Mango StorageBoost™ NVMe over TCP Initiator

Disaggregated Storage
with Near-local SSD Performance

OVERVIEW

Mango StorageBoost™- NVMe/TCP Initiator (NTI) is a high-performance NVMe/TCP initiator solution that unlocks the full potential of storage disaggregation. NTI delivers unmatched full-duplex line-rate performance with zero CPU consumption without requiring any modification on the existing systems, resulting in lower total cost of ownership (TCO).

KEY HIGHLIGHTS

UNPARALLELED PERFORMANCE NVME-OF INITIATOR SOLUTION

By offloading the entire NVMe/TCP layers to optimized hardware instead of relying on general purpose cores, NTI significantly accelerates the NVMe/TCP protocol and achieves full-duplex line-rate performance with zero CPU consumption.

EASY TO USE SOLUTION: NO MODIFICATION ON INFRASTRUCTURE AND APPLICATION

NTI is seamlessly integrated into NVMe/TCP initiator servers by presenting the DPU as a standard NVMe PCIe device. This enables existing storage systems to utilize NTI using the standard NVMe driver without any software modification. On the NVMe/TCP target side, NTI is compatible with any NVMe/TCP target server, thanks to its full compatibility with standard TCP/IP network and Ethernet switches.

CONFIGURABLE AND ADAPTABLE STORAGE SYSTEM

Powered by FPGA technology, NTI can be tailored to meet a wide range of customer requirements (e.g., # of queues, queue depth, etc.).

SPECIFICATIONS

HARDWARE INTERFACES

- PCIe Gen3 x16 or 2x Gen4 x8
- Dual Ethernet ports of 100GbE
  - 2x 100GbE QSFP28
  - Direct-attach copper or optical transceiver
- Full-height, half-length (FHHL)

CAPABILITIES

- Up to 2 NVMe PFs
- Up to 31 NVMe IOQs per PF
- Up to 256 NVMe namespace per PF
- Up to 2 NVMe/TCP target subsystems
- Maximum Data Transfer Size: 128KB
- Active-active multipath support
  - Asynchronous Namespace Access
- PFC and ECN support
- Jumbo Frame support (9000B)
- PCIe in-band management channel

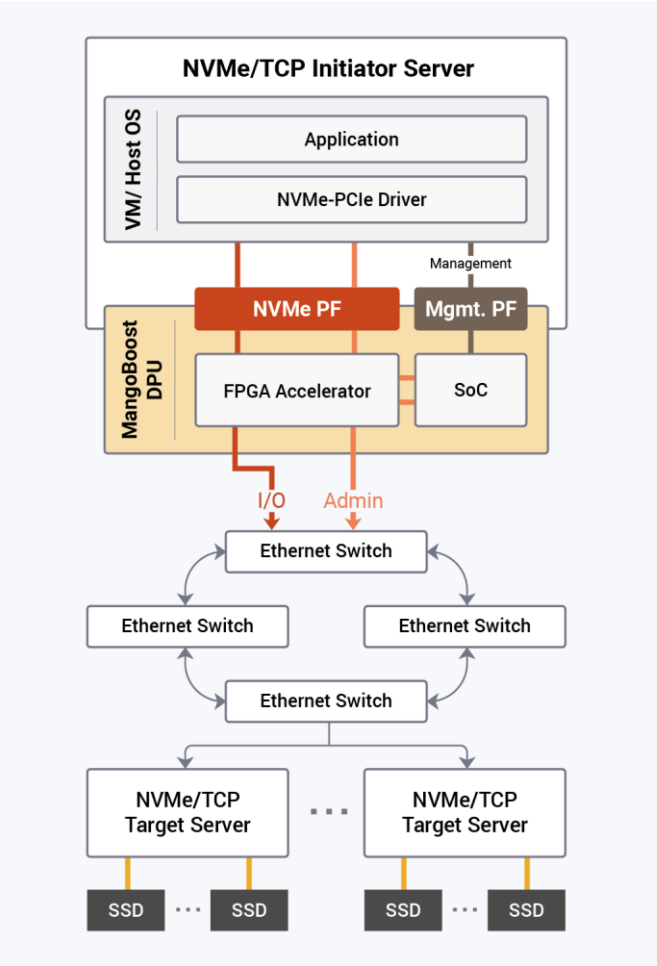
NVME COMPATIBILITY

- NVMe 1.3/2.0 compatible
- NVMe-oF 1.0c compatible

PERFORMANCE SUMMARY

4KB (IOPS)	Random Read	5.7M
	Random Write	5.7M
	Random RW Mix	10.3M
128KB (GB/s)	Sequential Read	23.6
	Sequential Write	23.5
	Sequential RW Mix	41.6

DESIGN OVERVIEW



NVMe-PCIe Virtualization

The MangoBoost DPU (Data Processing Unit) is seamlessly integrated into NVMe/TCP Initiator servers, presenting itself as a standard NVMe-PCIe device on the initiator side. This design allows users and data center managers to easily adopt NTI without requiring any software modifications. On the target side, the DPU connects to target servers via standard TCP networks. Its compatibility with TCP enables seamless integration into existing storage networks, allowing for the efficient utilization of storage disaggregation benefits.

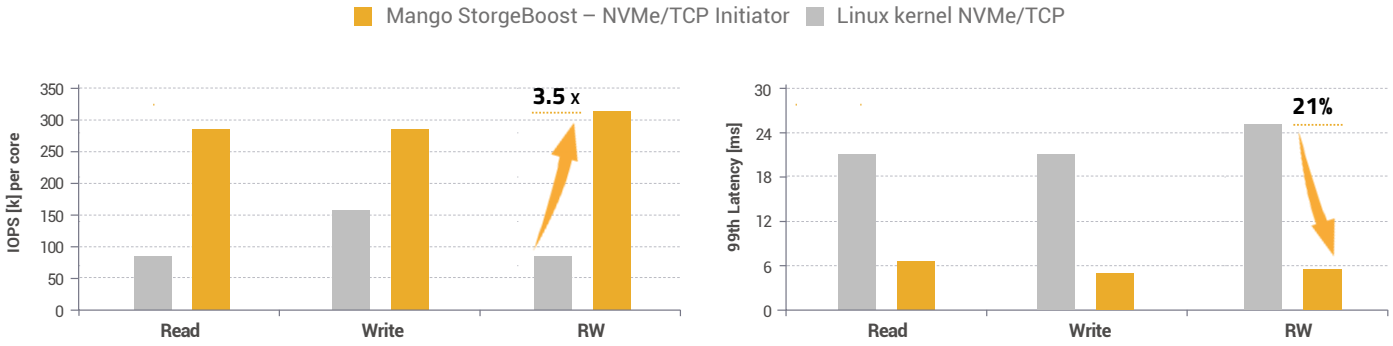
NVMe-TCP Acceleration

The MangoBoost DPU enhances NVMe/TCP performance by offloading the entire NVMe/TCP protocol stack to specialized hardware components. Within the DPU, the System on Chip (SoC) manages NVMe/TCP initialization and administrative command processing, handling tasks that are less computationally intensive but require support for a variety of command types. For more compute-demanding operations, such as protocol conversion from NVMe-PCIe to NVMe-TCP and TCP packet processing, the DPU utilizes an FPGA with highly optimized engines, significantly reducing CPU load.

Additionally, the DPU features a PCIe in-band communication channel, enabling the host to manage network configurations and retrieve logs via the SoC. An integrated NVMe/TCP agent on the SoC further streamlines the process by automating NVMe/TCP initialization.

PERFORMANCE COMPARISON OVER SOFTWARE-BASED NVME/TCP

3.5x Higher Bandwidth | 4.7x Faster 99th Tail Latency



* On the initiator side, the standard Linux kernel NVMe/TCP and NVMe/PCIe drivers were used for the software baseline and our solution, respectively. On the target side, we utilized SPDK LTS for both configurations and employed a Null device to measure the ideal performance independent to SSD models. The initiator and the target server are connected via a network switch (1-hop).

** For the latency measurement, we used the following FIO parameters; bs = 4k, iodepth = 128, numjobs = 64.