GPUs for real-time processing in HEP trigger systems

GAP Collaboration

R Ammendola[1], A Biagioni[1], L Deri[2], M Fiorini[3], O Frezza[1], G Lamanna[4], F Lo Cicero[1], A Lonardo[1], A Messina[1,5], M Sozzi[2,4], F Pantaleo[2,4], PS Paolucci[1], D Rossetti[1], F Simula[1], L Tosoratto[1] and P Vicini[1]

[1] INFN Roma,P.le A.Moro,2 - 00185 Roma - Italy
[2] Pisa University, Largo B.Pontecorvo,3 - 56127 Pisa - Italy
[3] Ferrara University, Via Saragat,1 - 44122 Ferrara - Italy
[4] INFN Pisa, laro B.Pontecorvo,3 - 56127 Pisa - Italy
[5] Roma University, P.le A.Moro,2 - 00185 Roma - Italy and CERN - Geneve - Switzerland

E-mail: Gianluca.Lamanna@cern.ch

Abstract. We describe a pilot project (GAP - GPU Application Project) for the use of GPUs (Graphics processing units) for online triggering applications in High Energy Physics experiments. Two major trends can be identified in the development of trigger and DAQ systems for particle physics experiments: the massive use of general-purpose commodity systems such as commercial multicore PC farms for data acquisition, and the reduction of trigger levels implemented in hardware, towards a fully software data selection system ("trigger-less"). The innovative approach presented here aims at exploiting the parallel computing power of commercial GPUs to perform fast computations in software not only in high level trigger levels but also in early trigger stages. General-purpose computing on GPUs is emerging as a new paradigm in several fields of science, although so far applications have been tailored to the specific strengths of such devices as accelerators in offline computation. With the steady reduction of GPU latencies, and the increase in link and memory throughputs, the use of such devices for real-time applications in high energy physics data acquisition and trigger systems is becoming relevant. We discuss in detail the use of online parallel computing on GPUs for synchronous low-level triggers with fixed latency. In particular we show preliminary results on a first test in the CERN NA62 experiment. The use of GPUs in high level triggers is also considered, the CERN ATLAS experiment being taken as a case study of possible applications.

1. Introduction

The scientific project described in this paper is based on the use of Graphic Processing Units (GPUs) for scientific computation. In recent years, the use of massively parallel computation is gaining ground in several fields of scientific research, in order to overcome some shortcomings of the present microprocessor technology. Modern super-computers are structured as arrays of several distributed computing units sharing data through dedicated high-bandwidth connections. Even processors in commodity personal computers (PCs) contain several computing cores.

Table 1. Characteristics of computing dedicated GPUs.

Video Card	n. of cores	Processing Power (GFLOPS)	Memory Bandwidth (GB/s)
NVIDIA TESLA C1060 (2009)	240	933	102.4
NVIDIA TESLA C2050 (2011)	448	1288	144
NVIDIA TESLA K20 (2013)	2496	3520	208
NVIDIA GFORCE GTX680 (2012)	1536	3090	192
AMD RADEON 5870 (2010)	1600	2720	153.6
AMD RADEON 7970 (2012)	2048	4096	288

GPUs, developed to speed up graphics-related computing tasks, are naturally organized around a parallel architecture. Because of such architecture, and the fact that most of the chip resources are devoted to computation - differently from a normal processor (CPU) in which a large fraction of those is required for other functions such as caching and handling of peripherals - allow to achieve a large computing power using a limited amount of space and power. GPUs are recently used also in applications very different from those for which graphic processors were originally conceived. So-called GPGPU (General Purpose computing on GPU) is nowadays common in several fields of scientific research requiring large computing power, such as hydrodinamical modeling, complex system simulation, image reconstruction for medical diagnostics. Major GPU vendors are strongly favouring this trend by exposing the raw computing power of graphic processors to users within a general-purpose paradigm. The two most widespread development frameworks and languages to program GPUs are $CUDA^{®}$ by NVIDIA® and OpenCL by the Kronos consortium. While the first one is devoted to the use of NVIDIA® devices, the latter was developed with the support of several vendors who want to allow a common approach to the programming of multi-core devices (both GPUs and CPUs).

Mainly due to the technological advances required by the massive computer game and video editing markets, current video processors provide computing power, in terms of operations per second (FLOPs) which are vastly larger than those offered by CPUs.

Table 1 shows the characteristics of some of the most recent GPUs on the market which are particularly suited to scientific computation. Besides the large number of cores, these processors have a large bandwidth to dedicated on-board memory, which is a key factor to allow a large computing throughput. On the other hand, one of the limitations for the use of GPUs in scientific computation is due to the need to access such devices through a PC bus which - despite the significant bandwidth provided today by the new PCI-express gen 3, up to $\sim 16GB/s$ - is often the most important bottleneck for real-time applications.

2. Real-time triggering in HEP

In High Energy Physics (HEP) experiments, the trigger system has a central role, which often determines the significance and the discovery potential. The trigger system must decide, usually based on limited and partial information, whether the physics event observed in a detector is interesting. The use of such real-time selection allows to optimize the usage of the finite data acquisition bandwidth, and to limit the amount of disk space required to store the data for the subsequent offline analysis stage. Online selection is performed by arranging the trigger system in a cascaded set of computation levels. The first, or the lowest, trigger level is usually implemented in hardware, often based on custom electronics tailored to the experiment and normally distinct from the data acquisition system. Nowadays these systems are often implemented using programmable logic (FPGA), which offer quite some flexibility and reconfiguration capability. In some cases, however, these kind of computations are performed on dedicated VLSI chips developed for the specific project [2], sacrificing flexibility and scalability for speed.

The following, "higher", trigger levels are today commonly implemented in software, using dedicated farms of commodity PCs. The use of commercial technology, computers and networks, allows to have very flexible and scalable trigger systems. The event rate which these trigger levels must handle has been already largely reduced by the upstream hardware trigger levels, allowing for complex decisions, based on more complete event informations, just by increasing the computing resources.

The time required to complete an event selection is not a big issue in higher trigger levels. The large amount of inexpensive memory (RAM) in today's PCs allow to store events for times of the order of seconds, thus relaxing the latency requirements for the algorithms. This is usually not the case for the lowest hardware trigger levels, since the memory buffers in the readout electronic boards, where incoming events are temporary stored before the trigger decision, often limit the maximum latency to tens of microseconds.

A better approach would be therefore one in which all data are brought directly to PCs, eliminating the hardware trigger levels. However, present HEP experiments must handle data rates so large to make such "triggerless" architecture unrealistic from a resource point of view: the bandwidth of general-purpose network systems is often insufficient to bring all data on PCs, and the raw computing power of individual PCs does not match, for complex computations, the required elaboration rates, to keep the number of PCs to affordable and manageable levels.

3. The use of GPUs in low-level triggers

Even if a fully "triggerless" approach is at the moment unrealistic, graphic processors represent a viable alternative to fill the gap between a multi-level trigger system with a hardware lowest level and a system which does not require any real-time hardware processing on reduced event information. Indeed GPUs do provide a large raw computing power on a single device, thus allowing to take complex decisions with a speed which can match significant event rates.

In a standard multi-level trigger architecture GPUs can be easily exploited in the higher software levels: being powerful computing devices, they can boost the capabilities of the processing system, thus allowing more complex computations to be performed without increasing the scale of the system itself. This is the case of the use of GPUs in the software trigger level (LVL2) of the CERN ATLAS experiment, as discussed later on.

The use of GPUs in lowest trigger levels, on the other hand, requires a careful assessment of their real-time performances. A low total processing latency and its stability in time (on the scales of "hard" real-time) are indeed requirements which are not of paramount importance in the applications for which GPUs have been originally developed. As mentioned above, the issue is related to the fact that in common usage of GPUs as graphics co-processors in computers, data is transferred to the GPU - and results are transferred back - through the PCI-express computer bus. Moreover, in order to better exploit the parallel GPU architecture, computing cores must be saturated, thus requiring the computation on a significant number of events after a buffering stage.

Considering a system in which data transfer occurs through a standard ethernet network, figure 1 shows the steps required for GPU computing, in a standard approach. Data from the detector reaches the Network Interface Card (NIC) which copies them periodically on a dedicated area in the PC RAM, from which they are copied to the user space memory where applications can process them. Here events are possibly buffered in order to prepare a sufficient data load for the following stages, and they are copied to GPU memory through the PCI express bus. The host (the PC on which the GPU card is plugged) has the task of launching the GPU kernel, which operates on the data. Computation results can be sent back to the host for further

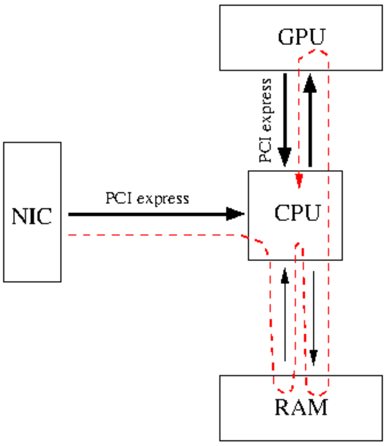

Figure 1. Schematic view of data transport from NIC to GPU

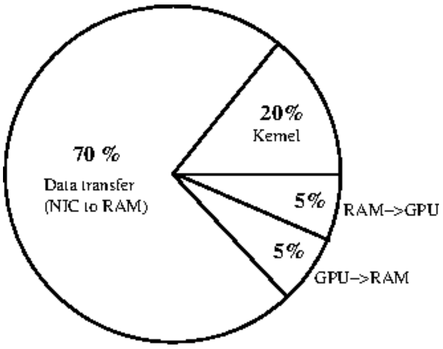

Figure 2. Time budget of data processing on GPU.

processing or distribution to the detectors, to ultimately trigger the reading of the complete data. Figure 2 shows, for a data packet of 1404 bytes, the budget for the processing time for a use case in which the kernel fits the data to a circle which, as discussed later, is a typical example of a computation required to extract significant information from raw detector data. In this example the most important contribution to the total latency is actually due to the data transfer latency from the NIC to the GPU memory. In order to reduce this contribution, in the project described here we studied two approaches: the use of a dedicated NIC device driver with very low latency (PFRING, by NTOP[1]) and the use of a direct data transfer protocol from a custom FPGA-based NIC to the GPU (Nanet); these two approaches will be described in detail in the following. The use of a custom NIC driver allows to perform the low-level trigger processing directly on a standard PC, but the results - in terms of total latency - might be affected by the PC characteristics (mother-board, CPU, etc.) because the data transfer time and its fluctuations are controlled by the host computer. In the second approach, instead, the data transfer is fully controlled by the custom NIC without any participation from the PC, at the price of using a non-standard hardware device.

3.1. Driver DNA-PFRING

Socket transmission on standard drivers does not guarantee the highest packet capture speed, especially if transmission is realized with small-size packets and high rate. As described above, the copy of packets from the NIC to the system buffers and then to the userland, is done by the CPU in two steps. PFRING is a new kind of socket that works in connection with the DNA (Direct NIC Access) driver, both developed by NTOP [1], in order to allow direct copy of packets from the NICs buffers to the memory through DMA (Direct Memory Access). A schematic drawing of the PFRING working principle can be found in Fig.3: using the DNA driver, data transfer is managed by the NICs NPU (Network Processing Unit) processor, through circular buffers that are directly accessible for the applications, and therefore for the copy to the GPUs memory. The scheduler that manages data transfer from the NIC to the GPUs memory has been implemented in a partially preemptive way in parallel streams. This allow to hide data transfer latency by exploiting concurrent copy-execution allows by last generation GPUs. Preliminary tests (Figure 4) show that this scheme improves by a factor higher than 2 the data transfer time and, most importantly, reduces fluctuations to a negligible level.

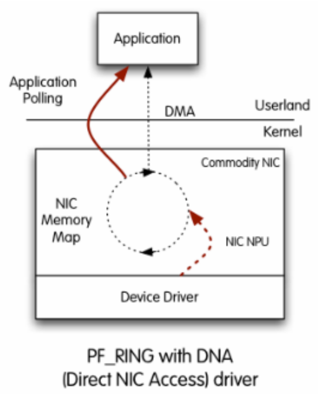

Figure 3. Scheme data processing using DNA driver (*courtesy of NTOP*).

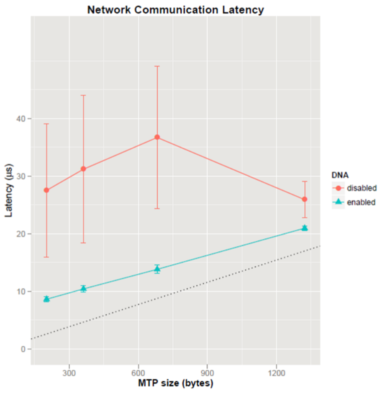

Figure 4. Comparison between standard data transmission and by using PFRING socket.

3.2. Nanet

The NaNet board [4] is a customized version of the APEnet+ [3] NIC designed to be integrated in the GPU-based low level trigger system of the NA62 RICH detector. Adding to the APEnet+ design the logic to manage a standard GbE interface, NaNet is able to exploit the GPUDirect P2P capabilities of NVIDIA Fermi/Kepler GPUs equipping a hosting PC to directly inject into their memory an UDP input data stream from the detector front-end, with rates compatible with the low latency real-time requirements of the trigger system.

In order to render harmless the unavoidable OS jitter effects that usually hinder system response time stability, the main design rule is to partition the system so that the hosting PC CPU can be offloaded from any data communication or computing task, leaving to it only system configuration and GPU kernel launch tasks. Within NaNet, this meant that data communication tasks were entirely offloaded to a dedicated UDP protocol-handling block directly communicating with the P2P logic: this allows a direct (no data coalescing or staging is performed) data transfer with low and predictable latency on the GbE link → GPU data path.

The `UDP OFFLOAD` block comes from an open core module [1] built for a Stratix II 2SGX90 development board. Focus of that design is the unburdening of the `Nios II` soft-core microprocessor onboard the Stratix II from UDP packet management duties by a module that collects data coming from the Avalon Streaming Interface (`Avalon-ST`) of the Altera Triple-Speed Ethernet Megacore (`TSE MAC`) and redirects UDP packets along a hardware processing data path. The `Nios II` subsystem executes the InterNiche TCP/IP stack to setup and tear down UDP packet streams which are processed in hardware at the maximum data rate achievable over the GbE network.

Bringing the open core into the NaNet design required some modifications, first of all the hardware code was upgraded to work on the Stratix IV FPGA family; this upgrade made available the improved performances of an FPGA which is two technology steps ahead in respect to the Stratix II.

The synthesis performed on a Stratix IV achieves the target frequency of 200 MHz (in the current APEnet+ implementation, the `Nios II` subsystem operates at the same frequency).

Current NaNet implementation provides a single 32-bits wide channel; it achieves 6.4 Gbps at the present operating frequency, 6 times greater than what is required for a GbE channel.

Data coming from the single channel of the modified `UDP OFFLOAD` are collected by the `NaNet CTRL`. `NaNet CTRL` is a hardware module in charge of managing the GbE flow by encapsulating

[1] NIOS II UDP Offload Example, `http://www.alterawiki.com/wiki/Nios_II_UDP_Offload_Example`

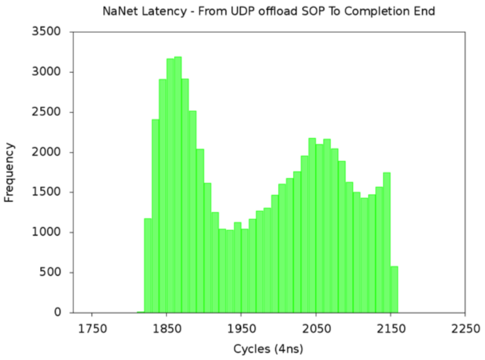

Figure 5. Distribution plot over 60000 samples of a NaNet packet traversal time.

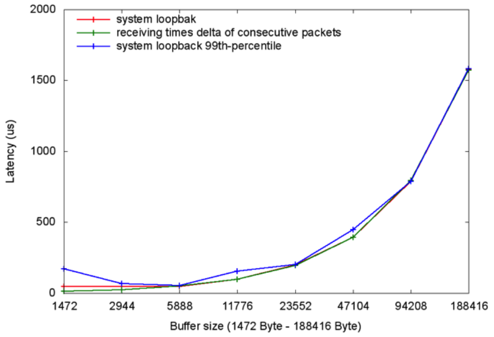

Figure 6. Average and 99-percentile NaNet latency vs. buffer size.

packets in the typical APEnet+ protocol (`Header`, `Payload`, `Footer`).

Benchmarks for latency and bandwidth were carried out. In order to measure the system latency and its fluctuations a “system loopback” configuration was used: connecting one GbE interface of the hosting PC to the NaNet, it was possible able to generate and receive a UDP stream in a single host process, measuring latency as the difference of host processor Time Stamp Counter register at send and receive time of the same UDP packet (see figure 6).

Latency inside the NIC was measured adding 4 cycles counters at different stages of packet processing; their values are stored in a profiling packet footer with a resolution of 4 ns; for a standard 1472 bytes UDP packet, traversal time ranges between 7.3 us and 8.6 us from input of `NaNet CTRL` to the completion signal of the DMA transaction on the PCIe bus (see figure 5).

For the same packet size, saturation of the GbE channel is achieved, with 119.7 MB of sustained bandwidth.

We foresee several improvements on the `NaNet` design:

- Implementing a custom logic dedicated to destination address calculation for the data receiving circular buffer in GPU memory, currently implemented by the `Nios II`, in order to lower latency and its fluctuations, especially at lower receiving buffer sizes.
- Adding a buffering stage to the `NaNet CTRL`, enabling the coalescing of consecutive UDP payload data into a single APEnet+ packet payload, improving bandwidth figure also with small-sized UDP packets.
- Increasing the number of supported GbE channels, to sustain eventually the higher bandwidth demands from the experimental requirements.
- Implementing a 10-GbE data link interface.

3.3. A physics case: NA62

The NA62 particle physics experiment at CERN [6] is considered as a use case for the study of the use of GPUs at the lowest trigger level. The NA62 trigger is organized in three levels: the first level is implemented on the very same FPGA-based boards which perform detector data readout [5], while the following two levels are implemented on PCs. The first hardware level (Level 0, L0) must handle an input event rate of order 10 MHz and, with a rather long maximum latency of 1 ms. It applies a rejection factor around 10, to allow a maximum input rate of 1 MHz to the second trigger level (Level 1, L1) implemented in software. This level, together with the following and last one (Level 2, L2), must reduce the rate to about 10 kHz in order to allow permanent data storage for later offline analysis.

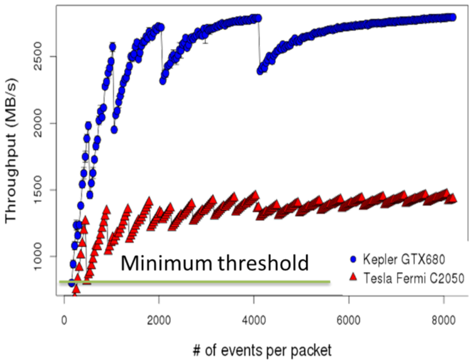

Figure 7. Throughput as a function of number of events for last generation GPUs.

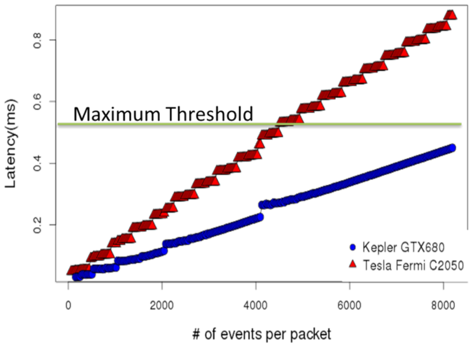

Figure 8. Total latency (including data transfer and computing).

In the standard implementation of L0, the trigger primitives contributing to the building of the final trigger decision are computed on the readout board FPGAs, and are very simple conditions, mostly based on the event hit pattern or multiplicity. The use of processors, such as GPUs as discussed in this paper, would rather allow to build more complex physics-related trigger primitives, such as energy or direction of the final state particles in the detectors.

As a first use case, we studied the possibility of reconstructing, in GPUs, the ring-shaped hit patterns in a RICH Cerenkov detector. Such detector, described in ref.[7], can provide a measurement of the velocity and direction of charged particles (such as muons and pions) which traverse it, thus contributing to the computation of other physical quantities such as the decay vertex of the K^+ and the missing mass. The use of such information allows to implement highly selective trigger algorithms also for other interesting decay modes.

We studied different ring reconstruction algorithms in order to assess which ones are the most suited to a GPU parallel implementation.

As described in ref.[8] the “math” algorithm, based on a simple coordinate transformation of the hits which reduces the problem to a least square procedure, was found to be the best one in terms of computing throughput (for single rings). This algorithm was implemented and tested on different GPUs, such as the NVIDIA Tesla C1060, Tesla C2050 and GeForce GTX680 (in increasing order of processing core generation). The computing performance of the C2050 and GTX680 proved to be a factor 4 and 8 higher than that of the C1060. In figure 7 we show the computing throughput for these devices as a function of the number of events processed in one batch. The effective computing power is seen to increase with the number of events to be processed in one go; the horizontal line shows the requirement related for an online trigger based on the RICH detector in NA62.

Figure 8 shows instead (for NVIDIA Tesla C2050 and GeForce GTX680 devices) the total latency, which includes data transfer times to and from the GPU and the kernel execution time. The significant reduction of the latency for the newer GTX680 GPU is due to the faster data transfer due to the presence of the gen.3 PCI express bus. Also in this case the maximum latency allowed by the NA62 application is seen to be attainable when a reasonable number of events is processed in one batch.

4. Application of GPUs in High Level Triggers

The ATLAS trigger [9] has performed remarkably well so far. The constant increase of the LHC luminosity up $3 \times 10^{34} Hz/cm^2$ and the successive upgrade at $1 \times 10^{35} Hz/cm^2$ requires a constant renovation of the trigger strategy. Currently, a first upgrade is foreseen in 2018 [10], when realtime tacking capabilities will also be available, followed by a complete renovation of the trigger and detector systems in 2022. The ATLAS trigger system is organized in 3 levels. The first-level trigger (LVL1) is built on custom electronics, while the second-level (LVL2) and the event-filter (EF) are implemented in software algorithms ran on commodity PC farms. The LVL2, based on the concept of the Region-of-iterest (RoIs), offers a natural study case for the deployment of GPUs in the realtime environment of a LHC experiment. The LVL2 algorithms are now implemented as approximated solutions of complex primitives such to stay within the time budget. The deployment of GPUs will allow to better exploit the potentiality of the detectors, implementing refined algorithms with higher selection efficiency. Such a result will improve the sensitivity to interesting physics signals and the ability to cope with an higher rate of multiple pp interactions (pileup). A fundamental aspect to exploit the computational performances of the GPUs is the possibility of parallelizing the execution of the algorithms.

5. Conclusions

The use of graphics processing units in scientific computing has become very common in the last years. The GAP Project [11], recently funded, will study the GPUs in real-time HEP trigger systems and for medical imaging. In particular, for the lowest trigger levels, work has to be done in order to reduce the contributions to the total latency due to data transfer from the detectors to the GPU. Two strategies are being pursued at the moment: the first makes use of a special driver that allows direct copy of the data from the NICs buffers avoiding redundant copies; the second one foresees to use a FPGA-based board for establishing a peer-to-peer connection with the GPU. Possible applications of this technique are the low level trigger of the NA62 Experiment at CERN, in particular for the reconstruction of photon rings in the RICH detector. Preliminary results show that current GPUs are suitable for sustaining the rate and minimizing the latency to an acceptable level for the experiment. As possible applications to high trigger, as a study case, we will consider a possible extension of the muon trigger of the ATLAS Experiment at CERN.

Acknowledgments

The GAP project is partially supported by MIUR under grant RBFR12JF2Z "Futuro in ricerca 2012". This work is partially supported by the EU Framework Programme 7 project EURETILE under grant number 247846.

References

[1] *http://www.ntop.org*

[2] Annovi A, Beretta M, Volpi G, Beccherle R, Bossini E, Crescioli F, Dell'Orso M and Giannetti P et al. 2011 *IEEE Nucl. Sci. Symp. Conf. Rec.* 141

[3] Ammendola R, Biagioni A, Frezza O, Lo Cicero F, Lonardo A, Paolucci PS, Rossetti D and Simula F 2012 *J. Phys. Conf. Ser.* 396

[4] Lonardo A et al. 2013 *http://on-demand.gputechconf.com/gtc/2013/presentations/S3286-Low-La tency-RT-Stream-Processing-System.pdf*

[5] Avanzini C et al. 2010 *Nucl. Instrum. Meth. A* 623

[6] Lamanna G 2011 *J. Phys. Conf. Ser.* 335 .

[7] Angelucci B, Anzivino G, Avanzini C, Biino C, Bizzeti A, Bucci F, Cassese A and Cenci P et al. 2010 *Nucl. Instrum. Meth. A* 205

[8] Collazuol G, Lamanna G, Pinzino J and Sozzi MS 2012 *Nucl. Instrum. Meth. A* 662

[9] Atlas Collaboration 2008 *JINST* 3 P08003

[10] Atlas Collaboration 2012 CERN-LHCC-2011-012

[11] *http://web2.infn.it/gap*