ATLAS TDAQ DataCollection

Routing and Streaming in TDAQ

Document Version:	2.0
Document EDMS ID:	ATL-DQ-ES-0076
Document ID:	DataCollection Note 068
Document Date:	26 February 2007
Document Status:	Draft

Abstract
This note describes the routing of data streams in ATLAS TDAQ.

Keywords
Routing, Streaming, StreamTag, Partial Event Building.

Authors:

S.Klous, HP.Beck, S.Gadomski, C.Haberli, K.Kordas, Y.Nagasaka, A.Negri, James Schlereth, S.Sushkov, S.Wheeler, H.Zobernig

Send comments to: sander@nikhef.nl

Table 1: Document Change Record

Title: ATLAS TDAQ DataCollection Routing and Streaming in TDAQ			
ID: DataCollection Note 068			
Issue	Revision	Date	Comment
0	1	8. Sep. ’05	Y. Nagasaka: Birth - following discussion with HP. Beck, S. Gadomski, C. Häberli, and K. Kordas
0	3	9. Nov. ’05	HP. Beck: Further refinements.
0	4	30. Mar. ’06	S.Klous: Rewritten after streaming discussions.
0	5	13. June ’06	S.Klous: Modifications after TDAQ week.
0	6	15. Sep. ’06	S.Klous: Modifications to include latest insights.
0	7	5. Oct. ’06	S.Klous: Modifications after TDAQ week, added Event-Header
0	8	11. Oct. ’06	S.Klous: Included comments. Last revision before 1.0
0	9	18. Oct. ’06	S.Gadomsky: Included string based StreamTag modifications.
1	0	31. Oct. ’06	S.Klous: Version 1.0 ready to be reviewed.
2	0	26. Feb. ’07	S.Klous: Version 2.0 update after review. Variable Length Header, Use Cases, Requirements, Removed RoutingTag definition.

1 ToDo list

Section	Action	Name	Date
All	Review revision 2.	HP et al.	27 February 2007

2 Introduction and Scope

The event fragments of each subdetector are gathered by an SFI and built into events, following accept decisions by the LVL1 and LVL2 trigger systems. For fully build events, the standard data flow is described in [6] and the message format and the event format are described in [7] and [2] respectively. Here, the routing aspects are addressed for all event types in the higher level trigger. Furthermore, we discuss the partial event building and/or stripping features available for calibration events in the HLT, allowing bandwidth, processing and storage resource optimization for TDAQ.

This document is in principle intended as a conceptual design document that provides a procedural description of Routing and Streaming in the Higher Level Trigger and verifies the feasibility. However, in some cases implementation details are added to provide an accurate description of the detailed discussions with LVL2, EF and Algorithm experts. The Routing configuration schemes and the description of the error conditions are examples of such details. Based on the contents of this document, we updated the Message Format document [7], to bring the header and message definitions in line with the Routing and Streaming requirements.

2.1 Vocabulary

2.1.1 Related to trigger decision

LVL1 TriggerType (TT) = Part of the event header containing a 8 bits summary of the trigger information of the event.

TriggerDecision (L2TD and EFTD) = Decision from the algorithms to be stored in the TriggerInfo. Supported decisions for an event are Accept and Debug, a decision is taken independently for each algorithm category. Two algorithm categories are foreseen: Calibration and Physics.

TriggerInfo (L2TI and EFTI) = Variable length array of words in the EventHeader containing the TriggerDecision.

- Separate TriggerInfo exists for the LVL2 trigger system and the EF.
- The first word of the TriggerInfo defines the size of the TriggerInfo in words (including this one).
- The second word of the TriggerInfo contains the TriggerDecision.
- ...

DataFlow StatusWords (DF-SW) = Part of the StatusWords in the EventHeader. 16 bits ID number and 16 bits available for infrastructural (PSC/DF) decisions, misbehaviors and/or errors.

- StatusBit generic field value: 32.
- 8 bits available for LVL2 trigger system and 8 bits available for the EF.
- Examples: L2PU/PT crash/timeout, DataFlow ForcedAccept, etc.
- Complementary to the AlgorithmBits and filled **outside** the algorithms.

TTC[5] = The Trigger, Timing and Control system distributes the LVL1 Accept together with the LHC clock, other trigger information and control commands to the detector frontend electronics.

TrigConfDB[3] = Relational Trigger configuration DataBase. This DataBase resides in the HLT software release.

DAQConfDB[4] = OKS based object oriented configuration DataBase for the configuration of DataFlow componentes. This DataBase resides in the TDAQ software release.

2.1.2 Related to routing

Routing = (process of) sending an event from one DAQ/HLT "unit" to the next on its DataFlowPath.

- Examples: from SFI to EFD, from EFD to PT, etc.
- Possible routes in the EFD are: physics (to physics PT), calibration (to calibration PT) and EFOut (to output task).

RoutingDecision (RD) = a decision (internal for *e.g.* EFD) about the next processing and/or DataFlow step based on the contents of the TriggerDecision and the DataFlow Status Bits.

- The mapping from TriggerDecision and DataFlow Status Bits to a RoutingDecision is configurable in the DAQConfDB.

DataFlowPath (DFP) = particular global path which an event may pass, or already passed, through the DataFlow System.

- Note that all RoutingDecisions in the DataFlowPath are known explicitly.

2.1.3 Related to streaming

StreamingDecision (SD) = A decision made somewhere on the DataFlowPath about the particular **SFO stream** (file) destination of the event. These SFO streams will have standard names based on information from the TrigConfDB.

StreamTag (ST) = an object that contains the StreamingDecision. The SFO will route an event into the appropriate output streams based on the contents of the StreamTag.

- A StreamTag is constructed by the SFI with a default value based on the information in the TriggerInfo and the DataFlow bits.
- The StreamTag is part of the event header.
- The contents of a StreamTag may be overwritten by the EventFilter. This will be the usual case for the processing of physics events.

Streaming = This term only exists in a global sense. It provides a view on a DataFlowPath that finishes in a particular SFO stream (file). Streaming is a two step process.

- Step 1 is the actual StreamingDecision: a "unit" with access to the TrigConfDB (*i.e.* the ProcessingTask of the EventFilter) maps the event to one, or a number of, logical stream(s) and stores this mapping in the StreamTag.
- Step 2 is the routing of events done by the SFO: The SFO reads the StreamTag from the event header and finds in it the names of streams to which the event belongs. An event can belong in multiple streams, *e.g.* Physics Muon, Express and Calibration InnerDetector. The names of streams as well as their types are encoded in the StreamTag.

3 Use Cases, Requirements and Implementation

The functionality proposed in this document is based on requirements extracted from Use Cases as identified in discussions with LVL2 and EF experts. In this section we summarize the collected Use Cases based on the route an event follows through the system. The Routing and Streaming schemes have to be flexible enough to accommodate a wide variety of event types. The requirements section clearly shows the criteria for these schemes and includes external criteria imposed by the Streaming Working Group on streaming configurations. In the implementation section we discuss configurations for all considered Use Cases, these configurations are studied in detail in the rest of the document.

3.1 Use Cases

Events marked for calibration at LVL1

These events are flagged as calibration events in the LVL1 TriggerType. The L2SV has to bypass the L2PU and the EFD has to bypass the PT. The SFO writes these events into a default stream.

Events accepted by LVL1

These events are flagged as physics events in the LVL1 TriggerType. The L2SV assigns these event to an L2PU and in the EventFilter they are assigned to a PT. The SFO writes the accepted events to a specific streams. The stream selection in the SFO depends on the decision of the EventFilter algorithms.

Calibration events from LVL2 (partial event building)

These events are flagged as physics events in the LVL1 TriggerType. The L2SV assigns these events to an L2PU that runs its algorithms. The L2PU algorithms decide that the events qualify for calibration (independently, these events can be accepted or rejected for physics). The SFI will build the events fully or partially based on the available partial event building information. In the EventFilter these events are processed by a specialized PT dedicated to calibration events. Next, events pass through the OutputTask and are sent (fully or stripped) to the SFO and written in the appropriate stream. The stream selection in the SFO depends on the decision of the EventFilter calibration algorithms. Note that stripping of events in the EFD is performed on the fly, when sending the event to the SFO. The full event remains in the EFD until the SFO confirmed reception of the event. This guarantees recoverability of events.

Physics events from LVL2

These events are flagged as physics events in the LVL1 TriggerType. The L2SV will assign these events to an L2PU that runs its algorithms. The L2PU algorithms decide that the events are accepted for physics. The same event may also classify for calibration, buth they will first be processed by the physics PT in the event filter. If an event is interesting for calibration as well, it has to be processed subsequently by the calibration PT. Note that an event can be accepted for physics and for calibration. In that case, the OutputTask has to send the event twice, because events accepted for calibration might need to be stripped.

Debug events from LVL2

These events are flagged as physics events in the LVL1 TriggerType. The L2SV assigns these events to an L2PU that runs its algorithms. The L2PU malfunctions for whatever reason and the the L2SV registers this error. The EFD bypasses the PT and the SFO writes these events into a default stream.

Streaming of events

Two offline streaming models are under discussion, inclusive and exclusive streaming. The difference between these models applies to events that fulfill the selection criteria of more than one stream. In the inclusive model these events will be written to each of these streams, this implies that multiple copies of such events are written to disk. In the exclusive model an overlap stream for such events exists.

Note that events selected for calibration and express streams are always stored in inclusive mode, *i.e.* in case such events are selected for physics streams as well, (sometimes partial) copies of these events are stored in the appropriate calibration and express streams. The discussion on inclusive vs. exclusive streaming concerns only the physics streams. We expect a choice of the streaming model to be made in the first halve of 2007.

In principle identification of the appropriate stream can happen either in the LVL2 trigger system or in the EF. Currently it is foreseen that the LVL2 trigger system identifies a default stream based on the LVL2 DataFlow Status and the LVL2 TriggerDecision. The EF ignores this information and makes an independent identification of the appropriate stream based on the EF DataFlow status and the EF TriggerDecision. Hence, the event will only end up in a default stream as identified by the LVL2 trigger system when it is not processed by the EF processing taks (*e.g.* the very first Use Case).

Luminosity blocks

A data file either contains events from a single luminosity block, or it can potentially contain events from multiple luminosity blocks. In case files have to respect luminosity block bounderies, they will on average be smaller. Hence, it is not expected that this option is chosen for all streams. In particular calibration streams will most likely contain events from multiple luminosity blocks.

Debugging configurations

The above Use Cases indicate that there are a number of decisions to make on the DataFlow path to route and/or stream events in the right direction. These decisions are either made in the DataFlow system or in the processing tasks (L2PU and PT). In the latter case, the decision has to be propagated to the DataFlow system and combined with other information (*e.g.* about errors in DataFlow components). We have to debug all stages of the decision making process (Algorithm level as well as DataFlow level).

3.2 Requirements

1. The LVL2 trigger system has to make a decision about assigning an event to an L2PU or to bypass the L2PU. This decision is based on the LVL1 TriggerType and has to be configurable.

2. The L2PU algorithms have to pass the TriggerDecision to the DataFlow system.

3. The L2PU has to pass Partial Event Building information to the DataFlow system.

4. The Partial Event Building information has to be stored in the event for future reference.

5. The LVL2 TriggerDecision has to be defined clearly and the definition must be stable.

6. The LVL2 DataFlow system has to register status information (*e.g.* error information or bypass of the L2PU).

7. Events that arrive in the EF have to be routed to either the Physics PT, the Calibration PT, or directly to the OutputTask.

8. The routing decision is based on information in the LVL2 TriggerDecision combined with information from the DataFlow system and has to be configurable.
9. The PT algorithms have to pass the TriggerDecision to the DataFlow system.
10. The EF TriggerDecision has to be defined clearly and the definition must be stable.
11. The EF DataFlow system has to register status information (*e.g.* error information or bypass of the PTs).
12. Events processed by a Physics PT or a Calibration PT have to be routed to either a Calibration PT, or to the OutputTask.
13. The routing decision is based on information in the EF TriggerDecision combined with information from the DataFlow system and has to be configurable.
14. The SFI has to be able to build an event based on the provided Partial Event Building information by the LVL2 trigger system or by the SubDetector type field in the LVL1 TriggerType word.
15. The SFI has to supply information about the default stream in an easily accessible location for the SFO.
16. The PT has to supply information about the appropriate streams in an easily accessible location for the SFO.
17. The OutputTask must be able to strip an event "on the fly" when it is sent to the SFO.
18. The SFO has to read and interpret the stream information provided by the PT or the SFI and write the event into the appropriate stream.
19. Each stream must be configured independently to either respect or ignore luminosity block boundaries.
20. We need different levels to configure the system in order to debug errors in routing and streaming information.. These levels match with the Algorithm and DataFlow software layers.

3.3 Implementation

- The L2SV interprets the LVL1 TriggerType and routes events based on a configurable mapping in the DAQConfDB. Hence, the LVL1 TriggerType must have a fairly stable definition.
- The L2PU SteeringController provides methods to access the TriggerDecision. This TriggerDecision is written in the EventHeader by the SFI as part of the L2 TriggerInfo.
- The L2SV provides status information about the DataFlow system. This information is written in the EventHeader by the SFI as part of the StatusWords.
- A RoutingTask in the EF interprets the LVL2 TriggerDecision and LVL2 DataFlow StatusWords and routes events based on a configurable mapping in the DAQConfDB. Hence, the LVL2 TriggerDecision must have a fairly stable definition.
- The PT SteeringController provides methods to access the TriggerDecision. This TriggerDecision is written in the EventHeader by the SFI as part of the EF TriggerInfo.

- The ExtPT task provides status information about the DataFlow system. This information is written in the EventHeader as part of the StatusWords.
- The ExtPT task interprets the EF TriggerDecision and EF DataFlow StatusWords and routes events based on a configurable mapping in the DAQConfDB. Hence, the EF TriggerDecision must have a fairly stable definition.
- The EF TriggerDecision and EF DataFlow StatusWords are reevaluated each time a PT processes the event. This can be multiple times, since an event can subsequently be processed for physics and for calibration purposes.
- The SFI writes the appropriate default stream for an event in the EventHeader. This variable length field is called the StreamTag. The information in the StreamTag is self contained, *i.e.* all information needed to write the stream, such as their name, type (physics, calibration or debug), inclusive or exclusive and the behaviour with respect to luminosity blocks, is defined is in the StreamTag.
- The PT writes the self contained information on the selected streams in the StreamTag. As streams are defined in terms of the Trigger menu, the TrigConfDB will describe the configuration of data streaming. The database has to contain a mapping of trigger chains to data streams as well as the stream configuration details.
- The EFD works with virtual events. These events contain a reference to the original event, with a list of modifications. Only when the event is shipped to the SFO, the appropriate parts are collected and combined. After the SFO acknowledged reception, the original event is removed.
- The SFO reads the StreamTag from the EventHeader and obtains from it all the information it needs for what concerns streaming of data. Based on this information, the event is written in the appropriate stream(s).
- The Algorithm and DataFlow software are configured independently for routing, providing the required flexibility for debugging. The TriggerDecisions are configured in the TrigConfDB and mapped to routes in the DAQConfDB.
- Streaming is performed after the DataFlow system so there is no associated configuration in the DAQConfDB. Streaming is configured entirely in the TrigConfDB (with exception of the default streams, see section 4).

4 Routing in the LVL2 trigger system

The LVL1 trigger system can either flag an event for physics, for calibration, or reject the event. The L2SV extracts this information from the 8 bit LVL1 TriggerType in the event header (see [8]) and decides to either process the event in an L2PU or to bypass the L2PUs.

- The LVL2 DataFlowPath in case of a bypass of the L2PUs is shown in Fig. 1. In case of a LVL1 calibration trigger flag, the event is flagged for partial event building. The event bypasses the LVL2 trigger system and is sent via the DFM to the SFI, where it will be built partially according the SubDetector type field (this field has a granularity of individual TTC zones) found in the LVL1 TriggerType word. Note that the SubDetector type field is used as a key to define which detector elements need to be read out, which, depending on the configuration of the actual data taking run, could mean any set of detector fragments, including reading out the full detector.

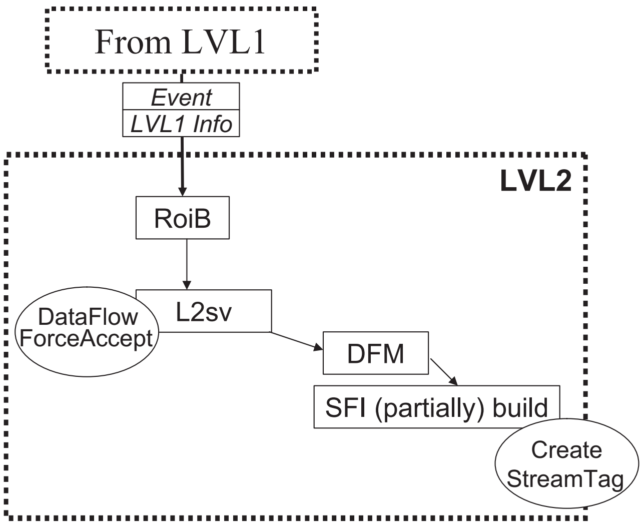

Figure 1: Bypass LVL2 trigger system.

- The "normal" LVL2 DataFlowPath is shown in Fig. 2. In case of a LVL1 physics trigger flag, the L2SV assignes the event to an L2PU where it is analyzed by the LVL2 trigger algorithms. Subsequent routing of the event is determined by the result of the L2PU analysis. The TriggerDecision provides the following options:
  - The event is rejected and deleted.
  - The event is accepted. The L2PU creates the LVL2 AlgorithmResultObject and adds the TriggerDecision (Accept or Debug) to the LVL2 TriggerInfo (see section 4.1) for physics and/or calibration. In case of an error in a DataFlow component, a description is added to the DataFlow StatusWords. Furthermore, the event can be flagged for partial event building (L2PU decision). A variable length array is available to store the associated TTC information. It is It is up to the LVL2 trigger system to fill the TTC information correspondingly (see section 4.1.3).

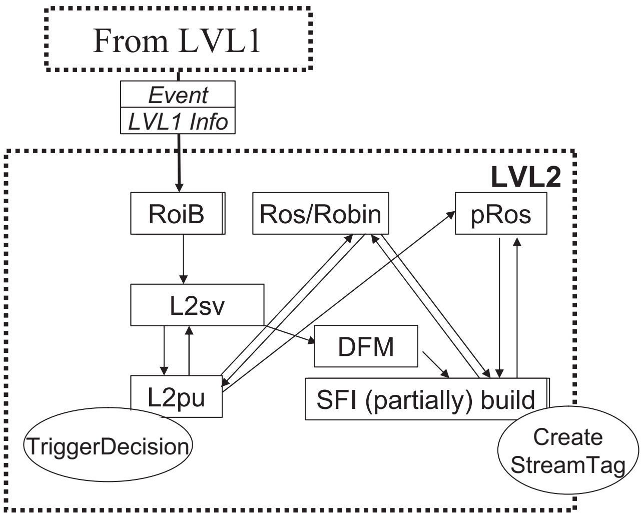

Figure 2: Routing through LVL2 trigger system

A StatusWord in the EventHeader is reserved for DataFlow information. These bits are mainly intended for the identification of error conditions, but include the possibility to

ForceAccept an event in the L2SV. The LVL2 TriggerInfo as well as the Partial Event Building information are transferred to the SFI, where they are added to the event header together with a default StreamTag based on the TriggerDecision and the DataFlow bits.

The RoutingTask in the EventFilter will read the LVL2 TriggerDecision and DataFlow StatusWords in the event header and handle the event accordingly. On of the possibilities is that the event bypasses the EF (EF Out). In that case it is the responsibility of the SFI to fill the StreamTag with the correct default stream information. Note that these values only have meaning in case of an EF bypass. In (normal) cases where the EF is processing the event it will ignore the values in the StreamTag and replace them (see section 5.2).

4.1 LVL2 information in the EventHeader

The LVL2 information in the EventHeader consists of 8 bits in the DataFlow StatusWord (DF-SW) for error conditions plus the LVL2 TriggerInfo (L2TI) and the Partial Event Building (PEB) information in the EventHeader specific words. The structures are shown in Table 2. The first word of the LVL2 TriggerInfo specifies its length in words (including this one). The TriggerDecision is written in the second word. This definition is consistent with [7].

Table 2: Structure of the LVL2 information in the EventHeader

Word	Usage	Remarks
DF-SW	ForceAccept	DataFlow decision: True/False
	Problem filling TriggerDecision	
	L2PU Timeout/Crash	Note: no TriggerDecision available
	Duplication warning	Due to DFM communication errors
	Duplication warning	Due to SFI/EFD EFIO communication errors
	3 bits reserved	
L2TI	Number of Words	Including this one
	TriggerDecision	Physics: Accept/Debug, Calibration: Accept/Debug
	...	Variable length
PEB	Number of Words	Including this one
	SubDetector IDs	Variable length

4.1.1 LVL2 DataFlow information and errors (DataFlow StatusWord)

L2SV ForcedAccept The L2SV ForcedAccept bit indicates that the event has not been processed by the L2PU or caused an L2PU crash, *i.e.* this is an event that may not contain a LVL2 TriggerDecision. In a normal configuration these events will bypass the EF as well, which means these events are routed directly to the OutputTask (EF Out). In that case, the LVL2 trigger system has to fill the StreamTag with the appropriate default SFO stream (see section 3).

Problem filling TriggerDecision This bit indicates that the L2PU was not able to fill the TriggerDecision due to *e.g.* inconsistent algorithm information or a DataFlow error. Such events need to end up in a Debug stream.

L2PU Timeout/Crash This bit allows to distinguish an L2PU crash from a ForcedAccept initiated by the L2SV. In both cases the ForcedAccept flag will be set as well.

Duplication warning This bit is set in case of DFM/SFI communication errors and/or failures that might result in accidental duplication of an event. The DataFlow ignores this warning, however in the offline analysis such events need to be treated with care.

Duplication warning This bit is set in case of EFIO communication errors and/or failures between SFI and EFD that might result in accidental duplication of an event. The DataFlow ignores this warning, however in the offline analysis such events need to be treated with care.

The rest of the bits are reserved for future extensions.

4.1.2 LVL2 TriggerInfo

The TriggerAlgorithms provide L2PU accessible methods to deliver both the TriggerDecision for the LVL2 TriggerInfo. The DAQConfDB contains a mapping of the bits in the LVL2 TriggerDecision and in the DataFlow StatusWord to routes in the EventFilter. Four routes are foreseen at the moment, an overview of a possible mapping is shown in Table 3. The mapping configuration can be changed to *e.g.* route all events directly to the OutputTask, which allows for debugging of the TriggerDecisions made by the LVL2 algorithms. Note that routing of events that arrive in the EventFilter is done with a RoutingTask, based on configuration in the DAQConfDB.

Table 3: EF Routing Scheme based on LVL2 information.

TriggerDecision	**DF-SW**	**Route**	**Comments**
Physics Debug || Calibration Debug ||	Any	EF Out	The SFI fills the StreamTag
Physics Accept &&	None	Physics	
Physics Reject && Calibration Accept &&	None	Calibration	
None	None	Delete	

4.1.3 Partial Event Building

On top of the physics triggers, the LVL1 trigger system provides calibration triggers for testing and calibration purposes for various detectors. Furthermore, the LVL2 trigger system, *i.e.* the L2PU, may identify events to be routed into separate calibration and/or debug streams. Events that will end up in dedicated calibration and/or debug streams could need so-called partial event building. Partial Event Building information is stored in the header as a variable length array. The first word of this array contains the length of the array in words (including this one). A length of 0 means that this event does not require partial event building. The rest of the words contain the Partial Event Building information itself. Every one of these words contains up to four bytes corresponding to four SubDetectors. A padding word is introduced to allow for 32-bit alignment of the final vector of words.

5 Routing in the EF

The InputTask of the EFD will receive events from the SFI. These events are stored in a read-only area of the SharedHeap (see [1]). A RoutingTask will perform the routing of events that arrive in the EF. The available routes in the EFD are specified in a configuration object in the DAQConfDB (see section 3), where the the LVL2 TriggerDecision and DataFlow StatusWord are mapped to the available routes in the EF. The mapping configuration can be changed to *e.g.* route all events directly to the OutputTask, which allows for debugging of the TriggerDecisions made by the LVL2 algorithms., see section 3.

5.1 EF bypass

The RoutingTask in the EF can decide, based on the LVL2 TriggerDecision and the DataFlow StatusWords, that an event will bypass the EF (route: EF Out), as explained in section 4.1.2. The appropriate DataFlowPath is shown in Fig. 3. The RoutingTask directly appoints the event to an OutputTask, without further processing by the EF. Next, the OutputTask sends the event to an SFO, which writes the event into one of the output streams.

The StreamTag should normally be set by the EF Processing Task. In case of the EF bypass the events do not go to the Processing Task and the StreamTag keeps it's initial state, as created by the SFI. The StreamTag will be implemented as a list of strings and the initial state is set by the SFI. Events with a default StreamTag will be sent to a special stream. The names of these special streams should be configured in the DAQConfDB.

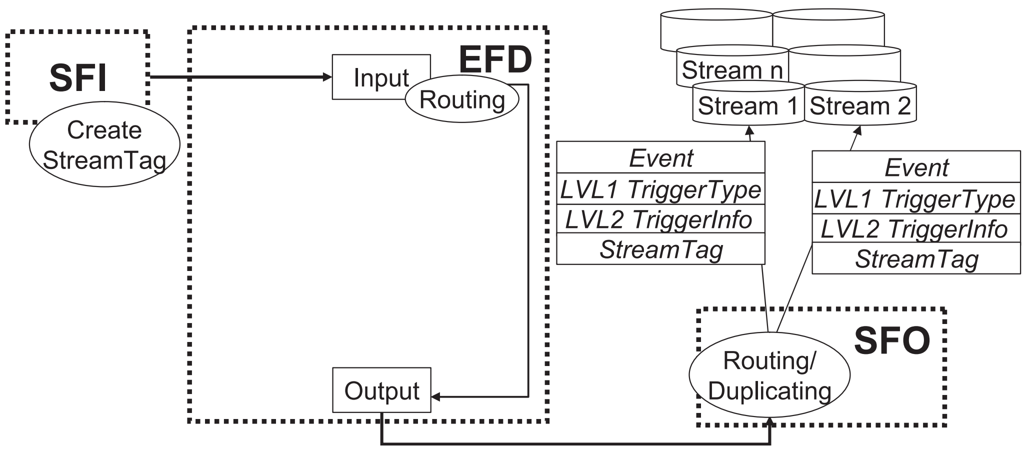

Figure 3: Event Filter, routing in case of an EF bypass

5.2 EF physics processing

When the RoutingTask decides that the event has to be processed for physics, it routes the event to an ExtPT task, which in turn assigns the event to a PT (see Fig. 4). Note that routing in the EF is based on virtual events. These virtual events contain a reference to the original event. Any modification to an event is listed in the virtual event, but the original remains untouched. Only when an event is sent to the SFO these modifications are applied on the fly. As long as the event is in the EventFilter, the original event will be available (*e.g.* for error recovery).

The PESA algorithms in the PT will create the AlgorithmResultObject in the virtual event and calculate the TriggerDecision. Two distinct decisions in the DataFlowPath are based on the outcome of these algorithms: subsequent routing in the EF and identification of the appropriate SFO stream(s). The EF overwrites the default StreamTag as created by the SFI when it creates a new one in the PT. The procedures for both decisions are described below.

Routing The TriggerAlgorithms provide PT accessible methods to deliver the TriggerDecision for the EF TriggerInfo. The DAQConfDB contains a mapping of the bits in the EF TriggerDecision and in the DataFlow StatusWord to consequetive routes in the EventFilter. After the physics PT three routes are foreseen at the moment, an overview of a possible mapping is shown in Table 4. After the calibration PT only two routes are available (EF Out and Delete). The mapping configuration can be changed to *e.g.* route all events to the OutputTask, which allows for debugging of the TriggerDecisions made by the EF Algorithms.

Table 4: EF Routing Scheme based on EF information.

TriggerDecision	**DF-SW**	**Route**
Physics Accept || Physics Debug || Calibration Debug ||	Any	EF Out
Calibration Accept	None	Calibration
None	None	Delete

Streaming The PT calculates the StreamTag based on the information in the relational TrigConfDB and stores it in the virtual event. The StreamTag, implemented as a list of strings, contains all the information needed for the SFO: stream name, stream type (physics, calibration or debug) and the behaviour with respect to luminosity blocks. As all the information is in the StreamTag, there is no need to have streaming configuration information in the DAQConfDB. Data streaming is configured in the TrigConfDB.

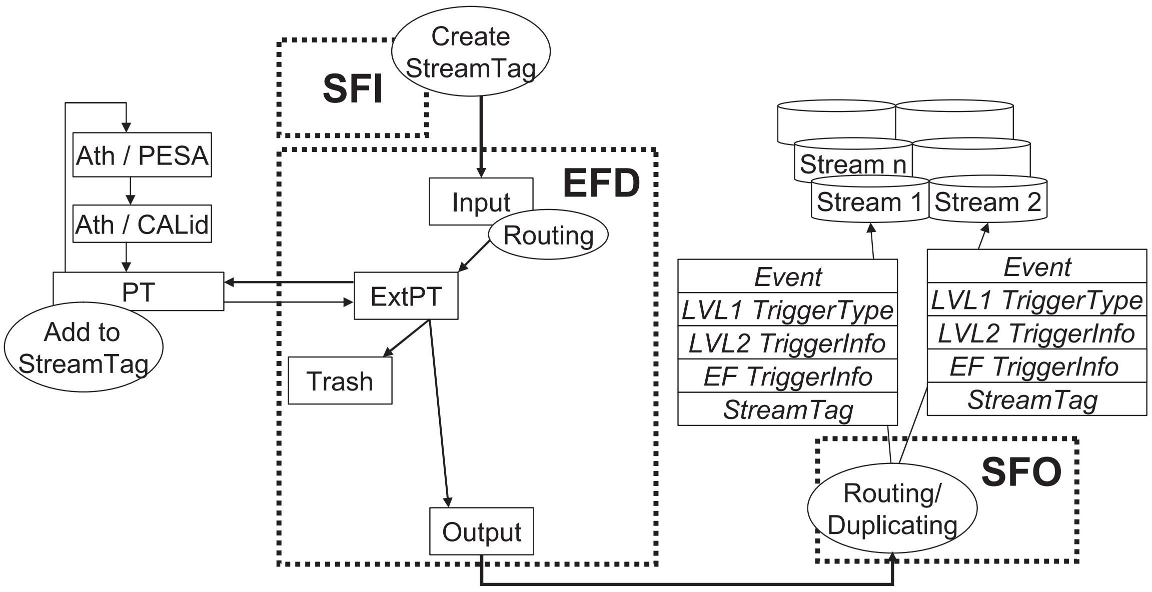

Figure 4: Event Filter, routing in case of physics processing

5.3 EF calibration processing

Routing for calibration processing in the EF is very similar to routing for physics processing as long as an event is identified by the LVL2 trigger system as either physics or calibration (exclusive OR). Instead of routing the event to the physics PT, it will be routed to the calibration PT (see Fig. 5). The construction of the EF TriggerInfo and the StreamTag then proceeds as described in section 5.2. The EFD requires some additional functionality, since

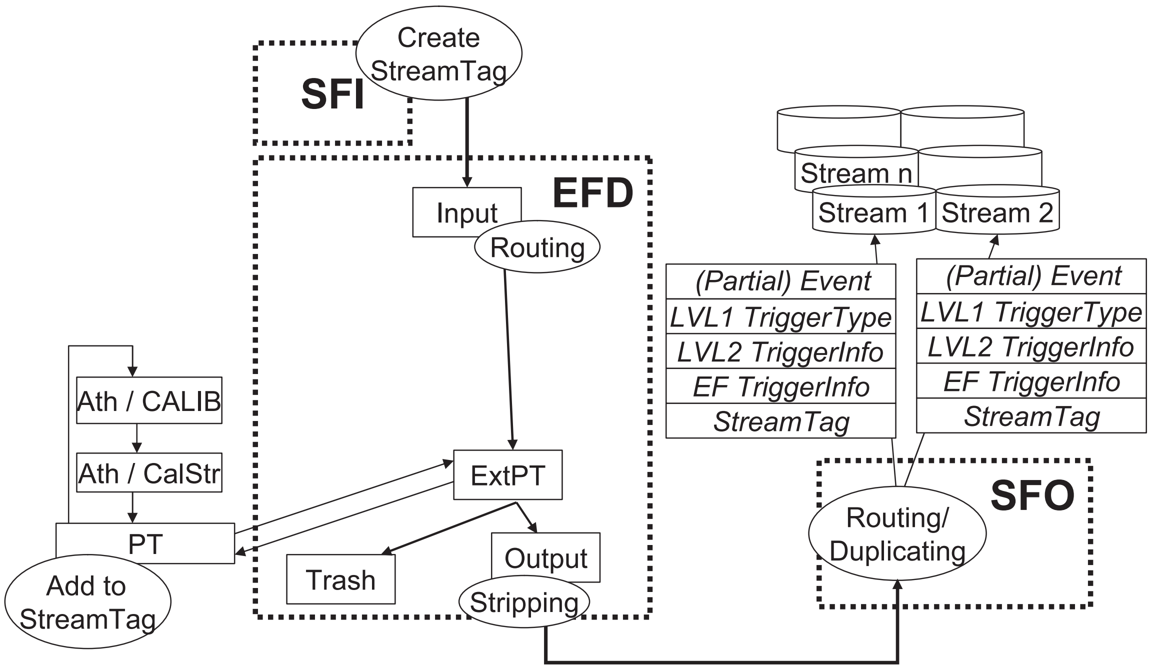

Figure 5: Event Filter, routing in case of calibration processing

most calibration events require only part of the raw event data. This additional functionality is called stripping and shown in Fig. 5 as well.

Other DataFlowPaths can involve the calibration PT as well. A full scheme of all available DataFlowPaths in the EFD is shown in Fig. 6. The physics PT can for example reject an event for physics, but decide that it might still be interesting for calibration. The event will then be routed to the calibration PT after it passed through the physics PT and processed as described in the previous paragraph. An additional complication arises when an event is accepted as a physics event but is also flagged by the PT as interesting for calibration. In this case, the virtual event is duplicated after physics processing by a duplication task in the EFD. The duplicate will be routed to the calibration PT where the decision can be made that the event should be stripped at a later stage (note that this is the reason to send a duplicate). The TriggerInfo and the DataFlow StatusWord in the duplicated virtual event are updated in the calibration PT with the results of the calibration algorithms. Note that two instances of the same event are shipped to the SFO, one of them might be stripped. The physics version has no notion of the result added by the Calibration PT and the two instances have different StreamTags. This allows the SFO to route both the physics and the calibration version of the event into the appropriate offline streams.

5.4 EF information in the EventHeader

The EF information in the EventHeader consists of 8 bits in the DataFlow StatusWords and the EF TriggerInfo:

- The DataFlow StatusWord (DF-SW) contains DataFlow specific information/error bits.
- The EF TriggerInfo (EFTI) contains the EF TriggerDecision (1 word).

The structure of these words is shown in Table 5.

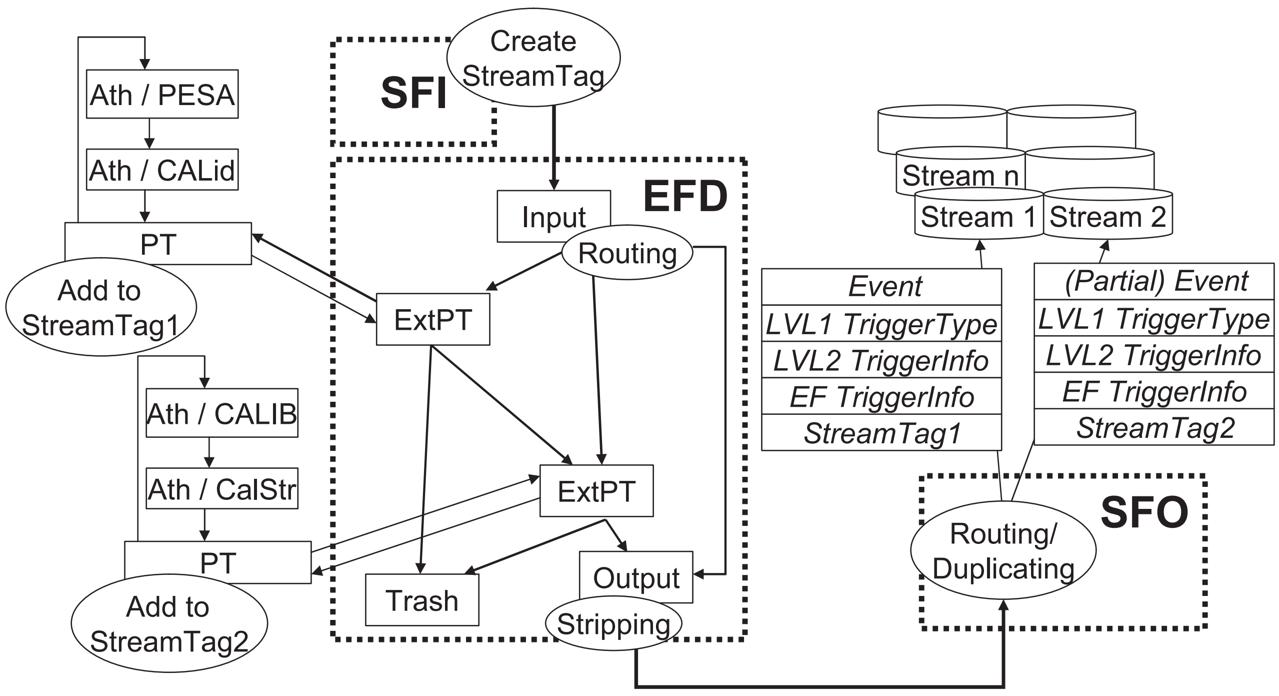

Figure 6: All possible DataFlowPaths in the Event Filter

Table 5: Structure of the EF information in the EventHeader

Word	Usage	Remarks
DF-SW	ForceAccept	DataFlow decision: True/False
	Problem filling TriggerDecision	
	Invalid Full Event	Note: no TriggerDecision available
	Athena/Gaudi problems	
	EF Result problems	
	SharedHeap recovered event	Event may be incomplete
	Duplication warning	Due to EFD/SFO EFIO communication errors
	1 bit reserved	
EFTI	Number of Words	Including this one
	TriggerDecision	Physics: Accept/Debug, Calibration: Accept/Debug
	...	Variable length

5.4.1 EF Dataflow information and errors

EFD ForcedAccept The EFD ForcedAccept bit indicates that the event has not been processed by the PT or caused a PT crash/error, *i.e.* this is an event that may not contain an EFResult. In this case the default StreamTag as created by the SFI is untouched by the EventFilter.

Problem filling TriggerDecision This bit indicates that the PT was not able to fill the TriggerDecision due to *e.g.* missing Algorithm information. Such events need to end up in a debug stream.

Invalid Full Event This bit indicates that the received event contains an invalid FullEventFragment. The PT should flag these events for debugging and continue with the next event.

Athena/Gaudi problems In specific cases, a failure in the Athena/Gaudi services will not result in a PT crash (e.g. failing to put EventInfo in StoreGate). The PT will be configured to flag such events for a debug stream.

EFResult problems This bit represents a collection of problems that can occur with the EFResult object on the way from the Athena algorithm to the PSC and the PT. Again, the PT will be configured to flag these events to a debug stream

SharedHeap problems The event may be corrupted after SharedHeap recovery. Such events need to end up in a Debug stream. Note that this is recovery from a fatal error. The EFD will exit and a brand new EFD will recover the raw events and ForceAccept all of them.

Duplication warning This bit is set in case of EFIO communication errors and/or failures between EFD and SFO that might result in accidental duplication of an event. The DataFlow ignores this warning, however in the offline analysis such events need to be treated with care.

6 Component functionality

In this section we present the actions that each application involved in streaming and routing of the data will need to perform. We also explain what information it needs and how it expects to receive it.

6.1 LVL2 Supervisor (L2SV)

Currently the L2SV passes LVL2 Decisions to the DFM in a very simple way. The oldest valid Event ID is sent followed by a list of Event IDs for accepted events, followed by a list of Event IDsfor rejected events. In the proposed routing scenerio, additional information must be supplied for each accepted event, namely:

- L2SV ForcedAccept flag and ErrorBits
- LVL2 TriggerDecision
- Partial Event Building flag
- SubDetector IDs

There are two possible modes described in the document, namely 1) LVL1 calibration type events, where the L2PU is bypassed and 2) LVL1 physics type events where the L2PU returns the LVL2Result to the L2SV. The LVL2 structures in the EventHeader, as defined in Table 2 are designed to handle these parameters.

This implies that the format of the L2SV_LVL2Decision message send to the DFM will need to be revised. In cases where the event either bypasses the L2PU or when there is no answer coming back from the L2PU (*i.e.* timeout or crash), then the L2SV will have to fill default values for the streaming into the message it sends to the DFM.

If the LVL1 TriggerType implies a test/calibration event, then the SubDetector ID found with the LVL1 TriggerType needs to be translated into a list of SubDetector IDs. The L2SV could in principle do that, with some corresponding configuration parameters from the DAQ-ConfDB, *i.e.*with every SubDet ID found in the LVL1TT a specific list of SubDetector IDs would need to be configured. In case the L2PU decides that a certain event shall be built partially, then it will also create the list of SubDetector IDs which it will send back to the

L2SV with the L2PU_LVL2Decision message. The L2SV forwards these bits to the DFM, from where they are sent to the SFI.

6.2 LVL2 Processing Unit (L2PU)

The L2PU receives from the L2SV the L1 information in form of 8 ROB fragments. These fragments contain *e.g.* the Region of Interest (RoI) information and they are made available to the HLT steering code and the HLT algorithms through the PESA Steering Controller (PSC). After decoding the L1 fragments the HLT algorithms will retrieve the event data fragments within the RoI and provide a trigger decision. The HLT code provides methods which allow the PSC to obtain the algorihtm decisions for both physics and calibration. The PSC sets the corresponding bits in the LVL2 TriggerInfo.

In the case of an accepted event the HLT code provides also extended algorithm processing information in form of an LVL2Result. Inside the PSC this LVL2Result gets serialized and packaged in a ROB fragment. In case of an inconsistent LVL2Result (*e.g.* the event is flagged as accepted but no LVL2Result object is available), or in case of an error during LVL2Result packaging in a ROB fragment (*e.g.* the LVL2Result gets truncated during serialization) the PSC will flag these cases in the LVL2 DataFlow StatusWord. To perform all these actions the LVL2 PSC needs to receive from the L2PU application the L1 TriggerType and from the TrigConfDB a mapping of the Algorithm results to the TriggerDecision. It will return to the L2PU application the LVL2 DataFlow StatusWord, the TriggerDecision and the LVL2Result in form of a ROB fragment. The L2PU will then form the corresponding messages with this information and send them to the L2SV and the pROS.

6.3 DataFlowManager (DFM)

In the proposed architecture all the event building nodes (SFI) are equal. All of them build all types of the events, there is no specialization. For this reason the DFM does not have any active role with respect to the routing of the event data. The DFM only passes some routing information, which will come in messages from L2SVs, to an SFI. This SFI then needs to build each event accordingly. As the DFM has all the information about the event building process it may accumulate statistics, also related to routing of events, if that would be useful (*e.g.* number of physics events assigned, number of calibration events built, number of partly built events). In some cases, luminosity block information might be interesting for these statistics.

6.4 Event Building Node (SFI)

In the TDAQ architecture each event building node (SFI) is assigned to "build an event" by means of DFM_Assign messages arriving from the Data Flow Manager (DFM) in the partition. The SFI node then asks all ROSs (and the pROS which holds the detailed L2Result) for the specific event's fragments. It then collects all fragments and puts them together in a complete event, which it holds until requested by an Event Filter node (EFD application). In its current implementation the SFI builds all events fully, *i.e.* using information from all the ROSs in the partition. Nevertheless, we want to introduce the ability to build partial events (*e.g.* for calibration or events for debugging). In the proposed scheme for data streaming and partial event building, all SFI nodes are equal; *i.e.* there are no dedicated SFI subfarms dealing exclusively with *e.g.* calibration, or physics, or force-accepted events. Thus, in order for an SFI to perform its event building task, it has to know the list of ROSs to be asked. The SFI will be able to find out the list of relevant ROSs to build the event given specific flags coming in a message from the DFM (*e.g.* build event 123, which is a calibration event with relevant info in TTC zone xyz). Note that the SFI constructs a default StreamTag based on the LVL2 TriggerDecision and the DataFlow bits according to a configurable mapping in the DAQConfDB.

6.5 Event Filter Dataflow (EFD)

The InputTask of the EFD receives the events and stores them in the SharedHeap. Next, a RoutingTask decides if the event needs to be assigned to a physics PT via the physics ExtPTs task and/or to a calibration PT via the calibration ExtPTs task or to the OutputTask in case of an EF Bypass. The routing is based on the LVL2 TriggerDecision combined with the information in the DataFlow StatusWords. The explicit configuration of routes is done in the DAQConfDB. If an event is routed to an ExtPTs task it is assigned to an associated PT (physics or calibration). The PT writes a TriggerDecision in the TriggerInfo as described below and the ExtPTs task determines the next required task based on the TriggerDecision and the DataFlow StatusWords. The explicit mapping is defined in the DAQConfDB. Eventually an event is either assigned to the DeleteTask and removed from the SharedHeap or assigned to an OutputTask and sent (fully or stripped) to an SFO. Stripping is indicated in the virtual event and not related directly to any routing and streaming functionality.

6.6 Processing Task (PT)

The PT handles events assigned by the ExtPTs task of the EFD. It forms a bridge to the physics and calibration event selection algorithms and the DataFlow. The algorithms provide methods that allow the PT to obtain the TriggerDecision for both physics and calibration. The PT writes the TriggerDecision in the EF TriggerInfo in the EventHeader. The ExtPTs task of the DataFlow will route the events according to these bits combined with the information in the DataFlow StatusWords. The explicit mapping of TriggerDecision and DataFlow StatusWord bits to routes is done in the DAQConfDB.

Furthermore, the PT will overwrite the StreamTag of the event. The StreamTag is delivered by the algorithms as a list of strings. No intelligence is required in the PT for the processing of the StreamTag.

6.7 Output node (SFO)

The SFO is expected to write the actual data streams. The requirements on the streaming of ATLAS data are given by the recommendations of the Streaming Study Group[9]. We expect to have several physics streams and possible also a few calibration streams. The classification of events will be based on the Event Filter decision. For diagnostic purposes the DAQ will also be able to classify some problematic events as belonging to certain special streams.

The SFOs will not be specialized, each one will write all the data streams in parallel. A data stream is a sequence of raw data files created by the SFO. In addition the SFO will produce an index, which will contain some meta-data per event. This meta-data would contain trigger and streaming information. The index could be sent to a special file. Another possibility, which needs to be investigated, is to send the index information via IS into the conditions data base.

In order to do the streaming of the raw data the SFO needs to receive for each event the information to which streams the event belongs. The same event can belong to multiple calibration streams and to multiple physics streams. In case of the calibration streams the SFO will make separate copies of the events, *i.e.* the total data output will be increased and the events will be multiplied.

In case of the physics streams the requirements are not yet defined. Depending on the outcome of ongoing ATLAS-wide discussions and tests [9], we will have either overlapping streams (also called inclusive), or non-overlapping ones (also called disjoint or exclusive). In the latter case a single overlap stream, containing all the events that belong to more then one physics stream, will be created.

The SFO expects to find the following information on the event header.

- Luminosity Block number,
- list of strings to which the event belongs,
- type of each stream (physics, calibration, debug),
- behavior of each stream with respect to Luminosity Block.

The Luminosity Block number is an integer in the full event header. The StreamTag will be stored in the full event header as a list of strings. The SFO will obtain all the information from these strings using helpful methods of the event format library.

It will be possible to identify the streams, including their type, by the names of the data files produced by the SFO.

In order to preserve physically meaningful information in the StreamTag, the overlap stream will not be represented there. If ATLAS chooses exclusive streaming, the SFO will do the corresponding action - *i.e.* save an event into an overlap stream if more then one physics stream is set.

Appendix A: EventHeader modifications

Implementation of routing and streaming in the HLT puts additional requirements on the EventHeader. Hence, the EventFormat has to be changed to comply with these requirements. An update of the generic definition of the EventHeader as specified in [2] is shown in Table 6. The header size is updated as well as the Format version. Furthermore, there are modifications to the StatusWords and the EventHeader specific words. Changes with respect to the old EventHeader are in *italic*.

Table 6: Generic definition of the EventHeader

Word	Content	Definition
1	0xaaa1234aaa	Start of HeaderMarker
2	0x000000FF	Total Fragment size in words
3	0x000000HH	Header size in words
4	0xVVVVVVVV	Format version number
5	0xSSSSSSSS	Source identifier (from SFI)
6	0xNNNNNNNN	Number of StatusWords
7	0xSSSSSSSSS	StatusWord(s)
X	0xNNNNNNNN	Number of EventHeader specific words
Y	0xSSSSSSSS	EventHeader specific words

The list of possible StatusWords contains a DataFlow StatusWord (identified with Generic Field Value 32), as shown in Table 7.

Table 7: Values and meaning for the Generic field of the first Word

Generic field value	**Description**
0	no known error
1	An internal check of the Bunch Crossing ID failed
2	An internal check of Extended LVL1 ID failed
4	A time out in one of the modules has occurred. The fragment may be incomplete.
8	Data may be incorrect. Further explanation in Specific field.
16	An overflow in one of the internal buffers has occurred. The fragment maybe incomplete.
32	*DataFlow intervention/error. Explanation in specific field.*

The specific part of the StatusWord (2 bytes) contains the DataFlow specific information of both the LVL2 TriggerSystem and the EventFilter. The list is shown in Table 8.

Table 8: Information in the specific part of the DataFlow StatusWord

Bit number	**Byte 2 (LVL2 system)**	**Byte 3 (EventFilter)**
1	ForceAccept	ForceAccept
2	Problem filling TriggerDecision	Problem filling TriggerDecision
3	L2PU Timeout/Crash	Invalid Full Event
4	Duplication warning	Athena/Gaudi problems
5	Duplication warning	EF Result problems
6	3 bits reserved	SharedHeap recovery
7		Duplication warning
8		1 bit reserved

The number of event specific words in the EventHeader will be variable. The proposed list of EventHeader specific words is shown in Table 9.

Table 9: EventHeader specific words

Word	**Contents**	**Definition**	**Comments**
1+2	*Bunch Crossing time*	*64-bit integer*	*Extracted from CTP info*
3	Global event ID	32-bit integer	Number assigned by Event Builder
4	Run Number	0xRRRRRRRR	*Flat Run Number*
5	*Luminosity Block Number*	*0x0000LLLL*	*16 bit Luminosity Block ID*
6	Extended LVL1 ID	0xNNNNNNNN	8 bit ECR count + 24 bit LVL1 ID
7 *W-1*	*Partial Event Building* *DetectorMask*	*0xWWWWWWWW* *0xDDDDDDDD*	*Nr. of Partial Event Building words* *SubDetector IDs*
8+W	LVL1 TriggerType	0x000000TT	8 bits status summary from LVL1 CTP
9+W *1* *X-2*	*LVL2 TriggerInfo* *LVL2 TriggerDecision* *...*	*0xXXXXXXXX* *0xTTTTTTTT* *0xTTTTTTTT*	*Nr. of LVL2 TriggerInfo words* *One word defined in TrigConfDB* *Variable length*
10+W+X *1* *Y-2*	*EF TriggerInfo* *EF TriggerDecision* *...*	*0xYYYYYYYY* *0xTTTTTTTT* *0xTTTTTTTT*	*Nr. of EF TriggerInfo words* *One word defined in TrigConfDB* *Variable length*
11+W+X+Y *Z*	*StreamTag* *Stream descriptions*	*A list of strings.* *0xSSSSSSSS*	*Overlap stream not included.* *Defined in eformat*

References

[1] Christophe Meessen Andrea Negri. Sharedheap. Technical Report v2, CERN, https://uimon.cern.ch/twiki/pub/Atlas/EventFilterDataflow/SharedHeapV2.pdf.

[2] L. Mapelli R. McLaren G. Mornacchi J. Petersen-F. Wickens C. Bee, D. Francis. The raw event format in the atlas trigger and daq. Technical Report ATL-D-ES-0019 and ATL-DAQ-98-129, CERN, 2005.

[3] Andre dos Anjos et al. Trigger configuration workgroup - http://isscvs.cern.ch/cgi-bin/viewcvs-all.cgi/lvl1/trigconf/?cvsroot=atlastdaq. Technical report, CERN, 2006.

[4] Igor Soloviev et al. Configuration database - http://atlas-onlsw.web.cern.ch/atlas-onlsw/components/configdb/welcome.html. Technical report, CERN, 2005.

[5] R. Spiwoks et al. Trigger, timing and control system - http://atlas.web.cern.ch/atlas/groups/daqtrig/level1/ctpttc/l1ttc.html. Technical report, CERN, 2002.

[6] C. Häberli HP. Beck. High-level description of the flow of control and data messages for the atlas tdaq integrated prototype. Technical Report DataCollection note 012 and ATL-DQ-ES-0028, CERN, 2002.

[7] Y. Nagasaka HP. Beck, F. Wickens. The message format used by datacollection in the atlas tdaq integrated prototype. Technical Report DataCollection note 022 and ATL-DQ-ES-0035, CERN, 2006.

[8] Per Gallno Georges Schuler Ralf Spiwoks Thomas Schoerner-Sadenius Thorsten Wengler Nick Ellis, Philippe Farthouat. Definition of the trigger-type word. Technical Report ATL-DA-ES-0022 and EDMS Id 325753, CERN, 2004.

[9] Hans von der Schmitt et al. Atlas streaming study group - http://atlas.web.cern.ch/atlas/groups/software/commissioning/streaming.html. Technical report, CERN, 2006.