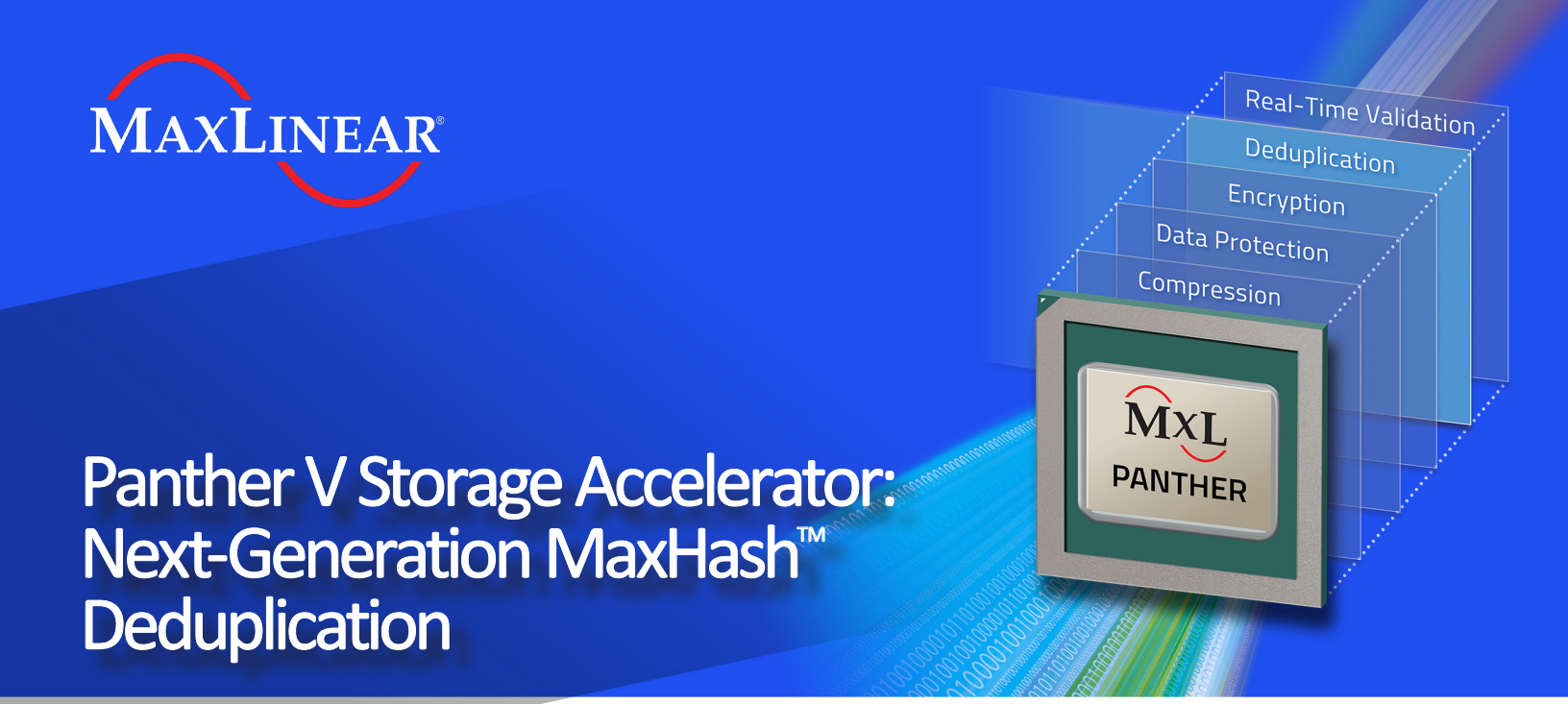


Panther V Storage Accelerator: Next-Generation MaxHash™ Deduplication

WHITE PAPER

Author:

Pinaki Chanda
Director of Software Engineering

KEY BENEFITS

- MaxHash enables you to achieve a data reduction ratio (DRR) of 12:1 for unstructured data and 15:1 for structured data when used with advanced data compression algorithms supported in Panther.
- MaxHash can be combined with compression, encryption, NVMe Protection Information (PI) processing, and other operations in a single command, making it a powerful primitive to use in a storage appliance.
- MaxHash can compute more than one set of slice hashes on data with multiple (up to four) hash engines. Each hash engine generates one set of hash results. Each hash engine can be independently configured with a unique slice offset.
- Each hash engine can skip non-data fields in the data so only data is hashed to improve deduplication hit rates.

Introduction

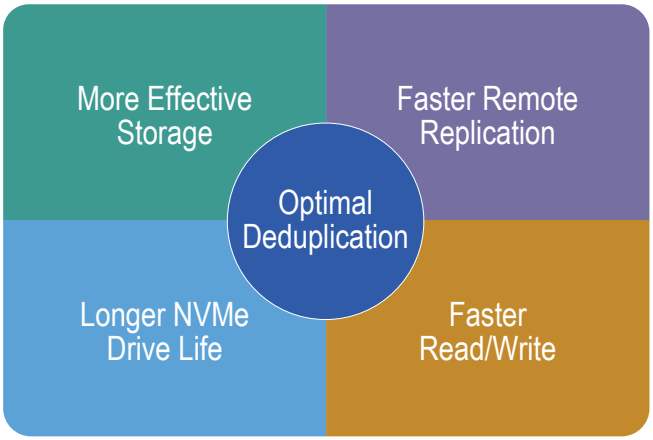
Data reduction methods in storage appliances combine compression and deduplication.

To improve the deduplication rate, it is necessary to decouple the input data block size for the compression operation from the block size for deduplication hash computation (slice size). It may also be important to apply user-defined offsets on the data and selectively skip fields while computing the hashes.

MaxLinear's patented MaxHash™ technology in the Panther V storage accelerator enables you to:

- Compute hashes on many data blocks or slices by decoupling the hash block size from data block size for compression.
- Compute hashes from unique programmable offsets in the data block using up to four parallel hash engines.
- Skip non-data fields in the data before hashing.

MaxHash, when combined with advanced single-pass compression algorithms in Panther V, delivers 12-15 times more effective storage, 12-15 times longer NVMe drive life, 12-15 times faster read/write speed, and 12-15 times faster remote replication.



What is Deduplication?

Deduplication is a method to identify and eliminate duplicate data based on predefined block sizes for data reduction in storage systems. Data reduction methods in storage appliances combine compression and deduplication. If data reduction using compression achieves a DRR of X:1 and data reduction using deduplication achieves a DRR of Y:1, the combination of data compression and deduplication achieves a DRR of XY:1. Efficient deduplication achieves the following results:

- **Increased effective storage**—With the same number of storage drives, a larger effective storage is achieved.
- **Faster read and write speed**—As less data is read or written, data reduction using deduplication enables faster drive processing.
- **Longer life of NVMe drives**—As less data is written, this increases the Drive Writes Per Day (DWPD) rating.
- **Faster remote replication**—Data reduction speeds up network transfers and reduces the cost of replicated storage.

Deduplication is commonly used in the following use cases:

- **Backup systems**—Deduplication in backup systems avoids storing multiple copies of the same file, object, or data block.
- **Database and data warehouse**—In databases or data warehouses, deduplication is used to ensure data uniqueness and reduce redundancy.
- **Networking**—Deduplication eliminates redundant data transfer in networking protocols.

For each data block, the deduplication system computes the hash value of the data block using a hash function. Common hash functions are SHA-160, SHA-256, SHA-384, and SHA-512. These hash functions work properly for populationg a deduplication table because they have low probability to produce same hash value across different data.
A distinct hash value is created for each block of data, file or object. When two pieces produce the same hash value, they are recognized as duplicates (deduplication hit). Instead of storing multiple copies of the same data, the system keeps only one copy, using pointers, or references to this stored version for the duplicates, which enhances storage efficiency. An efficient deduplication algorithm strives to increase the deduplication hit rate.

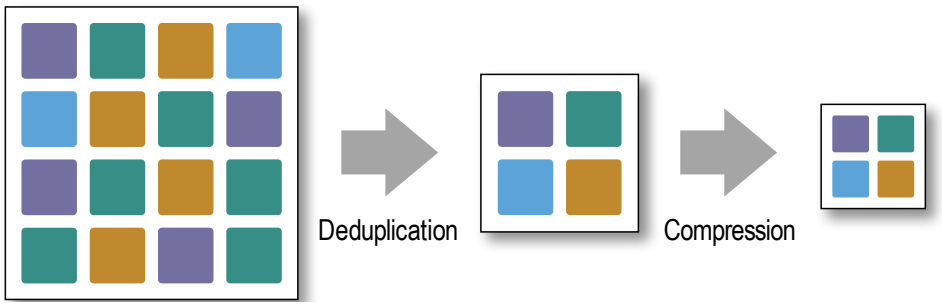


Figure 1: Data Deduplication and Compression Enabling Data Reduction

The MaxLinear Panther V storage accelerator delivers breakthrough data reduction and enhanced deduplication enablement using MaxHash technology.

With Panther, storage solutions deliver a 12:1 DRR using MaxHash, along with data compression and a comprehensive security suite, reduce capital storage expenses (CAPEX) while providing ultra-reliable data protection. Panther's powerful data reduction technology intelligently offloads the CPU by providing multiple independent parallel data processing unit transform engines.

Challenges in Deduplication

- **Decoupling data compression and deduplication block size**

Two of the general data reduction methods, compression and deduplication, have conflicting block size requirements. A large data block size is generally more suitable for data compression.
Dictionary-based compression algorithms (such as deflate, ZLIB, XP10) depend on identifying repeated patterns and replacing the patterns with more compact representations before applying lossless coding such as Huffman coding, arithmetic coding, or asymmetric numeral systems. Larger blocks, when input into these algorithms, help exploit redundancies in the data block and generate a higher compression ratio.

Small block sizes enhance deduplication hit rates primarily by enabling more granular comparisons of data. Smaller blocks are more likely to contain similar or identical segments across different files, increasing the chances of a match and leading to a higher deduplication hit rate. Many files exhibit similar structures or repeating patterns, and smaller blocks can capture these nuances more effectively than larger ones, which may overlook smaller, repeated segments. Additionally, using small blocks allows deduplication systems to identify duplicates at a finer level; even if two files differ slightly, small blocks can still match identical parts, thereby improving overall deduplication efficiency. However, it can lead to more metadata overhead and a larger deduplication table size.

For this reason, a large data block size is chosen in a storage appliance, but a smaller block size is used for the deduplication function. Traditionally, it requires that the compression operation is processed separately from the hash operations which needs more CPU memory bandwidth, more command processing, and more CPU or accelerator operations. Panther solves this problem by introducing the MaxHash feature.

- **Presence of metadata or unique key in the data degrades deduplication hit rate**

In a storage appliance, the input source data may contain metadata embedded in predefined locations in the source data —for example, but not limited to, T10-DIF NVMe PI at the beginning or end of each sector for data integrity in storage systems. Incorporating T10-DIF fields into deduplication hashing can lead to a zero deduplication hit, even when the data across sectors are identical. This occurs because the REF field, which contains a unique logical block address, differs between the sectors. If you are using a relational database, you can build queries that only consider specific columns for deduplication, excluding the primary keys.

Skipping fields during deduplication hashing enables you to focus on specific aspects of the data that are relevant for identifying duplicates. By carefully selecting which fields to include in the deduplication process, you can improve the accuracy and efficiency of data deduplication.

- **Deduplicating data that are shifted version of each other requires preprocessing before deduplication hashing**

There are cases where the data is shifted related to each other. For example, consider text or data arrays whose elements are shifted but not modified. Deduplicating data that are shifted versions of each other can be particularly challenging, because hashing may need to be done after applying an offset to the beginning of the data block.

Applying a programmable offset to the data block before slicing the data and computing the hash on the slices is also challenging because it can be computationally intensive in a storage appliance system.

MaxHash Technology in the Panther Storage Accelerator

Data to be stored must be compressed and deduplicated. Additionally, if an accelerator cannot compute deduplication hash based on the final data to be written and compressed, it becomes a multi-step process consuming multiple command cycles, sequential transactions, and overhead in latency. By supporting Max Hash, Panther allows combining deduplication hash generation with compression while adding intelligent operations such that only data is hashed in the desired block size being used by the deduplication function. This includes the decoupling of compression block size size versus deduplication block size, recognizing and skipping data protection fields and metadata. Also, addition of three additional engines allows for setting offsets for additional hashes to improve deduplication hits (matches).

To enable deduplication hash with a high hit rate, MaxHash consists of the following tools which can be exercised independently:

- Hash on input data block size.
- Alternatively, chunk the input data block into multiple slices and compute hash on the slices. This enables decoupling data block size used for compression from the hash slice size.
- Parallel execution of up to four hash engines with unique offsets to compute hash on the slices at different points in the data transform pipeline.
- Skip NVME Protection Information such as T10-DIF from the data before computing the hash on the slices.
- Skip start and end region in the data and also skip user defined fields (for example: non-data) before computing hash on the slices.

Note that a skip before the hash operation is beneficial before the data is transformed by a stage in the data transformation pipeline. Specifically, this skip should occur prior to the compression operation.

MaxHash can use one of the following hash algorithms.

- SHA1
- SHA224
- SHA256
- SHA384
- SHA512

Figure 2 shows four hash engines operating in parallel on input data before compression in encode pipeline. Figure 3 shows slicing and hash computation by one hash engine that applies an offset before the data is sliced for hash computation.

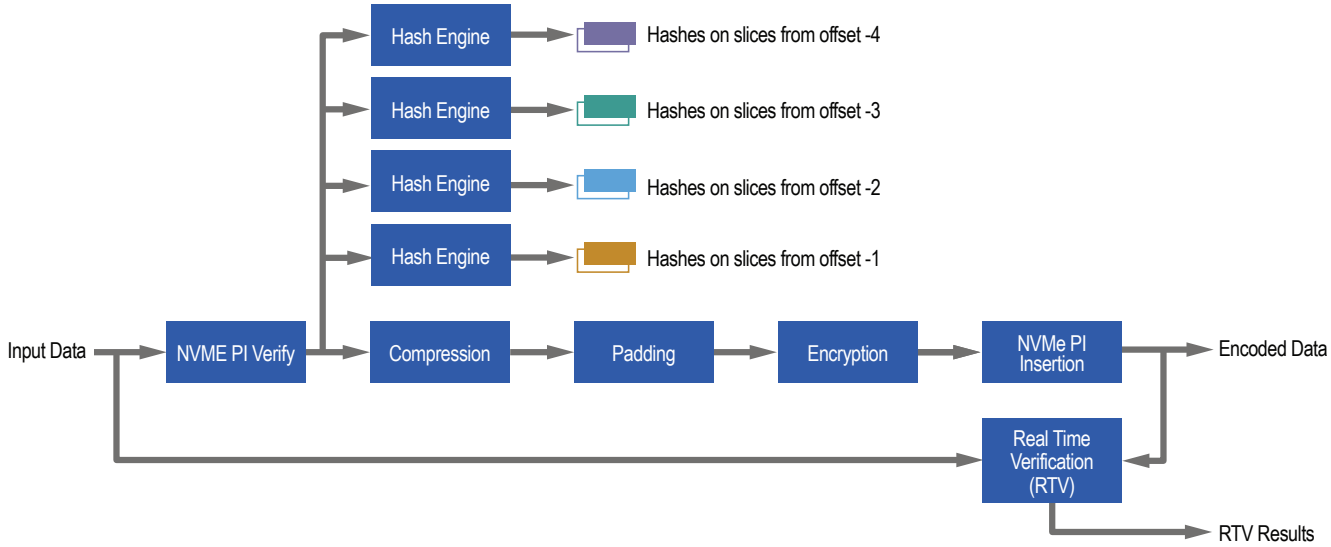


Figure 2: MaxHash Using four Engines before Compression Operation in Panther Encode Pipeline

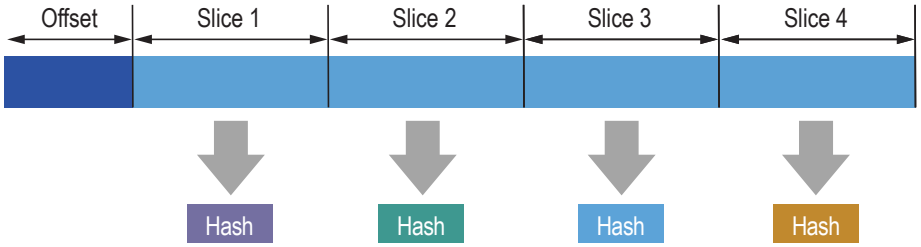


Figure 3: Applying an Offset before Computing Hash on Data Slices using One Hash Engine

Figure 4 shows four hash engines that apply a unique offset by each hash engine before slicing the data for hash computation

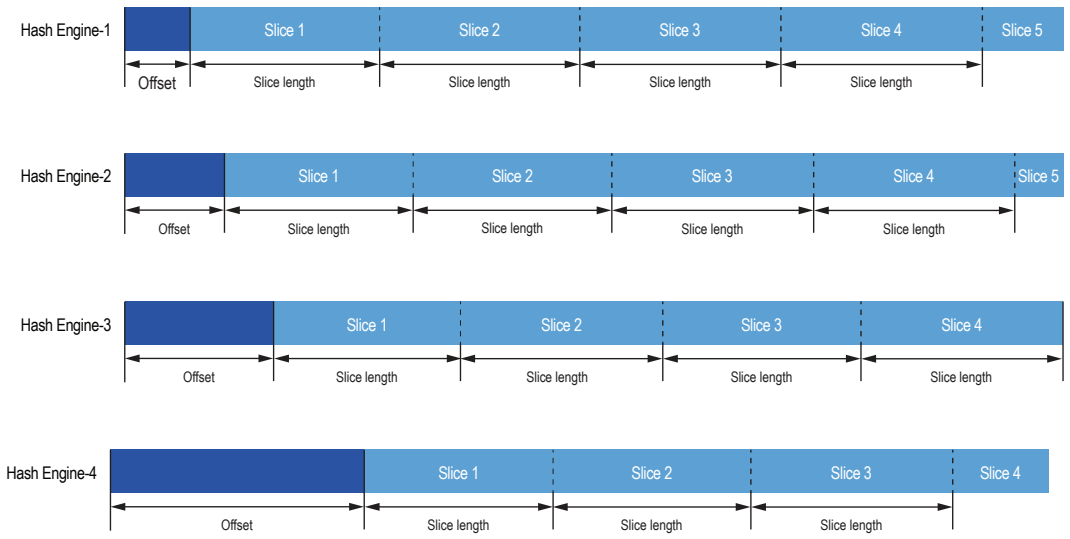


Figure 4: MaxHash Operation with Independent Offsets on Four Hash Engines

MaxHash operates in the following way as shown in Figure 5:

a. If the input data contains embedded T10-DIF (NVMe Protection Information), Panther encode pipeline optionally skips these T10-DIF fields as shown in Figure 5 (a).

b. When enabled, skip a region at the beginning and/or end of the data block by all hash engines on a per-command basis. This is shown in Figure 5 (b).

c. The hash engines optionally check if there are other user defined metadata. These intermediate user defined metadata regions are of fixed size and may be embedded in the data at the fixed interval. These parameters are programmed globally across all hash engines to skip. This is shown in Figure 5 (c).

d. The input data is chunked into several slices. The slicing can start from an offset defined uniquely for each hash engine. The slice size is configured on a per-command basis. After slicing, the hash values are computed with up to four hash engines. The result of the hash operation is written into the hash buffer passed as an argument of the command submitted to Panther. For each hash engine, one record is created in the hash buffer. This is shown in Figure 5 (d).

Note that the option to enable the skip feature and related parameters is only applicable when the hash values are computed on the data before transformation using a compression operation.

A skip before the slice hash operation is not useful if the data is already transformed by a stage in the data transform pipeline and the slice hash is computed after the transform operation.

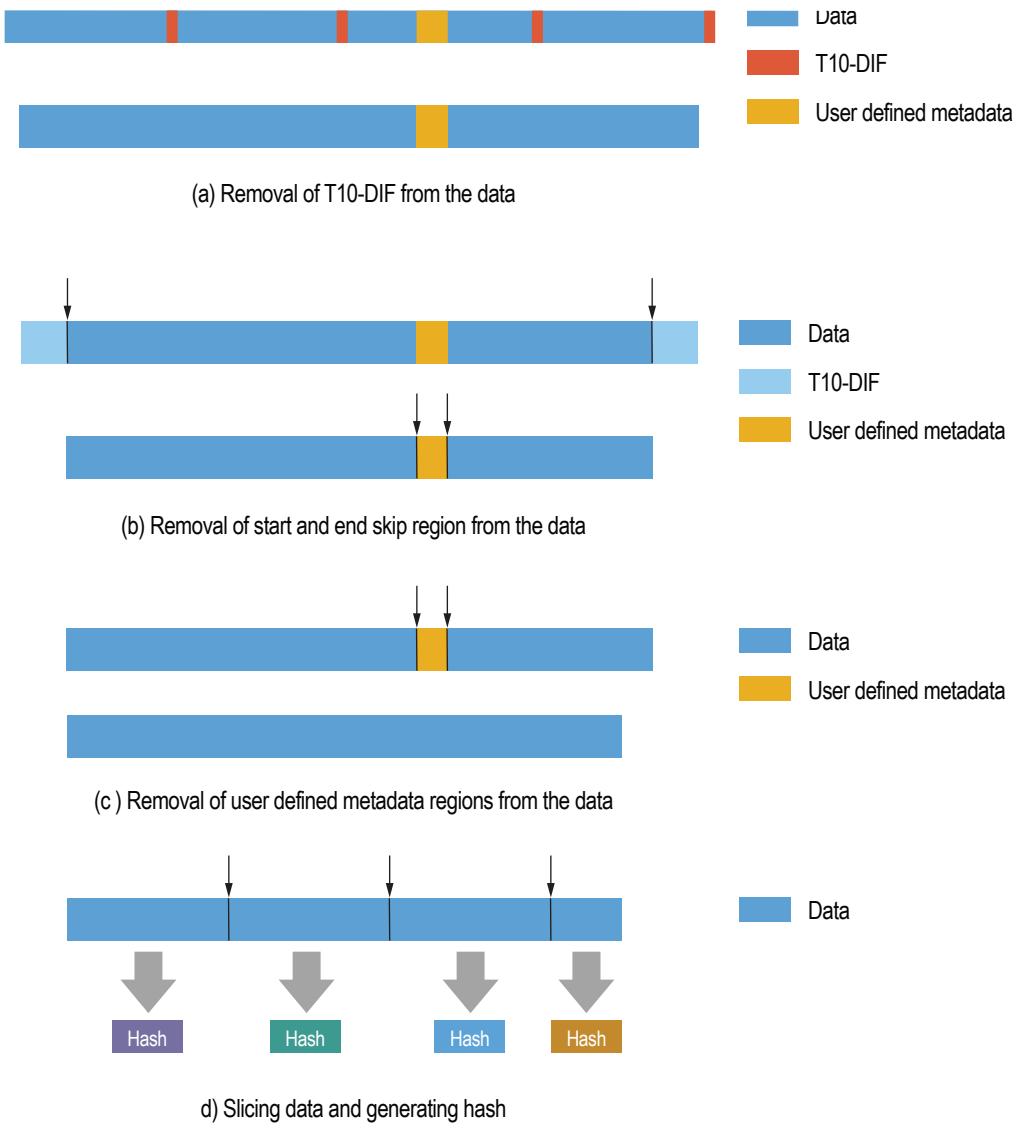


Figure 5: MaxHash Operation on Data

If MaxHash is enabled at any point in the data path, you can only use AES-GCM for authenticated encryption in conjunction with the MaxHash operation. For authentication purposes alone, you can use GMAC.

What does MaxHash do for you?

Standard hash produces an average DRR of 2:1. MaxHash increases this an average DRR of 3:1.

Considering the DRR of compression algorithms for unstructured data as 4:1, a storage appliance can achieve a DRR of 12:1. For structured data using the XP10 data compression algorithm supported in Panther, you can achieve a compression ratio of 5:1. Combining this with MaxHash gives you a DRR of 15:1.

This means:

- 12x-15x increase in effective storage
- 12x-15x longer NVMe drive life
- 12x-15x faster read/write speed
- 12x-15x faster remote replication

Conclusion

Deduplication using MaxHash is a powerful method to efficiently identify and eliminate duplicate data. Adding MaxHash to your storage appliance delivers 12-15 times more effective storage, 12-15 times longer NVMe drive life, 12-15 times faster read/write speed, and 12-15 times faster remote replication.

You can combine this technology with data compression, data encryption, NVMe PI processing, and other transform operation in a single command executed by the Panther storage accelerator to process data at a blazingly fast speed, ultra-low latency, and with minimal CPU core consumption.

References

- *MxL890x Raw Acceleration Application Program Interface (API) Reference Guide* (200AG).
- *MxL890x Storage Accelerator User Guide* (204-UG).

MaxLinear, Inc.
5966 La Place Court, Suite 100
Carlsbad, CA 92008
Tel.:+1 (760) 692-0711
Fax: +1 (760) 444-8598
www.maxlinear.com

The content of this document is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment, representation, or warranty by MaxLinear, Inc. or any of its affiliates (collectively, "MaxLinear"). MaxLinear assumes no responsibility or liability for any errors, omissions, or inaccuracies that may appear in the informational content contained in this document.

Reproduction, distribution, modification, or creation of derivative works, in part or in whole, without the express prior written consent of MaxLinear is prohibited. MaxLinear, the MaxLinear logo, and any other MaxLinear trademarks (including but not limited to MxL, Full-Spectrum Capture, FSC, AirPHY, Puma, AnyWAN, VectorBoost, MXL WARE, and Panther) are all property of MaxLinear and/or its subsidiaries in the U.S.A. and other countries. All rights reserved. All third-party marks and logos are trademarks™ or registered® trademarks of their respective holders/owners.

© 2026 MaxLinear, Inc. All rights reserved.