Real-time FPGA design for the L0-trigger of the RICH detector of the NA62 experiment at CERN SPS

2017 JINST 12 C01023

(http://iopscience.iop.org/1748-0221/12/01/C01023)

Published by IOP Publishing for Sissa Medialab

Received: *October 28, 2016*
Accepted: *December 20, 2016*
Published: *January 9, 2017*

Topical Workshop on Electronics for Particle Physics,
26–30 September 2016,
Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

Real-time FPGA design for the L0-trigger of the RICH detector of the NA62 experiment at CERN SPS

M. Barbanera[a,c,1] **and F. Gonnella**[b,c]

[a] *Università di Pisa e INFN (IT),*
3, Largo B. Pontecorvo, Pisa, Italy
[b] *School of Physics and Astronomy, University of Birmingham,*
Edgbaston Birmingham B15 2TT, United Kingdom
[c] *CERN, European Organization for Nuclear Research,*
CH-1211 Geneva 23, Switzerland

E-mail: mattia.barbanera@cern.ch

Abstract: The NA62 experiment aims at measuring rare kaon decays, in order to precisely test the standard model. The RICH (Ring Imaging CHerenkov) detector of the experiment is instrumental in charged-particle identification and in measurement of their crossing time, with a resolution better than 100 ps. Here we describe the design of the Level-0 trigger system for the RICH, which provides a precise time reference by counting the input hit multiplicity within programmable fine-time windows. Since the design does not use spatial information and stands the maximum input rate of TDC-based NA62 systems, it can be deployed also in other subdetectors.

Keywords: Cherenkov detectors; Trigger algorithms; Trigger concepts and systems (hardware and software); Trigger detectors

[1]Corresponding author.

doi:10.1088/1748-0221/12/01/C01023

Contents

1 Introduction

A real time FPGA trigger design for the RICH detector [1] of the NA62 experiment [2] is described, that provides precise time reference for the Level-0 trigger system. The NA62 experiment at the CERN Super Proton Synchrotron (SPS) aims to measure the branching ratio of the very rare kaon decay $K^+ \to \pi^+ \nu\bar{\nu}$, in order to precisely test the standard model by collecting O(100) events with 10% background. The RICH detector of the experiment is fundamental in charged particles identification and in measuring their crossing time, with a resolution better than 100 ps. The development of a high performance RICH L0-trigger primitive-generating firmware addresses the issue of the high rate in input to the trigger and data acquisition board, the TEL62. The aim of the algorithm is to count the input multiplicity within programmable time windows, providing a time of the trigger which is the averaged values of the input times. Detector-specific features are not used, so that the firmware can be deployed in all the L0 detectors. The design foresees also the use of InterTEL boards [3], which interconnect more TEL62s in a daisy chain link. These boards are part of the next upgrade for the L0-trigger system of the experiment.

2 Trigger and data acquisition system

The NA62 apparatus [4] consists of 16 subdetectors along the beam line, as in figure 1. Since the amount of data produced by the subdetectors is huge, a three-level Trigger and Data AcQuisition (TDAQ) system was designed, in order to reduce the rate of stored events. The lowest level, *Level-0 trigger* (L0), is implemented in hardware and selects 10% of the 10 MHz input rate. The other two levels are software based and further reduce the rate of events saved to disk from 1 MHz down to 10 KHz. As in figure 2, the TDAQ system foresees dedicated Front-End Electronics (FEE) for each subdetector, to send data to the readout boards. Many NA62 subdetectors share the same readout board, called TEL62, an upgraded version of the LHCb TELL1. All the TEL62s are then connected to the PC farm, while some of them are also connected to the *L0 Trigger Processor* (L0TP). L0

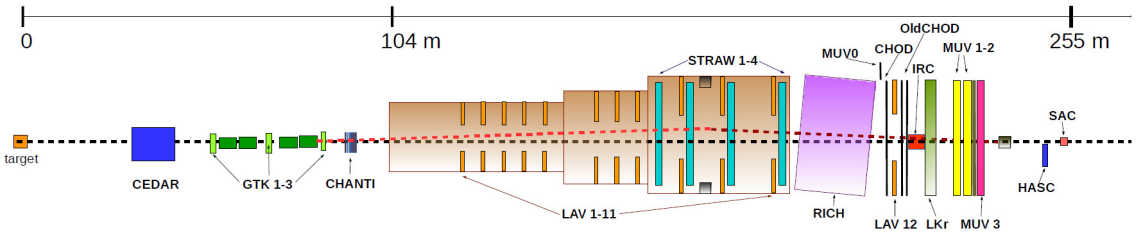

Figure 1. The NA62 apparatus with its 16 subdetectors along the beamline.

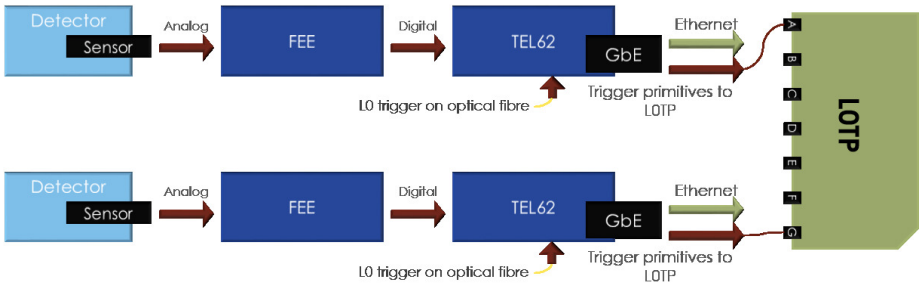

Figure 2. NA62 Trigger and Data AcQuisition system (TDAQ).

triggers are based on small-sized information, called *trigger primitives*, which are sent to the L0TP by the TEL62s and contain time information and an ID that specifies the type of sent event. When certain predefined conditions on the primitive IDs are satisfied, the L0TP dispatches a trigger to the readout boards, that send the data to the PC farm where the L1 and L2 triggers will run.

Figure 3 shows a block diagram of the TEL62 boards, which are equipped with 5 Altera Stratix-III [5] FPGAs each, divided into two subgroups: there are 4 *Pre-Processing* (PP) FPGAs receiving data from 4 daughter cards (Time-To-Digital Converter Boards, TDCBs) of the FEE of the specific detector. The 4 PP FPGAs, then, send their output to a *Sync-Link* (SL) FPGA. The SL combines the inputs into primitives and dispatches them to the L0TP. The latter, in turn, evaluates the L0 trigger directives, returning the indices of events that pass all the requirements. The SL then dispatches these events to the PC farm as Ethernet packet. Since TEL62 boards are used both for trigger and data acquisition, the logic occupation of the trigger part must be kept as low as possible. The *InterTEL* boards implement two 16-bit buses between two or more SLs. If the interconnected SLs are two, InterTELs implements a full-duplex bus, otherwise SLs are connected in a daisy chain way. In the latter case, only the last SL of the chain sends primitives to the L0TP, while the others only send data to the PC farm and to the successive SL in the chain.

3 RICH L0 firmware

This contribution describes the L0 firmware developed for the RICH subdetector. This piece of firmware was designed with the guideline of keeping it as general as possible, so that it can easily be ported to other primitive-generating subdetectors, and that it would be possible to use it with the foreseen InterTEL boards. The aim of the L0 firmware is to assemble time clusters of hits belonging

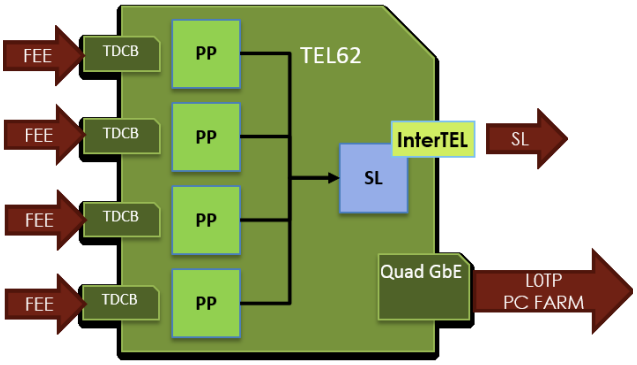

Figure 3. Block diagram of the TEL62 board.

to the same event and detector. Clusters are given a precise time reference and passed on to the next trigger levels, based on the multiplicity of hits forming the cluster. In case of the RICH detector, events consist in Cherenkov circles originated by charged particles but no geometrical information is used by the firmware, in order to maintain the firmware generality.

The logic occupation of the TEL62 FPGAs is already high due to the readout modules. In order to fit the Stratix-III, then, the firmware design should use as few logic resources as possible, use the clock frequency provided by the TEL62, and avoid timing violations. Attention was paid also to delay in primitives production, which has to be as stable as possible.

3.1 RICH data format

In order to maintain generality and optimise the design efficiency, a common 32-bit data-format called RICH format is used. Each module implemented in the firmware must be able to use the same format as input and output, in order to be able to reuse modules with little modification to serve different purposes. As shown in figure 4, the RICH format foresees two types of 32-bit words identified by a word ID: time stamps and data words. The ID is split into two 2-bit words to allow the split of one 32-bit word into two 16-bit word. This will be mandatory when InterTEL boards will be used, since they connect SLs with 16-bit buses. The time stamp is a 28-bit word with 400 ns resolution, containing the MSBs of the time of the experiment, while data words, on the other hand, specify the time of the clusters. Data words contain: cluster seed fine time (T), composed of 12 bits representing the time of the cluster in units of 100 ps; hit multiplicity (N), that counts in 8 bits the number of hits belonging to that cluster; and 8 bits of Cluster Time Sum (S), which is the sum

31 30	29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14	13 12	11 10 9 8	7 6 5 4 3 2 1 0
T.S. 1/2	Time stamp (27:14) 400 ns		**T.S. 2/2**	Time stamp (13:0) 400 ns		
Data 1 1/2	N_{hits} (7:2)	Cluster Time Sum (7:0) (signed) LSB = 100 ps	**Data 1 2/2**	N_{hits} (1:0)	Fine Time (11:0) LSB = 100 ps (up to 400 ns)	
Data 2 1/2	N_{hits} (7:2)	Cluster Time Sum (7:0) (signed) LSB = 100 ps	**Data 2 2/2**	N_{hits} (1:0)	Fine Time (11:0) LSB = 100 ps (up to 400 ns)	

Figure 4. RICH data format.

of the differences between cluster seed time and the hits belonging to that cluster. The S is used to efficiently compute the weighted average of the time of the hits belonging to a cluster and, being it signed, its value should remain small even if the cluster is made of a significant number of hits. Since the LSB of the time stamps has 400 ns resolution, and since the basic time divisions of the experiment are time frames of 6.4μs, there will be 16 time stamps in each frame. It can happen that there are no data corresponding to a given time stamp: in that case, the time stamp is followed by a fake cluster with $N = 0$, $S = 0$ and the maximum possible T. These data are called *speed-data* and are used to decrease the latency of some modules in the firmware.

3.2 L0 firmware: PP and SL

Figure 5 shows the block diagrams for PP and SL firmwares. All the FIFOs between the modules are used to compensate the latency of the modules themselves and possible delays coming from the communication with L0TP. Data coming from the FEE arrives at the PP firmware sorted in bunches of 25 ns, i.e. the time of the hits is sorted from the MSB down to the 9$^{\text{th}}$ bit, while the 8 LSBs are not sorted. As soon as the hits enter the PP, they are converted to RICH format: proper time stamps are sent and the hits are formatted into clusters with $N = 1$, $S = 0$ and T the time of the hit. Consistency of the RICH format is vital for the proper behaviour of the whole firmware: for that purpose Data Validator (DV) modules check it, together with consistency of the data flow, i.e. if after a time stamp there is always at least one cluster, and whether time stamps and words are sorted at 25 ns. If a check is not satisfied, the module raises an error flag, that is sent to the PC farm together with the data. There are 2 DVs in the PP and 2 in the SL, with similar logical positions: one checks the input of the FPGA while the other checks the output of the clustering module.

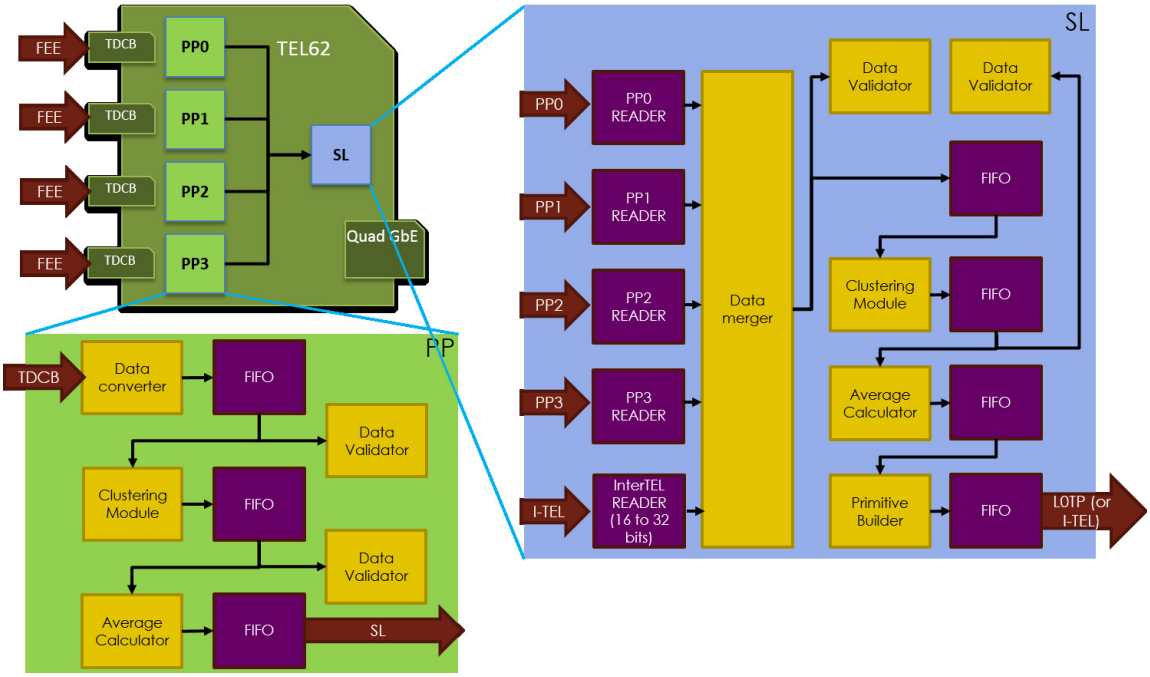

Figure 5. PP and SL firmwares block diagrams.

The goal of the Data Merger (DM) is to read data coming from the 4 PPs at a rate of 1 word per clock cycle and merge them into a single data flux, preserving the 25 ns sorting. Time stamp and special words must not be repeated. To achieve that, the module is purely combinatorial and divided in 3 identical submodules: a first level merges the data from PP0 and PP1 and the data from PP2 and PP3, while a second level combines the merged data. The output of the first layer is connected to two FIFOs, read by the second layer. This was done in order to reduce the length of the paths, as well as comparing only the bits down to 25 ns, disregarding the 8 LSBs. When InterTEL boards will be used, the DM will have another input connected after the second layer: this will be the only difference between the two firmwares. DM submodules start working only if both the input FIFOs are not empty, in order to be able to compare the input words. This could lead to starvation of data contained in one of the two FIFOs, e.g. when one of them is filled faster than the other one. In order to avoid starvation, the *speed-data* words are used: as they contain the maximum possible value of T, these words will always give the priority to the other FIFO, until the *speed-data* of the other PP will reach the merger input. The Average Calculator (AC) computes the mean value of the time of the cluster. In order to do so, all the fields of the cluster word are used: T is increased by the ratio S/N and S is reset to zero. Finally The Primitive Builder (PB) composes the trigger primitives in the standard NA62 Multi Trigger Packet (MTP) format.

4 Clustering module

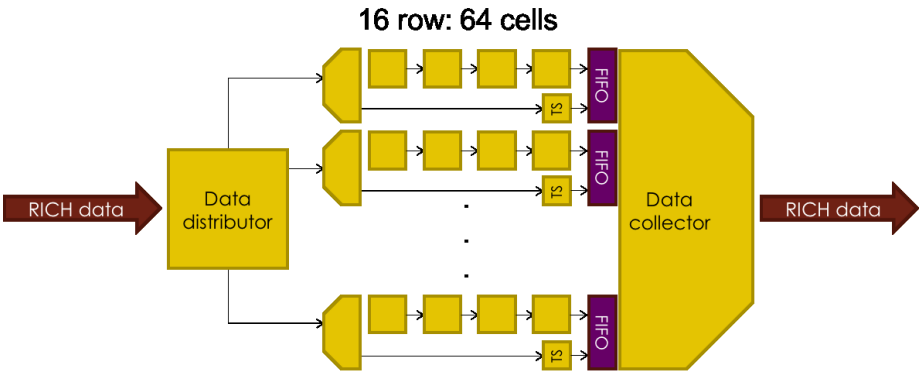

Figure 6. Schematic block of the clustering module.

The RICH Clustering Module (CM) is the most important block of this firmware. It creates clusters of hits or of other clusters if they are closer than a certain time value. As said above, the input is sorted down to 25 ns, in order to save logical cells and to reduce path lengths. The comparison between clusters is performed only on the T value (the time seed), while the other fields are only taken into account once the merging decision has already been taken. The CM is divided into sub-blocks as shown in figure 6.

The data distributor (DD) handles incoming data and feeds them into the so-called clustering rows. There are 16 rows, each made of 4 cells and a Time Stamp (TS) register, and taking in input hits belonging to a specific 25-ns frame. The DD rearranges data into 8-bit fine time (with 100-ps LSB) and 32-bit time stamp (with 25-ns LSB), which are sent directly to the TS register. In this

way, all the operations in the row are done on 9 bits only: 8 bit plus one to handle adjacent time frames. In order to handle border effects, the DD is designed to send hits in two rows at a time. In this way, hits belonging to clusters split into two adjacent time frames can be sent to the same row and merged together. To this aim, (see figure 7) the DD stores the value of the greatest time received in a given 25-ns frame into a register, called *current biggest*. When it switches to the next row, it compares every time to the *previous biggest* register. When a time hit is close enough to *previous biggest*, with respect to the clustering time, the DD sends it to the previous row instead of to the current one.

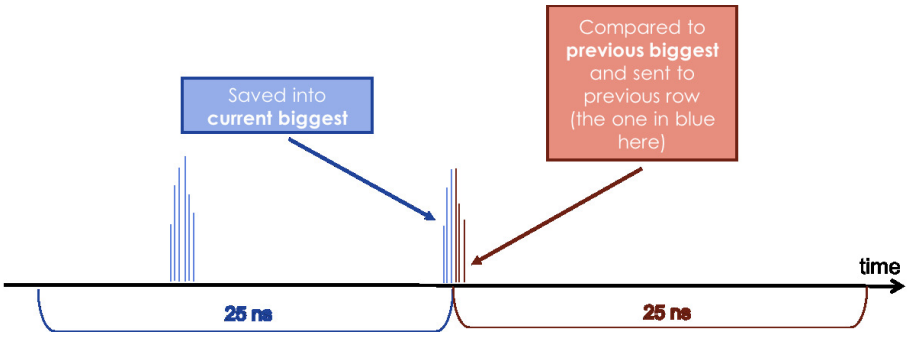

Figure 7. Data Distributor behaviour.

The comparison of the Ts and the merging of the clusters are performed by the cell. Each cell stores the first T received that will become the new cluster seed of a cluster. If the T value of the next clusters received matches the stored T within a programmable time window, the cell stores the cluster values for merging. N is increased by the input N and S is updated with N in input multiplied by the difference of the reference T and the input T. An Altera MegaWizard multiplier handles the multiplication and each cell is divided in two parts connected by a FIFO. The first one is the comparison part, while the other is the multiplier and correction one. If the incoming T does not match the stored value, the cluster is sent to the next cell that performs the same operations. By construction, the clustering module cannot handle more than 4 clusters per 25-ns frame. Clusters after the fourth are discarded, if any, and an error bit is raised. Each cell has an internal position field that is appended to the output cluster when the flush-mode is enabled. This field increases when a cluster with time bigger than the cluster seed passes through the cell. Clusters that cannot fit in the row, i.e. after the 4^{th} one, are subtracted from each position field, in order to not be taken into account.

When the n^{th} row is filled, the $(n-2)^{\text{th}}$ is ready to be read out. This is done by *flushing* the row. When the flush-mode is enabled, the cells act as a shift register, giving as output all the cluster times and number of events per cluster. The data are then sent to an output FIFO at the output of the row. The throughput of the clustering module, like every other module in the RICH firmware, is kept at 1 word per clock cycle. Since the emptying operations on a row should be finished before the row needs to be written again and taking into account the filling time of a row, 16 clustering rows are needed in order to process all data. Moreover, two rows are filled at the same time by the DD in order to take care of border effect, as previously explained. Once the second row is completed, the first one can be read out. 16 rows are enough to compensate for the latency of the cell being given by: $2d + f + m$ where d is the depth of the row, f is the delay of the FIFO and m is the latency of the multiplier. In our case $2 \cdot 4 + 3 + 3 = 14 < 16$, so there will always be a free row to fill.

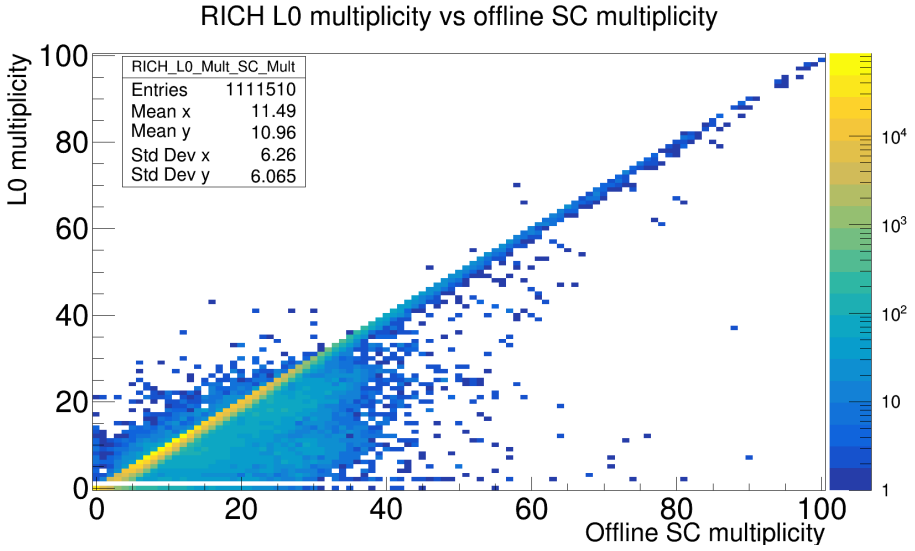

Figure 8. Correlation between the multiplicity computed in the off-line analysis and by the RICH L0 firmware.

Finally, the Data Collector (DC) retrieves data from the FIFOs of the rows and converts them into RICH format. The DC module is divided into four independent parts. The first one reads the clusters coming from the rows. In order to read one word per clock cycle, the collector has to react to the empty signals from two consecutive rows. The second part sorts the clusters of a row exploiting the position field appended to the clusters themselves. To stand the full rate, it uses eight RAM blocks with the same address as the position of the cluster. After the sorter, clusters with multiplicity outside a customisable range are discarded. This permits to reduce the noise coming from events with too low or too high multiplicity. The fourth and last part of the DC reads the sorted and not discarded clusters and converts them to the RICH format.

5 Tests and performance

After simulation, the design was implemented and used during the whole 2016 NA62 physics run. In order to check the behaviour of the RICH firmware, data triggered by other subdetectors have been analysed and compared to the output of the firmware itself. The carried-out analysis counted the number of events in the RICH, computing their multiplicity and their time. The plot in figure 8 shows the correlation between the multiplicity computed in the analysis and by the RICH L0 firmware.

The region with firmware multiplicity equal to 0 and the one with analysis multiplicity equal to 0 can be used to evaluate the inefficiency and the probability of false positive can be computed. The inefficiency is computed by adding all the events that have firmware multiplicity 0 but non-zero analysis multiplicity, for a total count that leads to a inefficiency of the firmware of 1.24%. The probability of false positive is computed by adding all the events that have analysis multiplicity 0 and non-zero firmware multiplicity, and has the value of 0.005%. The left plot in figure 9 shows the time contained in a trigger primitive versus the time of production. One can clearly see that the firmware latency is very stable and lies between 2 and 3 frames of 6.4μs.

2017 JINST 12 C01023

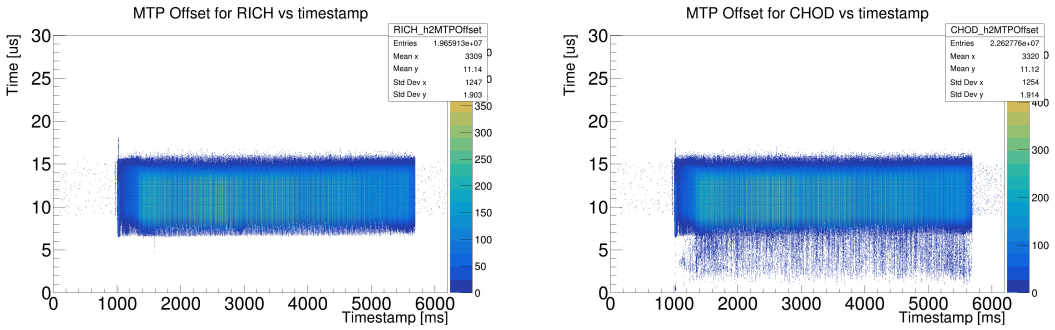

Figure 9. Delay of the production of primitives for RICH (on the left) and CHOD (on the right) detectors.

Thanks to its generality, the RICH firmware was deployed also for the L0 of the CHOD [6] detector. An analysis similar to the one concerning the RICH has been carried out, giving similar results. The only sizeable difference is the delay in primitive production, which is reported in the right plot of figure 9. The latency is lower than for the RICH detector, and oscillates between one and two frames of 6.4μs. The reason for that, is that the CHOD rate is about 15% higher that the RICH and this reduces the latency of the DM in the SL and of the DD which produce an output only when an input word is received.

6 Conclusion

A real time L0-trigger firmware for the RICH detector has been developed. It has been deployed in the NA62 TDAQ system and it works with an efficiency of 98.76%. Moreover, it can stand the full RICH data output rate, with a delay in the production of primitives that is very stable and significantly lower than the design upper limit. Moreover, with a higher input rate the delay of primitive production decreases, leading to a more reliable behaviour of the firmware. The design doesn't take into account any detector-related information, and thanks to this it can be used for all the detectors contributing to the L0 trigger of NA62. It was already implemented for the L0 trigger of the CHOD detector with the same performance as for the RICH. The firmware is ready for use with the InterTEL boards, which are not yet used in the experiment but are foreseen for the upcoming upgrade.

References

[1] G. Anzivino et al., *Construction and test of a RICH prototype for the NA62 experiment*, *Nucl. Instrum. Meth.* **A 593** (2008) 314.

[2] NA62 collaboration, *Proposal to Measure the Rare Decay $K^+ \to \pi^+ \nu\bar{\nu}$ at the CERN SPS*, CERN-SPSC-2005-013 (2005).

[3] M. Barbanera et al., *InterTEL assessment procedures for the LKr calorimeter of the NA62 experiment*, NA62-2015-03 (2015).

[4] NA62 collaboration, *Technical Design Document*, NA62-10-07 (2010).

[5] Altera Corporation, *Stratix III Device Handbook, Volume 1*, SIII5V1-2.2 (2011).

[6] NA48 collaboration, V. Fanti et al., *The Beam and detector for the NA48 neutral kaon CP-violations experiment at CERN*, *Nucl. Instrum. Meth.* **A 574** (2007) 433.

2017 JINST 12 C01023