Approaches that Combine Intel® Optane™ SSDs with Intel® QLC technology Offer Solutions Competitive with TLC

Kapil Karkra, Intel
Michal Wysoczanski, Intel
Piotr Wysocki, Intel
Santa Clara, CA
August 2019
Notices and Disclaimers

Intel technologies’ features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer.

No computer system can be absolutely secure.

Performance results are based on testing as of July 2019 and may not reflect all publicly available security updates. See configuration disclosure for details. No product can be absolutely secure. Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors.

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase.

Intel, the Intel logo, Intel Optane, Xeon, and others are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others.

© Intel Corporation.
Intel® Optane™ SSD + Intel® QLC 3D NAND SSDs

Intel® QLC 3D NAND SSDs
- Compelling cost and density
- Good read performance
But...
- Write bandwidth dependent on workload (WAF)
- Low endurance

Intel® Optane™ SSD:
- Write bandwidth independent of workload (no WAF, in-place overwrites)
- Very low latency and great QoS
- Maximum performance even at low queue depth
- High endurance

Why not to combine both to create an optimal solution?
A Note on Workloads Suitable For Intel® QLC Technology

- Workloads with Write Amplification Factor equal to unity (WAF=1) are not always suitable for QLC due to its low endurance
  - e.g., a streaming workloads with a Time to Live (TTL=18 days) policy might be suitable, but not a workload with TTL = 1 minute

QLC Media Writes = WAF * Host Writes

With NVMe* features like Streams and ZNS, the best software can achieve is WAF =1, at which point:

QLC Media Writes = Host Writes

- How do we make QLC Media Writes << Host Writes?
- Hint: Trap some host writes upstream in the memory/storage hierarchy in the Intel® Optane™ SSD

The key question that we answer next is: What kind of data is suitable for Intel® Optane™ SSDs?

*Other names and brands may be claimed as the property of others.
How to combine Intel® Optane™ SSDs and Intel® QLC Technology into O+Q?

- **Goal:** Reduce amount of writes that goes to Intel® QLC technology ($\alpha$ – Write Reduction Factor).
- **How:** Place on Intel® Optane™ SSD data that is small enough and generates a lot of writes ($\beta$ – Write Invalidation Factor).
- **Key:** PRISM – Data classification and separation according to WIF. Need to separate different data classes that meet WIF and WRF requirements.
- **Extra:** Shape/reduce the workload part that goes to QLC to improve WAF (e.g., classify data based on data lifetime into classes in Intel Optane SSDs and place them on separate zones(streams) in QLC; or stage data on Intel Optane SSDs, compress it to reduce it, and place compressed data on QLC.

There is a need for software that would provide PRISM. But what kind of data should we look for?
What kind of data should go to Intel® Optane™ SSDs?

- Through several more macro and micro benchmarks, a thumb rule emerged: Place TML+H on Intel® Optane™ SSDs

- **Temp data** – The intermediate data that’s discarded upon arriving at final result (e.g., data swapped out from memory)
- **Metadata** – Metadata (e.g., indexes that are updated and read a lot)
- **Logs** – Journals, write ahead logs, redo logs, undo logs, binary logs, transaction logs, Parallel raft or Paxos logs
- **Hot data** – The data that is read frequently (e.g., popular songs)
**Use case 1: RocksDB**

**PRISM** for RocksDB:

- Write lifetime hints – Open CAS in block layer consumes WLTH provided by RocksDB
  - WAL and levels 0-3 are placed on Intel® Optane™ SSD
  - Lower levels go directly to Intel® QLC technology
- Additionaly, file system metadata is also placed on Intel Optane SSD

<table>
<thead>
<tr>
<th>Rocksdb workload through the Prism</th>
<th>Occupancy (GB)</th>
<th>Writes (GB)</th>
<th>WRF (α)</th>
<th>WIF (β)</th>
<th>Type of data (HTML)</th>
</tr>
</thead>
<tbody>
<tr>
<td>XFS Metadata + Journal</td>
<td>2.0</td>
<td>5.7</td>
<td>0%</td>
<td>3</td>
<td>M, L</td>
</tr>
<tr>
<td>WAL</td>
<td>4.0</td>
<td>688.9</td>
<td>5%</td>
<td>172</td>
<td>L</td>
</tr>
<tr>
<td>L0</td>
<td>1.0</td>
<td>689.6</td>
<td>5%</td>
<td>690</td>
<td>T</td>
</tr>
<tr>
<td>L1</td>
<td>1.0</td>
<td>1012.3</td>
<td>8%</td>
<td>1012</td>
<td>T</td>
</tr>
<tr>
<td>L2</td>
<td>11.8</td>
<td>3157.0</td>
<td>25%</td>
<td>268</td>
<td>T</td>
</tr>
<tr>
<td>L3</td>
<td>100.0</td>
<td>4511.9</td>
<td>36%</td>
<td>45</td>
<td>T</td>
</tr>
<tr>
<td>L4/L5</td>
<td>1000.0</td>
<td>2601.9</td>
<td>21%</td>
<td>3</td>
<td>X</td>
</tr>
<tr>
<td>Total (GB)</td>
<td>1119.8</td>
<td>12667.3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

79% of α can fit within Intel® Optane SSD

**Endurance/Performance Metrics**

<table>
<thead>
<tr>
<th>Benchmark: fillseq+readwhilewriting</th>
<th>TLC</th>
<th>QLC</th>
<th>O+Q vs. TLC</th>
<th>O+Q vs. QLC</th>
</tr>
</thead>
<tbody>
<tr>
<td>EPBW</td>
<td>24.92</td>
<td>8.86</td>
<td>37.22</td>
<td>149%</td>
</tr>
<tr>
<td>EDPWD</td>
<td>1.71</td>
<td>0.63</td>
<td>2.65</td>
<td>156%</td>
</tr>
<tr>
<td>write bandwidth (MB/s)</td>
<td>29.83</td>
<td>25.96</td>
<td>40.32</td>
<td>135%</td>
</tr>
<tr>
<td>p50</td>
<td>163.93</td>
<td>7961.48</td>
<td>74.07</td>
<td>45%</td>
</tr>
<tr>
<td>p75</td>
<td>423.68</td>
<td>1263.73</td>
<td>240.07</td>
<td>57%</td>
</tr>
<tr>
<td>p99</td>
<td>5041.73</td>
<td>7961.48</td>
<td>1281.36</td>
<td>25%</td>
</tr>
<tr>
<td>p99.9</td>
<td>17866.50</td>
<td>18953.32</td>
<td>4213.98</td>
<td>24%</td>
</tr>
<tr>
<td>p99.99</td>
<td>30199.82</td>
<td>73848.21</td>
<td>13682.56</td>
<td>45%</td>
</tr>
</tbody>
</table>

See „Notes on benchmarks“ for benchmark specification
Use case 2: MongoDB

**PRISM for MongoDB:**

- Use Open CAS file path classification to put directory with MongoDB journal on Intel® Optane™ SSD

<table>
<thead>
<tr>
<th>Mongodb workload through the prism</th>
<th>Occupancy (GB)</th>
<th>Writes (GB)</th>
<th>WRF (α)</th>
<th>WIF (β)</th>
<th>Type of data (HTML)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Journal</td>
<td>3.0</td>
<td>27478.3</td>
<td>97%</td>
<td>9159</td>
<td>L</td>
</tr>
<tr>
<td>Data</td>
<td>750.0</td>
<td>776.7</td>
<td>3%</td>
<td>1</td>
<td>X</td>
</tr>
<tr>
<td>Total</td>
<td>753.0</td>
<td>28255.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mongodb (journal on Intel® Optane™ SSD)</th>
<th>Endurance/Performance Metrics</th>
<th>Relative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark: YCSB Workload A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rd:wr 50:50 (zipf θ = 0.99)</td>
<td>TLC</td>
<td>O+Q</td>
</tr>
<tr>
<td></td>
<td>0.27</td>
<td>3.01</td>
</tr>
<tr>
<td></td>
<td>1120%</td>
<td></td>
</tr>
<tr>
<td>OVERALL Throughput (Ops/s)</td>
<td>28935</td>
<td>50703</td>
</tr>
<tr>
<td></td>
<td>175%</td>
<td></td>
</tr>
<tr>
<td>UPDATE 99thPercentileLatency (us)</td>
<td>909</td>
<td>419</td>
</tr>
<tr>
<td></td>
<td>46%</td>
<td></td>
</tr>
<tr>
<td>OVERALL Throughput (Ops/s)</td>
<td>6155</td>
<td>1124</td>
</tr>
<tr>
<td></td>
<td>18%</td>
<td></td>
</tr>
<tr>
<td>Document Size 400B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OVERALL Throughput (Ops/s)</td>
<td>34853</td>
<td>41621</td>
</tr>
<tr>
<td></td>
<td>119%</td>
<td></td>
</tr>
<tr>
<td>UPDATE 99thPercentileLatency (us)</td>
<td>705</td>
<td>543</td>
</tr>
<tr>
<td></td>
<td>77%</td>
<td></td>
</tr>
<tr>
<td>OVERALL Throughput (Ops/s)</td>
<td>1752</td>
<td>1399</td>
</tr>
<tr>
<td></td>
<td>80%</td>
<td></td>
</tr>
</tbody>
</table>

See „Notes on benchmarks“ for benchmark specification
- Transparent to existing applications – middleware that uses regular block interface API
- Intel® Optane™ SSD and Intel® QLC technology accessed through regular block API
- Uses Open CAS as O+Q solution vehicle (PRISM, shaping, reducing, and read caching)
- Could be implemented as Linux kernel block device or SPDK bdev
Build PRISMs (data classifiers) with Open CAS to identify data classes in the workload with high $\beta$ (typically, this is TML+H).

Place as many higher $\beta$ data classes as possible on Intel® Optane™ SSDs to meet $\alpha$ criteria.

Use the remaining capacity of Intel Optane SSD to shape/reduce writes or as a read cache.

**Conclusion**

- **HDD**
- **Tape**
- **Persistent Memory**
- **DRAM**
- **CPU GPU caches**

- **Intel® Optane™ SSD**
- **Intel® QLC 3D Nand SSD**

**Shorter lifetime data e.g., TML+H**

**Longer lifetime data**
Extra: Shape/Reduce/Read Cache

**Shaping**
- Writes that are not pinned on Intel® Optane™ SSDs go to Intel Optane SSD write buffer portion
- Data in buffer partitioned, based on streams classifier
- Flushing of data performed in buckets of size equal to NAND drive band size
  - Only one bucket at a time
  - Band filled with data with same stream (e.g., same lifetime)
- Reads are handled directly from NAND drive (except for data that has not been flushed yet)

**Reducing**
- Compression
- Triple replication in Intel Optane SSD and Erasure Coding in QLC
Open CAS

- Open source caching engine
- Environment independent core and platform specific adapters
- Available adapters:
  - Linux kernel block device driver
  - Storage Performance Development Kit (SPDK) bdev
- Available on GitHub (https://github.com/Open-CAS)
- Tools for building solution:
  - Classifiers (PRISM)
  - Shaping/cleaning policies
Notes on benchmarks

RocksDB

CPU Intel(R) Xeon(R) CPU E5-2699 v4 @ 2.20GHz, 2 sockets, 22 cores, memory 256GB, BIOS Version: SESC610.86B.01.01.0016.033120161139. Release Date: 03/31/2016
Fedora 25 kernel 4.13.16, RocksDB v 5.17.2.

Drives Used:
- Intel® SSD DC P4500 8TB
- Intel® SSD D5 P4320 7.68 TB
- Intel® Optane™ SSD DC P4800X 375GB

Preparation phase:
- db_bench -db=mem rioclouds -num_levels=6 -key_size=32 -value_size=1024 -block_size=4096 -cache_size=$(8 * GiB) -cache_numshardbits=6 -compression_type=None -compression_ratio=0.5 -hard_rate_limit=2 -rate_limit_delay_max_milliseconds=1000000
- write_buffer_size=$(1024 * MiB) -max_write_buffer_number=4 -target_file_size_base=$(128 * MiB) -max_bytes_for_level_base=$(1024 * MiB) -max_bytes_for_level_multiplier=10 -sync=0 -verify_checksum=1 -level0_stop_writes_trigger=24 -level0_slowdown_writes_trigger=16
- level0_stop_writes_trigger=24 -benchmarks=filesiq -use_existing_db=0
- num=$(key_no) -threads=1

Benchmark phase:
- db_bench -db=mem rioclouds -num_levels=6 -key_size=32 -value_size=1024 -block_size=4096 -cache_size=$(8 * GiB) -cache_numshardbits=6 -compression_type=None -compression_ratio=1 -hard_rate_limit=2 -rate_limit_delay_max_milliseconds=1000000
- write_buffer_size=$(1024 * MiB)
- max_write_buffer_number=4
- target_file_size_base=134217728
- max_bytes_for_level_base=1073741824
- sync=0 -verify_checksum=1 -pin_l0_filter_and_index_blocks_in_cache=false -cache_index_and_filter_blocks=false -mmap_read=0
- max_background_compactions=32 -max_background_flushes=32 -level0_file_num_compaction_trigger=7 -level0_slowdown_writes_trigger=16
- level0_stop_writes_trigger=24
- benchmarks=readwrite
- use_existing_db=1 -stats_interval=5000000 -num=$(key_no) -threads=4

MongoDB

CPU Intel(R) Core(TM) i7-4960X CPU @ 3.60GHz, 1 socket, memory 16GB
CentOS 7.6 (kernel 3.10.0-957), MongoDB v 4.0.6, YCSB 0.15.0

Drives Used:
- Intel® SSD DC P4900
- Intel® Optane™ SSD DC P4800X 375GB

Preparation phase:
- ycsb load mongodb -s -p recordcount=100000000 -threads 16 -P workloads/workloada -p fieldlength=1024 -p fieldcount=8 -p requestdistribution=zipfian -p mongodbs=localhost:27017
- ycsb run mongodb -s -t -p operationcount=1000000000 -threads 16 -P workloads/workloada -p fieldlength=1024 -p fieldcount=8 -p requestdistribution=zipfian -p mongodbs=localhost:27017

Benchmark phase:
- ycsb load mongodb -s -p recordcount=100000000 -threads 16 -P workloads/workloada -p fieldlength=1024 -p fieldcount=8 -p requestdistribution=zipfian -p mongodbs=localhost:27017
- ycsb run mongodb -s -t -p operationcount=1000000000 -threads 16 -P workloads/workloada -p fieldlength=1024 -p fieldcount=8 -p requestdistribution=zipfian -p mongodbs=localhost:27017