Requirements to the L1 front-end electronics

LHCB Technical Note

Issue: Second release
Revision: 2.0

Reference: LHCB 2003- 078
Created: June 2001
Last modified: October 7, 2003

Prepared By: Jorgen Christiansen

Abstract

The data flow architecture and requirements to the L1 front-end electronics of LHCb sub-detectors are defined. (Supersedes LHCb note 2001-127 [17])

Document Status Sheet

Table 1 Document Status Sheet

1. Document Title: Requirements to L1 front-end electronics			
2. Document Reference Number: LHCb 2003- 078			
3. Issue	**4. Revision**	**5. Date**	**6. Reason for change**
Draft	0.1	September 2001	Preliminary version distributed for Rio LHCb week.
Draft	0.2	November 2001	2. nd. Preliminary version distributed for LHCb week
Draft	0.3	December 2001	3.rd. Preliminary version. L1 trigger spacing of 900ns for trigger accepts to allow 36-word event fragments in dual port memory implementations. Appendix 1 added.
First release	1.0	January 2002	Released as LHCb note 2001-127 L1 buffer depth changed from 1926 events to 1820 events to be compatible with up to 36 words per event in dual port memory mode. 100KHz L1 trigger rate changed from being a defined requirement to be a feasible upgrade path with limited additional costs. Propose L1 throttle monitor made with 32bit time counter. Minor text corrections
Second release		October 2003	Released as LHCb note 2003-078 L1 buffer size increased from 1820 to 58254 events. Upgrade by an additional factor of 2 must be possible. L1 trigger accept distribution spaced by 20 us to

			ease implementation. Maximum L1 rate decreased from 100kHz to 40 kHz. Interface to link card updated (Not anymore S-link) Data flow and requirements to L1 trigger interface included. FEM and RU as interface to DAQ removed. DAQ redefined as combined HLT and L1 trigger system Data formats to HLT and L1 trigger updated, interfacing directly to commercial switching equipment Multi event packaging for HLT and L1 included Definition of buffering requirements related to event packaging and link interface. L0-ID in L1 front-end changed from 16 bit to 32 bit. L1 trigger distribution with only accepts (appendix 1).

Table of Contents

1. Introduction

The L1 front-end electronics is defined as the last stage of the sub-detector specific front-end electronics, before data is sent to the common DAQ system [18] (in this document defined as combined HLT and L1 trigger implementation). Event data from the L0 derandomizers, in the L0 front-end electronics [2], must be collected and stored in the L1 buffer during the L1 trigger latency. Events accepted by the L1 trigger must be extracted from the L1 buffer, derandomized in the L1 derandomizer, zero-suppressed and finally formatted to be sent to the DAQ system on standardized readout links. In the DAQ system, event data will not be pre-processed before it arrives in the CPU's of the High Level Trigger (HLT) and L1 trigger processor farm. All special sub-detector specific processing of detector data must therefore be performed in the L1 front-end electronics.

The data flow and basic requirements to the interface to the L1 trigger system is also shortly described as the extraction of event data and its transmission to the L1 trigger system are implemented on the L1 front-end modules of the detectors contributing to the L1 trigger.

The logical data flow architecture of the L1 front-end is described and key parameters related to its function are defined and specified. The definition of the architecture and its parameters are based on extensive simulations of the LHCb front-end architecture [3]. The L1 front-end electronics architecture described in this document is a logical data flow model used to define requirement parameters. The physical implementation of this architecture may possibly look quite different, as long as it conforms to the defined requirements.

A large set of general requirements given in the L0 requirements document [2] will not be repeated in this document. The L1 front-end electronics must also comply with the general requirements given in this document (Radiation hardness, testability, reliability, ECS interface, etc.). Continuously updated information about the front-end electronics can be found on the web [1].

2. L1 Front-end Data Flow Architecture

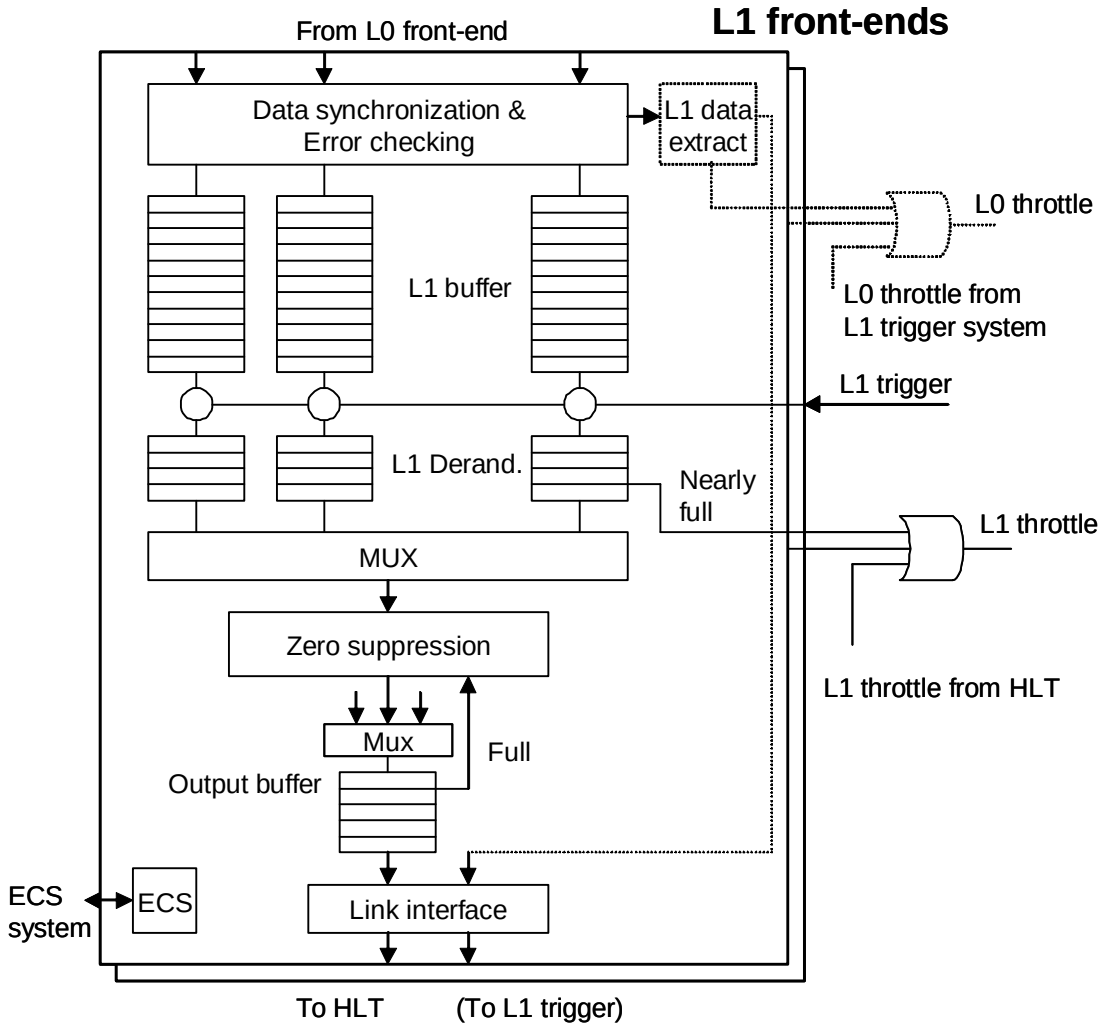

Figure 1. General data flow architecture of L1 front-end electronics.

The general data flow architecture of the L1 front-end electronics is shown in Figure 1. Event data accepted by the L0 trigger is received from the L0 front-end electronics on a set of sub-detector specific links. If data have not yet been digitised, it must be converted into a proper digital format, as sensitive analogue data cannot be stored reliably during the relatively long L1 trigger latency. For sub-detectors contributing to the L1 trigger a special data extraction is performed and a minimized data set is sent to the L1 trigger system. Received data from the L0 front-end is stored in the L1 buffers, waiting for the L1 trigger decision to be distributed to the front-end via the TTC system. Event data accepted by the L1 trigger is transferred to the L1 derandomizers, where it waits until it can be processed by the zero-suppression. After zero-suppression and proper event formatting the data is finally sent to the DAQ system on standardized links. Control and

monitoring of the module is performed via the ECS (Experiment Control System, [14]) interface.

Buffer overflow prevention in the L1 front-end electronics is made as a combination of central control and a hardwired L1 throttle signal. The L1 buffer occupancy is controlled and monitored centrally in the Readout Supervisor [3], [7] based on a set of strictly defined parameters. Buffer overflows after the L1 accept are prevented by a hardwired L1 throttle signal, as local buffer occupancies can not be predicted centrally when event data are zero-suppressed. The L1 derandomizer must handle incoming trigger accepts during the effective delay of activating the throttle mechanism. Following data buffers are assumed to back-propagate their full status to the L1 derandomizer. To handle possible buffering problems in the L1 trigger data extraction the L0 throttle signal is used. In a centralized fashion the L1 trigger and the HLT can also assert the throttle signals via the ECS interface of the Readout Supervisor.

The L1 front-end electronics modules interface directly to a commercial readout network of the DAQ system [18], as shown in Figure 2. For low rate links carrying data for the HLT small concentration switches are used to reduce the number of ports needed on the main readout network. A similar scheme will be used to collect information from sub-detectors participating in the L1 trigger. The standardized data links will be based on the Gigabit Ethernet standard, which has become the de-facto standard copper/optical link technology used in the computer industry.

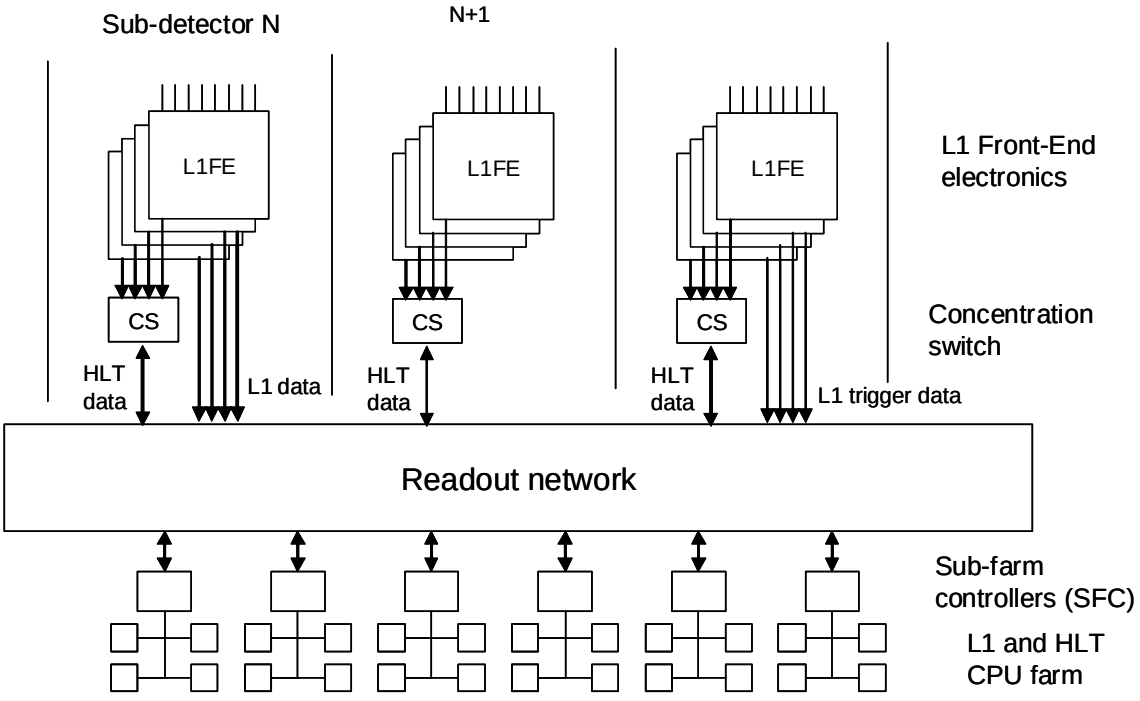

Figure 2. L1 Front-end interface to DAQ.

Strict definitions of data rates and event formats must be followed to have a reliable system where buffer overflows are prevented and event synchronization is maintained across the whole system.

3. Input data synchronisation and checking

The input synchronization stage of the L1 front-end electronics must receive event data, from the L0 derandomizers, that has been accepted by the L0 trigger. Event data received on different input links will have some timing skews related to the fact that they arrive from different detector parts and have variations in cable lengths. The input skew can in most cases be limited to a few clock cycles and the data alignment between different sources can therefore be performed with small data buffers. The correct reception of event data must be continuously verified. Data tags in the event fragments must be used to verify the correct function and synchronization of the L0 front-end electronics. After appropriate data verifications, the event fragment must be properly formatted before being stored in the L1 buffer. Analog input data must be converted to a digital format, if this has not already been done in the L0 front-end.

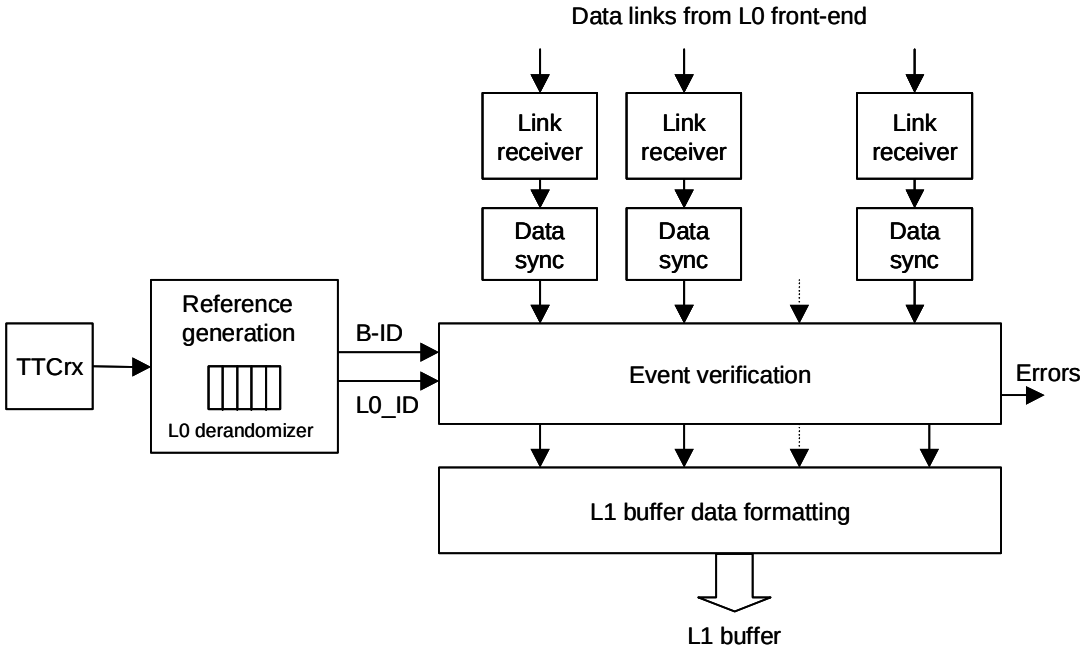

Figure 3. Functional diagram of input synchronization and checking stage.

Event data from the L0 derandomizer is specified to consist of maximum 36 words at a rate of 900ns per event (25ns per word). The 36-word event fragment consists in the general case of 32 detector channel samples and a maximum of four data tags. The data tags contain event information that must be used to verify the correctness of data generated by the L0 front-end electronics.

The data tags are required to contain a bunch crossing identification (B-ID) of the event (or other equivalent information) to be capable of verifying the correct function and synchronization of the L0 front-end. The bunch crossing identification of the event fragments received must be verified by comparing it with a local reference. Any

discrepancy must be reported as an error condition. Other data tags (e.g. L0-ID) must also be used to verify the correctness of the received event fragments. The L0 front-end electronics are in most cases located in an environment with radiation that may corrupt its correct function by Single Event Upsets (SEU). The L0 front-end electronics should therefore in general be assumed to be somehow “unreliable”.

The data tag containing error information from the L0 front-end must be maintained and any further errors detected must be appended. All detected errors must also be made available to the ECS system via error status registers, so detailed error diagnosis can be performed when needed.

After a reset of the L0 front-end electronics (L0-reset) the data synchronization stages must re-synchronize themselves with new data streams. When the L0-reset is issued, without the L1 reset, the Readout Supervisor will ensure that event data from correctly working L0 front-ends are not truncated (but this may not be the case for L0 front-ends which have suffered a SEU or other mal-function). No new valid event data will arrive before 160 clock cycles after the L0 reset and the first new valid event fragment will have the L0-Event-ID set to zero.

Event data from the L0 front-end is specified to arrive at a maximum event rate of 900ns per event (36 words @40MHz). Event data will in most cases be organized as a small event header followed by 32 data samples. The data tags in the event header from the L0 front-end may have to be reformatted before being stored in the L1 buffer. Data tags, which have already been used for data verifications between individual sources, can at this point be shared between several groups of channels.

Data to be stored in the L1 buffer must include event data error information and the L0 event ID and the Bunch ID. Error flags must be carried along with the event data to determine if it can be considered to be correct in following processing stages. The L0 event ID is needed to verify, at the output of the L1 buffer, that a given L1 trigger decision matches the event data available at the output of the L1 buffer. To ensure that each event in the L1 buffer have a unique L0-ID identification the L0-ID used in the L1 buffer must be minimum 24 bits (32 bit recommended). The L1 buffer will in many cases be made as a wide memory, with a single controller handling event data from several input links, where event tags can be shared across multiple channel groups as indicated in Figure 4. The B-ID (for common L0 and L1 electronics see: 15.1) and other additional data tags should be maintained in the data stream.

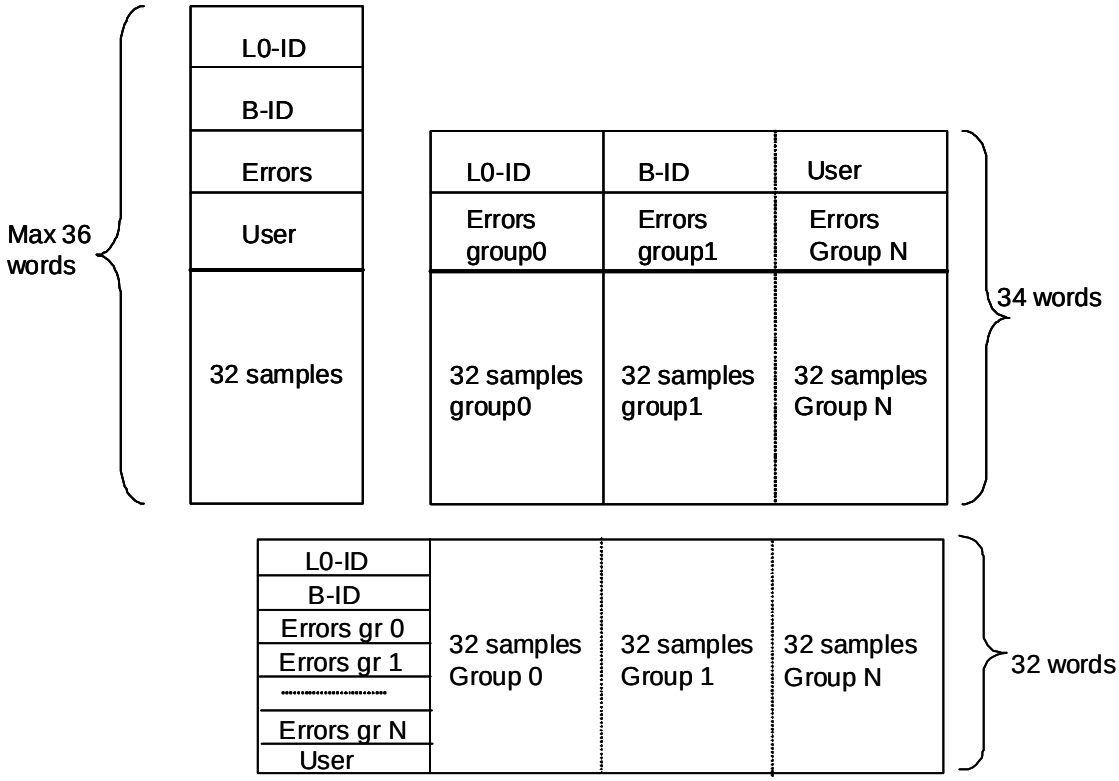

Figure 4. Alternative data formatting in L1 buffer to match memory configuration.

The L1 front-end electronics will normally receive event data from multiple L0 front-end units. In case a specific L0 front-end unit has been found to be malfunctioning it must be possible to ignore event data from any L0 front-end branch, controlled from the ECS.

Requirements summary:

- Minimum event spacing: 900ns.
- Detection of correct event reception.
- Verification of B-ID, L0-ID and other event tags from L0 front-end.
- Inclusion of (shared) event tags to L1 buffer: Error flags, 24bit (32bit) L0-ID, B-ID.
- Ignore input data in the first 160 clock cycles after a L0 front-end reset.
- Possibility to ignore (mask) data from any input source.

4. Data extraction and Interface to L1 trigger

Sub-detectors contributing event data to the L1 trigger must extract a minimal data set with information needed by the L1 trigger system. The high event rate (1.1 MHz) at this location in the general data flow puts strong constraints on the extraction processing and the amount of data that can be transmitted to the L1 trigger. All event processing must be done with a guaranteed minimum rate of one event per 900ns. This will in most cases require heavy use of parallel processing both at the event level (event pipelining or multiple processing units) and possibly also at the clock level (traditional clocked pipelining).

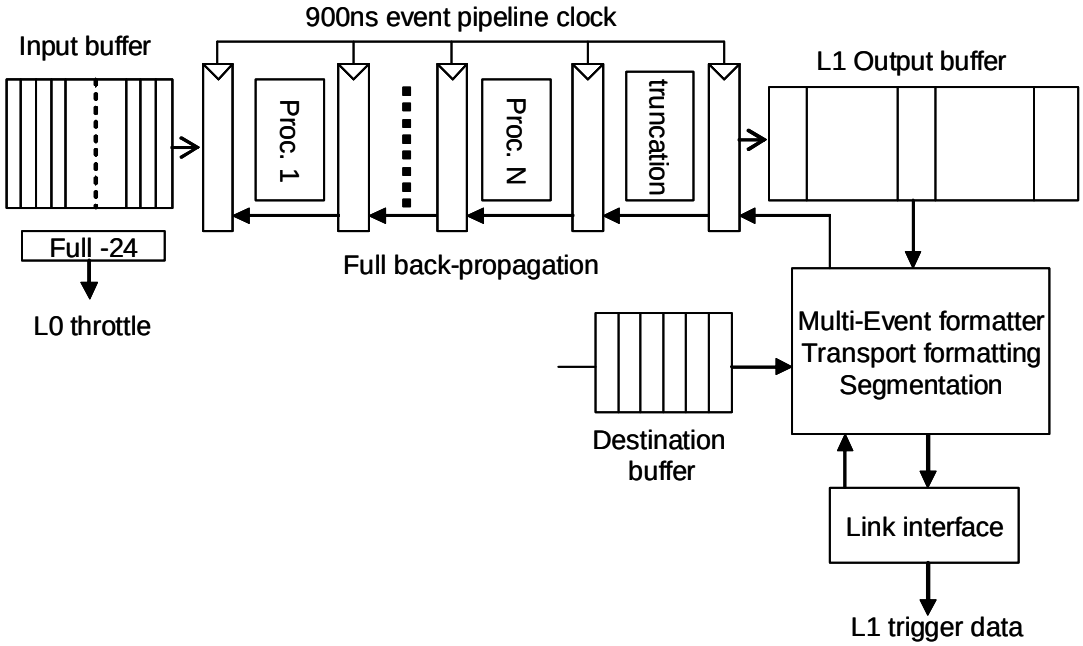

Figure 5. Generic data extraction for L1 trigger

The data extraction needed for the L1 trigger will in many aspects resemble the processing done in the zero-suppression for the HLT data (see chapter 9). Options must be available to separately control processing parameters for the L1 data extraction from ECS. It may for instance be needed to mask certain noisy channels in the L1 trigger data stream to maintain a minimal event size. The same channels may though still need to be read out to the HLT, as higher level processing may still extract useful information from a not too noisy channel.

It is strongly recommended to use a processing scheme with a guaranteed maximum processing time per event, compatible with the required rate, without the use of internal derandomizer buffers. The event rate itself has at this point already been derandomized by the L0 derandomizers, to a guaranteed event spacing of 900ns. Event synchronization

must never be lost as this will require the front-end and DAQ system to be reset and restarted.

At the final output stage of the processing, where extracted data is transferred to the link interface card (see chapter 14) to be sent to the L1 trigger system, some data derandomization will though be needed because of the fixed bandwidth of the link, event size fluctuations after data extraction and the multi-event packaging required by the L1 trigger system [18].

A combination of two schemes should be used to prevent buffer overflows. Data frames sent to the L1 trigger must be limited to a well defined maximum size. This will require event data truncation for certain events that must be clearly identified in their event headers. The truncation size should be programmable from ECS to allow this to be optimized for a given running scenario (background, trigger rate, etc.). When internal buffers start to fill up, the L0 throttle signal must be asserted sufficiently early to prevent the readout supervisor to generate further L0 trigger accepts. When the L0 throttle signal to the readout supervisor is asserted, multiple events may though still arrive from the L0 front-end electronics. Up to 16 events can be waiting in the L0 derandomizer and multiple (maximum 4) L0 trigger accepts may occur during the latency of the throttle signal network, readout supervisor and the TTC distribution system. A few events must also be accounted for in the links from the L0 front-end to the L1 front-end and in the input synchronization stage (set to maximum 4). This implies that the L0 throttle must be raised quite early at a time where there is still space for 24 full events. It is therefore reasonable to require an input buffer of 32 events. The packaging of multiple events before sending them to the L1 trigger system, plus the sharing of a Gigabit Ethernet plug-in card, will also require significant output buffering (see also chapter 10). It is strongly recommended to have a buffer strategy where intermediate buffers and output buffers back-propagates until the input buffer. When the input buffer only has space for additional 24 events the L0 throttle must be asserted. The water mark levels where the L0 throttle is asserted and where it is released must be programmable from the ECS interface. The generation of the L0 throttle must follow the same general rules as defined for the L1 throttle (see chapter 11)

The buffering strategy described above may in some cases imply the use of large buffers. To handle extremely rare cases of data congestion it can be envisaged to truncate all event data when buffering resources are close to be exhausted. Event headers must though always be maintained to ensure event synchronization. This scheme can reduce buffering requirements, but is in conflict with the principal rule that it must be possible to reconstruct the L1 trigger processing performed, based on event data sent to the HLT system (se below).

An important requirement to the local data extraction comes from the fact that it must be possible to regenerate fully the L1 trigger processing for all events that have been accepted by the L1 trigger. Dedicated random triggers will be generated for this to crosscheck the correct function of the local data extraction units and the global trigger decision processing (to verify events that would normally have been rejected by trigger system). This is very important to be capable of calculating trigger efficiencies and

possible physics biasing. It must therefore be possible to reconstruct on/off – line the data that have been extracted locally and sent from each data source to the L1 trigger. Doing event truncation based on the filling of local derandomizer buffers can by definition not be fully reconstructed as it depends on previous events that may have been rejected by higher level triggers. An acceptable handling of such a problem is to use a single bit per event to flag to the HLT if some "unpredictable" data truncation has occurred. This single bit per event truncation flag must be stored in a small separate L1 buffer, or alternatively it must be possible to access the event data stored in the normal L1 buffer and set the truncation flag.

Simple data truncation of events based on a fixed maximum event size can normally be reconstructed based on HLT data when knowing the exact algorithm used and taking certain precautions (e.g. must not have any dependency on previous or later events). This relies on the assumption that the zero-suppression performed on the HLT data stores sufficient information to fully reconstruct what has been extracted to the L1 trigger. To ensure this, it can in certain cases be required to repeat the zero-suppression algorithm for the L1 trigger (now at much lower rate) together with the normal zero-suppression for the HLT. In case the zero-suppression for the HLT is equivalent to the L1 trigger zero-suppression then this is obviously easily accomplished (with a possible exception that for the HLT amplitude information is kept for channels with detected hits where for the L1 trigger only binary information is kept).

To allow the system to monitor the extent of data and event truncation individual counters for the occurrence of the two types of truncations must be made accessible from ECS.

Because of the high event rate, and the limited available bandwidth of the links to the L1 trigger, it will be required to use a very compact data format. On a single Gigabit link an absolute maximum average fragment size of ~120Mbytes/s / 1.1 MHz = ~109 Bytes per event including overheads must be maintained. This makes it reasonable to define an absolute maximum size per event of 256 (or 512) bytes. It is important to define a maximum event size at this level to be capable of defining requirements for the output buffer (see end of this chapter). To ensure an absolute minimum data size it will for the L1 trigger links be required to use a very compact event format [19]. The option for future upgrades is to use a quad link interface card (see chapter 14).

Sending many small events at high rate on Gigabit Ethernet links is very inefficient because of the transport formatting overhead. Sending many small event fragments will also pose problems in the sub-farm controllers that would have to handle a very high IO interrupt rate [18]. It has therefore been decided to perform event packaging of multiple events into Multi-Event Packets (MEP). An event packing factor of up to 32 has been found appropriate for the L1 trigger (must be programmable from ECS). The fact of packaging multiple events also reduces the fluctuations on the size of data frames sent from the front-end. At an event packaging factor of 32 the average multi-event package must be smaller than 120Mbytes x 32/1.1M = 3490 bytes (one link used for L1 trigger data) which is a reasonable data size to handle with gigabit Ethernet. A three level data formatting similar to the HLT data is used for the data transport (see chapter 12). The multi-event formatting, transport formatting and fragmentation will be identical to the

HLT to use the same commercial readout network. The event data formatting itself must be strongly minimized and optimized for each detector to get the minimum overhead and must be defined in close collaboration with the L1 trigger coordination.

To ensure unique event identifiers of all events during their processing in the L1 trigger system a L0-ID of 32 bits is required at this level.

The multi event packaging and the limited link bandwidth will require a final level of output buffering. It is reasonable to require buffering for at least one full multi-event for active transmission and one for active assembly and formatting. With a maximum event packaging factor of 32 and a maximum event fragment size of 256 (512) bytes the output buffer must have a minimum size of 2x32x256 = 16kbytes (32kbytes). For applications where it may be needed to use the Quad link card to get sufficient bandwidth to the L1 trigger system this should be increased to 64kbytes

Requirements summary:

- L1 data extraction and transmission must keep up with 900ns event rate.
- Programmable masking of noisy channels (independent from HLT masking)
- Buffer overflow prevention schemes
  A: Guaranteed maximum processing time per event below 900ns
  B: Use of back-propagation to input buffer and assert L0 throttle when input buffer only have space for 24 additional events (programmable)
  C: Data truncation above a certain event size (programmable)
  (D: Only keep event headers when buffers close to overflow)
- Input buffer of minimum 32 events
- Data and event truncation counters accessible from ECS
- It must be possible to reconstruct off line the L1 trigger processing of stored events.
- Multi event packaging of up to 32 events (programmable)
- Output buffering of minimum two maximum sized multi-event packets.
- Minimal event formatting overhead.
- 32 bits L0-ID needed in output data formatting.

5. L1 buffer

The L1 buffer must store event data, while the L1 trigger system determines which events to accept for further processing in the high level trigger farm (HLT). The L1 trigger system has largely varying decision times for different events, and trigger decisions are therefore taken out of order. To simplify the front-end electronics it has been decided that the L1 trigger decisions are reorganized back into their original chronological order. This in practice means that the trigger latency seen by the front-end is close to the maximum trigger latency [3].

The L1 buffer has been defined to be minimum 58254 events deep (2M / 36), as a compromise between cost and complexity of the L1 trigger system and the front-end systems. Events will be written to the L1 buffer with a minimum event spacing of 900ns, given by the L0 derandomizer readout time. A future increase by a factor two must be possible by exchanging memory chips used to implement the L1 buffer.

L1 trigger decisions are defined to be distributed to the front-end in the same order as the events entered the L1 buffer. This initially allowed the L1 buffer to be implemented with a simple First in First out (FIFO) buffer scheme at the event level as real physical FIFO buffers needs the same time to read a rejected event and an accepted event. This implied a strict bandwidth handling at the output of the L1 buffer that enforced complications in the implementations of the L1 front-end modules (this was the case in the first version of the L1 front-end specifications [17]). With the significant expansion of the L1 buffer since its original definition (factor 32) it has become impractical and too expensive to use real physical FIFOs. The L1 buffers will now in practice be implemented with QDR or DDR SDRAM where direct access to each individual event is possible.

Accepted events will be read from the L1 buffer with a maximum rate of 40 kHz. The average time between L1 trigger accepts is 25 us but this interval may have large statistical variations. A standard derandomization scheme would transfer an accepted event from the L1 buffer into a separate derandomizer buffer as quickly as possible. This would require a fast data transfer from the L1 buffer to a separate, sufficiently large, derandomizer buffer. Such a scheme will be optimal in cases where the L1 buffer is relatively small (which was previously the case). An alternative is to perform the derandomization in the L1 buffer itself and transfer the accepted events at a modest speed. The effective derandomizer needed in the L1 buffer itself is in this case 1.1 MHz/40kHz = 28 times larger. For an effective L1 derandomization of ~16 accepted events this scheme needs of the order of 28 x 16 = 448 events in the L1 buffer itself. For a large L1 buffer of 58254 events this is only a marginal part of the total buffer (0.8%) and this scheme has therefore been adopted when the L1 buffer size was increased by a factor 32.

Accepted events only need to be dealt with at the maximum frequency of 40 kHz, but the total handling of events (accepted and rejected events) at the output of the L1 buffer must still be performed at the same average rate as on the input (1.1MHz). The data bandwidth

out of the L1 buffer will be determined by the accept rate but the event handling frequency must match the input event rate. The event handling rate will determine the speed and complexity of the memory controller. An additional complication to take into account is the time needed by the TTC system to transmit L1 trigger decisions to the front-end electronics. The minimum time spacing between two short TTC broadcasts is 16 clock cycles (400ns). If an enforced time spacing between a L1 trigger accept and the following trigger decision (accept or reject) is made of 25us then data can be read slowly from the L1 buffer, but there will not be sufficient time to transmit all the L1 trigger reject decisions to the front-ends. This problem can be corrected for by allowing sending trigger reject decisions during this time interval but the first coming trigger accept must comply with the time spacing requirement. Such a scheme will introduce a complication in the L1 buffer controller that must be capable of handling incoming trigger rejects while concurrently handle the (slow) readout of the latest L1 trigger accept. Even with this scheme it will be difficult to catch up with pending trigger decisions after a sequence of consecutive L1 trigger accepts. Using a time internal of 20us (instead of 25us) for the readout of accepted events will ensure an effectively working buffer. With this maximum readout time the buffer controller must be capable of handling up to 20us/400ns = 50 trigger rejects while slowly reading out an accepted event.

To ensure compatibility with the previous version of the L1 front-end specification [17] a time spacing of 900ns will be enforced from a L1 trigger accept to the following L1 trigger decision (readout option B in Figure 6). This will allow a very simple L1 buffer control if it can be guaranteed that accepted events can be read out in 900ns (36 x 25 ns).

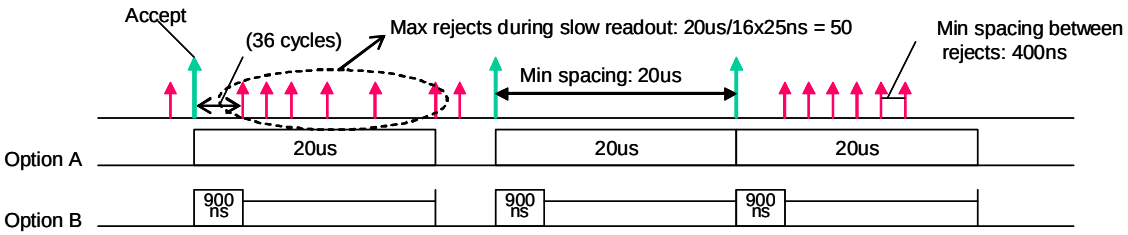

Figure 6. L1 trigger decision distribution to front-end electronics with two possible readout options.

The L1 trigger system must have a latency limit to prevent overflows in L1 buffers in the front-end, by using a system of timeout mechanisms. The L1 trigger system though does not have sufficient information, about when the trigger decisions actually arrive to the front-end, to directly prevent L1 buffer overflows. The Readout Supervisor will therefore continuously keep track of the occupancy of the L1 buffers, under the assumption that the front-ends only have a very limited local buffering of L1 trigger decisions (see chapter 6). When the Readout Supervisor estimates that the L1 buffers are in danger of overflowing, it will directly throttle L0 trigger accepts. For such a scheme to work reliably it is extremely important that all front-end systems comply strictly with the defined requirements. This L1 buffer monitor function actually allows running the L1 trigger system with larger timeout values, which would in principle imply L1 buffer overflows. The L1 buffer monitor will in this case throttle L0 triggers during periods when there is a

real risk of an overflow [3]. Local L1 buffer overflows, because of a malfunction, must be detected and an appropriate error status must be set. Features to allow monitoring of the local L1 buffer occupancy via the ECS interface will also be very useful to perform debugging.

It has been determined that the implementation of L1 front-end modules in fact do not necessarily need to get ordered trigger decisions when using a L1 buffer implementation with dual port memories. With dual port memories a circular buffer scheme is found to be more practical than a linear FIFO like buffer. With a circular buffer L1 trigger rejects does not need to be distributed, as events are automatically overwritten. Only L1 trigger accepts with a pointer to the location in the buffer is needed. With such a scheme L1 trigger decisions do not even need to be distributed in time order, which can give important simplifications in the L1 trigger system. L1 front-end modules should also support this alternative scheme as described in detail in appendix 1. If this poses problems for specific implementations then this must be discussed with the electronics and trigger coordinators to determine if it can be dispensated for. The common L1 front-end module used by most sub-detectors in LHCb will fully support both schemes [21].

Requirements summary:

- Minimum event spacing at input: 900ns
- Buffer architecture based on 36 (4 data tags + 32 data samples) words per event.
- Minimum L1 buffer depth: 58254 events.
- L1 trigger decision distribution spacing:
  L1 trigger reject to L1 trigger reject: 16 x 25ns = 400ns
  L1 trigger accept to L1 trigger reject: 36 x 25ns = 900ns
  L1 trigger accept to following L1 trigger accept: 20 us
- Local detection of overflow required to detect possible malfunctions.
- Support for alternative L1 trigger distribution scheme (appendix 1)

6. L1 trigger distribution

L1 trigger decisions will be distributed to the front-ends with a short broadcast message via the TTC system (see also appendix 1). Each trigger decision carries two LSB bits of the L0 event ID and a 3-bit trigger type [5], [6]. A trigger type equal to zero implies that the related event data from the L1 buffer must be rejected. A trigger type different from zero means that the related event data must be extracted from the L1 buffer and stored in the L1 derandomizer.

Bit[7]	Bit[6]	Bit[5]	Bit[4]	Bit[3]	Bit[2]	Bit[1]	Bit[0]
1	Type[2:0]			L0-ID[1:0]		X	X

Figure 7. Short broadcast used for L1 trigger distribution.

The L1 trigger type must be attached to the event data as further event processing may depend on this. A set of predefined trigger types, useful for the running of the L1 front-end electronics, has been defined as shown in Table 2. Reserved codes can only get assigned dedicated trigger types after an official acceptance by the front-end electronics, trigger and DAQ coordinators.

Type	
0	Reject
1	Normal physics trigger
2	Reserved for future use
3	Random trigger
4	Reserved for future use
5	Reserved for future use
6	Trigger for timing alignment (Isolated interaction)
7	Trigger on calibration pulse

Table 2. L1 trigger types distributed to front-end electronics.

Only two bits of the L0 event ID is distributed with each L1 trigger decision. These two LSB bits must be confirmed to be equal to the LSB of the L0 event ID tag of the event data from the L1 buffer. Any detected discrepancy between L0 event ID's must be considered an error.

The two LSB bits of the TTC short broadcast must not be decoded as a part of the L1 trigger distribution command as these bits are specifically used by the TTCrx as resets [6].

The Readout Supervisor will on purpose space the distribution of L1 triggers to the front-end, to allow all event data to have been copied from the L1 buffer into the L1 derandomizer before a new L1 trigger accept is distributed. The minimum spacing between the arrival of two trigger decisions will be 400ns with the additional spacing rules defined in the previous chapter. Only a limited number of L1 trigger decisions must be stored locally, as shown in Figure 8, before events are actually removed from the L1 buffer. The use of such a local buffer for L1 trigger decisions can simplify the L1 buffer controller. Without such a local buffer the L1 buffer controller must be capable of handling rejecting events while reading out slowly an accepted event. In case the readout of an accepted event can be done quicker than 20us (e.g. 20us – 50x 25ns = 18.75 us) then remaining cycles can be used to process quickly waiting trigger rejects (e.g. one event rejected per 25ns clock cycle). Such a local L1 trigger decision buffer must only be used to buffer incoming L1 trigger rejects during the slow readout of an accepted event. If it was used to buffer multiple incoming L1 trigger accepts then the L1 throttle mechanism cannot be guaranteed to work as intended. The Readout Supervisor must ensure that L1 buffers never overflow so it will include an additional margin for these trigger decisions waiting in this buffer. The absolute maximum number of locally stored L1 trigger decisions (rejects) is limited to 64 (slightly more than the maximum number of received rejects during the readout of an event = 50).

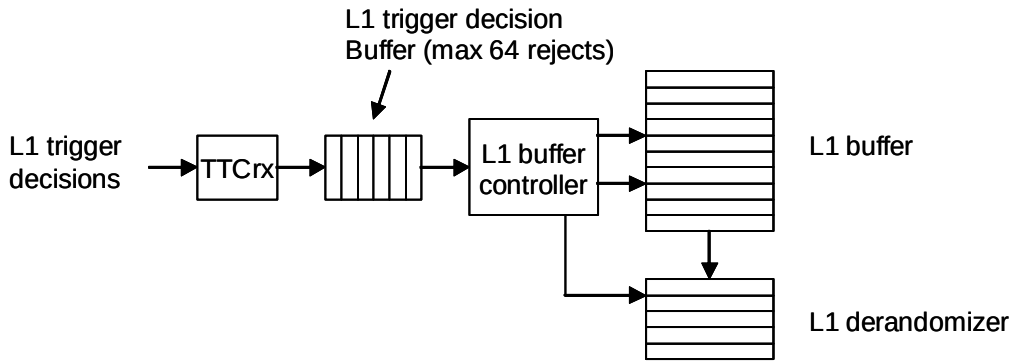

Figure 8. Local storage of L1 trigger decisions

Requirements summary:

- Trigger type = 0 -> trigger reject
  Trigger type <> 0 -> trigger accept

- L1 trigger type must be attached to event data.
- Check of L0-ID between L1 trigger decision and event data from L1 buffer.
- Maximum local storage of L1 trigger decisions: 64 (rejects)

7. L1 trigger rate

The maximum average L1 trigger rate is 40 kHz, as defined in the trigger TDR [18].

Requirements summary:

- Maximum L1 trigger rate: 40kHz

8. L1 derandomizer

Event data accepted by the L1 trigger must be stored in the L1 derandomizer buffer to smoothen statistical variations of the instantaneous trigger rate. The L1 derandomizer is also needed to allow a delay from the assertion of the L1 throttle until subsequent L1 triggers are converted into L1 trigger rejects.

Every event accepted by the L1 trigger must be assigned a continuous increasing 32bit L1 event ID (L1-ID) needed for further event processing in the HLT system. The first event accepted by the L1 trigger after a L1 front-end reset must be assigned a L1-ID of zero (see chapter 18.2.1).

The maximum instantaneous input event rate is defined to be one event each 20us (determined by the minimum spacing between L1 trigger accepts). The output rate is given by the average L1 trigger rate (40 kHz), with an additional margin to ensure an efficient derandomization.

For the L0 derandomizer, a set of strict requirements have been defined to ensure that the Readout Supervisor can centrally prevent overflows in all L0 derandomizers in the whole front-end system [2]. At level 1 the L1 derandomizer overflow prevention scheme is based on a physical throttle signal. This allows more room for flexibility in the different sub-systems, under the condition that they assert the L1 throttle correctly and that no specific sub-system enforces a significant dead time. It has been determined that the L1 throttle network will have a maximum latency of 2μs, including cable delays and the serialization of TTC broadcast commands. This is only a fraction of the guaranteed time spacing between L1 trigger accepts of 20us.

It is assumed that following data buffers in the front-end architecture back propagates their full status until the output of the L1 derandomizer is finally blocked. When this occurs, the L1 throttle must be asserted sufficiently early that the L1 derandomizer will not overflow. This will require the L1 throttle signal to be raised when there is still space for one additional event in the L1 derandomizer. This will in practice mean that the minimum configuration of the L1 derandomizer must have space for two full events. (A derandomizer of only one event could be envisaged but a minimum size of two events is required to ensure a stable system)

In the simple case of a guaranteed readout time of each event below 20us (e.g. constant processing time per event in zero-suppression) a 2 event deep L1 derandomizer is sufficient.

In case the processing time of individual events in the zero-suppression cannot be guaranteed to be shorter than 20us for all events, because of processing times in the following stages that depend on the event data itself, the L1 derandomizer must handle the derandomization needed. The derandomizer must in this case be made sufficiently

large that the total LHCb system never have dead times related to this above 1% at the defined maximum L1 trigger rate. This must be guaranteed with sufficient safety factors (factor ~2) on the occupancy of the detectors. A set of simple guidelines has been defined for the required derandomizer size as function of the average processing time. If the average processing time is below 15us a derandomizer of only 4 events can be considered sufficient. For an average processing time between 15us and 20 us a derandomizer of 8 events should be used. Finally for a processing time between 20us and 25us a derandomizer of 16 events must be used. The average processing time must obviously never exceed 25 us to be compatible with a L1 trigger rate of up to 40 kHz (unless having multiple parallel processing units).

When data is read out of the L1 buffer, the following information must be included to allow event verifications at later stages: B-ID, L0-ID, L1-ID, L1-TYPE and Error flags. This information is needed in the event formatting to the HLT system (see chapter 12). Notice that a separate derandomizer buffer may be needed for this information as indicated in Figure 9.

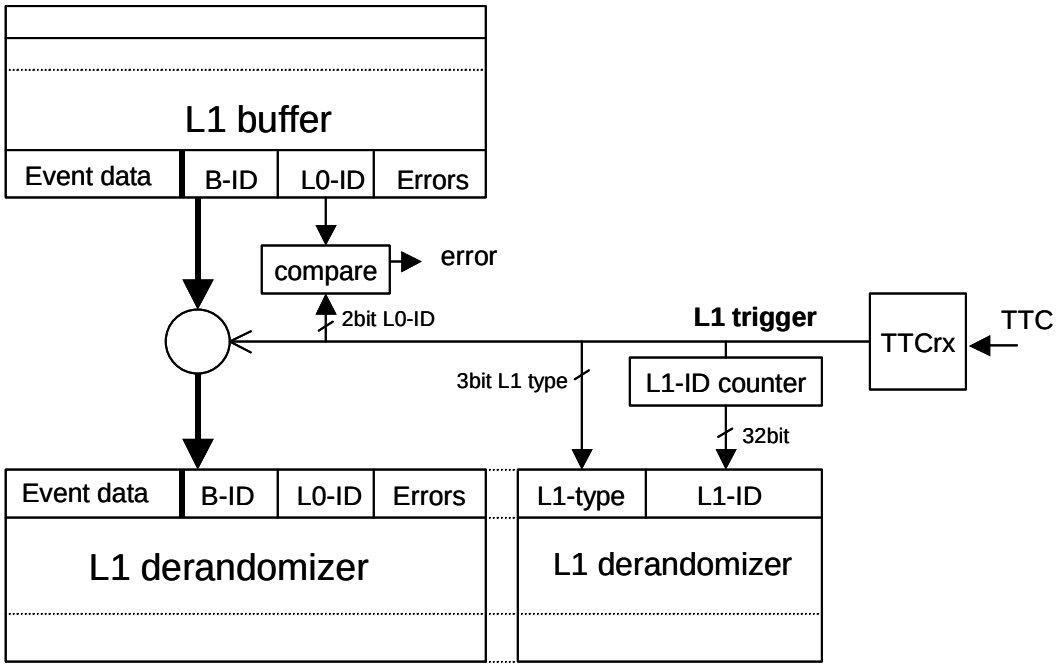

Figure 9. Attaching important data tags to event data at input to L1 derandomizer.

Any buffer overflow must be detected and the error flags of events and the status information to the ECS must be set accordingly.

Requirements summary:

- Minimum input event spacing: 20us
- Minimum dept:
  2 events if following processing is guaranteed to be shorter than 20us for all events.

4 events if average processing time is 15us or below.
8 events if average processing time is between 15 and 20us
16 events if average processing time is between 20us and 25us.

- Average output event spacing: 25μs @ 40KHz
- Assertion of L1 throttle when only place for one additional event.
- Local detection of overflow.
- Data tags to attach to event data at input to L1 derandomizer: L1-ID, L1-type, L0-ID, B-ID, and Errors.

9. HLT Zero-suppression

Event data must be properly zero-suppressed and formatted (sparcified) before sending it to the HLT system, to minimize the required bandwidth in the readout network. It must be possible to disable the zero-suppression for debugging or in case that unexpectedly high data occupancies make the zero-suppression worthless.

Zero-suppression will normally be based on a set of thresholds and pedestals. These parameters will in many cases be calculated and updated dynamically to obtain an efficient suppression factor without loosing information of interest. It should be possible to run the zero-suppression with fixed parameters, set by the ECS system, to help in early phases of testing and commissioning. Features to allow monitoring of dynamic zero-suppression parameters will also be useful during debugging and testing.

To limit system effects of noisy or malfunctioning detector channels it must be possible to mask channels individually. Such a channel masking will be registered in the detector conditions database.

For highly occupied events zero-suppression may actually increase the amount of data to transport. For sub-detectors where this is expected to occur for a significant fraction of events it can be considered to send non zero-suppressed event data for high occupancy events. This implies that HLT trigger routines will be forced to handle two different data formats for the same sub-detector and such a scheme can only be used after a confirmation from the HLT coordinator.

It is advantageous to fix a maximum event size as the multi event packing used for the link interface (see chapter 12) has a maximum defined size of 64kbytes. With a maximum event packing factor of 16, single events should be limited to 64k/16 = 4 Kbytes including all overheads. If such a hard limit on the data size per event can not be guaranteed a special agreement with the front-end electronics, trigger and DAQ coordinators must be defined and finally to be endorsed by the TB.

Requirements summary:

- Average zero-suppression time: < 25μs @ 40KHz
- Programmable disable of zero-suppression.
- Programmable zero-suppression parameters (pedestals, thresholds, etc.).
- Possible masking of noisy or malfunctioning detector channels.
- Maximum event fragment size: 4 Kbytes per front-end module

10. HLT Output buffer

There will be large variations in event size from one event to the next after zero-suppression. Some kind of output buffering will be required to smoothen the data bandwidth before sending event data over constant bandwidth links. When such a buffer runs full it must block the zero-suppression and the readout of the L1 derandomizer, without loss of any data (back propagation). If this situation is maintained for some time the L1 derandomizer will fill up, and finally throttle the L1 trigger preventing buffer overflows.

The major requirements to output buffering is determined by the packing of multiple single events into multi-event packets (see chapter 12). It is required to be capable of storing at least two such multi-event packets in the output buffer: one for transmission and one for assembly and formatting. The maximum size of a multi-event packet is determined by the maximum IP packet size of 64 Kbytes. The output buffer must therefore have a minimum buffering capability of 128kbytes.

The Gigabit Ethernet link card has an input buffer of 16 Kbytes in the Media Access Controller (MAC) that is used to derandomize the traffic to the link itself. The access to this buffer is though shared with data traffic to the L1 trigger (see chapter 14). The main purpose of this buffer is to ensure that the MAC has direct access to a complete Ethernet transport packet before starting its transmission.

Requirements summary:

- Back propagation of full status to zero-suppression and L1 derandomizer.
- Minimum 128kbytes output buffer.

11. L1 throttle

A simple Hardwired L1 throttle signal is used to prevent buffer overflows in the L1 front-end electronics and the DAQ system. Several hundred sources of L1 throttles must be ORed in a fashion compatible with the current system partitioning [16], to generate the final L1 throttle signal to the Readout Supervisor controlling this partition. A system of programmable throttle switches (ORs) [9] in the TFC system will be used in the last stage of the L1 throttle network to allow a flexible partitioning scheme. It is the responsibility of each sub-detector to perform the first level of merging of L1 throttle signals to a limited number of signals compatible with the partitioning scheme of their sub-detector. A standardized LHCb module will be available for making the local ORing [15]. It must be possible to mask the throttle signal from any front-end module to be capable of continuing data taking with parts of the system being malfunctioning.

With so many sources of the L1 throttles it becomes very important to have the means of monitoring the amount of throttling from each source. Each L1 throttle source must therefore measure the effective dead time introduced and make this information available to the ECS system.

The L1 throttle delay has been determined to be below 2μs including cable delays and the serialization of L1 trigger decisions for the TTC distribution system. This maximum delay is defined from a L1 throttle signal is asserted until it is ensured that no more L1 trigger accepts will be received by the L1 front-end electronics. The number of events that may still arrive to a module, that has asserted the L1 throttle, will depend strongly on its position in the front-end/DAQ hierarchy. In the L1 front-end it is assumed that a buffer full (or function busy) is back-propagated to the L1 derandomizer, which asserts the L1 throttle when needed. A DAQ unit asserting the L1 throttle may still have significant amounts of event data arriving before the data flow stops. Any module in the front-end or DAQ system that encounters buffer problems during such a situation must resolve the problem on its own. Event data can be discarded, if properly marked in the event, but event headers should never be lost, to ensure continuous correct event synchronization in the front-end/DAQ system.

A special case exists, when the front-end systems are used without any zero-suppression, during testing and commissioning. In this case large event fragments can be generated and the front-end and DAQ system may not be capable to cope with this under normal trigger conditions. The L0 and L1 trigger rates will in this case be regulated at the source (in the Readout Supervisor).

Requirements summary:

- L1 throttle must be asserted sufficiently early to prevent buffer overflows.
- Masking of all L1 throttle sources must be possible.

- Local monitoring of L1 throttle dead time.

- Modules with potential buffer overflow problems, that cannot be guaranteed to be resolved by asserting the L1 throttle signal, must handle this locally. Event data can be discarded if properly marked, but event headers must never be lost.

12. Data formatting

Data sent to the HLT and L1 trigger, via the common DAQ system, must be properly formatted to allow it to be correctly transported through the event-building network and finally be processed in the processor farm. The event data itself must be completely self-describing, as it will not be pre-processed during event building and its transport to the final processing in a CPU of the processor farm. Three levels of data framing and formatting of the data will be required to ensure its correct transport as indicated in Figure 10.

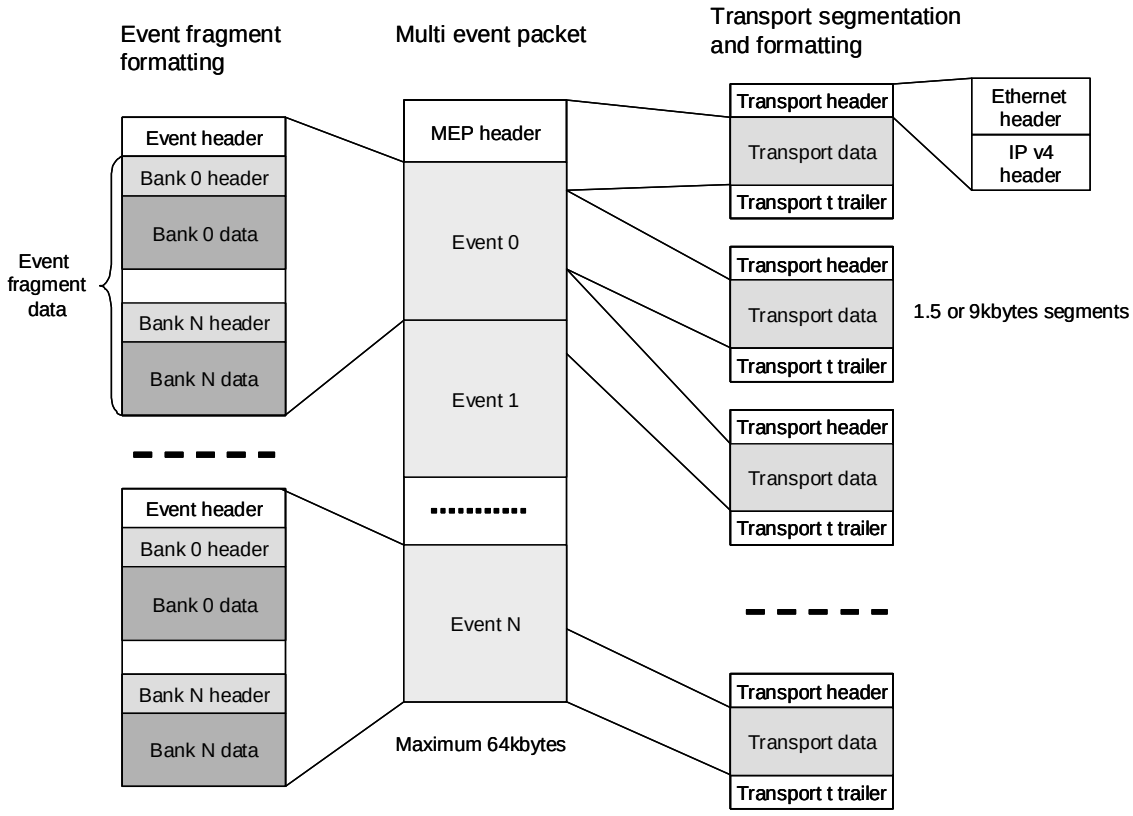

Figure 10. Data formatting and framing.

Data fragments from individual sources (L1 front-end modules) are sent to the Sub-farm controllers of the DAQ processing farm as shown in Figure 2. The event framing and formatting, containing vital event information, must be performed by the L1 front-end electronics modules. Multi-Event Packets (MEPS) have to be built from multiple individual event fragments, to reduce the network overhead and limit the IO interrupt rate in the sub-farm controllers. Finally transport framing and segmentation must be performed to adapt data to be transported using the Gigabit Ethernet link protocol.

The general outline of the data formatting and framing is sketched here in this document but for detailed information the reader is referred to a specific document on this [19]. It is brought to the attention of the reader that the byte ordering of header and data information is different because of different traditions of using big-endian notation in networking and little-endian notation in normal computing.

12.1. Event data formatting and framing

Event data itself must be formatted such that event processing routines in the CPU farm can access the required information in an efficient way. The event size must also be minimized to give a minimal load to the transport network. The data format should not make it difficult for the event processing routines in the farm to access data. Event data should be divided into bit fields that are byte aligned or 16/32 bit word aligned. Marginal event size optimizations should not be obtained at the cost of awkward data formatting as this may require data to be reformatted before processing. Event data size information must be included in the event to allow efficient memory management in the processing nodes. The identification of data sources (sub-detector, station, channel group, channel, etc.) must be fully contained in the event data using a hierarchical naming/numbering convention. Full event identification must also be contained in each event fragment (Bunch-ID, L0-ID, L1-ID, trigger type, etc.). As event data, after zero-suppression, has variable size, there must be relative address pointers that allow efficient access to data from specific detector parts. Event data will be transported as a black-box through the readout network. Sub-farm controllers will build complete events from the event fragments received from all data sources. The event data itself will only be looked at by the specific processing routines running on the CPUs of the farm.

Event headers of each event fragment contain a unique 16 bit event identifier (L0 event ID for L1 trigger, L1 event ID for HLT) and a length specifier that allow to find the individual event fragments in a Multi-Event Packet.

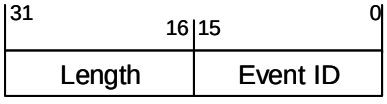

Figure 11. Event fragment header.

For the HLT processing event data is divided into data banks and a generic description of the event data formatting can be found in [4]. Error information from the front-end electronics must be contained in the data banks themselves and/or dedicated error banks, with detailed error information. The detailed event data formatting from each sub-detector must be defined in collaboration with the HLT trigger coordination.

The event data formatting for the L1 trigger is more optimized to obtain absolute minimum overhead which is critical at an event rate of up to 1.1 MHz.

12.2. Multi-Event Packet framing

Event fragments from sequential events must be merged into multi-event packets with a number of events defined by an ECS parameter. The event packing factor is defined to be in the range between 1 and maximum 32 for the L1 trigger and a maximum of 16 for the HLT. The maximum values of the event packaging factor have been determined from the maximum 64kbytes frame size defined by the IP protocol (HLT) and from limiting buffering requirements on the front-end module (L1 trigger).

Normally the bit field with the number of events per multi-event packet contains the programmed event packing factor. At the end of a run where a L1/HLT flush command is distributed this field must though contain the specific number of events available in the output buffer when the flush command was received.

A MEP header, as shown in Figure 12, contains key information about event ID (L0 or L1 event Id of first event in MEP) the number of events and the total data size. During the event building stage the sub-farm controllers need dedicated information to ensure that event fragments from different detector partitions are routed and built as separate event streams. The partition ID is also used to differentiate between L1 trigger multi-event packets and HLT multi-event packets so two independent ECS registers are required for the L1 partition ID and the HLT partition ID.

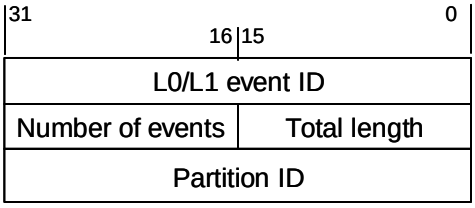

Figure 12. Multi-Event Package (MEP) header.

12.3. Transport framing and formatting

To transport event data over physical links, the data must be converted into a specific coding format (e.g. 8B/10B encoding) and transport framing and fragmentation must be applied. Gigabit Ethernet has been identified as the appropriate link technology, as it has become the de-facto standard in the computer industry. Gigabit Ethernet is the backbone link technology used in the DAQ system and extending the use of Gigabit Ethernet to the front-end interface ensures a smooth front-end interface. The Gigabit Ethernet encoding will be performed using an LHCb specific plug-in card with the PMC form factor [13]. A significant part of the required data transport framing must be done by the L1 front-end card itself [19].

A transport header, consisting of an Ethernet header and an IP header, is required to ensure conformity to the IP protocol for correct data transport over commercial Ethernet

switches and correct reception by commercial gigabit Ethernet interfaces in computers. For the use in LHCb several of the required fields in the header can be static values taken directly from ECS registers. The major dynamic values in the headers used in LHCb are the destination of the data packets, the number of gigabit Ethernet packets used to send the MEP, and the total data size.

It has been chosen to use a central destination assignment scheme for the multi-event packets controlled by the readout supervisor [22]. A centralized scheme ensures that there is a minimal risk of data sources losing event destination synchronization. The use of a central destination assignment scheme also allows dynamic load balancing between sub-farms if found necessary. The basis for the destination fields are a 10bit destination number distributed to the front-end modules with a long TTC broadcasts (see chapter 12.4). Only a limited address range will be needed in LHCb with the order of 100 sub-farms. A 10 bit destination address has been chosen to ensure sufficient addressing range for possible future system upgrades.

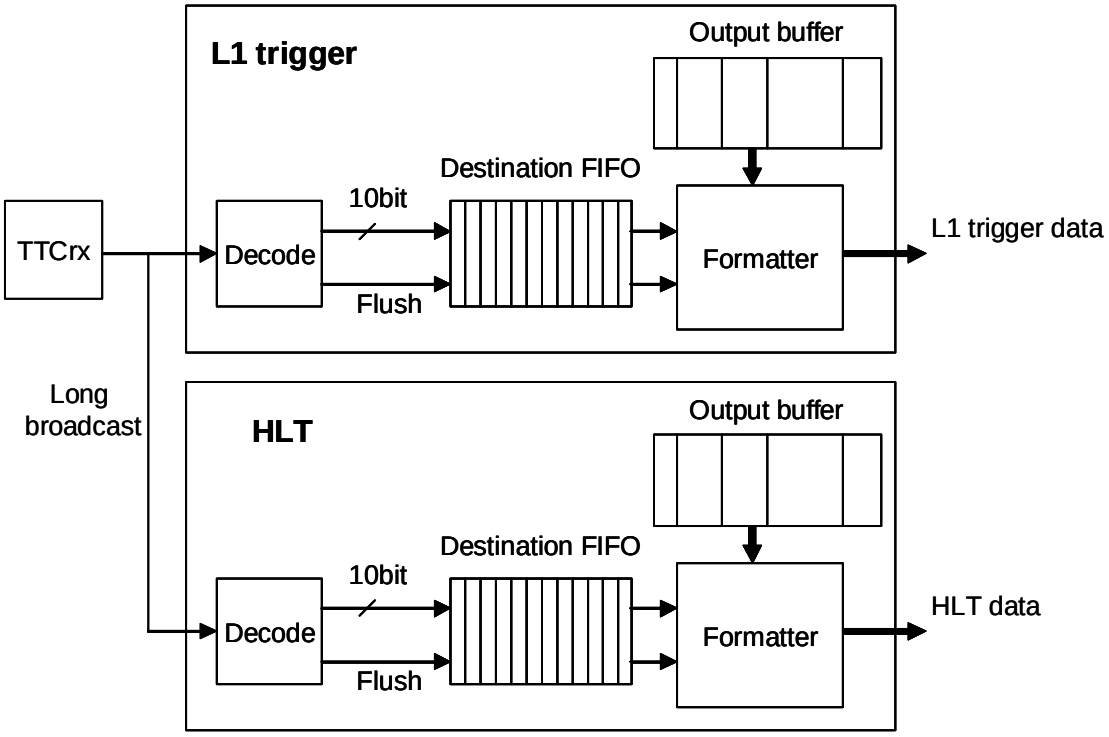

Figure 13. Reception and buffering of destination addresses from TFC system.

The central generation and distribution of destination addresses is made such that the front-end module is assured to have received the destination address for a MEP when all event fragments are available for readout (can in practice be slightly before or slight afterwards). The destination address is scheduled to be distributed by the Readout-Supervisor just after sending the trigger accept for the last event in the Multi-Event Package. Because of different delays in the processing of the events on the front-end module a destination address may be received before the MEP is fully available. For long local processing times and small packing factors, multiple destination addresses may need to be stored locally. It is therefore required that front-end modules have a small FIFO

buffer for minimum 16 destination addresses. For special cases with long local processing latency (Large L1 derandomizer and multiple levels of event buffering) and when using low packing factors this destination FIFO may need to be expanded.

The destination fields in the Ethernet header and the IP header are both based on the 10 bit destination address from the TTC broadcast. The 48 bit Ethernet destination field must be generated with the 38 MSB bits directly from ECS registers and the 10LSB bits being the distributed destination address. The 32bit IP destination field must be generated in a similar way with the 22 MSB bits directly from an ECS register (not same as for Ethernet address) and the 10 LSB bits being the destination address. This flexible way of generating the two destination fields ensures that any commercial switching equipment can be used.

The identification field in the IP header must use the MSB bit to distinguish between the L1 and HLT data flow and the remaining bits must carry the 15 least significant bits of the L0/L1 event identification. Remaining bit fields in the Ethernet and IP header shown in Figure 14 are static values that must be taken directly from a set of ECS registers (one set for L1 and one set for HLT).

The IP header also shown in Figure 14 contains a large set of bit fields that can be taken directly from a set of ECS registers (Version, IHL, Type of service, Flags, Time to live, Protocol, Source address). To allow different optimizations for L1 trigger and HLT data traffic independent ECS registers are required. To have a direct indication from which MEP packet the transport packet comes from a 16 bit identification field contains the 16 LSB bits of the MEP header L0/L1 ID. A fragment offset determines the location of the current transport data in the total Multi-Event Packet. Finally a 16 bit header checksum must be generated to detect transmission errors on the critical header.

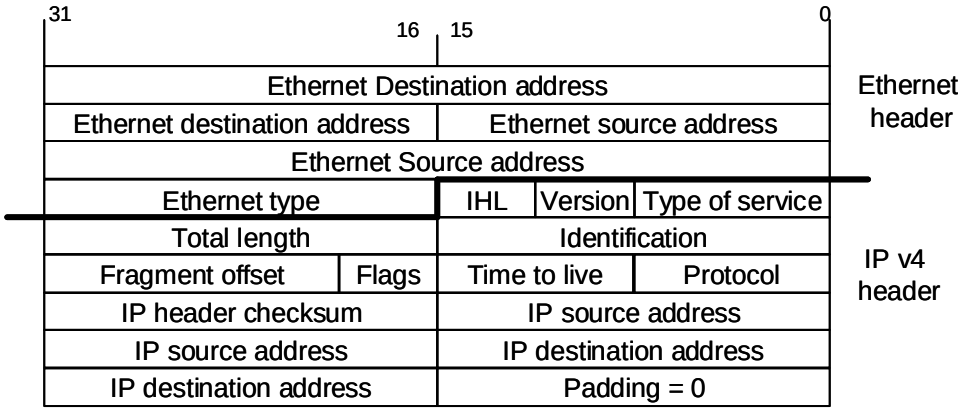

Figure 14. Combined Ethernet and IP header.

The official maximum Ethernet frame size is defined to be 1500 bytes. Most modern gigabit Ethernet equipment though supports frames up to 9000 bytes, called jumbo frames. This uncertainty in the maximum supported frame size requires the front-end electronics to be capable of performing segmentation into transport packets with

programmable size (defined by ECS register). A detailed algorithm for segmentation is described in [19].

The transport trailer, containing a CRC check sum and frame padding (for transport packets below 64 bytes), is generated by the Media Access Controller of the plug-in card. The interface to the Gigabit Ethernet plug-in card will follow a FIFO like protocol over a unidirectional 32 bit bus. When the input FIFO of the MAC is flagged as full (too high momentarily bandwidth or XOFF asserted by link receiver) it must be back-propagated to internal buffers that will finally assert the L0 or L1 throttle signal.

12.4. Destination assignment distribution and flush commands

The 10bit destination assignment address is generated by the readout supervisor and distributed to all front-end modules with a long TTC broadcast command [22]. Destination addresses for L1 trigger and HLT are distinguished by a different type field in the long broadcast command. This broadcast channel must be shared with short broadcasts handling resets and the distribution of L1 trigger decisions. This is handled by a priority scheme for different types of broadcasts:

1. Reset short broadcasts
2. **L1 trigger destinations**
3. L1 trigger accepts/rejects
4. **HLT destinations.**

The sharing of the broadcast channel will introduce some uncertainty when the destination addresses for a multi-event packet is available on the front-end card. Destination addresses will be sent on the first possible occasion after the trigger accept for the last event in a MEP has been sent. This ensures that the destination addresses will normally be quickly available in the front-end module when a complete MEP is ready in the output buffer. For exceptional running conditions with very low packing factors a MEP may have to wait for a limited time for the destination address to be available. For low packing factors the output buffer is though not extensively used as it is not needed to wait for many events to be assembled in the buffer.

Long broadcast	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Destination fields	Brd. type				Spare		Destination address									
L1 destination	1	0	0	0	Spare		L1 Destination address									
L1 flush	1	0	0	1	Spare		L1 Destination address									
HLT destination	1	0	1	0	Spare		HLT Destination address									
HLT flush	1	0	1	1	Spare		HLT Destination address									

Figure 15. Long broadcasts used for destination assignment distribution.

Under normal running conditions Multi-Event Packets all contain the same number of events as defined by ECS parameters. At the end of a run it may though occur that sufficient events are not available to fill up a complete MEP. To “clean” the system for such remaining event fragments a set of flush commands via TTC long broadcasts are available to force these remaining events to be shipped as incomplete MEPs. These flush commands contain the destination address of the non-complete MEP.

13. Data bandwidth to the HLT

Event data from several front-end modules must in some cases be concentrated before it can be given to the main event-building network of the DAQ system. The event-building network consists of a network of switching nodes interlinked by high speed Gigabit links. The absolute maximum bandwidth, which can be handled by the 1Gbits/s link is ~125Mbytes/s. To take into account statistical fluctuations in the event data, and protocol overhead in the readout network, each link should not be fed with more than two thirds of its maximum bandwidth on average (~80Mbytes/s). It is important not to concentrate event data too much in the L1 front-end electronics, to allow easy reconfigurations to handle unexpectedly high channel occupancies. By limiting the data rate per readout link to 30 - 40Mbytes/s, at nominal luminosity and nominal L1 trigger rate, it is always possible to handle higher channel occupancies by changing the configuration of concentration switches and the main readout network. This will ensure that the front-end system is made capable of handling data rates ~2 times higher than currently estimated, if needed. Such a reconfiguration will obviously need a larger event-building network and a larger HLT trigger farm, but the DAQ system is on purpose built to be scalable [14].

For modules having no interface to the L1 trigger it will also be possible to distribute the bandwidth between the two links available on the standard GE plug in card. Finally it can also be envisaged to use a four link interface card for future upgrades (see chapter 14).

Channel occupancies are not evenly distributed over the total surface of the sub-detectors. The particle density in LHCb in general follows a $1/r^2$ relationship. The difference in local channel occupancies must be taken into account when planning the structure of the concentration switches.

Requirements summary:

- Under nominal conditions the HLT data rate per output link should not exceed ~40 Mbytes/s.
- Concentration of event data from several sources in the concentration switches must take into account the variability of local channel occupancies.

14. Gigabit Ethernet plug in card interface.

A LHCb standardized gigabit Ethernet plug-in card of PMC size must be used as the interface to the HLT and L1 trigger links. This interface card uses an industry standard SPI3 unidirectional 32 bit bus with a FIFO like protocol. This bus can run at frequencies from 60 to 104 MHz and is shared to access two Gigabit Ethernet links via two independent Media Access Controllers (MAC). The connector definition of the plug-in card has an additional SPI3 bus available. This extra SPI3 bus can in some cases be used to get received data from the MACs (this is an exceptional utility, mainly for test purposes, and must be discussed in detail with electronics and DAQ coordinators if used). The second SPI3 bus is primarily available as an additional transmission bus for possible future system upgrades where up to four gigabit Ethernet links may be needed per L1 front-end module. If it is chosen to also connect the second bus to allow possible future upgrades it must be assured that this bus is in fact verified to work correctly to allow its future use.

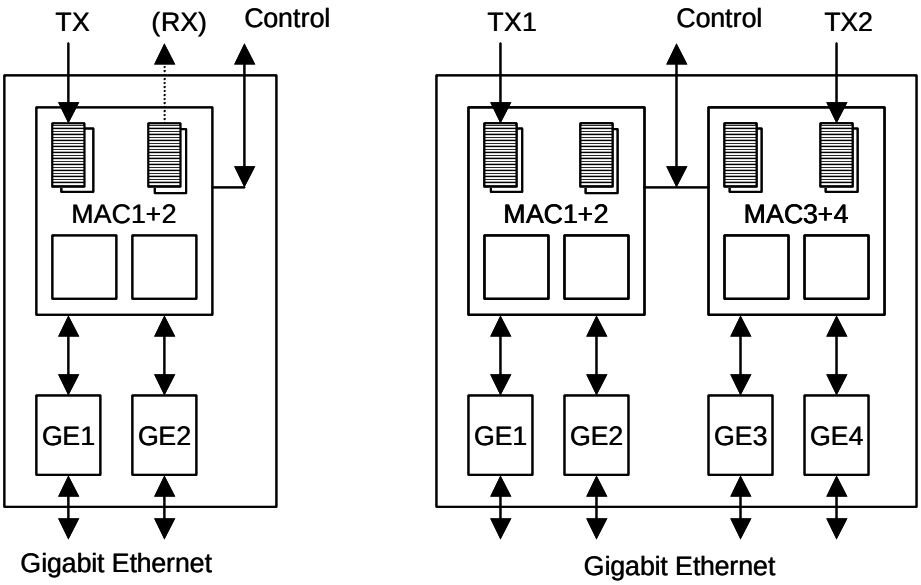

Figure 16. Two versions of gigabit Ethernet plug in cards for interfaces to L1 trigger and DAQ system.

It is foreseen to have two types of gigabit Ethernet plug-in cards available. A dual gigabit Ethernet card using one SPI3 bus to access the two links and the possible option of receiving data on the second SPI3 bus [13]. The users are made aware that the bit mapping and the control signal mapping on the second SPI3 bus is different in the transmit and in the receive configuration. A quadruple link card will be made available at a later stage for possible system upgrades. The quad link card will have two MAC's for each of the two SPI busses allowing data transmission on up to four gigabit Ethernet links. L1 front-end modules must be designed to initially use the dual link card. All

gigabit links can be tested from the DAQ system using the loop-back feature of the MAC's.

A simple microprocessor like control bus is used to configure and monitor the function of the MACs. The L1 front-end module must provide the means for the ECS system to access the MACs directly via this bus. The DAQ control software of the ECS system will take care of configuring and monitoring the functions of the plug-in cards.

The availability of the two different link cards allows quite some flexibility in the implementation of the L1 front-end modules and their use of available link bandwidth. For a L1 front-end module without an interface to the L1 trigger the situation is quite simple. A dual link card will in all cases be more than sufficient to cover the bandwidth needs. For low bandwidth applications one single of the two links can be used and thereby limit the number of links needed in the DAQ system. For higher rate needs it can be envisaged to use both links after consultancy with the DAQ coordinator on how to split the bandwidth (e.g. every second Multi-event package on each port or half of each on each port).

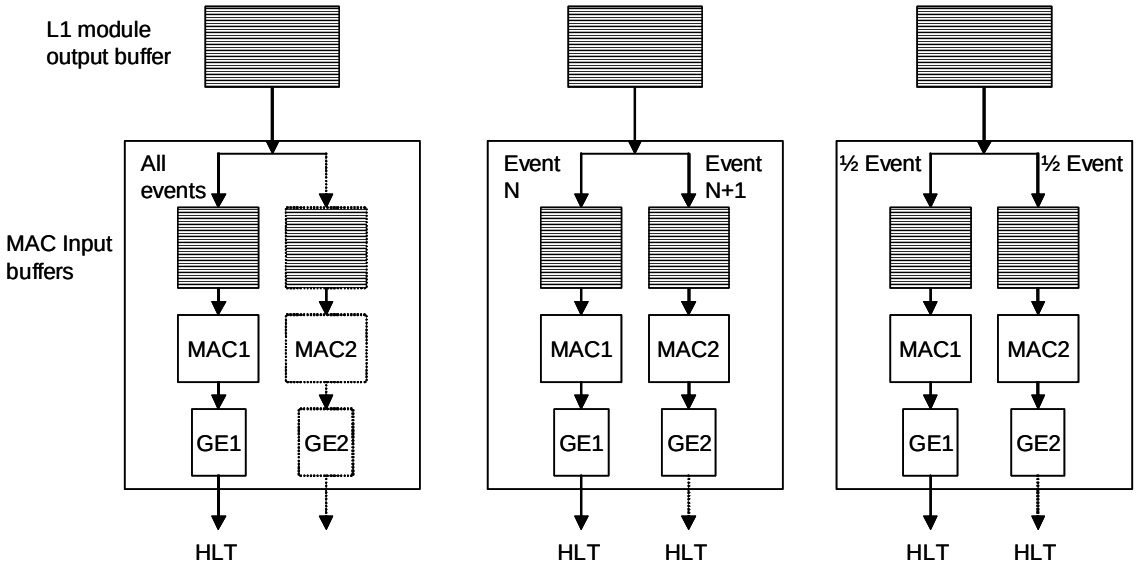

Figure 17. Configuration options for L1 modules with only HLT interface links using the dual link plug-in card.

Applications with a link interface to both the L1 trigger and the HLT system will have several alternatives available. The most obvious options are shown in Figure 18.The cheap and relative low bandwidth configuration is to use a single dual link card. One link is used for the HLT data and the other is used for the L1 trigger data. In this case the HLT data and the data to the L1 trigger must share one common SPI3 bus with a maximum bandwidth of 400Mbytes/s. A higher bandwidth option is to use a quad link card where one SPI3 bus is used for HLT and the other for L1 trigger data. It can even be considered to mix HLT and L1 trigger data on the same links in case of a combined DAQ system.

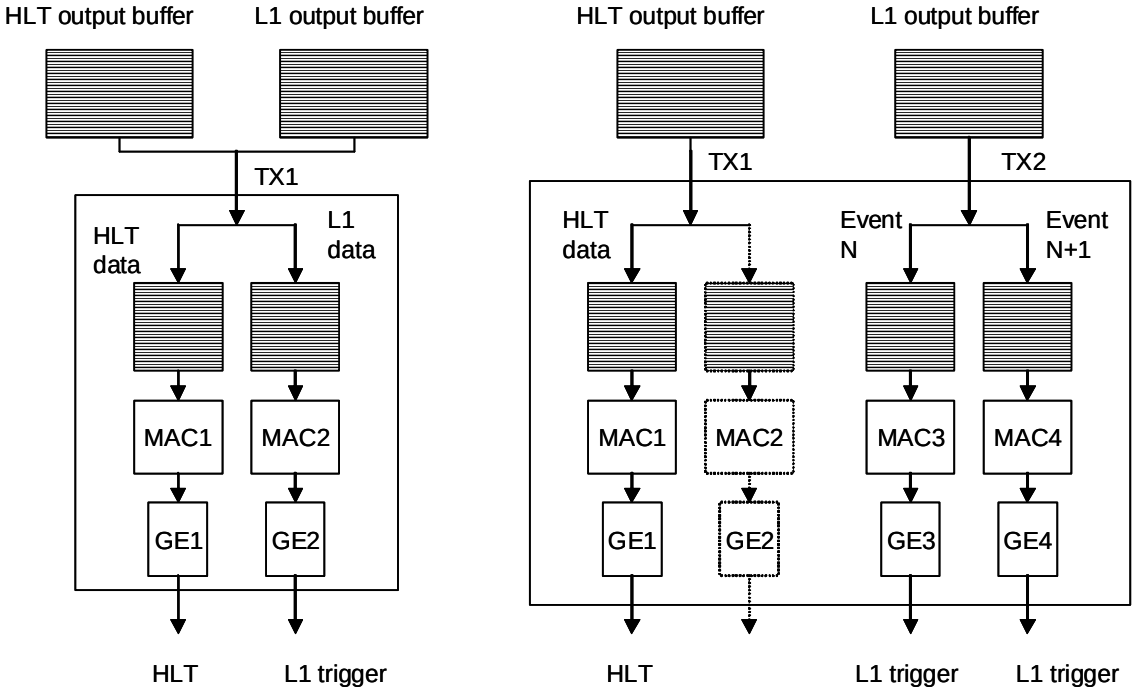

Figure 18. Configuration options for L1 modules with both HLT and L1 trigger interface links.

15. Error checking and monitoring

As for the L0 front-end electronics, it is important that extensive error monitoring is implemented in the L1 front-end electronics. Event data received from the L0 front-end electronics, located in a radiation environment, must be considered error prone and must be verified and monitored, based on the event data tags (e.g. B-ID, L0-ID).

The L1 front-end electronics must also monitor its own functionality for possible error conditions. This is especially important if it is located in the cavern and therefore subjected to errors caused by radiation effects (e.g. SEU). All internal buffers must be monitored for overflows. Important event information (e.g. event headers) should if possible be protected by parity error detection or Hamming error correction.

When error conditions have been detected, the event fragment(s) must be flagged as being error prone to inform following processing stages that data cannot be trusted. Detected error conditions must also set error flags in status registers and increment error counters, which can be used by the ECS system to monitor and trace error conditions. Fatal error conditions that will affect the processing of all sub-sequent events can set simple error flags. Error conditions that only affect the processing of a single event (or a few events) should increment error counters. For error counters it must be ensured that they saturate at their maximum value and do not overflow to zero.

The use of the L0 and L1 throttle signals are important to prevent buffer overflows. It is though also important to prevent “excessive” use of the throttles, generating a significant dead time for the whole experiment. It must therefore be possible to monitor the dead time introduced by each throttle source in the system.

15.1. Common TTCrx for L0 front-end and L1 front-end

In the baseline architecture of the front-end, it is assumed that the L0 and L1 front-end electronics are split into separate modules driven by separate TTCrx receivers. In this scenario, an error in the function of a single TTCrx (e.g. fake L0 trigger) will be detected by cross checking data tags (especially B-ID), in the events from the L0 front-end, with “reference” event identifiers from the TTCrx on the L1 module. In case of a common TTCrx receiver for both the L0 and L1 electronics, such errors can only be detected at a much later stage, as a fake L0 trigger will be seen by both the L0 and the L1 front-end. A fake L0 trigger (most likely failure of TTCrx in radiation environment) in a local L0-L1 front-end module will in this case not be detected in the local input synchronization and error checking stage before the L1 buffer. It will neither be detected with the L0-event ID check at the output of the L1 buffer, as this is just a simple sequence check. The result will be that the local L1 buffer will contain one additional fake event, but this cannot be detected locally. Checking of L1-ID’s in the event building will neither detect this kind of

error. The error can only be detected in the DAQ system, by making crosschecks between L0-ID, L1-ID and the B-ID event identifiers from individual event fragments. All in all, this means that front-end electronics based on a common TTCrx for the L0 and L1 front-end must carry all event identifiers (B-ID, L0-ID, L1-ID) to the DAQ system.

16. Testing and debugging.

Extensive testing and debugging features must be implemented in the L1 front-end electronics to allow in-situ testing and ensure a smooth commissioning of the large front-end and DAQ systems.

All control (not synchronous control which will come from the Readout Supervisor via the TTCrx) and monitoring of the L1 electronics will be made via the ECS interface so it is vital that the correct function of this interface can be verified. All setup and configuration registers in the front-end electronics must have read-back capability to verify that correct data has actually been down loaded (also for FPGA configuration data, DSP software, etc.).

The L1 front-end electronics is the critical interface between the sub-detector specific front-end electronics and the common DAQ system. It should therefore have sufficient test features that enable its connections to the L0 front-end electronics and its output links to be tested and verified. It should be possible to read event data that has been received from the L0 front-end electronics in a special debugging mode via the ECS interface. The generation of fixed (programmable) event fragments to the DAQ system for each trigger accept should be supported, without having received input data from the L0 front-ends. This will enable extensive tests of the DAQ interfaces to be made without having a complete front-end system with L0 front-end electronics. Similar functions should be available for the interfaces to the L1 trigger. Sending such test frames on request from the ECS system will also be useful.

As previously mentioned, it must be possible to disable zero-suppression and fix zero-suppression parameters to be capable of analyzing abnormal behavior of channel occupancies.

The L1 front-end electronics will contain large buffer memories. It must be possible to access these embedded memories from the ECS interface to perform extensive verifications of the memories. Having read (and write) access to the L1 buffer from ECS allows extensive error analysis to be made when needed. It will be very useful to be capable of freezing the L1 buffer (by sending no more L0 and L1 triggers), after some time of normal running, and then read the L1 buffer content directly via ECS.

For L1 front-end electronics modules with an intelligent ECS interface controller (e.g. credit card PC), it will have significant advantages if a complete self-test can be executed locally from a simple ECS request. Such features will also be extremely useful for production testing of modules.

For high density boards using surface mount devices it will in most cases be required to implement JTAG boundary scan testing to be capable of performing efficient production testing.

17. ECS interface

The Experiment Control System (ECS) interface is needed to configure and initialize the front-end electronics before data taking. It is also vital for efficient monitoring of the correct function of the front-end electronics modules during data taking. A high level of controllability and observability of the modules will be required to perform efficient testing and debugging of the electronics during commissioning and debugging. As the ECS interface is the only path to control and monitor the front-end electronics it must have high reliability and it must be ensured that it cannot get into a deadlock state where the control of the module is lost (e.g. SEU effects). The ECS interface must be capable of recovering the normal function of a module without the need of powering off and powering on the module.

To ensure a homogenous ECS system it has been agreed to limit the number of different ECS interfaces to a minimum. Three different ECS interfaces have been identified as viable solutions. The SPECS based ECS interface [9] is based on a serial protocol on twisted pairs to a radiation tolerant slave interface on the front-end modules. The SPECS slave interface will be made with extensive error checking features and will use triple redundant logic to be immune to radiation effects in the LHCb cavern. A special CAN slave interface called the ELMB [12] will also be accepted for used in the LHCb cavern. The use of these two ECS interfaces in the LHCb cavern is conditioned on their proper qualification for radiation tolerance. For applications in areas with no radiation (under ground counting room and surface buildings) a commercial Ethernet interface based on a Credit Card PC [11] has been found to be an attractive solution. The local intelligence of this Credit Card PC can be used for local monitoring and verification of the correct function of a front-end or DAQ module.

To ensure a correct identification and configuration of electronics modules, each ECS interface must have three identification registers. A board type identifier must define the module type (e.g. Adr: 0). A revision number must define the hardware revision number (e.g. Adr: 1). Configuration data for FPGA's can in this respect be considered as software versions, if it can be down loaded via the ECS interface. Finally a serial number must uniquely identify the module (e.g. Adr: 2). The content of these identification registers should in general be hardwired by jumpers or solder bridges on the module itself. These three identification registers should be found on the three lowest addresses in the local ECS address space, to ensure a consistent identification of modules across the whole system. In case a module has software or FPGA configuration data residing in permanent memories (e.g. Flash), it must be possible to read its revision number via the ECS interface.

An important part of the system monitoring performed by the ECS during running is to observe the error status of all front-end modules. It is therefore important that the front-end electronics makes extensive error status information available via its ECS interface. Error conditions that only have affected individual events should be counted with error

counters. Fatal errors can be signaled with simple error flags. All error counters and error flags must be reset by the L1 front-end reset (see later) but must also be resetable directly via the ECS interface itself.

Requirements summary:

- Only use of agreed ECS interfaces: SPECS, CAN ELMB or Credit Card PC
- ECS interfaces used in environments with radiation must be protected against SEU induced deadlock states.
- Read-back from all configuration registers.
- Hardware identification registers: Module type, Module version, Serial Number
- Possibility to read software revision.
- Extensive error status reporting to ECS system via error counters and error flags.

18. Reset of L1 front-end

The L1 front-end electronics must react to a set of reset signals in a well-defined fashion, to ensure that correct error recovery and synchronization can be obtained at startup or after a detected malfunction. Front-end resets are sent to the front-end electronics as short broadcast commands from the Readout Supervisor via the TTC distribution system with an encoding that can be found in [6].

The general reset of the front-end electronics have been split into two independent reset signals, to be capable of resetting the two parts individually. The L0 front-end reset can be asserted alone, under a well defined set of conditions [2], to limit the dead time related to this front-end reset. A reset of the L1 front-end will always be associated with a L0 front-end reset, as it does not make sense to reset the L1 front-end electronics alone.

18.1. L0 resets

The L0 reset signals (B-ID reset, L0 event ID reset, L0 front-end reset) are mainly related to the L0 front-end electronics [2], but the L1 front-ends must also react to these to be capable of receiving correctly event data from the L0 front-end and to be capable of doing consistency checks on the incoming data.

18.1.1. B-ID reset

The B-ID reset is sent to the L0 front-end electronics in each LHC machine cycle to maintain a correct synchronization to the bunch collisions. Bunch-ID information (or equivalent) is attached to event data accepted by the L0 trigger and this must be used in the input stage of the L1 electronics to verify the correct synchronization of the L0 front-end. The B-ID event tag must be compared to a locally (in L1 front-end) generated reference B-ID, buffered for each L0 trigger, or alternatively across multiple inputs with individual TTCrx receivers (see chapter 15.1). To have a local B-ID reference, a B-ID counter and a corresponding B-ID derandomizer buffer must react correctly to the B-ID reset and the L0 reset signals as defined in [2].

18.1.2. L0-ID reset

The L0 event ID data tag on event data is based on a simple event counter incremented for each L0 trigger. Each event accepted by a L0 trigger will have a unique pair of B-ID and L0-ID. For the L1 front-end to generate a correct L0 event ID reference, it must also

react correctly to the L0-ID reset as defined in [2] and have a local derandomizer. The first L0 trigger generated after a L0-ID reset (or L0 front-end reset) must have a L0-ID equal to zero.

18.1.3. L0 front-end reset

The L0 front-end reset is defined to reset the L0 front-end, including the L0 pipeline and the L0 derandomizer to recover from potential local errors in the L0 pipeline and L0 derandomizer control. When a L0 reset is issued alone, it occurs at a specified time in the LHC machine period and with a specific pre-conditioning [2]. The preconditioning ensures that the L0 derandomizers, in the L0 front-end electronics, are empty and no data is actively being transported to the L1 front-end electronics. A L1 flush command with a destination address for remaining event fragments to the L1 trigger will also have been generated before hand. The L1 front-end must, after a L0 reset, resynchronize itself to the data stream from the L0 front-end. After a L0 front-end reset it is ensured that no valid event data from the L0 front-end will arrive to the L1 electronics before 160 clock cycles (L0 latency). All data arriving from the L0 front-end should be ignored during this period. The L1 buffer and all following data buffers must not be affected by the L0 front-end reset.

18.2. L1 resets

The L1 reset signals must reset counters and buffers in the L1 front-end electronics itself and is not used by the L0 front-end electronics.

18.2.1. L1-ID reset

The L1-ID is used in the HLT system to verify that event fragments from different origins are in correct order and that all data sources have sent one and only one event fragment for all events accepted by L1. The availability of a correct L1-ID is an integral part of the event synchronization between the front-end systems and the HLT system. The L1-ID reset must reset the local L1-ID counters in the L1 front-end electronics and the event related to the subsequent L1 trigger accept must be sent to the HLT system with a L1-ID equal to zero. The L1-ID reset is related to the events at the input of the L1 derandomizer and the L1 event ID must therefore be attached to events at the time they are written into the L1 derandomizer.

18.2.2. L1 front-end reset

The L1 front-end reset must reset all buffers, counters, error flags and synchronization checks in the L1 front-end. A reset of the L0 front-end (L0 front-end reset) is always generated together with a L1 front-end reset, but without the standalone L0 reset pre-conditioning. Events in the process of being transported from the L0 front-end to the L1 front-end may therefore be truncated when a L1 reset is issued. All input data to the L1 front-end electronics, after a L1 reset (and L0 reset), must be ignored until 160 clock cycles have elapsed. The L1 buffer must be cleared and it is guaranteed that no L1 trigger decisions will be distributed to the front-ends before a given time interval (few ms). This will allow sufficient time to initialize the zero-suppression processing and output links to the DAQ system.

The L1 front-end reset will normally be pre-conditioned with no L0 and L1 trigger accepts in a given time period. A L1 flush command will have been sent to clear the interfaces to the L1 trigger system followed by a HLT flush command to clear interfaces to the HLT. The first event sent to the HLT system must have a L1-ID equal to zero, to allow the HLT system to resynchronize to a new data stream.

19. Initialization and running the front-end

A “cold” startup procedure of the front-end and DAQ after power up, after recalibration, or after a serious error condition that requires the front-end and DAQ system to be re-initialized, will require several minutes to be performed by the ECS system. Such a restart of a new run will be performed following a generic sequence of actions at the system level:

1. ECS stops L0 and L1 trigger accepts via Readout supervisor.
2. L1 flush command is sent to front-ends via Readout supervisor.
3. HLT flush command is sent to front-ends via Readout supervisor.
4. DAQ waits for event stream to stop (with some timeout).
5. ECS reads error status from all front-ends and DAQ.
6. ECS re-initializes TFC system if needed (Readout Supervisors, partitioning, etc.)
7. ECS re-initializes front-ends and trigger systems if needed.
8. ECS re-initializes DAQ if needed.
9. ECS resets all front-ends with a L1 front-end reset (and L0 front-end reset) via Readout Supervisor (TTC broadcast).
10. ECS restarts L1 trigger, DAQ and high level trigger systems
11. ECS enables L0 and L1 triggers via Readout Supervisor

In case of “simple” synchronization problems between the front-end systems and the DAQ system, the front-end and DAQ system may not need to be completely re-initialized by the ECS system. In such a scenario, a quick recovery from synchronization problems can be obtained with the following system actions:

1. ECS stops L0 and L1 trigger accepts via Readout supervisor.
2. L1 flush command is sent to front-ends via Readout supervisor.
3. HLT flush command is sent to front-ends via Readout supervisor.
4. ECS resets all front-ends with a L1 front-end reset (and L0 front-end reset) via Readout Supervisor (TTC broadcast).
5. ECS enables L0 and L1 triggers via Readout Supervisor

20. Features in Readout supervisor

The Readout Supervisor is the main front-end controller that ensures that the whole front-end system can run fully event synchronous without any buffer overflows. A short summary of Readout supervisor functions needed for the L0 front-end can be found in [2]. The functions needed to keep the L1 front-end running is described in the different chapters of this document and is shortly summarized below. A detailed functional description of the Readout Supervisor can be found in [7].

- Sending L1 trigger decisions to the front-end at a rate that can be handled by the L1 front-end modules.
- Including L1 trigger type and L0-ID in trigger decisions sent to front-end.
- Monitor L1 buffer occupancy and throttle L0 triggers in case of risk of overflows.
- Convert L0 trigger accepts into rejects in case of L0 throttle signal asserted.
- Convert L1 trigger accepts into rejects in case of L1 throttle signal asserted.
- Sending L1 trigger and HLT destination addresses to front-ends.
- Sending L1 flush and HLT flush command when requested by ECS.
- Enable/disable triggers from ECS.
- Generate and condition reset sequences.
- Generate programmable trigger and reset sequences for testing and debugging
- (Sending L0 event IDs of events accepted by L1 trigger, see appendix 1)

21. Qualification and radiation tolerance

The L1 front-end electronics must be qualified to work reliably over extended periods of time before it can be accepted for use in the experiment. The qualification must include link interfaces, which may be a major cause of problems in large scale systems. Modules must also have been shown to work correctly with direct interfaces to the final DAQ system.

For L1 front-end electronics located in the cavern the problems of radiation effects must be carefully taken into account. Radiation tests must be performed to demonstrate that the electronics can be expected to have a lifetime of more than 10 years, including sufficient safety factors (~10) on the expected radiation dose [20]. Error rates from single event effects must also be carefully evaluated and must be shown to be compatible with a fully working LHCb front-end system for extended periods of time (days). The ECS interface will be needed to recover normal functionality of a module after a SEU. The ECS interface itself must be proven to never enter into a hang-up state caused by a SEU. Single Event Latch-up (SEL) effects require special attention, as it in many cases will cause serious permanent hardware failures (unless using special Latch-up protection circuitry). It must be demonstrated that SEL is sufficiently unlikely that the whole LHCb experiment can work for several months without the need of hardware repairs. This will enforce very strict requirements to SEL at the component level. Power supplies located in the cavern will be a particular problem as standard commercial power supplies have been seen to be particular sensitive to radiation.

22. References

[1] LHCb electronics web: http://lhcb-elec.web.cern.ch/lhcb-elec

[2] LHCb 2001-14, Requirements to the L0 front-end electronics.

[3] LHCb 2001-126, Simulation of the LHCb L1 front-end.

[4] LHCb 2001-097, Online (raw) data format.

[5] LHCb 2001 – 16, Timing and fast control.

[6] LHCb 2001 – 17, TFC broadcast format.

[7] LHCb 2001 – 12, Readout Supervisor design specifications.

[8] CERN/LHCC 98-4, LHCb Technical proposal.

[9] LHCb 2001-018, TFC switch specifications.

[10] LHCb 2001-144, Serial Protocol for the Experiment Control System of LHCb,
LHCb 2003-005, Using the SPECS in LHCb.

[11] LHCb 2001-147. The use of Credit Card-sized PC's for interfacing boards to the LHCb ECS,
LHCb 2003-056. Glue light - a simple programmable interface between the Credit Card PC and board electronics.

[12] CAN ELMB,
http://atlas.web.cern.ch/Atlas/GROUPS/DAQTRIG/DCS/ELMB/elmb.html

[13] LHCb 2003-21. Gigabit Ethernet mezzanines

[14] LHCb TDR 7, LHCb Data Acquisition and experiment control.

[15] Throttle OR module.
No specific LHCb note available, partly covered in [5] and [9]

[16] LHCb 2001-116, Partitioning in LHCb.

[17] LHCb 2001-127. Requirements to L1 front-end electronics (now obsolete)

[18] CERN/LHCC 2003-031, LHCb TDR 10, LHCb Trigger System

[19] LHCb 2003-014. Raw data transport format.

[20] LHCB radiation hardness assurance web page: http://lhcb-elec.web.cern.ch/lhcb-elec/html/radiation_hardness.htm

[21] LHCB 2003-07. Common readout board for LHCb, specification.

[22] LHCb 2003-80. Implementing the L1 trigger path.

23. Appendix 1. Distribution of L1 trigger accepts with L0 event ID.

The distribution of only L1 trigger accepts together with an identification of the event location in the L1 buffer has become a feasible working principle after all L1 buffer implementations have direct access to individual events. Such a scheme reduces significantly the number of TTC broadcasts needed for the L1 trigger decisions and also allows L1 trigger accepts to be sent out of order. In such a scheme events rejected by the L1 trigger are explicitly "removed" when the write pointer in a circular buffer overwrites old events as shown in Figure 19.

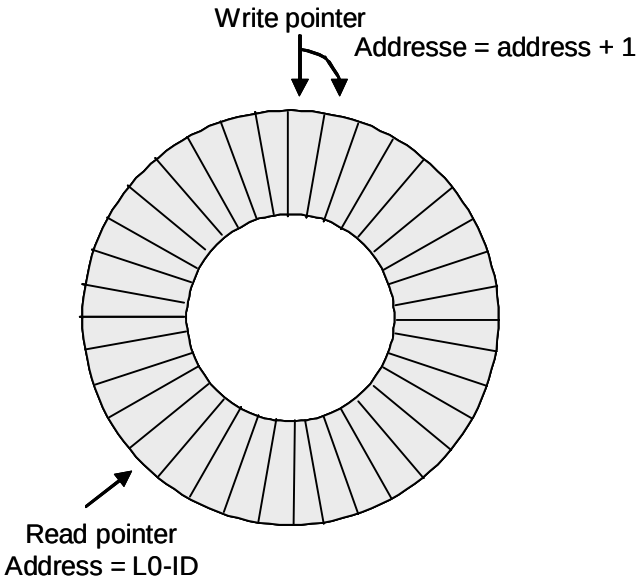

Figure 19. Principle of circular L1 buffer.

16 bit is in principle sufficient to address all events in a L1 buffer with ~58k events but to support possible future upgrades of the L1 buffer depth a 20 bit event identifier is used. This unfortunately requires using two long TTC broadcasts as shown in Figure 20 . Each L1 trigger accept therefore consists of two sequential long broadcast messages that the front-end must decode and assemble. The event identifier used is the sequential increasing L0-ID that can be used to address events directly in the L1 buffer. Direct addressing of events in the L1 buffer is most straight forward if individual events can be stored in blocks of 32 (possibly 64) words as indicated on Figure 4. Access to events in blocks of 36 words can though also be obtained quite easily with simple shift and add operations (L0-ID x 4 + L0-ID x 32).

Long broadcast	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L1 trigger fields	Brd. type															
L1 trigger (1)	0	1	0	R	L0 event ID [11:0]											
L1 trigger (2)	0	0	1	R	Type			R	L0 event ID [19:12]							

Figure 20. TTC long broadcasts for L1 trigger accept with L0-ID

It is the responsibility of the Readout supervisor to only distribute L1 trigger accepts for events which are still in the L1 buffer (not yet overwritten). The front-end should for each event extracted from the L1 buffer verify that the L0-ID matches the distributed L0-ID.

With this alternative scheme there is a strong link between the operation of the L1 buffer and the L0-ID, which is normally mainly related to the function of the L0 front-end. It is therefore not considered possible with this scheme to support a L0 front-end reset without a concurrent reset of the L1 front-end.

As in the original scheme it is guaranteed that there will be a minimum time spacing of 20us between the reception of complete L1 trigger accepts.

The use of this alternative distribution scheme should be relatively easy to support by front-end modules using directly addressable memories for the L1 buffer, controlled by FPGA's. It must just be ensured that the TTC broadcasts are correctly decoded and the event identification of the accepted event is communicated to the FPGA that takes care of controlling the dual port memories used to implement the L1 buffer.

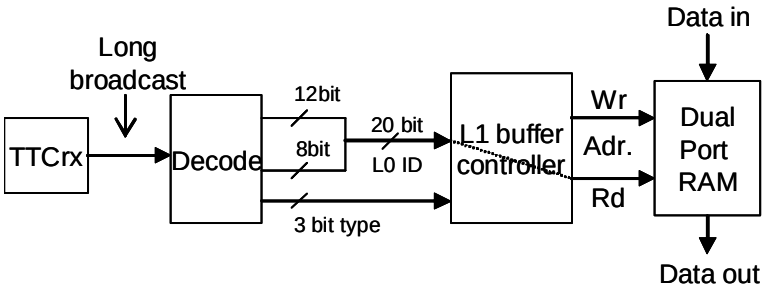

Figure 21. Required interface between TTCrx and L1 buffer controller.