LPHE 2005-08

Raw-data transport format

LHCb Technical Note

Issue: 4
Revision: 3

Reference: LHCb 2004-056 DAQ
Created: 11/02/2003
Last modified: 21/07/2004 18:02:03

Prepared By: LHCb DAQ Group
B. Jost and N. Neufeld

Abstract

This document describes the raw data transport format as it must provided by the front-end electronics boards, when transmitting data to the combined L1 trigger and HLT system based on Ethernet. Several events need to be packed into one packet to ease the load on the event-builder. The precise definition of the headers for both data paths is given and a fragmentation mechanism for large packets is described.

This document contains important changes and supersedes all previous versions of this note.

Document Status Sheet

Table 1 Document Status Sheet

1. Document Title: Raw-data transport format			
2. Document Reference Number: LHCb 2004-056 DAQ			
3. Issue	**4. Revision**	**5. Date**	**6. Reason for change**
4	3	12/07/2004	Remove padding byte
4	2	24/09/2003	Fix error with bit and byte ordering in Ethernet Frame
4	1	15/08/2003	Change definition of IP Identification field to ensure difference between L1 and HLT packet stream.
4	0	28/07/2003	Change to 32 bit formatting for all fields. Work in Jorgen's comments
3	Draft	11/02/2003	Change everything to IPv4
2	Draft	11/02/2003	Implement various comments Bring in line with L1 front-end electronics document [1] Rewrite to reflect decision on common implementation
1	0	11/02/2003	Initial Version

Table of Contents

List of Figures

List of Tables

1 Introduction

The front-end electronics boards [1] must transmit the data to the DAQ system. All detectors send their data upon reception of a L1 yes; this data is referred to as the Higher Level Trigger (HLT) data. Some detectors also participate and send (similar but different) data upon reception of a L0 yes. Both data streams are received by the same event building system, which is based on Ethernet.
The event-building system collects and merges data fragments from all participating detectors, forwards them to a CPU for the corresponding trigger algorithm to decide (L1 or HLT). The event-building system (or DAQ for short) never looks at the contents of the data fragments described in [2] for the HLT or in [4] for the Level 1. In order to assemble the many fragments belonging to one event, it needs additional information. This information must be put into a header which must be pre-pended to .each fragment sent by the board.
Some of the information to be put into the header comes from the Experiment Control System (ECS) (e.g. the partition ID); some information is transmitted via the TFC system [3] (e.g. destination addresses), finally some information must be generated by the driving entity on the board itself (e.g. the length of the event fragments).
In order to be able to implement the event builder system fully from commercial components it is necessary to reduce the high packet rate of 1 MHz for Level-1 events by packing several events. These packed events are then formatted as Internet Protocol (IP) packets and sent via the Gigabit Ethernet mezzanine card [4], which interfaces the board to the DAQ.
This note describes in detail the header format for the two data streams, the L1 and the HLT, the packing and the fragmentation of the MEP into IP packets.

1.1 Basic notations

It is assumed that the data is formatted by an FPGA or processor like entity, which can read some registers, which have been configured externally by means of the ECS [6]. In the following we will call this entity the *driver*. The driver will write its data directly to the interface described in [5] using a bus also there.

A word is understood to consist of 32 bits, with the least significant bit being bit 0. A half-word is understood to consist of 16 bits, with the least significant bit being bit 0.

If a number has less significant bits than reserved then it must be filled into the least significant part of the field, i.e. bit 0 must be the LSB. All remaining bits must be set to 0.

Several parameters are set by the ECS. Some of them are implemented differently for L1 and HLT. A complete list is given in Section 2.3.1.

1.2 MEP network protocol

When we apply the OSI seven layer model (see Figure 1) to the LHCb transport format, Ethernet is the Datalink (layer 2), IP is the Network (layer 3) and MEP is the Transport (layer 4). The MEP is implemented on top of IP like UDP or TCP are implemented on top of IP, MEP, which could in this sense also be read as (**M**ulti-**E**ventfragment-**P**rotocol). MEP consists of one MEP header of 12 bytes (described below) and several MEP fragments, each of which is preceded by a 4-byte fragment header (described below). The number of fragments in a MEP is called the *packing factor* (**pf**). MEP inherits from IP the maximum size of a datagram, which is 64 kB – sizeof (MEPheader) – 4 * pf. Like UDP MEP is an unreliable, connectionless protocol. It should be noted that numbers in MEP are stored in **little endian format**, ***contrary to Ethernet and IP***. That means that the lower bytes of a (half-) word are at lower addresses. This is the endianness of the IA32 and IA32_X64 machines, which will be the majority of the receivers.

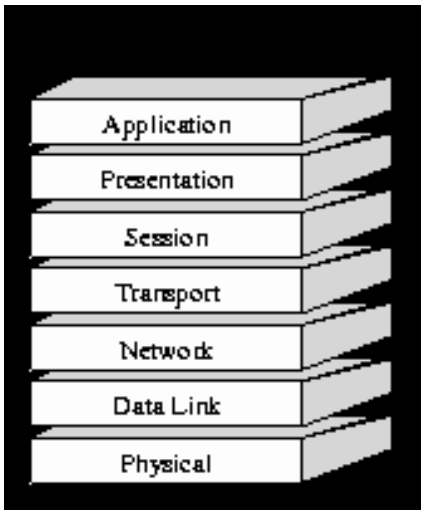

Figure 1The OSI Layer model for network protocols

2 Transport structure

The various layers of the event are described in Figure 2 In the following the formatting of the various parts will be described, starting from the innermost part, the raw event blocks, to the outermost part, the Ethernet frames.

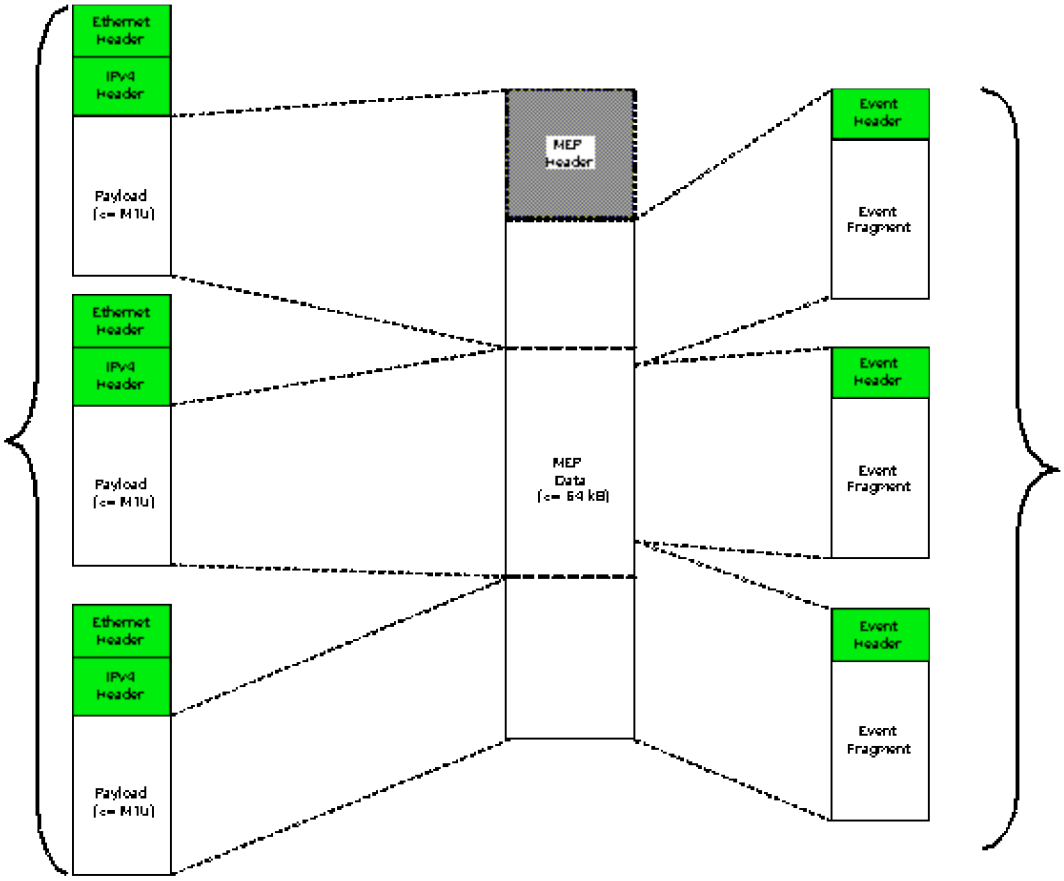

Figure 2 A MEP consisting of a MEP header and several events with an event header appended (right side). The left side indicates how the MEP will be segmented into several IP packets embedded in Ethernet frames.

Here a summary of the processing required for the formatting is given:
Once the required number (given by the Packing Factor set by the ECS) of events is ready a MEP is formed as described in Section 2.1. The resulting MEP gets a packet header; the MEP becomes the payload of the IP packet. The resulting packet is submitted to the fragmentation procedure described in Section 2.2 the packets smaller or equal to the maximum frame size (MTU) are sent as described in Section 2.3 to the Gigabit Ethernet Mezzanine.

2.1 Multiple Event Packets

In order to reduce the packet rate, the driver is required to pack adjustable amount of event fragments into one Multiple Event Packet (MEP). The information for the event-builder to dis-assemble and re-arrange the MEPs must be put into a header, described in this section. The header is 12 Bytes long. Since it has been decided that one MEP must always fit into one IP packet, this means that **the amount of data in the MEP can be at most 64 kB – 12 B – pf * 4 = 65524 – pf * 4 Bytes!.** For the nominal packing factor of Level-1, 25, this means 65424 / 4 = 16356 words are useable for the raw data in the 25 fragments themselves.

The use of IPv4 is necessitated by the fact that many high-end network routers are based on this protocol. They will ignore, and even overwrite the information contained in the lower level transport protocol, in our case Ethernet.

As has been said above, the maximum size of an IP packet is 64 kB (65536 Bytes). This is larger than the maximum size allowed for an Ethernet frame (maximum transmission unit MTU), which is either 1500 or 9000 Bytes, which is also what the Gigabit Ethernet Mezzanine will accept as the maximum length of a packet into its FIFO [5]. IP describes a fragmentation procedure to cope with this case. This is described below in section 2.2

The full header is shown in Table 2.

Table 2 Data format for the L1&HLT raw data. A MEP can span several frames, each of which has to have transport header shown in green. **Only the first packet** contains the 3x 32 bits word long MEP header shown in yellow, the other packets have a structure as shown in Table 3. An event-fragment in a MEP is always preceded by a 32 bit long event-header shown in blue.

Bits 31:24	Bits 23:16	Bits 15:8	Bits 7:0
DA[7:0]	DA[15:8]	DA[23:16]	DA[31:24]
DA[39:32]	DA[47:40]	SA[7:0]	SA[15:8]
SA[23:16]	SA[31:24]	SA[39:32]	SA[47:40]
L/T[15:8]	L/T[7:0]	Version/ IHL	Type of Service
Total Length[15:8]	Total Length[7:0]	Identification[15:8]	Identification[7:0]
Flags/ Fragment Offset[11:8]	Fragment Offset[7:0]	Time to Live	Protocol
Header Checksum[15:8]	Header Checksum[7:0]	IP-SA[31:24]	IP-SA[23:16]
IP-SA[15:8]	IP-SA[7:0]	IP-DA[31:24]	IP-DA[23:16]
IP-DA[15:8]	IP-DA[7:0]	L0ID / L1ID[7:0]	L0ID / L1ID[15:8]

L0ID / L1ID[23:16]	L0ID / L1ID[31:24]	Number of Events[7:0]	Number of Events[15:8]
Total Length[7:0]	Total Length[15:8]	Partition ID [7:0]	Partition ID [15:8]
Partition ID [23:16]	Partition ID [31:24]	EventID 1[7:0]	EventID 1[15:8]
Len 1[7:0]	Len 1[15:8]	Data1[7:0]	Data1[15:8]
Data1[23:16]	Data1[31:24]	…	…
…	…	Event ID2[7:0]	Event ID2[15:8]
Len 2[7:0]	Len 2[15:8]	Data2[7:0]	Data2[15:8]
Data2[23:16]	Data2[31:24]	…	…
…	…	…	…
…	…	…	…
…	…		

The meaning of the fields is explained in the following:

DA: 48 bits

Ethernet destination address. The 38 most significant bits are set by ECS; differently for L1 and The 10 least significant bits are set by the driver, based on information received by the TTC.

SA: Byte 5 4 3 2 1 and 0: 48 bits

Ethernet source address. Set by ECS; differs for L1 and HLT.

Ethernet Type L/T: 16 bits

Ethernet type field. Set by ECS.

Version: 4 bits

IP version number. Set by ECS 4

IHL: 4 bits

Internet Header Length is the length of the header in 32 bit words. Set by ECS.

Type of Service: 8 bits

Indication of desired service for this packet. Set by ECS; differs for L1 and HLT.

Total Length: 16 bits

Total length of this packet including the 5 word long IP header, but NOT including the Ethernet header (i.e. the first 14 bytes of the transport header), measured in Bytes. Set by driver

Identification: 16 bits

Assembly aid set by driver to the least significant 15 bits of the first event ID (L0ID or L1ID) of the MEP. **The MSB (Bit 15) must be set to 1 for L1 and to 0 for HLT**[1]. See below 2.2

Flags: 3 bits

Bit 0: always 0
Bit 1: (DF) 0 = may fragment, 1 = don't fragment. Must always be set to 0.
Bit 2 (MF) 0 = Last fragment, 1 = more fragments. Set by driver during fragmentation (see next section).

Fragment offset: 13 bits

This measures where the fragment belongs in the packet. It is measured in units of 8 bytes (64 bits). The first fragment has offset 0, set by driver

Time to live: 8 bits

Time in seconds for the packet to stay in the net: set by ECS.

Protocol: 8 bits

Indicates the next level protocol, set by ECS; differs for L1 and HLT.

Header checksum: 16 bits

The checksum is on the header only: It is the 16 bit one's complement of the one's complement sum of all 16 bit words in the IP header. These are the 10 16 bit words starting at the *Version / Type of Service* field up and including to the *IP Destination Address*. It is important that for the purpose of this calculation the 16 bit words are assumed to get the most significant byte (of the two bytes) from the lower address (assuming that the bytes are addressed linearly – this is sometimes referred to as big-endian byte ordering). For the calculation of the checksum the checksum field is assumed to be 0. Set by driver

IP SA: 32 bits

IP source address. Set by ECS. Differs for L1 and HLT.

IP DA: 32 bits

[1] This difference allows the SFC to discern between the two IP streams when the LSBs of the L0 and L1 event-ID are accidentally equal.

IP destination address. Set by driver. 22 bits are assigned by the ECS. 10 bits are received by the TTC for each MEP. Differs for L1 and HLT.

L0ID / L1ID: 32 bits

This field contains the L0ID for Level-1 MEPs and the L1ID for HLT MEPs. This is the event number of the first event in the MEP. Set by TTC / ECS

Total Length: 16 bits

The total length of the MEP including the header in Bytes. Set by the driver

Number of events: 16 bits

The number of events packed into this super-event. Set by ECS. Important: this value will not change during a run. When the run however terminates for any reason the MEP must be filled with the events collected so far and the Number of Events set accordingly by the driver.

Partition ID: 32 bits

The partition ID defines the partition this data source is currently part of. Set by the ECS at the start of RUN, valid throughout one run.

2.1.1 Event header for events in MEP

The event-builder needs information to perform the deconvolution of the events in the MEP. This information consists of two 16-bit words shown in blue in Table 2 and Table 3:

Length: 16 bits

Length of the event following event fragment (excluding the event header) in bytes. Set by driver

Event ID LSB: 16 bits

The least significant 16 bits of the L0ID (L1ID) of the event. The 16 most significant bits are given in the MEP header, which contains all bits for the L0ID (L1ID) for the first event in the MEP only. Set by the driver. Unused bits must be set to 0.

2.2 IPv4 Fragmentation

The fragmentation must be implemented by the driver. It is imperative that the procedure is followed to the point, because commercial IP equipment, will check the packets it receives, and if it finds them non-well formed it will drop them, because that is what the IP standard dictates.

The input to the algorithm described in the following is a full MEP formatted exactly as shown in Table 2. The fragmentation process treats everything after the transport header (shown in green) as payload. In particular, **the MEP header**, shown in yellow **is not repeated** in the course of the fragmentation procedure, while the transport headers are repeated. The first packet will have a structure like shown in Table 2; all subsequent packets resulting from the fragmentation of the MEP will have a structure as show in Table 3. **All packets created during the fragmentation process**

from a single MEP, have an identical Ethernet header – the first 14 Bytes of the transport header (in green).

The fragmentation procedure is as described in RFC 791 [6], with some simplifications.

Let TL be the total number in bytes of the MEP including the IP header and the MEP header. The maximum sized packet that can be transmitted over Ethernet is called the maximum transmission unit (MTU), which is set by ECS.

If the total length is less than or equal the MTU then the packet can be submitted to the Ethernet layer and from where to the network right way. Otherwise the packet must be split into two fragments, the first fragment being the maximum size, and the second being the rest of the packet. The first fragment is handed over to Ethernet, while the second is re-submitted to the procedure, in case it is still too large.

Notation:

FO	Fragment Offset
IHL	Internet Header Length
DF	Don't Fragment flag. Always 0
MF	More Fragment flag
TL	Total Length. Length of the MEP including all headers and the IP header (20 bytes).
OFO	Old Fragment Offset
OIHL	Old Internet Header Length
OMF	Old More Fragments flag
OTL	Old Total Length
NFB	Number of Fragment Blocks, a fragment block is 8 bytes = 64 bits
MTU	Maximum Transmission Unit. The maximum length for an Ethernet frame in bytes. Set by ECS.
A := B	A is set to B

Procedure

```
IF TL =< MTU THEN Submit this fragment to the Ethernet layer  (see
Section 2.3)
ELSE
        To produce the first fragment:
        (1)  Copy the original internet header;
        (2)  OIHL := IHL; OTL := TL; OFO := FO; OMF := MF;
        (3)  NFB := (MTU-IHL*4)/8;
        (4)  Attach the first NFB*8 data bytes;
        (5)  Correct the header:
             MF := 1;  TL := (IHL*4)+(NFB*8);
             Recompute Checksum;
        (6)  Submit this fragment to the Ethernet Layer
             (see Section 2.3);
        To produce the second fragment:
        (7)  Copy the internet header
        (8)  Append the remaining data;
        (9)  Correct the header:
```

```
        IHL := ((OIHL*4) + 3)/4;
        TL := OTL - NFB*8 - (OIHL-IHL)*4);
        FO := OFO + NFB;  MF := OMF;
        Recompute Checksum;
(10)    Submit this fragment to the fragmentation test; DONE.
```

Table 3 Data format for the L1&HLT raw data. A MEP can span several frames, each of which has to have transport header shown in green. **This table gives an example of the packet structure as it looks for all packets resulting from fragmentation except the first.** As indicated, the split need not be at an event boundary, but is solely given by the algorithm described in Section 2.2. The description of the fields is given in Section 2.1.

Bits 31:24	**Bits 23:16**	**Bits 15:8**	**Bits 7:0**
DA[7:0]	DA[15:8]	DA[23:16]	DA[31:24]
DA[39:32]	DA[47:40]	SA[7:0]	SA[15:8]
SA[23:16]	SA[31:24]	SA[39:32]	SA[47:40]
L/T[15:8]	L/T[7:0]	Version/ IHL	Type of Service
Total Length[15:8]	Total Length[7:0]	Identification[15:8]	Identification[7:0]
Flags/ Fragment Offset[11:8]	Fragment Offset[7:0]	Time to Live	Protocol
Header Checksum[15:8]	Header Checksum[7:0]	IP-SA[31:24]	IP-SA[23:16]
IP-SA[15:8]	IP-SA[7:0]	IP-DA[31:24]	IP-DA[23:16]
IP-DA[15:8]	IP-DA[7:0]	Data5	Data5
…	…	…	…
Event ID6[7:0]	Event ID6[15:8]	Len6[7:0]	Len6[15:8]

Data6[7:0]	Data6[15:8]	Data6[23:16]	Data6[31:24]
…	…	…	…
Event ID7[7:0]	Event ID7[15:8]	Len7[7:0]	Len7[15:8]
…	…	…	…

2.3 Sending packets to the Ethernet Mezzanine

In the LHCb front-end electronics Layer 2 / Ethernet **Error! Reference source not found.** is handled by the standard Gigabit Ethernet multi-link mezzanine card [5]. This card expects as an input into either of its two independent FIFOs a complete Ethernet frame, which is guaranteed by adding a header as prescribed in Table 2 and Section 2.2

The data block can be written assuming that there is a receiving FIFO at the other end, and the block will go out to the network exactly as it has been written with two notable exceptions, which are transparent to the driver (but visible to the receiving end, which is why they are noted here):

1. A 4 byte CRC checksum will be appended by the interface card

2. While the block written can be shorter, the interface will always pad the block with 0 bytes to the minimum Ethernet length of 64 bytes.

2.3.1 Example of writing a frame to the Gigabit Ethernet Mezzanine

In the following a short description is given what to write to the 32-bit bus of the Gigabit Ethernet Mezzanine given some initial values from ECS.
We assume the following frame to be written

Etherent Source Address:	00:11:22:33:44:55
Ethernet Destination Address:	66:77:88:99:AA:BB
IP Source Address:	127.0.0.1
IP Destination Address:	127.0.0.2
Partition ID:	0x12345678
Number of Events (Packing Factor)	2
Event length	12 Bytes (i.e. 3 32-bit words)
IP Identification:	0x0000
IP Time To Live:	64
IP Flags:	0x4 (don’t fragment, no more fragments)