TAGnet
a twisted-pair protocol for event-coherent DMA transfers in trigger farms

Hans Műller, Francois Bal, Sebastien Gonzalve, Angel Guirao, Filipe Vinci dos Santos

CERN EP-ED group[1]
1211 Geneva 23, Switzerland

Abstract

TAGnet is a link protocol developed for high-rate hardware eventbuilding in trigger farms. Its first implementation is in the level-1 VELO trigger system of LHCb where all data sources (Readout Units) need to receive destination addresses for their DMA engines at a trigger rate of nominally 1 MHz. TAGnet organises event-coherency for the source-destination transfers and provides the proper timing for best utilization of the network bandwidth. The serial TAGnet LVDS link interconnect all Readout Units in a twisted-pair ring, which is controlled by a TAGnet scheduler.

I. Introduction

TAGnet is a protocol for the creation of event-coherent transfers between hardware DMA engines of Readout Units (RU [5]) and CPUs of a trigger farm (Fig.1). A TAGnet ring interconnects slave DMA modules via twisted pairs to a TAGnet scheduler that collects all requests from the farm CPUs. The scheduler is under control of a host CPU which may be embedded in the trigger farm.

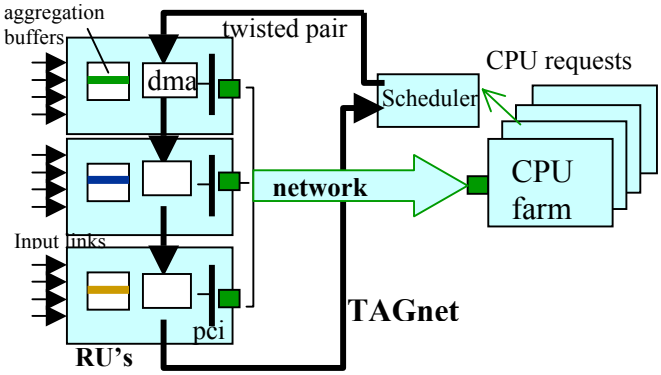

Figure 1. TAGnet in a readout system. Farm CPUs send their request to a scheduler which masters the TAGnet ring. Under TAGnet command, DMA engines transfer the next event buffers to the requesting CPUs.

The scheduler provides the proper timing of the transmission and organizes "event-coherent" DMA transfers from all RU buffers as follows: In the RU input buffers, detector data are received, aggregated and queued in increasing event-order like in a FiFo. TAGnet associates the event-number of the oldest event in the RU buffers with the address of a requesting CPU. The consecutive network DMA from all RU's transports all fragments of one event to a CPU and is hence called Event-Coherent DMA (E.C.D.).

This E.C.D. approach has the following advantages over conventional eventbuilding approaches as far as they applicable at these rates:

1. A farm CPU requests new events from the scheduler whilst processing a previous one (double buffer, no worst-case buffer requirements like for "round-robin")
2. A TAGnet implementation is based on twisted pair, FPGA logic, and one programmable PCI card, hence can be implemented at very low cost and real estate.
3. TAGnet provides added functionality via "message-TAGs" like for online RU buffer-checks, and error reporting from the DMA engines.
4. highest possible use of the raw network bandwidth due to hardware timing.
5. no spurious arrivals of event fragments since all arrive concatenated in time at the destination CPU
6. no problem with crashing farm CPUs (just 1 less)
7. no problem with very large variations in CPU-time per event (auto-balancing load).

The majority of all TAGs are Command-TAGs (C-TAGs), which contain realtime information for the DMA engines. The TAG transmission rate is considerably higher than the incoming trigger rate, leaving ample bandwidth for Message TAGs (M-TAGs), which transport non time-critical information for control or message exchange.

For the 1 MHz VELO trigger application of LHCb, a cluster with 32 CPUs has been built at KIP Heidelberg, in which CPUs and RU's are interconnected via a 6 Gbit/s network. A shared-memory paradigm is established [2] between all destination CPUs and the MCUs (processor card) on the RU's.The "hardware eventbuilding" in this cluster is based on direct copying of a fixed numbers of input fragments to the destination memory in requesting CPU. Event-fragments (Si-strip clusters) in the input buffers of all RU's get aggregated to average payloads of 128 bytes which get transmitted as DMA packets to the (remote) CPU memory, arriving here as E.C.D. burst. The timing and organisation is due to C-TAGs which, with the network technology used at Heidelberg, can reach event transfer rates well bejond the required 1 MHz.

C-TAGs transmit the requests from individual CPUs of the trigger farm, via an intermediate TAGnet scheduler, to the RU's. The CPU identifiers are converted within each RU into physical PCI addresses, prior to be loaded into the DMA engines. These PCI addresses map, via the shared memory

[1] http://ep-div-ed.web.cern.ch/ep-div-ed/

mechanism, directly to a destination memory of the requesting CPUs. There is no dead-time involved in the E.C.D. since when starting to process an event, the CPUs pipeline their request for new E.C.D. event for their alternate buffers. The transfer latencies are in the order of a few microseconds.

II. TAGs

The TAG is a 64 bit object are (Fig. 2) containing several bit-fields, of which the most important :

1.) Command: 3 bit Command field for TAGnet slaves
2.) CPU-Identifier: 12 bit Identifier of the requesting CPU
3.) Buffer address: 11bit address of the DMA eventbuffer

The other fields contain more advanced TAG features like:

4 TAG classes for validity and consume properties
4 TAG types differentiate C-TAGs and M-TAGs
7 bit Done counter counts down in each slave action
7 bit Source Module ID identifies the DMA module (RU)
7 bit Coded information for messages and error reporting
7 bit error correcting code (2 bit correction)

A 64 bit TAG contains 9 bit-fields, physically a TAG it is generated in 4 successive 16-bit words, followed by one idle word.

TAG: a 64 bit of transfer information

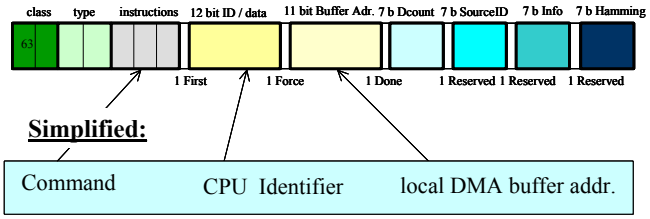

Figure 2. A TAG in its logical 64 bit representation

A. Heartbeats

A TAG is transmitted on an external bus as a fixed clocked sequence, called "heartbeat" (Fig. 3).

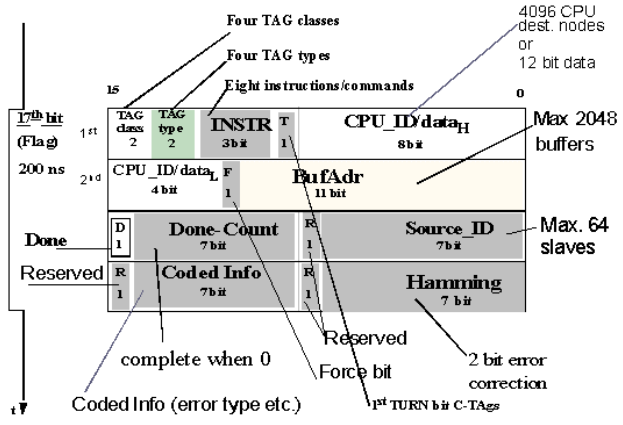

Figure 3. External TAG representation on a 16 bit bus. A 17th FLAG bit delimits four consecutive TAG words which correspond in their bit-fields and order exactly the logical 64 bit TAG representation.

A 17th bit is used to generate a "heartbeat" (Fig. 4) consisting of 4-words and 1 idle clock. The scheduler generates TAGs of class "valid or invalid" which are copied into the heartbeat. For example "invalid TAGs" may be used to empty the TAGnet ring whilst the heartbeat maintains the physical ring activity alive. The integrity of the 64 bit TAG within a heartbeat is protected by a Hamming code, the integrity of the 17th frame bit is verified via the returning heartbeat pattern by the TAGnet scheduler.

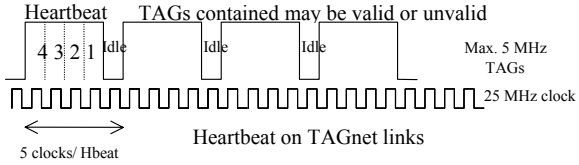

Figure 4. The TAGnet heartbeat is generated by the 17th flag bit of the 16 bit TAG representation. Heartbeats are permanent whether they contain valid TAGs or not. Shown here is that with a 25 MHz bus clock, TAGs are transmitted at 5 MHz.

B. TAGnet over twisted pair links

In order to link DMA engines over distances of up to 15 meters, the 16 bit TAG is serialized using commercial LVDS serializer / deserializer chips which contain four 7-bit converters (Fig. 5), each driving one of the four wire pairs of a standard network cable. The TAGnet clock is transmitted over one pair whilst the remaining 17 bits are divided up on the remaining 3 lines.

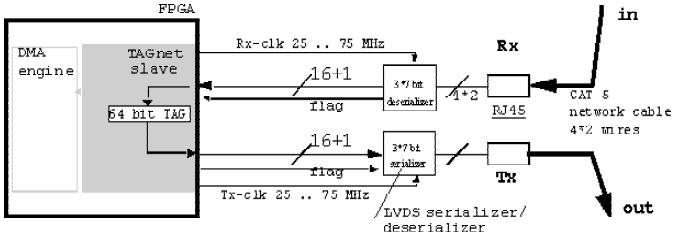

Figure 5. Using LVDS serializers / deserializer chips, TAGnet heartbeats are transmitted over standard CAT6 network cables which contain 4 shielded twisted pairs.

Using a 25 MHz TAGnet clock, 3*175 Mbit/s get transmitted across the CAT6 cables that interconnect TAGnet nodes in a ring.

C. TAG classes

The upper two bits of a TAG define four TAG classes generated by the "valid" and the "consume" bits and are explained in more detail in Table 1. The slaves from the TAGnet link strip consumable TAGs, and if valid, they get examined for class and type before being passed to command execution. An executed TAG gets re-transmitted to the TAGnet link by copying it into the heartbeat of the next invalid TAG on the link.

Non-consumable TAGs are bypassed by slaves without modification after a minimal delay of 1 heartbeat.

TABLE 1. TAG CLASSES

valid	consume	TAG classes
0	0	**invalid, not consumable.** These filler TAGs have no other purpose than allowing the scheduler to control the level of usable TAGs. In order to clear the TAGnet ring the scheduler transmits only these TAGs.
0	1	**invalid, consumable**. These TAGs are freely consumable, invalid TAGs which can be used by any TAGnet slave to create valid TAGs at it's output
1	0	**valid, not consumable.** These TAGs fall into the type of directed scheduler messages, created normally by a TAGnet slave. They contain message information (like errors) for the scheduler and hence must not be consumed by other TAGnet slaves.
1	1	**valid, consumable.** These TAGs fall into the types: undirected command, undirected message,directed slave message and hence contain important scheduler information (command / address / message) to be consumed by TAGnet slaves.

D. TAG types

The four TAG types (see Table 2) are:

a.) C-TAGs: these are the vast majority of all TAGs for generating event-coherent DMA
b.) M-TAGs: They exist in 3 types for messages between slaves and scheduler

The TAG types are defined by two bits in the TAG header as shown in the table below:

TABLE 2. TAG TYPES

directed / undirected	encoded messages	**TAG types** (valid TAGs only)
0	0	**undirected slave command (C-TAG)** of class VALID CONSUMABLE. C-TAGs convey command and data to all slaves. These realtime TAGs are the large majority of all TAGs
0	1	**undirected slave message (M-TAG)** of class VALID CONSUMABLE. These TAGs send an encoded message to all slaves
1	0	**directed slave message (M-TAG)** of class VALID CONSUMABLE. These TAGs send command and data to one slave
1	1	**directed scheduler message (M-TAG)** of class VALID NON CONSUMEABLE. These TAGs send an encoded message (error or other) from a slave to the scheduler

E. Message TAGs (M-TAGs)

Message TAGs (M-TAGS) coexist with C-TAGs for messages between slaves and scheduler. Generated by the scheduler software, M-TAGs are not time critical. For each of the three types of M-TAGs there are 8 possible actions defined by the 3 bit instruction field of the TAG. Some examples of M-TAGs are shown in tables 3, 4, and 5 below:

Table 3. M-TAG message examples of "Directed to 1 Slave"

0	0	0	**SELECT:** select a TAGnet slave operation mode contained in the 6 bit info field
1	0	0	**FLUSH ALL**: flush all buffers, reset pointers

Table 4. M-TAG message examples of "Undirected Slave" (to all slaves)

0	0	0	**DIAG**: request for Slave Info via M-TAG as specified in Coded INFO field
1	0	0	**FLUSH ALL:** flush buffers, reset their pointers

Table 5. M-TAG message examples of "Undirected Slave" (slave to scheduler)

0	0	0	**ERROR** (error type decoded in INFO field)
1	0	0	**THROTTLE request (** stop input to protect DMA buffers)

III. LHCB'S VELO TRIGGER TESTBED

The global I/O requirement for trigger-eventbuilding in the LHCb VELO trigger is 4 kbyte of trigger-events at a rate of up to 1.1 MHz. More specifically, only very small payloads are produced per link (30 - 40 byte) and the farm is to be constructed from COTS computers, but requires use of the 64bit@66 MHz PCI bus for achieving the needed I/O performance.

A first trigger prototype, a 32-CPU cluster, has been constructed at KIP in Heidelberg from COTs PCs, which are interconnected via a 6 Gbit/s standard network in a 2D torus architecture [4]. The I/O paradigm[2] implemented is "shared memory", i.e. for data transfers within the cluster there are no intermediate buffers, nor copy processes, nor messages. The memory-memory DMA transfer curve for this cluster has been measured for 3 different variants of DMA:

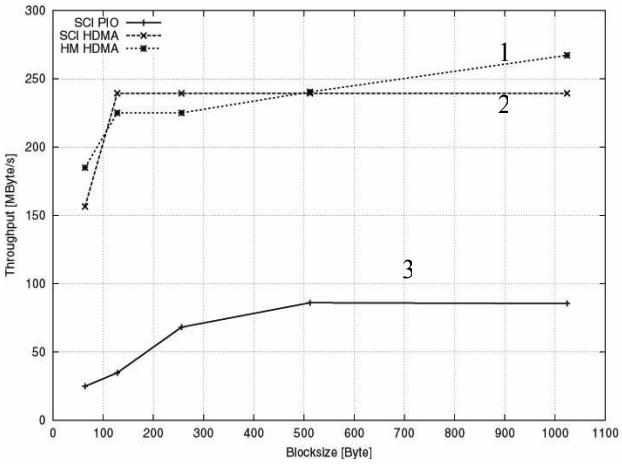

Figure 6. Measured p-p DMA transfer curve (from [4]): (1) Hardware DMA to local CPU memory, (2) Hardware DMA to remote CPU memory (3) Software DMA to local memory

The DMA transfer curves show that:

1. Hardware DMA is far superior to CPU-generated DMA
2. Hardware DMA to remote or local memory is similar
3. A threshold effect is at 128 byte payload for reaching the maximum p-p bandwidth of ca. 246 Mbyte/s .

[2] Thomas Kuhn uses the word paradigm to mean the model that scientists hold about a particular area of knowledge. Kuhn's famous book, *The Structure of Scientific Revolutions*, is his view of the stages through which a science goes in getting from one paradigm to the next.

As a consequence, the following decisions for the construction of a 1 MHz demonstrator system at CERN [8] with dual DMA engines in Readout Units have been taken:

1. aggregate 4 input links into output payloads of 128 byte and use a minimal overhead format [9] in order not to compromise the data payload.
2. implement "hardware eventbuilding", i.e. basically copy a constant number of input fragments to the destination memory . (whether the memory is local or remote does not make much difference).
3. use the PCI memory access mechanism for the memory-memory copy. The DMA engines need to be charged with a physical PCI address which maps to the (remote) destination memory.

TAGnet is required in this demonstrator for organizing the high rate eventbuilding on CPU demand. A first version of a scheduler (hardware + software) was used in order to measure the performance of E.C.D. transfers between Readout Units and a remote CPU.

A. TAGnet slave

A TAGnet slave is implemented in programmable hardware in each Readout Unit, sharing the same FPGAs as the DMA engines. A simplified diagram is shown in Fig. 7

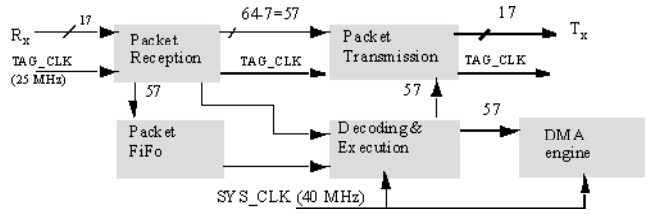

Figure 7. Simplified block diagram of the TAGnet slave logic: The "Paket Reception" stores all incoming TAGs in 64 bit bypass register. "Packet FiFo" only stores TAGs which are directed to the slave. The "Decoding& Execution" takes the desired action. The "DMA-engine" gets loaded with source/destination before execution. The "Packet-Transmission" block copies TAGs back into the TAGnet.

A TAGnet slave stores one full 64 bit TAG and decodes it's TAG class and type. "Consumable, valid" TAGs are either queued in an input FiFo (M-TAGs) or can be prioritized for immediate execution (C-TAGs) of the DMA engine which transfers the selected event buffer.

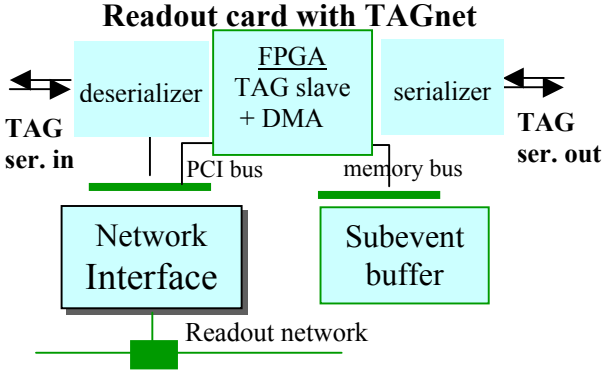

Figure 8. typical Readout-card implementation with TAGnet

The effect of M-TAGs depends on the application, for test purposes however M-TAGs may be used like C-TAGs.

B. TAGnet scheduler

The Scheduler consists of a.) hardware: FPGA logic in PCI card [6] b.) high performance software [1] in particular a C-TAG PCI driver, an M-TAG I/O unit and error handling.

IV. FIRST TAGNET TESTS

TAGnet may be used for E.C.D. in any DAQ or Trigger system. However it has been particularly designed [10] for high rate trigger farms where hardware eventbuilding via shared memory can be applied. In such systems, DMA engines in the RU can "write-through" to the CPU memory. Events get auto-closed when a fixed number of subframes have arrived via the E.C.D. transfers. Initial tests with "1-bit TAGnet" and Readout Unit "Mockups" at the VELO cluster at KIP indicate that TAGnet-generated DMA rates should reach 1.9 MHz for payloads of 128 byte [4].

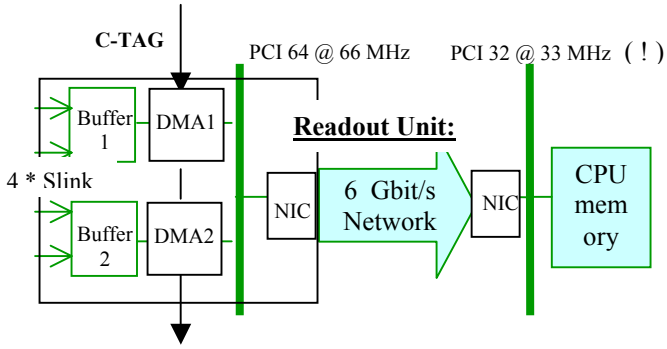

Figure 9. Test set-up for E.C.D. to shared memory

The test set-up is depicted in Fig. 9. The Readout Unit (RU) consists of two subevent buffers, each of which gets filled by 2 Slinks and one DMA engine serves each. The DMAs share a common 64-bit PCI bus with the network interface (NIC). The destination memory of the CPU has been exported though an embedded processor on the RU (not shown here) [7], hence the DMA engines can directly write, via PCI, to the destination CPU memory.

A. Payload aggregation on RUs

The aggregation of payloads from 4 links (each carrying 30-40 byte of Si-strip cluster data) into one output packet of at least 128 byte payload, the PCI write combining protocol has been tested: 64 byte sub-event payloads from both buffers are written in succession to a contiguous PCI buffer of the network interface (NIC) (Fig. 10). The expected result is a single packet with 128-byte payload on the network. Measured on the 6 Gbit/s network link, the network packet starts ca. 1us after the start of the first DMA burst on the 64 bit@ 66 MHz PCI bus.

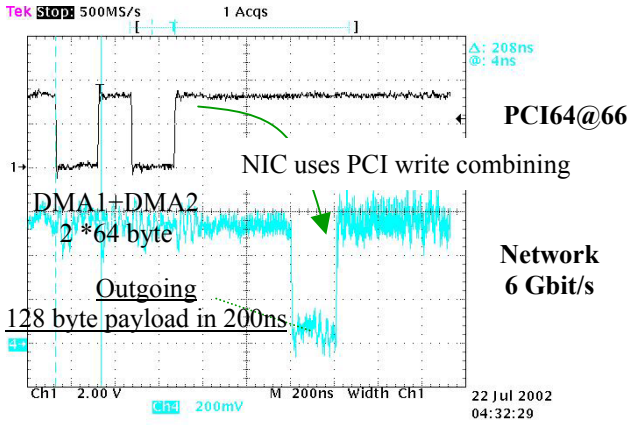

Figure 10. Write combining of two 64-byte DMA payloads via the PCI protocol in the Readout Units. After a latency of ca. 1us a 128-byte payload network packet starts and occupies ca. 200 ns of the 6 Gbit/s link.

A. Experimental TAGnet Scheduler

As scheduler we used the programmable PCI-FLIC [6], a 64-bit PCI card with FPGA, SDRAM and Slink mezzanine connectors. A dedicated TAGnet mezzanine card (Fig. 11) was developed, based on an LVDS Slink card [3].

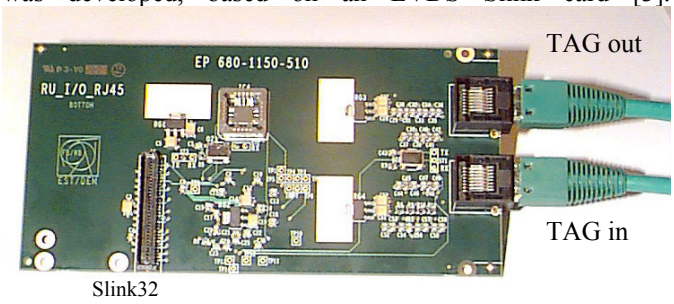

Figure 11. TAGnet mezzanine card

B. Scheduler software performance

A special PCI driver module was developed under Linux that collects CPU requests in the shared memory farm and copies up to 24 CPU requests per 32-bit PCI word to the Scheduler PCI module. On an "old" PC with 32 bit PCI bus, a 32 bit word transfer loop was measured as 2.3 microseconds, hence on a new PC with 64 bit@66MHz PCI bus, a factor 4 is expected, hence a worst case performance of 1.8 MHz per single CPU request can be extrapolated. A detailed discussion of this work is found in reference [1].

C. First E.C.D. Measurements

The succession of individual 128 byte DMAs of Fig.10 from several RU's within a TAGnet ring has been emulated in a single RU that was available at the time of writing. The result (Fig. 12) shows that with a convenient network duty cycle of 40% payload and 60% idle, 128 byte payloads packets are transmitted at an interval of 500 ns (= 2 MHz) corresponding to a p-p transfer bandwidth of 256 Mbyte/s. Another measurement shows that the PCI->PCI latency Readout Unit–>PC for ECD transfers corresponds to ca. 1.8 us + network cable latency.

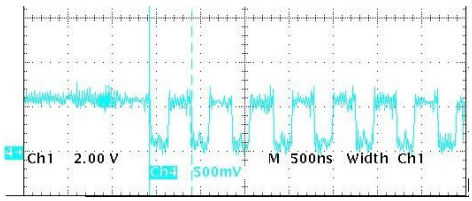

Figure 12. Measured 2 MHz E.C.DMA transfer of 128 byte payloads from Readout Units to a destination CPU memory

V. Summary

TAGnet has been prototyped as a new protocol for event-coherent DMA transfers between DMA-based Readout Units and CPUs in a trigger farm. The scheduler, under control of a host CPU, organizes the proper timing and source-destination buffer allocation for the DMA engines. TAGs exist both as high-rate C-TAGs or low-priority M-TAGs, the latter providing online communication between the scheduler and the DMA modules. In our first tests it has been demonstrated that C-TAG creation on CPU demand can work in shared memory clusters well beyond 1 MHz. The aggregation of low-payload links into network frames of 128 byte payload works well above 2 MHz at a load of 40 % for a 6 Gbit/s network

VI. References

[1] *Logiciel de haute performance pour TAGnet,* Sebastien Gonzales, rapport de stage 3eme annee, Institut Superieur de Modelisation et de leurs Applications (ISIMA) Aubiere France http://daltmann.home.cern.ch/daltmann/sheduler.htm

[2] *Installation of SCI-PCI adapters via an embedded CPU on Readout Units,* D.Altmann, H.Muller, LHCB Note TRIG 2002-xxxx (to appear)

[3] *LVDS link cards (SLINK) for multiplexers, VELO trigger and other applications in LHCb*, F.Bal, H.Muller, LHCb DAQ 2002-005 http://hmuller.home.cern.ch/hmuller/RU/Links/Slinknote.pdf

[4] *Architecture and Prototype of a Real-Time Processor Farm Running at 1 MHz,* Inaugural doctoral thesis, Alexander Walsch, Sept 2002, KIP Heidelberg (to appear)

[5] *Study and Design of the Readout Unit Module for the LHCb Experiment*, Doctoral Thesis, Jose Francisco Toledo, University of Valancia, Gandia, 2001 http://hmuller.home.cern.ch/hmuller/RU/curro_thesis.pdf

[6] *A plug & play approach to Data Acquisition,* NSS conference 2001, San Diego, J.Toledo, H.Muller, J.Buytaert, F.Bal, A.Guirao, IEEE Transactions on Nucl. Science, Volume 49, no. 3, June 2002

[7] *Design and Use of a PPMC Processor as Shared-Memory SCI Node, D.Altmann, A.Guirao, H.Muller, J.Toledo*, presented at the 8th Workshop on Electronics for LHC Experiments, Sept. 2002 Colmar, France

[8] *HS Project, horizontal slice of the Level-1 VELO Trigger*, Hans Muller, Filipe Vinci dos Santos, Angel Guirao, Sebastien Gonzalv, LHCb TRIG note 2002-xxxx (to appear) http://hmuller.home.cern.ch/hmuller/RU/L1Velo/HorizontalSlice/HSproject.pdf

[9] *STF Subevent Transport Formats*, H.Muller, LHCb Note, 2002-xxx TRIG to appear http://hmuller.home.cern.ch/hmuller/RU/STFormat/STF.pdf

[10] *TAGnet, 1 MHz eventbuilding protocol for the LHCb VELO trigger,* LHCb notre TRIG 2002-xxxx to appear http://hmuller.home.cern.ch/hmuller/~HMULLER/DOCS/Colmar/TAGnetNote.pdf