KIOXIA

Technical Brief

RAID Offload Technology: A New Paradigm Re-inventing RAID and Erasure Code Data Protection Using NVMe™ SSDs

Introduction

Data redundancy solutions (e.g., RAID[1] or Erasure Code[2]) are by nature compute intensive and consume high DRAM bandwidth in the write operation path. In particular, RAID solutions also contribute to CPU (Central Processing Unit) cache thrashing. With NVMe™ SSDs added to a system, read/write performance doubles with every PCIe® generation. The increase in SSD performance has shifted performance bottlenecks to these data redundancy solutions, both in hardware and software (Figure 1).

To utilize this shift, KIOXIA has developed RAID Offload technology, which offloads RAID compute and DRAM utilization to SSDs. RAID Offload technology is a scale out solution, so as the number of SSDs increase, performance can scale proportionally. It is also extremely flexible to use it with existing hardware and software RAID applications, delivering the following:

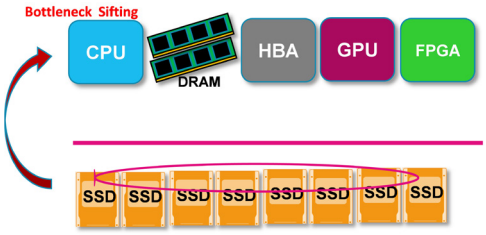

Figure 1 depicts the performance bottleneck shift of write operations to host/RAID applications

- *Enhanced performance*
- *Reduced memory wall issues*
- *Optimized CPU core and DRAM bandwidth usage*
- *Lowered total cost of ownership (TCO)*

All of these benefits are achieved while utilizing the existing, mature RAID stack and user interface.

RAID Issues

The dramatic increase in raw NVMe SSD performance has shifted bottlenecks from the storage to higher in the stack (hardware and software RAID implementations), which do not scale well:

- *Software RAID is memory and compute intensive.*
- *In a RAID 5[3] partial stripe write on one SSD, the data crosses the DRAM interface ten times.*
- *The amount of data that crosses the DRAM interface for one RAID 5 full stripe write = (n-1)*3*segment size.*
  - *Segment size = amount of data stored on one disk in a RAID stripe.*
  - *n = number of SSDs in RAID 5.*
- *RAID 6[4] and erasure coding add weight multiplication cost causing an increase in DRAM bandwidth, compute cycles and CPU caching.*

A number of years back, the industry shifted and offloaded compute and DRAM bandwidth to hardware RAID solutions. This strategy worked well with hard disk drives (HDDs), but with the advent of NVMe SSDs over a PCIe interface, the hardware RAID solutions faced the same compute and DRAM bandwidth issues and became a bottleneck. The graph below (Figure 2) describes the level of DRAM bandwidth that a hardware or software RAID application consumes (implementation specific) in order to deliver RAID write performance at 1 gigabyte[5] per second (GB/s).

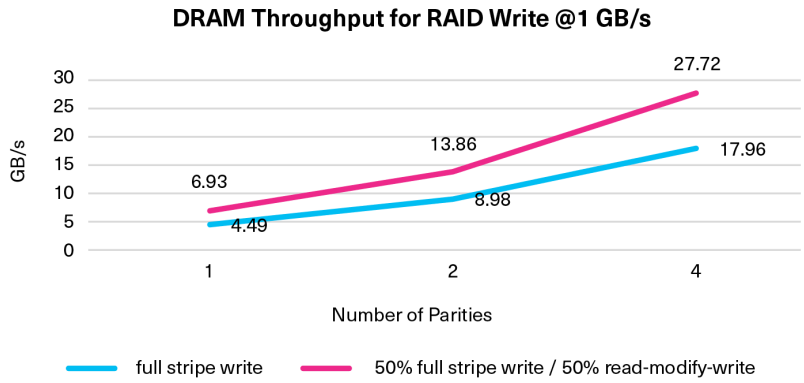

Figure 2 shows DRAM throughput for RAID write at 1 GB/s with parities

RAID Offload Construct

Using PCIe® interface features, NVMe™ SSDs can enable compute memory offload capabilities. KIOXIA may add the following offload capabilities for its NVMe SSD product lines to enable RAID Offload:

- **Parallel parity compute engine for Galois field computations:**
  (e.g., RAID 6 PQ (https://www.kernel.org/pub/linux/kernel/people/hpa/raid6.pdf), Erasure Code)
  The CPU normally computes parity computation of multiple buffers while the parity compute engine computes parity more efficiently, relieving significant CPU load.
- **Direct memory access (DMA) engine for buffer to buffer data copy:**
  The DMA engine facilitates bulk data movement between the buffers. An application is free to leverage external DMA engines from the CPU, data processing unit, smart network interface card (i.e., SmartNIC), RAID controller, etc. For RAID applications, use of the RAID Offload DMA engine is optional to reduce the host system's DMA engine usage.
- **Controller memory buffer (CMB):**
  The NVMe controller memory buffer functionality (version 1.2 upwards) exposes some of the NVMe subsystem's controller memory for host application use. KIOXIA enterprise SSDs may support 256 megabytes[5] (MB) for the CMB.
- **Proposal for the addition of two new NVMe commands for standardization:**
  Call to Action: TPAR has been introduced to RAID Offload technology and it is actively being discussed in the technical working group of NVM Express™. Contributing members may review and comment on this proposal.

To describe how the SSD RAID Offload is constructed, there are a few key points. For example, a DMA engine can access an entire host address space including the bar-mapped peer CMB. The host manages the CMB buffer, any virtual addressing, Galois generator equations and error handling (e.g., write holes). The parity compute offload does not affect the performance of conventional read/write operations. Shown below is the SSD RAID Offload construct (Figure 3):

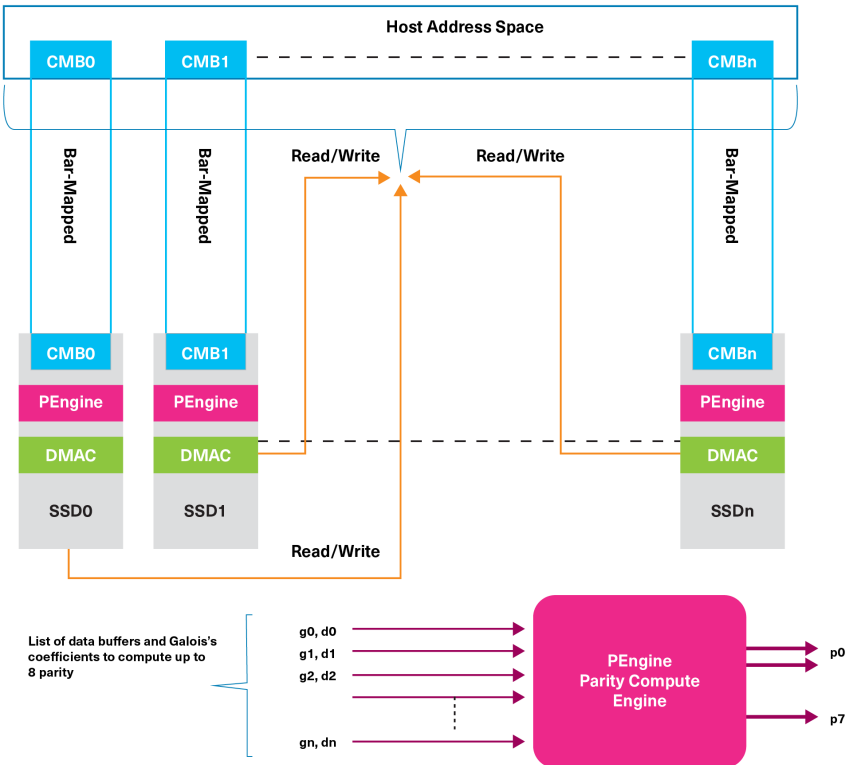

Figure 3 depicts the SSD RAID Offload construct

RAID Offload Benefits

The graph below (Figure 4) compares the data flow for a 4 KiB[6] write operation of conventional RAID 5 solutions with a RAID Offload solution. Note that the host has options on how to best utilize the peer-to-peer CMB data movement.

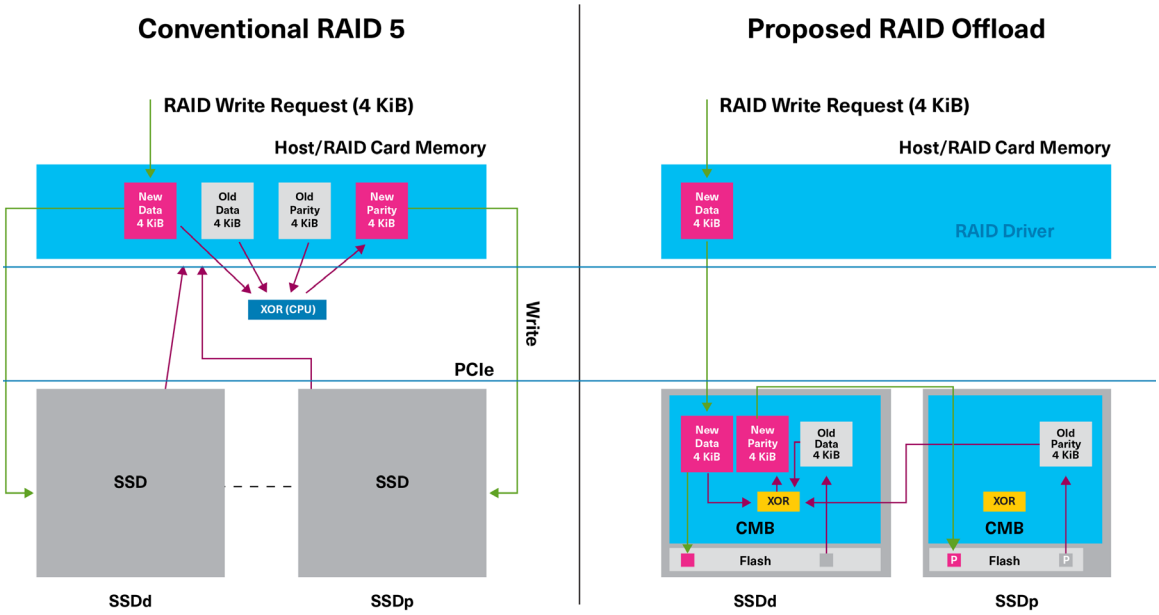

Figure 4 compares the data flow of 4 KiB write operations – traditional RAID 5 versus RAID Offload

A simple 4 KiB RAID 5 write operation requires two reads from SSDs and the generation of new parity followed by two writes to SSDs. The table below summarizes resource usages for 4 KiB RAID 5 write operations.

Steps	CPU Usage	Memory Usage
Read Old Data / Old Parity	Command handling	8 KiB
Calculate New Parity	Data processing	24 KiB
Write New Data and New Parity	Command handling	8 KiB

For each 4 KiB written in a partial stripe write, the DRAM usage is 40 KiB and 24 KiB for the CPU processes. The 24 KiB data processed by the CPU will be cached leaving less cache for other applications.

The table below (Figure 5) shows limited proof of concept (PoC) results with mdRAID 5[7]. In this example, write bandwidth was limited to 950 MiB[6]/s due to PoC platform limitation. For a given performance, mdRAID 5 software with the RAID Offload solution reduces valuable DRAM use by 91% and CPU use by 12%.

System	KIOXIA CM7 Gen4 x4 – mdRAID 5#	RAID Offload	% Benefit
Number of SSDs	5	5	
Full Stripe Write (512 KiB)			
Performance (in MiB/s)	950	950	N/A
CPU Utilization	42	37	12% reduction
DRAM Bandwidth (in MiB/s)	3,450	340	91% reduction

Figure 5 is a comparison of the RAID Offload technology versus traditional software RAID solutions

System set-up: *Dell® PowerEdge™ R650xs with Intel® Xeon® Gold 6338N 2.2 GHz (2 socket, 32 cores) PCIe® Gen4, SSDs: 5x KIOXIA CM7 Series SSD (1.92 terabytes[5]).*

Conclusion

KIOXIA NVMe™ SSDs offer a standards-based, host-orchestrated scale out and sustainable solution to offload RAID parity compute. This technology frees up valuable host CPU, memory and cache resources that can now be used to accelerate primary applications. RAID Offload technology can improve performance and increase power efficiency in server and storage systems. As the number of SSDs in a volume increases, RAID Offload technology scales with the number of SSDs (scale out). Existing RAID and erasure coding hardware and software solutions can leverage RAID Offload technology via two new proposed NVMe commands.

Beyond the offload tasks, existing RAID and erasure coding applications can continue to leverage the RAID algorithms, device management and error handling developed over the years. These RAID applications can use RAID Offload technology for optimizing throughput and latency, customizing data flows, and for erasure code algorithms. RAID Offload technology is a highly versatile solution and feasible for application adoption as described below (Figure 6).

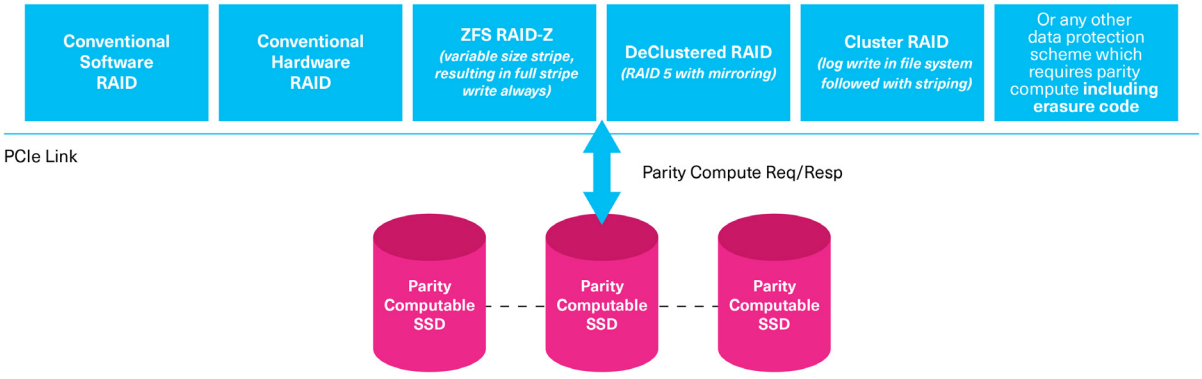

Figure 6 shows potential use cases for RAID Offload technology

RAID Offload technology developed by KIOXIA provides host flexibility when building a data redundancy solution optimized for latency and throughput. It can also initialize or rebuild a RAID volume at the maximum sequential write performance of an SSD. RAID Offload technology re-architects proven data protection technologies, enabling NVMe™ SSD storage to eliminate legacy performance bottlenecks.

FOOTNOTES:

[1] The term "RAID" was invented by David Patterson, Garth A. Gibson, and Randy Katz at the University of California, Berkeley in 1987. Reference : https://www.usenix.org/system/files/conference/atc12/atc12-final181_0.pdf
RAID (redundant array of independent disks) is a data storage technology that stores identical data in different locations on SSDs to protect the data in the case of a drive failure. It combines multiple SSD components into one or more logical units for data redundancy and/or performance improvement.

[2] Erasure code is a forward error correction code and a method of data protection that breaks data into sectors, expands and encodes the sectors with redundant data pieces, and stores the redundant pieces across different storage media. It improves data availability and durability, and when compared to RAID, can tolerate more SSD or server node failures, recover data faster, handle larger and more diverse data sets, and support different and flexible configurations.

[3] RAID 5 is currently one of the most commonly used RAID methods and uses disk striping with parity. Data and parity is striped across all SSDs so that no single SSD becomes a bottleneck. Striping also enables users to reconstruct data in case of a SSD failure.

[4] RAID 6, also known as double-parity RAID, places data on multiple SSDs allowing input/output (I/O) operations to overlap in a balanced way, improving performance.

[5] Definition of capacity - KIOXIA Corporation defines a megabyte (MB) as 1,000,000 bytes, a gigabyte (GB) as 1,000,000,000 bytes and a terabyte (TB) as 1,000,000,000,000 bytes. A computer operating system, however, reports storage capacity using powers of 2 for the definition of 1Gbit = 2^{30} bits = 1,073,741,824 bits, 1GB = 2^{30} bytes = 1,073,741,824 bytes and 1TB = 2^{40} bytes = 1,099,511,627,776 bytes and therefore shows less storage capacity. Available storage capacity (including examples of various media files) will vary based on file size, formatting, settings, software and operating system, and/or pre-installed software applications, or media content. Actual formatted capacity may vary.

[6] KiB: a kibibyte (KiB) means 2^{10}, or 1,024 bytes. MiB: a mebibyte (MiB) means 2^{20}, or 1,048,576 bytes.

[7] Mdraid (or MD/RAID) is Linux® software RAID that makes the use of RAID possible without a hardware RAID controller.

TRADEMARKS:

DISCLAIMERS: