BENEFITS OF THE INTEL® MEMORY DRIVE TECHNOLOGY FOR SCIENTIFIC APPLICATIONS

Vladimir Mironov, Senior Researcher, Lomonosov Moscow State University

Yuri Alexeev, Principal Project Specialist, Argonne National Laboratory

Andrey Kudryavtsev, SSD Solution Architect, Intel Corporation

Igor Chernykh, Head of the High Performance Simulation Lab, ICMMG SB RAS
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.

Intel, the Intel logo, Xeon, Intel Optane, 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.
Delivering an industry leading combination of low latency, high endurance, QoS and high throughput, the Intel® Optane™ SSD is the first solution to combine the attributes of memory and storage. This innovative solution is optimized to break through storage bottlenecks by providing a new data tier. It accelerates applications for fast caching and storage, increasing scale per server and reducing transaction cost. Data centers based on the latest Intel® Xeon® processors can now also deploy bigger and more affordable datasets to gain new insights from larger memory pools.

1. Responsiveness defined as average read latency measured at queue depth 1 during 4k random write workload. Measured using FIO 2.15. Common configuration - Intel 2U PCSD Server ("Wildcat Pass"), OS CentOS 7.2, kernel 3.10.0-327.el7.x86_64, CPU 2 x Intel® Xeon® E5-2699 v4 @ 2.20GHz (22 cores), RAM 396GB DDR @ 2133MHz. Intel drives evaluated - Intel® Optane™ SSD DC P4800X 375GB, Intel® SSD DC P3700 1600GB, Intel® SSD DC P4600 1600GB. Samsung drives evaluated – Samsung® SSD PM1725a, Samsung® SSD PM1725, Samsung® PM963, Samsung® PM953. Micron drive evaluated – Micron® 9100 PCIe® NVMe™ SSD. Toshiba drives evaluated – Toshiba® ZD6300. Test – QD1 Random Read 4K latency, QD1 Random RW 4K 70% Read latency, QD1 Random Write 4K latency using fio-2.15.
Predictably Fast Service

1. Common Configuration - Intel 2U PCSD Server ("Wildcat Pass"), OS CentOS 7.2, kernel 3.10.0-327.el7.x86_64, CPU 2 x Intel® Xeon® E5-2699 v4 @ 2.20GHz (22 cores), RAM 396GB DDR @ 2133MHz.

Optane Configuration – Intel® Optane™ SSD DC P4800X 375GB.

NAND Configuration – Intel® SSD DC P3700 1600GB.

QoS – measures 99% QoS under 4K 70-30 workload at QD1 using fio-2.15.
Intel® Optane™ SSD Use Cases

**Fast Storage and Cache**
- Intel® Xeon®
- DDR
- PCIe*
- Intel® Optane™ SSD
- Intel® 3D NAND SSDs

**Extend Memory**
- Intel® Xeon®
- DDR
- PCIe
- Intel® Optane™ SSD
- 'memory pool'
- Intel® 3D NAND SSDs

*Other names and brands names may be claimed as the property of others
INTRODUCING INTEL® MEMORY DRIVE TECHNOLOGY

• Use Intel® Optane™ SSD DC P4800X transparently as memory

• Grow beyond system DRAM capacity, or replace high-capacity DIMMs for lower-cost alternative, with similar performance

• Leverage storage-class memory today!
  • No change to software stack: unmodified Linux* OS, applications, and programming
  • No change to hardware: runs bare-metal, loaded before OS from BIOS or UEFI

• Aggregated single volatile memory pool

*Other names and brands may be claimed as the property of others
Paging in OS

Application

CPU

MMU

Phys. addr.

PN offset

TLB

Operating system

Page directory

Phys. addr.

Page

Disk

RAM

Disk

On-demand (Separately Managed)
How Intel® Memory Drive Technology works

Application

CPU

MMU

Phys. addr.

PN offset

TLB

Phys. addr.

Operating system (OS)

Phys. addr.

IMDT

RAM

Cache

Disk

Disk

backstore

Predict/Prefetch

Intel HPC DevCon 2017
When to use and when not to use Intel® Memory Drive Technology

✔ Your application is designed to use very large amount of memory
  • Benefits from the large memory pool
  • Virtually no performance decrease on benchmarks with high arithmetic intensity

✔ Your application does not handle memory-locality/NUMA well
  • Benefits from the intelligent control of NUMA memory access

✘ Your application is bound by the memory bandwidth
  • The memory-bandwidth of Xeon is >50GB/s; Optane is 2GB/s per SSD
  • Up to ~50% efficiency is expected, not more
What is important for Intel® Memory Drive Technology?

- Predictable accesses
  - If there is a pattern to the memory access, be it simple such as “sequential”, mid-complex like “fetch 1K every 72K”, or entirely complex like “if going to an ID field in a record in a table, fetch the whole record”

- High arithmetic intensity (FLOPs/byte ratio)
  - For every fetch from memory (in average) many compute cycles done

- High concurrency
  - Using at least 50% of the cores in a server platform concurrently, preferably more and even over-subscribed
Features of an ideal benchmark:
- Recognizable memory access patterns
- Ability to vary FLOPs/byte ratio
- Ability to vary benchmark size in memory

Used benchmarks:
- Level 3 BLAS functions liked GEMM ($O(N^3)$ compute and $O(N^2)$ data)
- Modified STREAM with delays

Used hardware:
- Dual-socket Intel® Xeon® E5-2699 v4 (2x22 cores, 2.2 GHz)
- First configuration (MDT):
  - 256 GB ECC DDR4
  - 4x320 GB Intel® Optane™ SSD (~10 GB/s aggregated bandwidth)
- Second configuration (lot of DRAM):
  - 1536 GB ECC DDR4
GEMM benchmark

All DRAM

DRAM + Intel® Optane™ SSD + Intel® Memory Drive Technology

Original GEMM application

Optimized application

2,322 GFLOPS

2,786 GFLOPS

2,605 GFLOPS

up to 1.1x faster matrix multiplication with optimized data locality with no changes\(^1\) in the application

up to 0.9x near DRAM performance of the optimized application

---

1. Optane + IMDT configuration – 2 x Intel® Xeon® CPU E5-2699 v4 @ 2.20Ghz, Intel® Server Board S2600WT, 128GB DDR4 + 4* Intel® SSD Optane® (SSDPED1K375GA), CentOS 7.3.1611. All DRAM configuration – 2 x Intel® Xeon® CPU E5-2699 v4 @ 2.20Ghz, Intel® Server Board S2600WT, 768GB DDR4 CentOS 7.3.1611. Test – GEMM(MKL), segment size 18689, factor 22, threads 42, dataset consumed ~650GB.
Polynomial benchmark

- Sequential-memory access benchmark: compute polynomial values over a large array of input data
- Three types of memory access patterns:
  - Read only (RO)
  - Read and write back (RW)
  - Read and write to another array (RW2)
  - No “triads”-like scheme
- Horner scheme
- Adjustable degree of polynomials

\[ N_{FLOP} = (2 \cdot degree) \cdot N_{data} \]

\[ \frac{FLOPs}{byte} = \frac{2 \cdot degree}{sizeof(real_t)} \]
Polynomial benchmark (Read Only)

**EFFICIENCY: INTEL® MEMORY DRIVE TECHNOLOGY VS RAM**

% RAM – workload size, FLOPs/byte – workload complexity, color – efficiency
Polynomial benchmark (Read&Write)

**EFFICIENCY: INTEL® MEMORY DRIVE TECHNOLOGY VS RAM**

| % RAM – workload size, FLOPs/byte – workload complexity, color – efficiency |

44 threads

88 threads
Polynomial benchmark summary

- RW2 is just 2 times x-axis rescaled RW

- If data size is larger than DRAM:
  - Arithmetic intensity (AI) requirements to get efficiency >80%:
    - RO access pattern (1 data stream): ~256 FLOPs/byte
    - RW access pattern (2 data streams): ~512 FLOPs/byte
  - AI should be measured on DRAM-LLC level!

- If data fits in DRAM:
  - No performance degradation
  - MDT can be faster for NUMA non-aware applications
LU decomposition

- Factorization of matrix $A$ into product of lower triangular ($L$) and upper triangular ($U$) matrices
  - A commonly used kernel in many scientific codes:
    - Solving systems of linear equations
    - Matrix inversion
    - Computing determinants
  - A kernel in LINPACK benchmark

\[ A = L \times U \]
Performance results

- DRAM maximum performance: 850 GFLOPs/s
- Intel® Memory Drive Technology max performance: 1,250 GFLOPs/s
- A huge performance degradation beyond 150% RAM utilization

Can we improve these results?
LU decomposition

- Memory access pattern is by column blocks
- Nearby elements are scattered throughout different memory pages
  - 4KB page = 512 double precision numbers
  - A huge data traffic for large matrices ($2 \cdot 10^5$ and above)
- There are tiled LU algorithms (e.g. PLASMA)
LU decomposition

- Memory access pattern is by column blocks
- Nearby elements are scattered throughout different memory pages
  - 4KB page = 512 double precision numbers
  - A huge data traffic for large matrices \((2 \cdot 10^5\) and above)
- There are tiled LU algorithms (e.g. PLASMA)
- We used a simple implementation from \textit{hetero-streams} code base
- Little performance degradation beyond 100% RAM usage

![Performance Diagram](image)

**Performance, GFLOP/s**

- **MKL LU**
- **Tiled LU**

- **Efficiency**
  - 33% → 86%

\[ N = 280000, 230\% \text{ RAM} \]
Fast Fourier transformation

- A common used kernel in physics and material science
- Compute bound, but AI grows very slow with problem size
- $O(N \log N)$ time complexity
- “Butterfly” memory access pattern – complex but predictable
Fast Fourier transformation

- Intel® Math Kernel Library DFT kernel
- 3D FFT benchmark, $N \times N \times N$ grid
- Results:
  - 80-130% of DRAM performance up to 200% of RAM utilization
  - 40% efficiency over 250% DDR utilization
- 3D FFT can be optimized for NUMA and MDT in a similar way to the LU decomposition
  - by dividing the total memory worked on by all threads at a given time
Lessons learned from benchmarks with Intel® Memory Drive Technology

- Data moving between Intel® Optane™ SSDs and RAM is very expensive (10 GB/s max):
  - Reuse data as much as possible
    - Arithmetic intensity on DRAM↔MDT level should be ≥500 FLOPs/byte
  - Redesign data structures in your program for locality
  - Work with large data chunks
    - Think about DRAM as a large L4 cache for MDT

- Same optimization principles as on NUMA architectures

- Data-oriented programming is a must
  - It also favors modern hardware architectures
Scientific applications

Computational chemistry:
- LAMMPS* (molecular dynamics)
- GAMESS (two-electron integral kernel)

Astrophysics:
- AstroPhi* (hyperbolic partial differential equation solver)

Sparse linear algebra problems:
- Intel® Math Kernel Library PARDISO

Quantum computing simulator:
- Intel-QS, formerly known as qHipster
LAMMPS*

Popular molecular dynamics package
- Mostly used in material science
- Force-field based molecular dynamics
  - Partitions the simulation domain (spatial-decomposition) into small 3d sub-domains, each assigned to a CPU
  - Processors communicate and store ghost atom information for atoms that border their subdomain

Scaled Rhodopsin benchmark:
- Hundreds of millions atoms
- Major bottleneck is calculation of electrostatic interaction between atoms
- Reasonable efficiency up to 150% RAM
  - 50% efficiency for high memory consumption
Two-electron integrals:

- An important kernel in quantum chemistry
- Used in many quantum chemistry methods
- Different types of two-electron integrals have different efficiency on MDT
- Benchmark details:
  - Rys quadrature ERI kernel from GAMESS
  - Compute and store ERIs to memory

<table>
<thead>
<tr>
<th>Int. type</th>
<th>27%</th>
<th>82%</th>
<th>109%</th>
<th>164%</th>
<th>273%</th>
</tr>
</thead>
<tbody>
<tr>
<td>(SS</td>
<td>SS)</td>
<td>97%</td>
<td>111%</td>
<td>104%</td>
<td>100%</td>
</tr>
<tr>
<td>(PS</td>
<td>SS)</td>
<td>100%</td>
<td>95%</td>
<td>98%</td>
<td>93%</td>
</tr>
<tr>
<td>(PS</td>
<td>PS)</td>
<td>95%</td>
<td>94%</td>
<td>97%</td>
<td>87%</td>
</tr>
<tr>
<td>(PP</td>
<td>SS)</td>
<td>104%</td>
<td>97%</td>
<td>96%</td>
<td>86%</td>
</tr>
<tr>
<td>(PP</td>
<td>PS)</td>
<td>124%</td>
<td>100%</td>
<td>117%</td>
<td>76%</td>
</tr>
<tr>
<td>(PP</td>
<td>PP)</td>
<td>146%</td>
<td>106%</td>
<td>106%</td>
<td>79%</td>
</tr>
<tr>
<td>(DS</td>
<td>SS)</td>
<td>98%</td>
<td>96%</td>
<td>94%</td>
<td>92%</td>
</tr>
<tr>
<td>(DS</td>
<td>PS)</td>
<td>114%</td>
<td>105%</td>
<td>103%</td>
<td>77%</td>
</tr>
<tr>
<td>(DS</td>
<td>PP)</td>
<td>140%</td>
<td>87%</td>
<td>98%</td>
<td>91%</td>
</tr>
<tr>
<td>(DS</td>
<td>DS)</td>
<td>112%</td>
<td>111%</td>
<td>133%</td>
<td>75%</td>
</tr>
<tr>
<td>(DP</td>
<td>SS)</td>
<td>105%</td>
<td>99%</td>
<td>93%</td>
<td>99%</td>
</tr>
<tr>
<td>(DP</td>
<td>PS)</td>
<td>130%</td>
<td>107%</td>
<td>101%</td>
<td>68%</td>
</tr>
<tr>
<td>(DP</td>
<td>PP)</td>
<td>109%</td>
<td>106%</td>
<td>88%</td>
<td>87%</td>
</tr>
<tr>
<td>(DP</td>
<td>DS)</td>
<td>88%</td>
<td>92%</td>
<td>109%</td>
<td>83%</td>
</tr>
<tr>
<td>(DP</td>
<td>DP)</td>
<td>124%</td>
<td>121%</td>
<td>115%</td>
<td>88%</td>
</tr>
<tr>
<td>(DD</td>
<td>SS)</td>
<td>119%</td>
<td>100%</td>
<td>100%</td>
<td>80%</td>
</tr>
<tr>
<td>(DD</td>
<td>PS)</td>
<td>130%</td>
<td>117%</td>
<td>116%</td>
<td>72%</td>
</tr>
<tr>
<td>(DD</td>
<td>PP)</td>
<td>88%</td>
<td>135%</td>
<td>109%</td>
<td>89%</td>
</tr>
<tr>
<td>(DD</td>
<td>DS)</td>
<td>128%</td>
<td>103%</td>
<td>126%</td>
<td>96%</td>
</tr>
<tr>
<td>(DD</td>
<td>DP)</td>
<td>118%</td>
<td>114%</td>
<td>104%</td>
<td>94%</td>
</tr>
<tr>
<td>(DD</td>
<td>DD)</td>
<td>111%</td>
<td>115%</td>
<td>113%</td>
<td>83%</td>
</tr>
</tbody>
</table>
AstroPhi*

- The hyperbolic PDE engine
- Numerical 3D finite difference kernel
- Code is not optimized for MDT

---

AstroPhi

- The hyperbolic PDE engine
- Numerical 3D finite difference kernel
- Code is not currently optimized, opportunities for MDT optimization have been identified

PARDISO

- Intel® Math Kernel Library PARDISO – Parallel Direct Sparse Solver
- Solves huge linear algebra problems: hundreds of millions variables
- Test cases:
  - Cholesky factorization of square $N \times N$ matrices
  - Matrix dimensions: $N = 10 \cdot 10^7$, $20 \cdot 10^7$, $25 \cdot 10^7$
  - Number of nonzero elements: $O(N)$

![Graph depicting Intel® Memory Drive Technology provides better than DRAM performance](image)
Quantum computing simulation
- Application requires more memory as more qubits are simulated
- Without MDT, scaling beyond a node's capability requires MPI on a cluster

Test cases:
- Quantum Fourier transform
- $N_{qubits} = 30 - 35$

Good performance up to 4×RAM utilization
- 35 qubits would require > 1.5TB
- with MDT a single node can runs 35 qubits – enabling the move to HPC/HTC Cloud, instead of HPC cluster with MPI
Conclusions

- Efficiency of optimized applications is close to 100% with Intel® Memory Drive Technology
- Efficiency of non-optimized applications can vary from 20% to more than 100%. Typical efficiency of bandwidth-bound applications is up to 50%.
- Optimal performance is expected on next generation of Intel® Optane™ SSDs
Acknowledgements

We thank:

- ScaleMP team for technical support
- RSC Group and Siberian Supercomputer Center ICMMG SB RAS for providing access to certain hardware
- Gennady Fedorov (Intel) for help with Intel PARDISO benchmark
- Justin Hogaboam (Intel) for Intel QS code
- all of you for your attention!