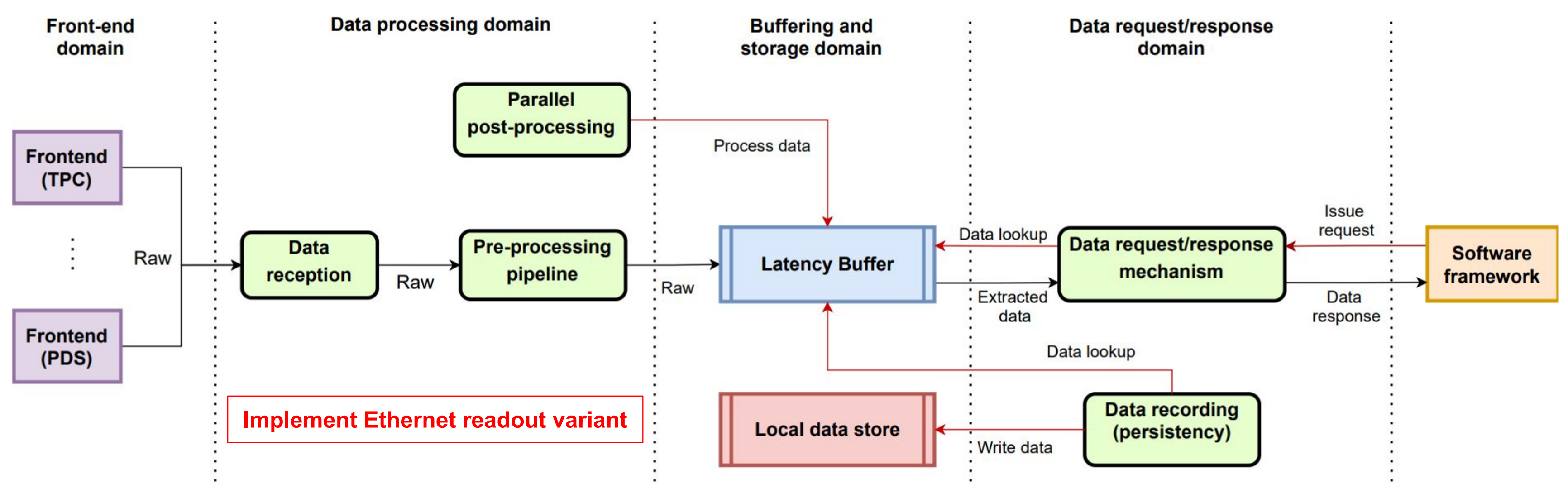
The Ethernet readout of the DUNE DAQ system

Roland Sipos for the DUNE Collaboration

Introduction

In 2023 the Deep Underground Neutrino Experiment (DUNE)[1] Data Acquisition (DAQ) system transitioned to a fully Ethernet based readout. While most of the modular readout system design could be preserved, the adoption of Ethernet required a modified data reception block. This work included a completely new I/O device library implementation, interfacing with the detector electronics via a transmitting firmware module that is provided by the DAQ.



Data flow diagram of the DUNE DAQ generic readout subsystem. Transitioning to Ethernet readout requires the introduction of a new detector data format, and new implementations of the data reception component and processing pipelines and stages.

In addition to data reception the readout is processing all incoming data to carry out hit-finding and generate trigger primitives, is buffering data in DRAM while the trigger takes its decision, and upon command persist up to 100 seconds of all raw data in high-performance NVMe drives.

Front-end domain

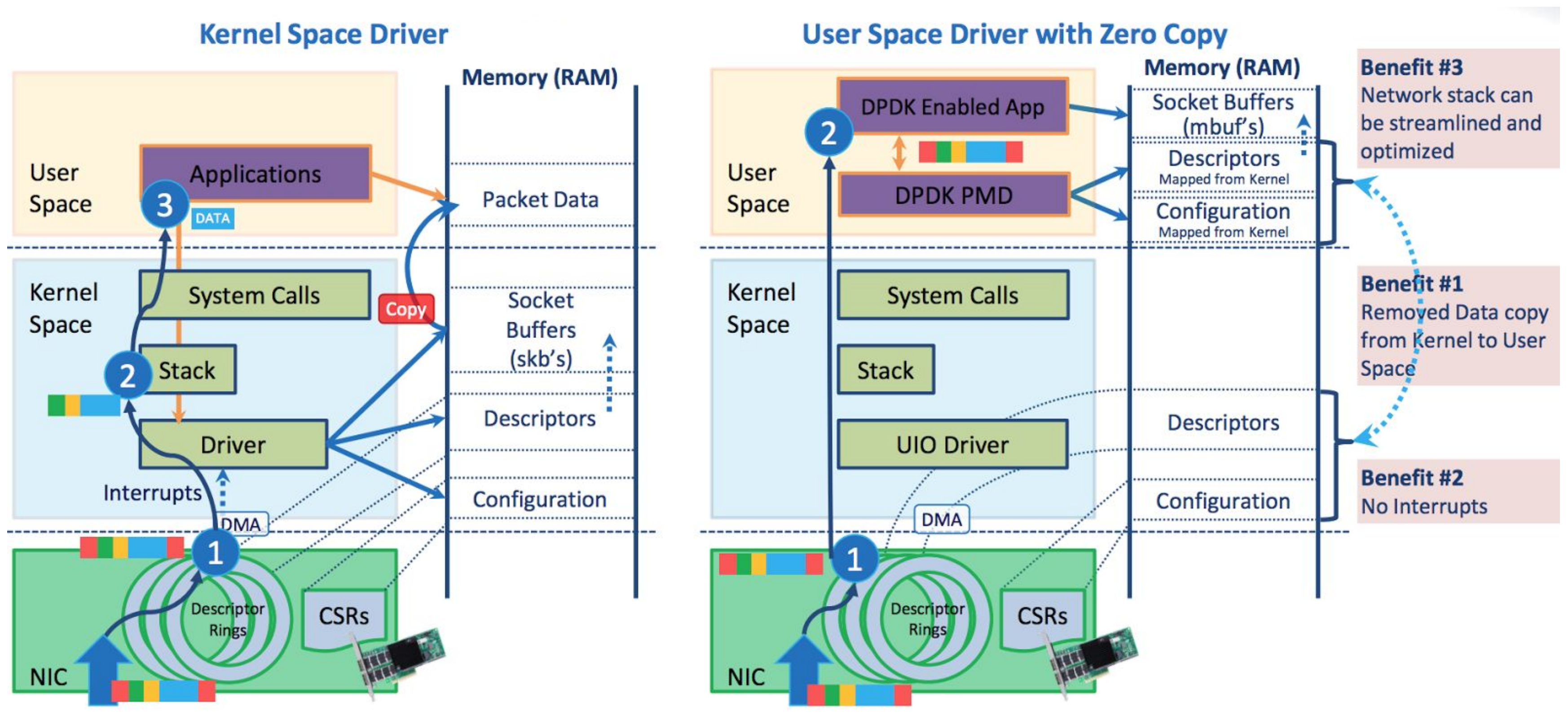
The detector electronics transmits data over 10 Gbps links. Those are aggregated into 100 Gbps links via network switches and are fed to the readout unit servers. The overall aggregated data throughput for each of the 4 DUNE Far Detector modules is ~30 Tbps. In this poster results refer to the TPC electronics of the Horizontal Drift Far Detector module, consisting of 1500 x 10 Gbps links, each streaming data over 4 UDP streams at ~2 Gbps. Each UDP payload is equipped with a uniform DAQ header that carries information about stream and sequence identification.

Single detector components for charge readout	Links and Data Streams	Payload size and arrival rate	Total throughput (incl. IPv4 and UDP headers)
Anode Plane Assembly (APA)	10 links, 4 streams each Total: 40 streams	7200 Bytes @ 30.5 kHz x 40 streams	~82.5 Gbit/s

Numbers of physical links and data streams from the TPC electronics of the Horizontal Drift Far Detector.

Data Plane Development Kit (DPDK)[2]

In order to sustain the multiple 100 Gb/s aggregated input data streams' high-throughput and low latency requirements, the new software stack for the I/O device control, configuration, monitoring and readout of the NICs in the readout units is built upon DPDK.It enables more efficient computing than the standard interrupt processing that is available in the kernel. It uses a PMD (poll-mode driver) that eliminates data copies from kernel to user space.



Comparison of the kernel space network driver and the DPDK provided PMD driver, highlighting the benefits of poll-mode drivers: no interrupts via the OS scheduler, lack of context switching, and eliminated extra memory copies between the kernel and user space. Picture source: DPDK Summit presentation.

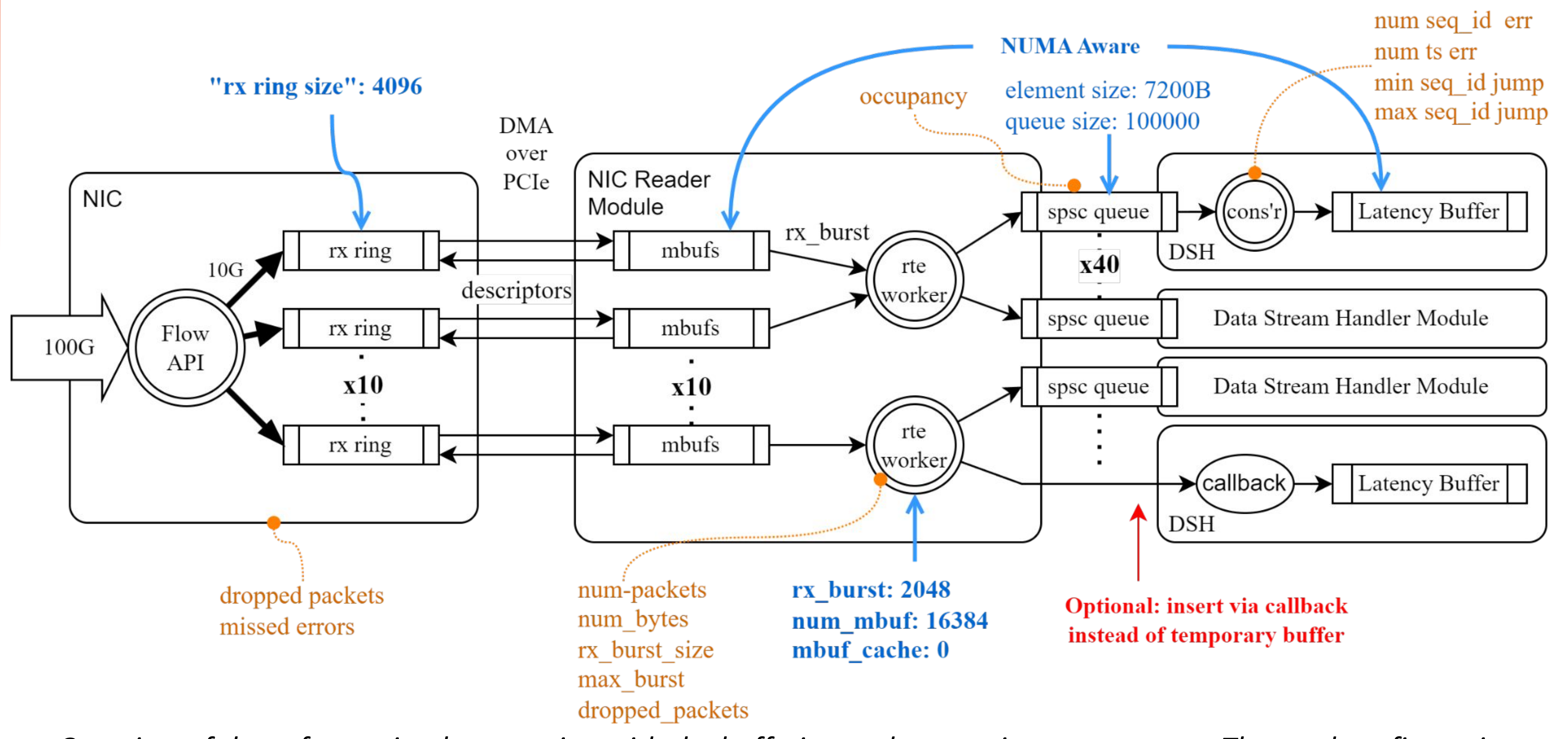
The readout units' system configuration is automated to allocate huge-pages of memory on the same NUMA location where the NIC is connected, and that the 100Gb NIC interfaces can be bound with the PMD driver. The readout subsystem uses the following key DPDK features:

- EAL (Environment Abstraction Layer) - Provides the main entry point for configuring and controlling the interfaces, like their RX/TX descriptors. Provides access for resources on the system (CPU cores, NUMA aware DMA buffers).
- Mempool and Mbufs - This library is dedicated to creating and allocating memory pools, and data structures that carry network packets as messages.
- Flow API - A programming interface for defining rules for balancing and routing network packets to dedicated RX/TX descriptors based on configurable rules.

Software implementation

The DPDK based implementation consists of a new dynamically loadable module implemented within the DUNE DAQ Application Framework[3]. The module implements the standard DAQ module interfaces for configuring, controlling, and monitoring the underlying resources, in this case the network interfaces. The main functional steps are the following:

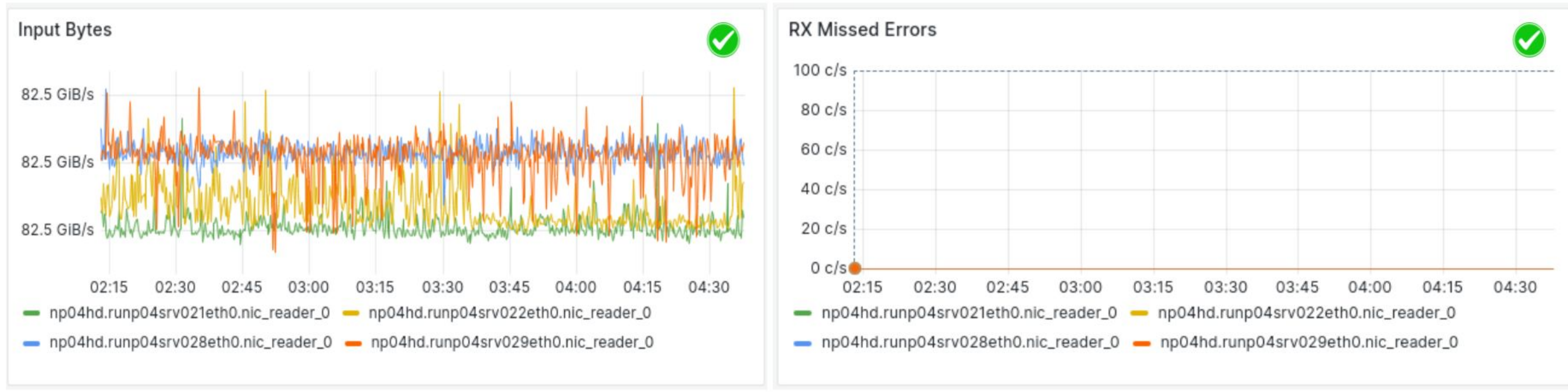
- Initialization: Based on a detector readout map a connection topology is established. For intra-process message passing of the data, either intermediate buffering or callback (zero-copy) mechanism is used between the modules.
- Configuration: Through EAL the selected interface is initialized and configured with the provided parameters for MTU, number of RX rings and their depth to use, and allocation of memory pools, each with given number of memory buffers. Using the Flow API a load balancing policy is configured, based on the topology provided during the initialization step. Each source IP is routed to a dedicated RX ring, and they are distributed among the configured set of CPUs to be used. The CPU set defines the number of DPDK threads that can be launched for reading the DMA buffers and process a burst of packets.
- Start/Stop: The worker threads are spawned on each CPU from the set defined in the configuration. They are polling the assigned descriptors for acquiring a burst of network packets, which are reinterpreted and copied out from the DMA buffers, and forwarded to intra-process connections towards the generic readout modules.



Overview of the software implementation with the buffering and processing components. The used configuration parameters (arrows) of these elements are also highlighted with the operational monitoring metrics (dotted lines). The last stream handler module shows the callback mechanism instead of the intermediate buffer.

Performance evaluation

There is an extensive operational monitoring metric set provided by the NICs' hardware counters that is periodically acquired through the DPDK extended statistics API. These contain information about throughput, packet burst occupancy rates, and different error conditions like CRC, dropped and missed packets occurrences. Several low-level processor counters are also gathered for analysis, like cache misses and memory bandwidth utilization. These data allowed finding the optimal configuration parameters for the system. Hardware topology oriented tuning for NUMA awareness, last-level cache (LLC) efficiency, and processing threads CPU affinity control is in place for achieving deterministic quasi-real time performance without packet loss.



The left plot shows the data rate from 4 detector elements (APAs) of the NP04 prototype detector at CERN. The data is received by 4 readout units, each equipped with a 100Gb NIC. On the right plot the total number of missed packets are shown, which can occur if there is processing backpressure. Both plots indicate that for several hours the full NP04 detector's data was acquired via the Ethernet readout without any data loss.

These optimization techniques are essential for scaling up configurations, with multiple 100Gb interfaces and detector elements being read out by a single CPU server, including all processing functionalities like the Trigger Primitive Generation and the Supernova Burst persistency (on NVMe drives for over 100 seconds).

Summary

The Ethernet readout is successfully integrated into the DUNE DAQ system and used in standard operations for the DUNE detector prototypes at the Neutrino Platform at CERN, and in ICEBERG at Fermilab. The full readout feature set and requirements were validated and demonstrated using multiple generations of CPU servers. Scalability studies and further performance evaluation with different hardware components and topologies are ongoing in order to finalize the readout units' technical specification.

References

[1] [2] DPDK [3] DUNE DAQ AppFwk

Contact

roland.sipos@cern.ch

Contribution ID: 5765004

Affiliations

CERN