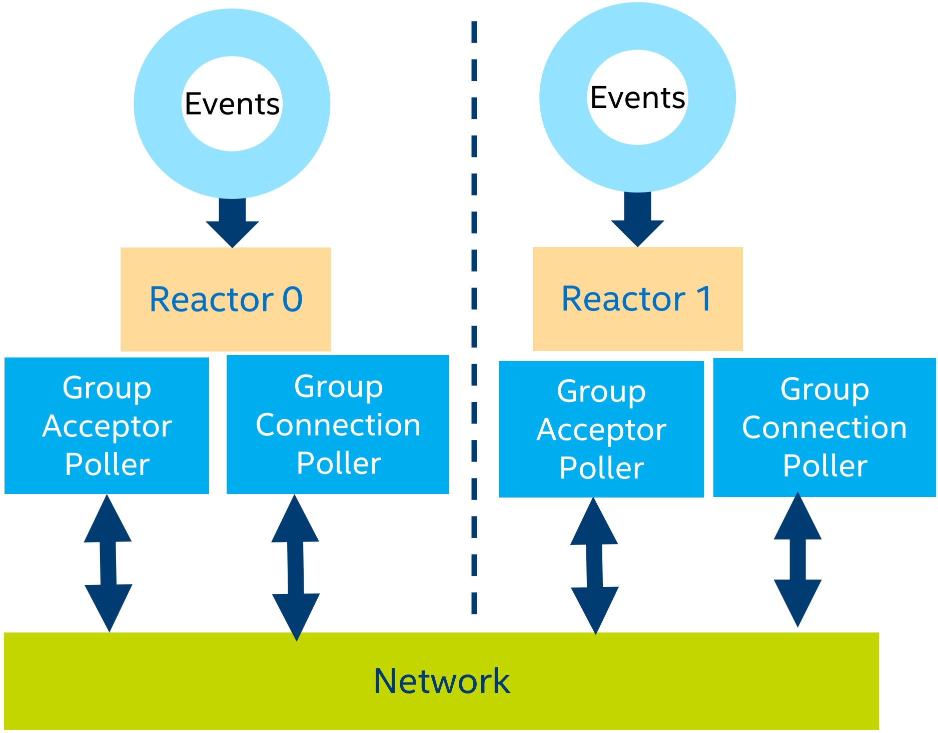
I/O channel creation drives poller creation

Pervasive in SPDK



NVME OVER FABRICS TARGET EXAMPLE

- nvmf_tgt_advance_state
  - spdk_nvmf_parse_conf (listen on transport)
  - NVMe-oF tgt I/O channel creation: spdk_nvmf_tgt_create
  - Group data poller creation in each core: Trigger the create_cb (spdk_nvmf_tgt_create_poll_group) of I/O channel, then we will have spdk_nvmf_poll_group_poll in each core
  - Group Acceptor network poller creation: spdk_nvmf_tgt_accept will be used to connect events in each core



NVME OVER FABRICS TARGET EXAMPLE

- Group Acceptor network poller handles connect events
- New qpair (connection) is allocated to different cores via Round Robin manner. Asynchronous message passing is used, then spdk_nvmf_poll_group_add is called.
- I/O request arrives over network, and handled by the group poller in the designated core.
- I/O submitted to storage
- Storage device poller checks completions
- Response sent

ALL ASYNCHRONOUS WORK IS DRIVEN BY POLLERS

Agenda

- SPDK programming framework
- Accelerated NVMe-oF via SPDK
- Conclusion

SPDK NVMe-oF Components

NVMe over Fabrics Target

- Released July 2016 (with spec)
- **Hardening:**
  - Intel test infrastructure
    Discovery simplification
    Correctness & kernel interop
- **Performance improvements:**
  - Read latency improvement
    Scalability validation (up to 150Gbps)
  - Event Framework enhancements
  - Multiple connection performance improvement (e.g., group transport polling,)

NVMe over Fabrics Host (Initiator)

- New component added in Dec 2016
- Performance improvements
  - Eliminate copy: now true zero-copy
  - SGL (single SGL element)

SPDK NVMe-oF transport work

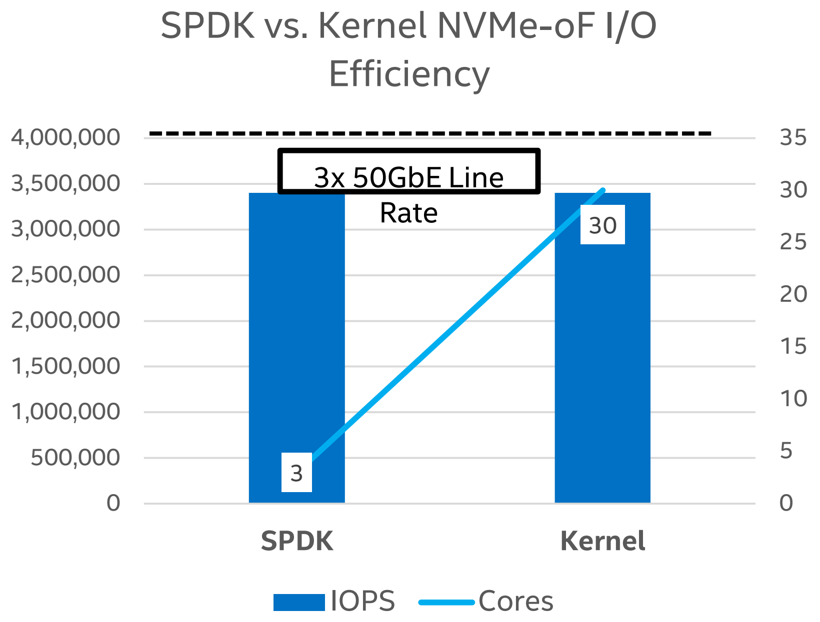
Existing work: RDMA transport

- **DPDK components used which is encapsulated in libspdk_env_dpdk.a, e.g.,**
  - PCI device management
  - CPU/thread scheduling
  - Memory management (e.g., lock free rings)
  - Log management

Upcoming work: TCP transport

- Kernel based TCP transport
- VPP/DPDK based user space TCP transport
  - Use DPDK Ethernet PMDs
  - Use user space TCP/IP stack (e.g., VPP)

NVMe-oF Target Throughput Performar

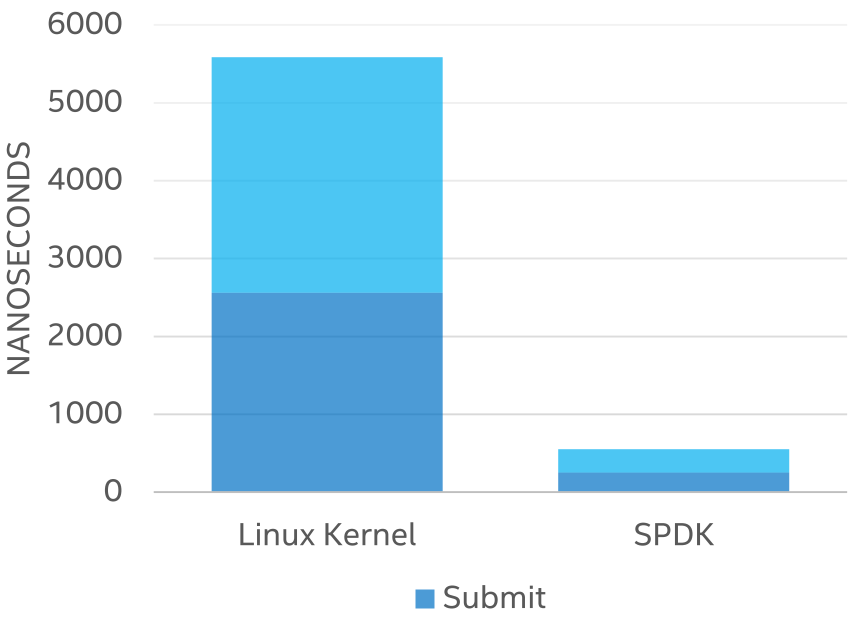


NVMe* over Fabrics Target Features	Realized Benefit
Utilizes NVM Express* (NVMe) Polled Mode Driver	Reduced overhead per NVMe I/O
RDMA Queue Pair group Polling	No interrupt overhead
Connections pinned to CPU cores	No synchronization overhead

SPDK reduces NVMe over Fabrics software overhead up to 10x!

System Configuration: Target system: Supermicro SYS-2028U-TN24R4T+, 2x Intel® Xeon® E5-2699v4 (HT off), Intel® Speed Step enabled, Intel® Turbo Boost Technology enabled, 8x 8GB DDR4 2133 MT/s, 1 DIMM per channel, 12x Intel® P3700 NVMe SSD (800GB) per socket, -1H0 FW; Network: Mellanox* ConnectX-4 LX 2x25Gb RDMA, direct connection between initiators and target; Initiator OS: CentOS* Linux* 7.2, Linux kernel 4.10.0, Target OS (SPDK): Fedora 25, Linux kernel 4.9.11, Target OS (Linux kernel): Fedora 25, Linux kernel 4.9.11 Performance as measured by: fio, 4KB Random Read I/O, 2 RDMA QP per remote SSD, Numjobs=4 per SSD, Queue Depth: 32/job. SPDK commit ID: 4163626c5c

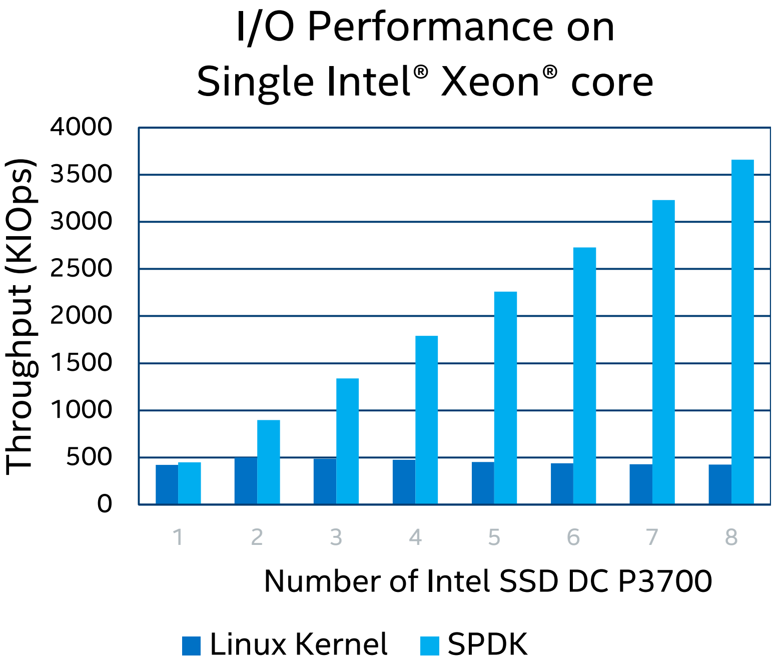
NVM Express* Driver Software Overhead



Kernel Source of Overhead	SPDK Approach
Interrupts	Asynchronous Polled Mode
Synchronization	Lockless
System Calls	User Space Hardware Access
DMA Mapping	Hugepages
Generic Block Layer	Specific for Flash Latencies

SPDK reduces NVM Express* (NVMe) software overhead up to 10x!

NVM Express* Driver Throughput Scalab



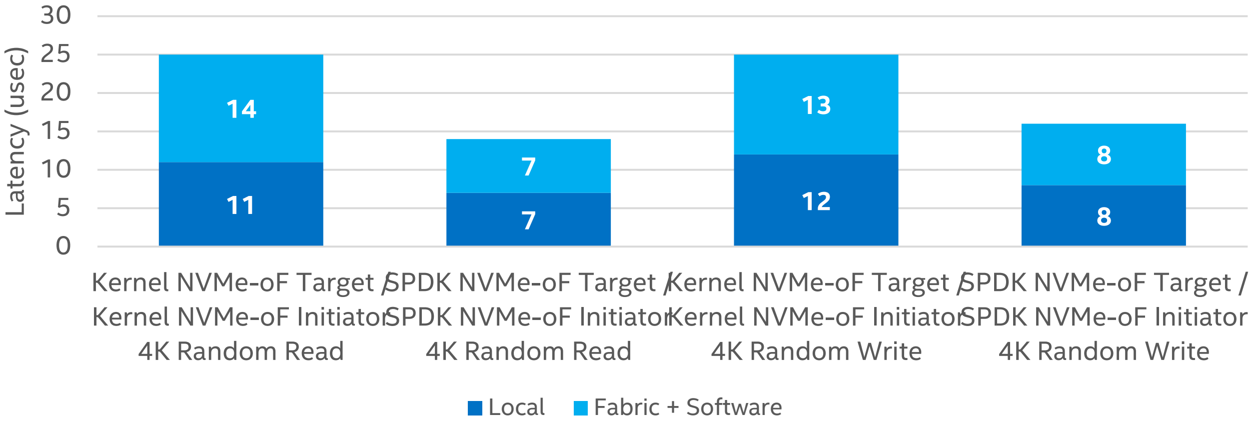
- Systems with multiple NVM Express* (NVMe) SSDs capable of millions of I/O per second
- Results in many cores of software overhead with kernel-based interrupt-driven driver model
- SPDK enables:
  - more CPU cycles for storage services
  - lower I/O latency

SPDK saturates 8 NVMe SSDs with a single CPU core!

System Configuration: 2x Intel® Xeon® E5-2695v4 (HT off), Intel® Speed Step enabled, Intel® Turbo Boost Technology disabled, 8x 8GB DDR4 2133 MT/s, 1 DIMM per channel, CentOS* Linux* 7.2, Linux kernel 4.10.0, 8x Intel® P3700 NVMe SSD (800GB), 4x per CPU socket, FW 8DV101H0, I/O workload 4KB random read, Queue Depth: 128 per SSD, Performance measured by Intel using SPDK perf tool, Linux kernel data using Linux AIO

SPDK Host + Target vs. Kernel Host + Ta

Avg. I/O Round Trip Time
Kernel vs. SPDK NVMe-oF Stacks
Coldstream, Perf, qd=1



SPDK reduces Optane NVMe-oF latency by 44%, write latency by 36%!

System Configuration: 2x Intel® Xeon® E5-2695v4 (HT on, Intel® Speed Step enabled, Intel® Turbo Boost Technology enabled, 64GB DDR4 Memory, 8x 8GB DDR4 2400 MT/s, Ubuntu 16.04.1, Linux kernel 4.10.1, 1x 25GbE Mellanox 2P CX-4, CX-4 FW= 14.16.1020, mlx5_core= 3.0-1 driver, 1 ColdStream, connected to socket 0, 4KB Random Read I/0 1 initiators, each initiator connected to bx NVMe-oF subsystems using 2P 25GbE Mellanox. Performance measured by Intel using SPDK perf tool, 4KB Random Read I/O, Queue Depth: 1/NVMe-oF subsystem. numjobs 1, 300 sec runtime, direct=1, norandommap=1, FIO-2.12, SPDK commit # 42eade49

Agenda

- SPDK programming framework
- Accelerated NVMe-oF via SPDK
- Conclusion

Conclusion

- In this presentation, we introduce
  - SPDK library
  - The accelerated NVMe-oF target built from SPDK library
- SPDK proves to be useful to accelerate storage applications equipped with NVMe based devices
- Call for action:
  - Welcome to use SPDK in storage area (similar as using DPDK in network) and contribute into SPDK community.

Q&A

Notices & Disclaimers

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Perform
configuration.

No computer system can be absolutely secure.

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. For more complete information about performance and benchmark results, visit http://www.intel.com/benchmarks .

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit http://www.intel.com/benchmarks .

Benchmark results were obtained prior to implementation of recent software patches and firmware updates intended to address exploits referred to as "Spectre" and "Meltdown." Implementation of these updates may make these results inapplicable to your device or system.

Intel® Advanced Vector Extensions (Intel® AVX)* provides higher throughput to certain processor operations. Due to varying processor power characteristics, utilizing AVX instructions may cause a) some parts to operate at less than the rated frequency and b) some parts with Intel® Turbo Boost Technology 2.0 to not achieve any or maximum turbo frequencies. Performance varies depending on hardware, software, and system configuration and you can learn more at http://www.intel.com/go/turbo.

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Cost reduction scenarios described are intended as examples of how a given Intel-based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.

Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.

intel

IT大咖说 知识共享平台

intel®

experience what's inside™