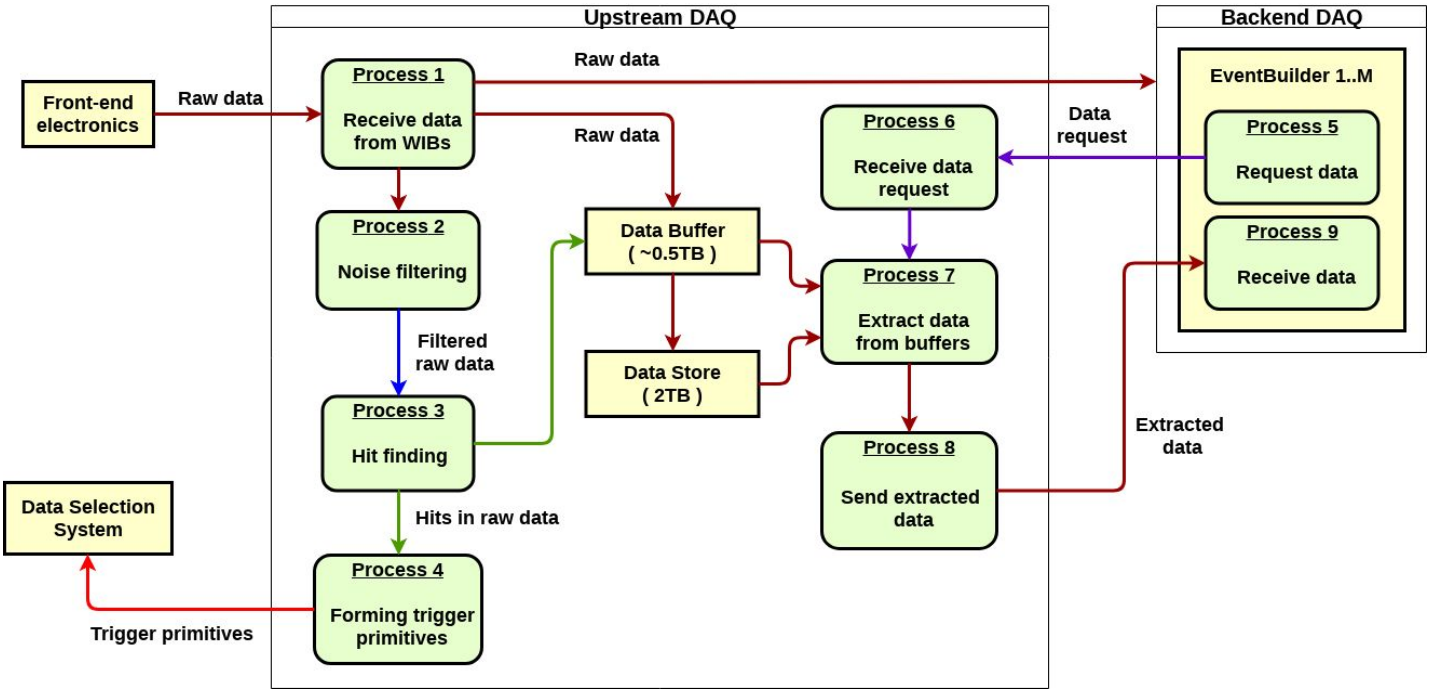
DUNE Upstream DAQ – Readout Technology Evaluation

The review will cover possible implementations for key functions of the Upstream DAQ :

- Trigger primitive generation
- Latency (10s) buffer
- SNB (100s) buffer
- Data request handling

These elements and their interfaces are shown in the diagram below. For each component, firmware and software implementations have been studied, and demonstrated to varying degrees.



For each component, we will review the current status of development, as well as the proposed final design. The discussion of the different components and how they may be combined into a fully functional system (see *Readout system variants* section), should be carried out taking into account the layout of the DAQ (see *Readout Infrastructure* section) as well as the foreseen operation modes (see *Readout Operation Modes* section). The criteria through which options will be compared are described in the *Criteria-based Assessment* section.

This review is the first step of a process which aims to arrive at a consensus on:

- The *baseline implementation* for the readout system
- Alternative solutions that bring potential future benefits, that should be pursued at a lower priority than the baseline

Readout infrastructure

The physical setup for the DAQ at SURF foresees 1 barrack (max 125 kW, 16 racks) underground, for each detector module, which will house equipment for UD, DS, SC, CCM and networking. Triggered data will be transferred to the surface where the dataflow storage and the high level filter are located. It is foreseen that the readout system for the first and second SP modules will be based on ~80 servers (at least 2U) each, supporting the readout of 150 APAs as well as of the photon detectors. Thus, one server will need to handle 2 APAs (TPC). A FLX card may support one or two APAs (i.e. a server may house one or two FLX cards).

Readout Operation Modes

In the following we describe three types of data taking that the readout shall support concurrently for DUNE.

1) Trigger requests for windows of few [us] to few [s]

This is the main way in which the readout will participate in the data taking. The data selection system will issue trigger decisions and the readout units will be asked to provide data for some or all channels with a variable readout window and offset. At present, we consider the granularity to be at the level of a full APA for those data requests. This may change in a scenario in which we move to a RoI based readout, but is not the baseline.

There are some differences with respect to ProtoDUNE that shall be taken into account for DUNE:

- There is no requirement for data requests to be time ordered: since the data selection system is by nature asynchronous, the decision of capturing some data may be taken faster or slower depending on the signature.
- There is no sequence number allowing to reorder data requests in a readout unit: since trigger records will often be formed with a subset of detector data, not all readout units will receive data requests for all trigger decisions.

- The readout window and offset will be variable on an event by event basis.
- It may be possible (though unlikely) for data requests to partially overlap.

2) Debug / calibration data streaming

Based on information contained in the WIB (or PDS data) headers it will be necessary to be able to stream out data from the readout units for the purpose of debugging and/or calibration. The granularity here is at the level of a single data link.
This type of data taking is “triggered” by the content of the data headers and the streaming shall be configurable (e.g. if calibration flag X is set in a WIB frame, then stream out 10 frames before and 25 frames after it to calibration stream X; if error Y is detected stream out the problematic frame, applying a throttle filter to avoid a data flood if all frames start being corrupted). Those data records will need to be surrounded by a header and sent to the dataflow for storage. The details of the data transfer are still to be finalised with the DFWG, but the logic in the readout units to be able to check WIB headers for specific calibration flags, as well as errors, and of spontaneously forming these data fragments is clear.

In ProtoDUNE we did not exercise this type of data taking, which nevertheless shall be fully supported for ProtoDUNE II.

3) SNB

If the data selection system detects a potential supernova burst, then the readout will receive a data request for an extremely large readout window, for all APAs. The TDR foresees a window of ~100 s with an offset of ~10 s with respect to the trigger timestamp.
Due to the high-value of those data the readout will persist them locally and then transfer them to the dataflow system at a low pace. This implies that the readout shall be able to sustain a continuous data input of ~10 GB/s for each APA and concurrently be able to persist data with a similar throughput for about 100s. In parallel to this activity, the readout shall of course continue supporting the other two data taking modes outlined above.
The details on how the stored data will be sent out to the dataflow need to be finalised, but it is important to highlight that besides storing the full data, the readout will have to be able to rapidly and reliably extract them from storage and forward them to the DF.

Known Failure Scenarios

There are certain conditions in which the data integrity is compromised hence these have implications on underlying functionalities of the upstream DAQ. Early recognition, reporting, and mitigation of these errors will be crucial for efficient data-taking. Based on the experience of ProtoDUNE-SP, there are three critical conditions that directly affected raw data integrity:

1) Low level FE failure

The lowest level of possible failure is a single logical unit being in an error state, which is a single WIB link (2 FEMBs). Failure of a single link should not compromise normal data taking conditions of other functional links in the readout.

2) Link alignment error

Link alignment errors between the WIB and FELIX may occur. This leads to missing data blocks and non-consecutive timestamps. All processing stages are required to be resilient against these faults in data.

3) Incompressible data

ProtoDUNE-SP's APA3 had a "faulty" FEMB that forced the RCE readout to exclude data compression for certain front-end links of APA3. This required a special firmware variant to cope with the problematic FE. Data compression cannot be guaranteed under all conditions, and should be a dynamic feature that can be toggled on or off. This is also important for the calibration and debug streams.

Readout system variants

In order to maintain unidirectional flow of data (from FE to backend) in the readout system, only certain viable combinations of the functional elements are feasible. Any other combinations would require additional data transfers between the FPGA and the hosting server, which would lead to extra processing cycles and substantially increased throughput between the card and the server. The following table shows the viable combinations of functional elements for a readout system.

Combination	*HitFinding*	*10s buffer*	*SNB store*
A	Host	Host	Host
B	FPGA	Host	Host
C	FPGA	FPGA	FPGA

Criteria-based Assessment

Criteria-based assessment is a qualitative assessment of a development project in terms of sustainability, maintainability, and usability. This can inform high-level decisions on specific areas for project improvement. The assessment is based around 5 broad
criteria, listed below, with specific points to be addressed in the review.

Criterion	Questions
Features	• *Which features have existing implementations and/or demonstrators?* • *What features are missing and how much further development is required?* • *How does the solution interface to other components of the Upstream DAQ, and wider DAQ? (e.g. Dataflow, CCM, Data Selection interfaces)*
Adaptability	• *Were any components that rely on this technology integrated in existing DAQ systems? (e.g.: within ProtoDUNE-SP DAQ)* • *How would the solution adapt to potential new requirements*

	of the DUNE DAQ? (for example, different TP algorithms, RoI-based readout) ● *Is it possible to tune and align the solution to support new ideas (e.g.: additional interfaces) or are there intrinsic structures that constrain potential extensions/modifications?*
Reliability	● *Can you ensure data integrity using this technology?* ● *How do you handle and mitigate the known failure scenarios (above) and other errors in order to avoid their propagation to other components?*
Long term support & maintenance	● *What is the balance between in-house development and COTS components?* ● *Does the solution require specific hardware products? (E.g.: only works with a specific SSD variant, or with any kind? Requires specific FPGA/CPU models or features? Are there implications for spares?)* ● *Can this solution profit from research and development outside DUNE (eg. by manufacturers, other experiments, etc.)?* ● *How strong and connected is the community of users and partners around the technology?* ● *Do we have sufficient engineers and developers with the necessary expertise to support this solution?*
Resource requirements	● *What are the resources (FPGA, cpu-cycles, memory) required by the solution as it stands now, per APA?* ● *What are the future prospects for reducing resource use?*