16

High-Efficiency Digital Readout Systems for Fast Pixel-Based Vertex Detectors

Alessandro Gabrielli, Filippo Maria Giorgi and Mauro Villa
University of Bologna and INFN Bologna
Italy

1. Introduction

Particle physics is one of the science branches which heavily relies on most advanced technologies due to the increasing complexity of the problems it has to face. In future colliders, luminosities and beam energies are scaling upwards. These are necessary conditions for the discovery of new physics which both result in a larger amount of data that need to be brought out of the detector. That's why one of the crucial points for new experiments is the evolution of data acquisition systems. Data acquisition systems employed in particle physics experiments followed the global technology trend and moved towards digital electronics and transmission lines, in this chapter we will describe how the effort of our work has been applied in this direction trying to extend digital processing on the very front-end of the detector. We will show how digital elaboration on the very front-end can help coping with new stringent requirements.

One possible scenario for the discovery of new physics is the chance to investigate with high-precision some apparently known processes instead of brutally scaling the $\sqrt{s}$ energy foreseeing to achieve the threshold for new heavy particle discovery. High luminosity e^+e^- accelerators can provide clean signals at very fast rates in order to provide in a reasonable amount of time the required statistics for high-precision new physics investigations. The next-generation flavour factories are aiming at luminosities up to $10^{36} cm^{-2} s^{-1}$ (refer to SuperB Collaboration (2007)) which imply a very high particle rate especially in the first layer of the innermost detector: the vertex tracker.

This perspective opened new challenging researches for the realization of very fast and efficient sensors and readout electronics capable to take advantage of these super-luminous facilities. In this chapter we would like to present our works on data acquisition chains, which is focused on the front-end side of the detector but involves also external DAQ boards. We will show how we have expanded digital signal processing of a classical DAQ systems outside the walls of the counting rooms to the front-end chips. What we present is far from being a complete and definitive DAQ for a tracker design, but it provides viable solutions and technicalities for what concern the readout electronics world.

The front-ends targeted by our data acquisition system are silicon sensors and, in particular, wide matrices of pixels. The huge improvements of the last decade in the world of the silicon industries, and the new technology processes that emerged recently, have stimulated the curiosity of the scientific community. Several types of pixel sensors for particle physics applications have been produced and integrated on large particle detectors, and some other are still under investigation or produced as prototypes. Hybrid pixel sensors for example, consisting of a sensitive layer bump-bonded to a standard CMOS layer, have been employed on the two major experiments operating at LHC:[1] The ATLAS silicon pixel detector [R. Klingenberg for the ATLAS pixel collaboration (year 2007)] integrates 80 million channels on a total active surface of 1.8 m^2, while CMS [S. Schnetzer for the CMS Pixel Collaboration (year 2003)] has a total pixel active area of 1 m^2 and 66 millions of channels.

Other kind of silicon pixel sensors exploit the deep N-well sub-micron technologies, that allow, with a standard CMOS process, the integration of the sensor with the analog front-end and the digital readout electronics on the same substrate. This monolithic sensors are very promising in terms of granularity and material budget (down to 50 microns of pitch and thickness); this pixel variant is known as DNW MAPS (Deep N-Well Monolithic Active Pixel Sensors). [A. Gabrielli for the SLIM5 Collaboration (year 2009)],[G. Rizzo for the SLIM5 collaboration (year 2007)].

Furthermore, a lot of attention is being dedicated to the possibility of exploiting new 3D integration technologies. In this perspective the aim is to stack many ultra-thin (15 μm) silicon layers with a high interconnection density given by 1 μm-wide through-silicon via. L. Gaioni et Al. (year 2009), R. Lipton (year 2007).

It is clear that the new frontier is then the possibility to exploit to the maximum level the integration of sensor, analog front-end and the digital logic on the same sensor chip. Our work proceeded in this direction, and proposes a series of digital sparsification and readout digital architectures that can be paired with several kinds of pixel technologies. These digital circuits are highly speed optimized, we aimed at sustaining a hit flux of 100 MHz/cm^2 which is, by order of magnitude, the value foreseen on the innermost detector layer of a flavor factory like SuperB.

Our goal is to read the hits from the matrix as soon as they form, in order to promptly reset the pixels and then reducing the sensor dead time. In this context we present different readout architectures comparing the achievements in terms efficiencies and time resolutions. We also give a brief description of the developing process consisting of conceptual design, VHDL description model, model validation, estimation of the efficiencies and finally digital ASIC design.

Silicon sensors that implement nontrivial sparsification circuits and a digital data interface can provide a more robust interconnection with the data acquisition system. In addition the chance of having a data-push event-formatted stream directly out of the front-end chip, can simplify the processing algorithm on the data acquisition boards. This can be translated into wider margins on triggering latencies and less stringent computational power requirements on the DAQ boards. We present for example a data-push front-end architecture that sends out encoded events where hits are already time sorted & space aligned (divided on the base of their spatial origin on the matrix). This allowed to remove the hit sorting algorithms on the DAQ firmware, resulting in a smoother data flow.

It is evident that the possibility to exploit more and more complex digital architectures in direct contact with the sensor readout opens a new field of research: how can be improved the DAQ performance by preprocessing data on the very front-end? We are investigating

[1] Large Hadron Collider, a proton/ions accelerator built at CERN (Geneva, CH).

these opportunities working both on the digital front-end architectures, and DAQ counterpart electronics.

Several chip submissions, within fruitful collaborations like SLIM5 and VIPIX, adopted the architectures that we have developed and allowed us to test them on an accelerated particle beam. In this occasions we could also test the interaction between these digital front-end architectures with a powerful DAQ system. We present here also the external DAQ infrastructure used to test the pixel detectors, a multi-FPGA VME board called EDRO (Event Dispatch and Readout) capable of an integrated I/O of 30 Gbps. It features also an associative memory interface, in order to exploit fast triggers ($< 1\mu s$) coming from an external board that compare the incoming hits to a bank of predefined patterns.

In summary, we think that now the silicon technologies allow the data acquisition to expand inside the walls of pure sensors world, increasing the synergy between these two elements that for long have been considered different fields of research.

2. Silicon vertex trackers

In a high energy collider experiment, the innermost detector is appointed to perform an accurate reconstruction of the particle tracks coming out of the interaction vertices, from which the name ”tracker”. Several coaxial layers of sensors are displaced around the beam pipe in the interaction region, with the detected crossing points it is possible to track a particle trajectory. Resolution and efficiency are the two main parameters to optimize but typically one has to trade off between these two. A higher number of channels and a smaller radius around the interaction point can improve the resolution at the cost of an increasing rate that worsen the efficiency.

Another crucial point is the total amount of material that we use for the construction of a tracker since the particles we want to measure actually interact with the detector itself. Depending on its momentum, a particle can be deflected at a non negligible angle each time it crosses a layer of the detector, making difficult to reconstruct the original trajectory. This undesired effect is known as multiple-scattering. In order to reduce the probability for a particle to scatter at large angles, it is very important to keep low the material budget of the entire detector.

Nowadays silicon detectors are widely used for this kind of application since the integrated circuit technology allows the integration of high-density micron-scale electrodes on large wafers providing an excellent position resolution. A silicon sensor grants also an easy integration with the semi-conductor-based readout electronics and, as we will discuss later, they can be fitted also on the same silicon substrate. This is a great advantage in terms of material budget if we consider that we can thin the silicon substrate roughly down to a hundred of microns. That is made possible because the density of silicon and its small ionization energy[2] can produce an adequate signal with a sensitive layer of that scale.

A typical silicon detector is composed by:

- **The Sensor:** It is the sensitive part of the detector. It is a capacitive element appointed to collect the charge that forms in the silicon substrate translating it into a tension signal (for minimum ionizing particles the most probable charge deposition in a 300 micronthick silicon detector is about 3.5 *fC* (22000 electrons) [W-M Yao et Al. (year

[2] It is the minimum energy required to extract a bounded electron

2006)]. It is typically implemented as a reverse-biased *p-n* junction which forms a region depleted of mobile charge carriers and sets up an electric field that sweeps the charge generated by radiation and diffusing in the substrate.

- **The Analog Front-end:** It is the analog electronics directly connected to the sensor, its task is to amplify, adapt and discriminate the sensor signal with a voltage threshold. Keeping the front-end noise low is a critical issue either to improve the energy resolution (which depends on the collected charge) and to allow a low detection threshold. For certain energy values, particles are more reluctant to ionize and release less charge, the electronics *ENC* (Equivalent Noise Charge) should be below this value. A scheme of a typical front-end circuit is presented in Fig. 1.
- **The Latch:** It is the memory element that keeps track of a threshold crossing. It is reset after the channel has been read out. The longer it takes to read and reset the latch, the longer the sensor is ″blind″ to new incoming particles.
- **The Readout:** It is the electronics appointed to extract the hit information from each pixel latch. It can be implemented in very different ways depending on the optimization targets. This is the element on which we focused our work.

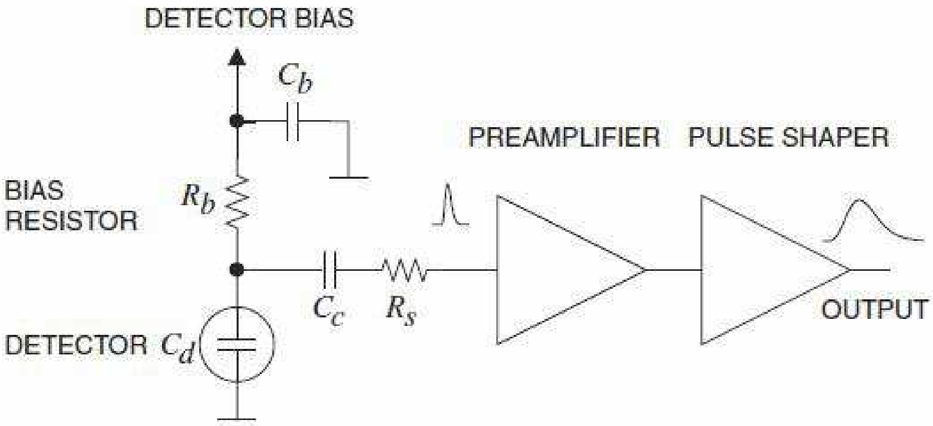


Fig. 1. Typical detector front-end circuit.

The silicon sensors can be implemented with different granularities and form factors, for example the *Silicon Strip Sensors* are long and thin *p-n* junctions that extends for several centimeters and they are about 50 microns wide. The longer the *p-n* junction is, the higher the capacitive load C_d, which means slower signals and higher power consumption. Pixel devices instead, are matrices of square-shaped sensors that improve granularity and provide faster signals. In this way the same area is covered by a greater number of channels, giving a more precise spatial information.

In a particle tracker, the error on the reconstructed position of the vertex is dominated by the spatial resolution of the innermost layers, therefore they are typically instrumented with pixel sensors due to their higher resolution. Moreover, since the area to be instrumented increases with radii, and since pixels sensors present a higher cost-per-area, the outer layers of the tracker are typically instrumented with silicon strips.

3. Pixel detectors

In the digital era the word ″pixel″ is very diffused, since it embodies the concept of digital quantization in the field of *imaging*. Nowadays a large variety of electronic devices based on silicon incorporate ″pixel sensors″. The most common and diffused semiconductor pixel sensors are those employed in modern digital cameras, mobile phones and, more generally, in almost every portable device. This kind of silicon sensor detects visible-light photons and it is designed to have a wide and optimized dynamic range in order to exalt, for example, the brightness and contrast of the subject. A statistical number of photons is collected in the sensor array making some pixels ″brighter″ than others. The whole matrix has to be read out in order to provide the final image.

The pixel sensors adopted in particle physics experiments instead, should detect traversing charged particles or photons. These detectors should be sensitive even to the crossing of single particles. By means of this, and due to the high flux of particles nearby the interaction point of a collider (our goal is to sustain 100 MHit s^{-1} cm^{-2}), tracker sensors are optimized in terms of readout speed rather than dynamic range. Moreover, in some cases the readout phase is continuous, overlapped to the acquisition phase, and concentrated only on the hit regions, since what the physicist are looking for is not a *photo* of the event but the spatial position of a trace produced by an impinging particle. The superimposition of several layers gives the spatial information to reconstruct a particle trajectory as it was discussed in section 2.

The information about the quantity of charge collected in a fired pixel can be read out. It is useful to enhance the sensor resolution in case of clustered events where the reconstructed crossing point of the particle can be evaluated with a centre of mass algorithm (a spatial weighted average where the charge acts as a weight). This information is also useful to reconstruct the amount of energy lost by the particle in the detector. This would give a calorimetric information dE/dx, that can be used for particle identification. Though, the extraction of this information is not for free, it can be rather very time consuming especially for pixels since the density of channels is very high (400 channels/mm^2 with a 50-micron pitch). When a pixel get fired by a crossing particle, it is unable to detect any other impinging particle until it is read out and reset. This time laps, during which the pixel is latched, is called *dead time*. In our specific case-study, the dead-time introduced by charge extraction wold be unaffordable, consequently the readout we developed extracts only the hit/not hit information from the pixels.

A very simple readout structure for a CMOS APS - *Active Pixel Sensor* - is shown in Fig. 2, and it is know as the 3T (three transistor) configuration.

A 3T APS matrix is read out with the so called *rolling shutter* procedure. Each row is read out one after the other driving a column bus. A the other end of the column bus the front-end electronics processes the pixel signals. The advantage of this method is that the sensor matrix can collect charge during a continuous acquisition process.

A pixel detector can be implemented with different fabrication technologies. The most common and diffused at the moment foresees the interconnection of a sensor silicon layer to a standard CMOS-process integrated circuit (that hosts the front-end electronics) by means of an array of micro solder bumps. This kind of sensors are known as hybrid pixel sensors. They are employed in both the major experiments taking place at CERN: ATLAS and CMS (ref. R. Klingenberg for the ATLAS pixel collaboration (year 2007) and S. Schnetzer for the CMS Pixel Collaboration (year 2003)).

It is possible to get rid of the delicate bump-bonding procedure integrating both sensor and readout on the same substrate processed in standard CMOS technology: this kind of device are known as MAPS (*Monolithic APS*). The *p-n* sensitive junction can be obtained by an n well implanted in the p substrate. The use of this technology for the detection of charged particles is challenging since only the very thin epitaxial layer (10-20 microns) of the silicon

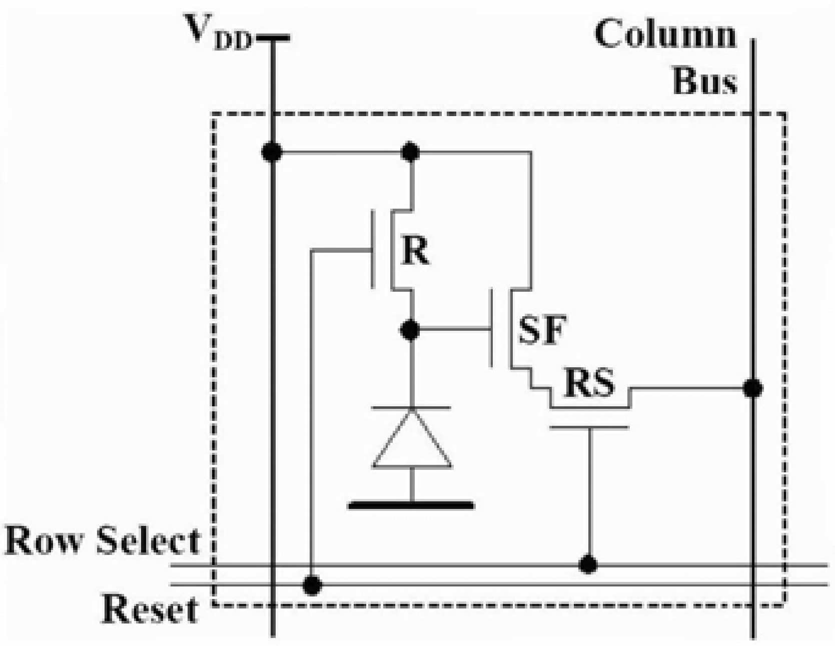


Fig. 2. Three transistor readout for a matrix of pixels. *Reset* transistor **R** clears the pixel of integrated charge, *Source Follow* transistor **SF** amplifies/buffers the signal and *Row Select* transistor **RS** selects the row for readout.

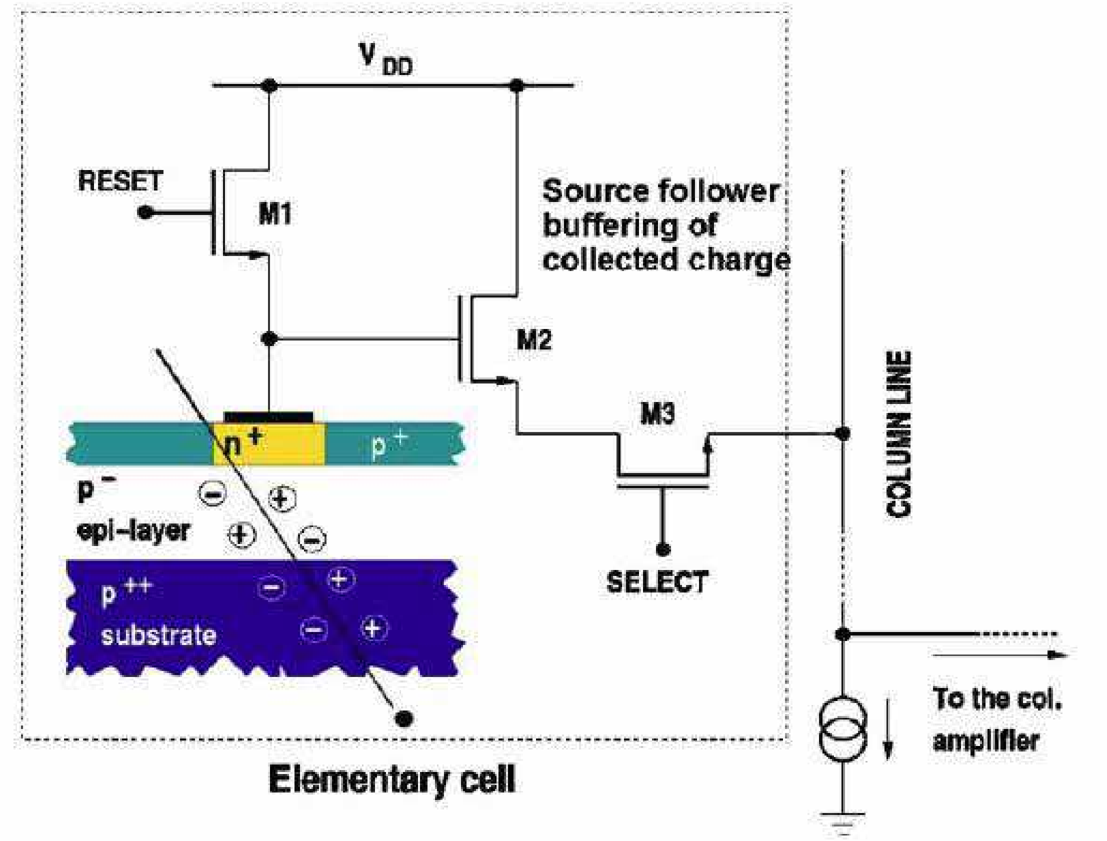


Fig. 3. Schematic view of a CMOS MAPS device with typical 3T readout structure. G. Rizzo for the SLIM5 collaboration (year 2007)

is available as sensitive volume. On the other hand, this allows to thin down the substrate to its mechanical limits and to build vertex detectors with an extremely low material budget.

Modern CMOS processes allow triple well structures, a feature that has been explored to increase the collection efficiency and to implement in-pixel front-end electronics S. Bettarini et Al. (year 2007). A deep and extended N-well is used as the collecting electrode, wherein a *p* layer is deposited to host the NMOS transistors of the front-end electronics. The large electrode area improves the collection efficiency, and the charge to voltage conversion, which generally decreases with the capacitive area, is enhanced by the in-pixel active amplifier. The front-end PMOS transistors are enclosed in additional N-wells, that actually steal charge to the main collecting electrode, therefore the in-pixel analog and digital electronics is quite limited in order to keep a high collection efficiency. The enclosure of the analog front-end at pixel level in the deep N-well brings a significant noise improvement, N. Neri et Al. (year 2010) report a measured ENC of 75e$^-$. Moreover, the possibility to include also digital components at pixel level allows to develop faster readout, improving the speed limits of the typical rolling shutter architecture used for 3T APS structures.

Another promising processing technology, that captured the attention also of the physics community, allows the integration of several ultra-thin silicon layers (~15 μm thick) in a 3D structure, interconnected by micron-scale through-silicon vias L. Gaioni et Al. (year 2009)R. Lipton (year 2007). This means in principle that a silicon detector could stuff the sensor layer, the analog front-end electronics and dense digital logics at pixel level for enhanced readout capabilities, all on independent substrates (low noise, almost 100% active area) within a silicon stack only few hundreds of microns thick (very low material budget).

There are also ongoing researches that aim to integrate deep N-well MAPS structures in 3D vertically integrated IC [V. Re et Al. (year 2010)] represented in Fig. 4.

Our work is intended to exploit the new opportunities brought by these technological innovations, in order to provide readout architectures characterized by higher efficiencies. The main aspect we are trying to optimize, is the reduction of the average pixel dead-time. We are investigating different ways to extract the hits as fast as possible from the sensor matrix, in order grant a high detector efficiency. In second place we want to compress the large amount of data produced in high luminosity experiments, in order to reduce the on-chip memory and the output data bandwidth, with a consequent improvement of the static and dynamic power consumption.

4. Tools and procedures

In this section we want to present the working procedures, the typical project flow and also the tools that we use for the design and implementation of an embedded readout in a sensor chip.

In first place we start a new project taking into account the structural parameters, like pixel resolution and the total sensitive area, and considering the typical working conditions in terms of hit rate, time resolution and so on. We deal with our partners that provide the sensor matrix in order to find a routable structure that can improve the hit extraction algorithms but, at the same time, that can be scaled up to the desired dimensions. This step was found to be crucial since it requires to be quite forward-looking. The point is to establish the demarcation line between the full-custom design of the matrix and the world of standard-cells. The pinout of the whole matrix is then defined. In addition, a precise and sharp edge between these two blocks is fundamental for an accurate set up of the logical test benches that are performed along the implementation phase.

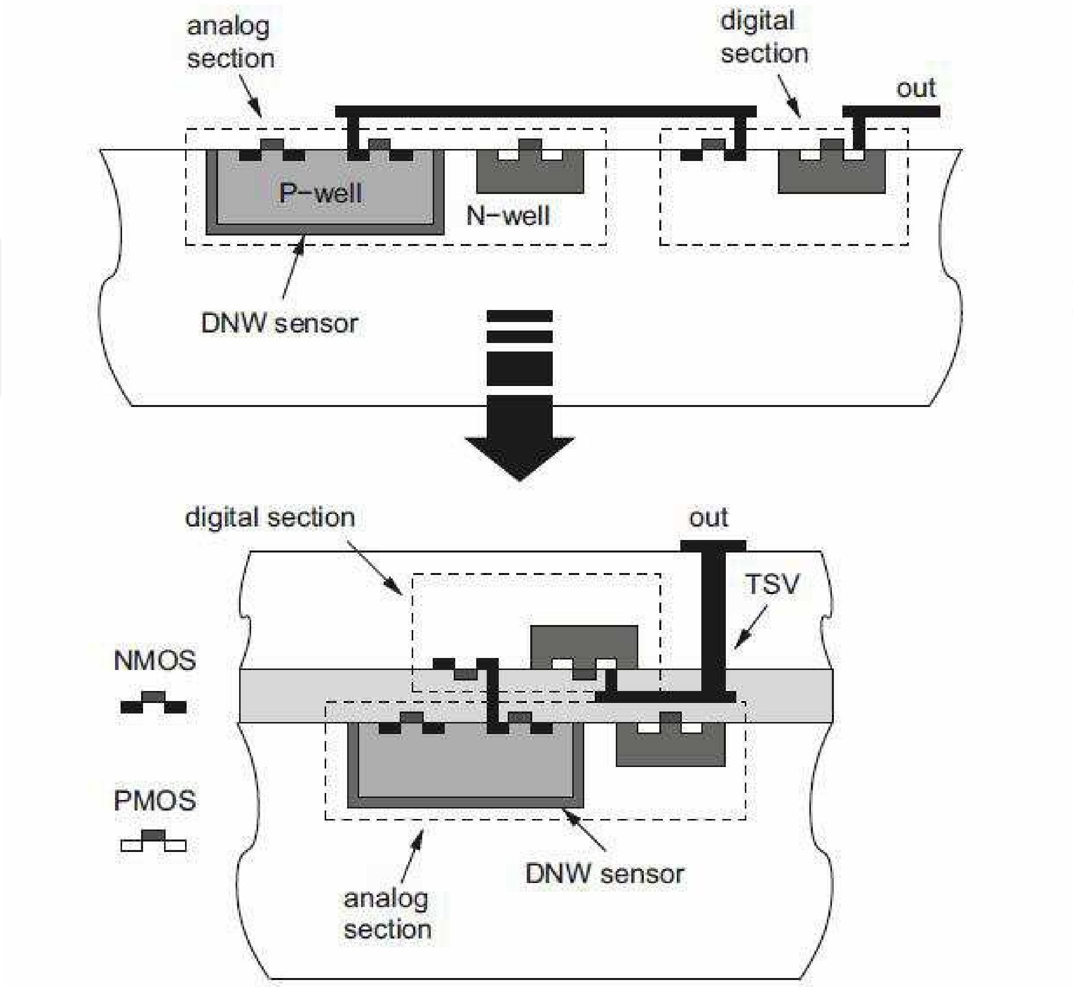


Fig. 4. Section view of a 2-TIER 3D MAPS structure.

Thereafter we try to project the readout architecture that fits at best in these requirements, and that optimizes the average pixel dead-time. We want to get as closer as possible to the pixel physical limit, mainly due to the front-end shaping-time (ref. to section 2). The architecture is developed in blocks, each one with specific and dedicated tasks. Once we have the complete conceptual design of each block, and of its task, we start to code the architecture with a specific hardware description language called VHDL (Very high speed Hardware Description Language).

VHDL can look like a sequential compiled language like C at first sight: it has a defined syntax, statements, functions and so on. But, at a closer look, it reveals the differences: since VHDL is used to describe digital architectures, the code has not a sequential flow from the beginning to the end, but it is divided in concurrent statements. Each of them is parallel, and it represents the equivalent of an independent circuit. Only the statements that are included inside special code blocks like *processes*, *functions* or *procedures* are sequential. The sequential execution of the statements inside a process is a high-level logical representation of the behaviour of the corresponding gates net. VHDL syntax is suitable both for a high-level behavioural modeling of electronic devices, and also for a gate-level net-list description. Moreover, in VHDL it is possible to give a hierarchical structure to the code, describing small components to be incorporated and interconnected into a higher level entity; this simplifies the maintenance and re-use of code. We took also a great advantage of VHDL by describing the architectures in a parameterized way, so that it could be easily adjusted to fit with different matrix dimensions and granularities.

A high-level hardware description in VHDL (or in any other HDL language like Verilog) can be translated into a net-list by specific EDA tools (Electronic Design Automation) that compile the code and implement the desired functions with the physical components found in a library. These libraries must be provided by the foundry where the designer wants to submit the IC. For our applications we used the *Synopsys Design Compiler* tool, a high-end product synthesizer for ASIC design (Application Specific Design Circuit).

But VHDL is intended also for circuit simulation, providing the designers with a set of non-synthesizable functions that can be used to build powerful test benches: for example text file I/O capability has been extensively used to load matrix patterns, and store simulation results. This constructs can be included in a top-level hierarchical entity that describes the stimuli and interconnects them to the top-level entity of sythesizable logic. We compiled and run our test benches with Mentor Graphics ModelSim, another EDA application that perform a logical simulation of the architecture giving the designers a plenty of tools for architecture debug and optimization.

Several steps of simulations take place during the implementation of the readout, a first logical model of the matrix sensor is connected to a hit file loader and integrated in the readout test benches. This is the starting point for every logical simulation of the high-level VHDL code since it allowed us to stimulate the components of readout as we pleased. Once each readout block has been coded and interconnected in the top hierarchical entity, we start a a dedicated simulation campaign in order to evaluate the efficiencies of that architecture. For this purpose a VHDL Monte Carlo hit generator stimulate the matrix and several millisecond of system working are simulated and analysed.

The top readout entity is then synthesized by the EDA tool. The produced net-list can be simulated in turn exploiting the cell models library furnished by the foundry within their design kit. This models includes the timing characterization of each component so that the post synthesis simulation can take into account also the propagation delay of signals as they go through the standard cells.

The following step is the physical implementation: in this phase the produced net-list of standard components should be placed on a predisposed area and routed. We adopted SoC (System on Chip) Cadence Encounter tool, a CAD developed for IC floor-planning, standard-cells placement/routing, and timing analysis. The floor-plan of an IC typically starts with the geometrical definition of the IC area, then we define the disposition of I/O pads. At this point we can import the matrix layout as an independent block and we define the readout core area as shown in Fig. 5.

The design placement and routing are performed by semi-automatic algorithms that leave to the designers the possibility to set a wide set of parameters and constraint. A delicate constraint is that on core interconnection to the matrix block.

The production flow foresee several iterations of implementation followed by timing extraction and analysis in order to find an optimal configuration. When an optimum is reached a DRC (Design Rule Check) is run on the design in search of constraint and rule violations. The final step is the extraction of the GDSII file, that contains the graphic layout of the IC to be sent to the foundry.

Now we will describe the main features of some of the matrix and peripheral architectures that we have developed, in conjunction with the efficiency evaluation studies that we have performed on them, focussing on those that have been implemented on silicon.

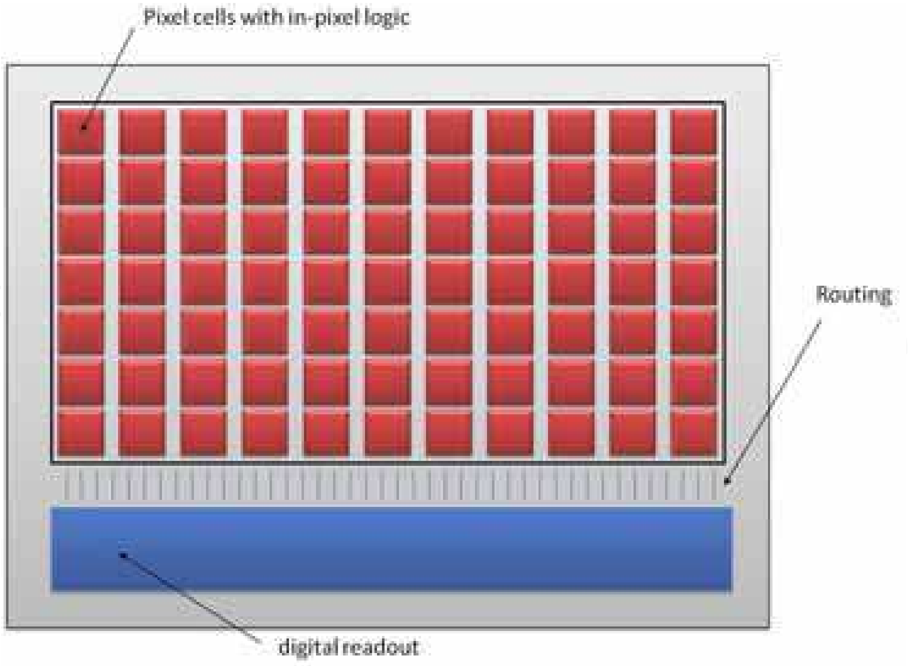
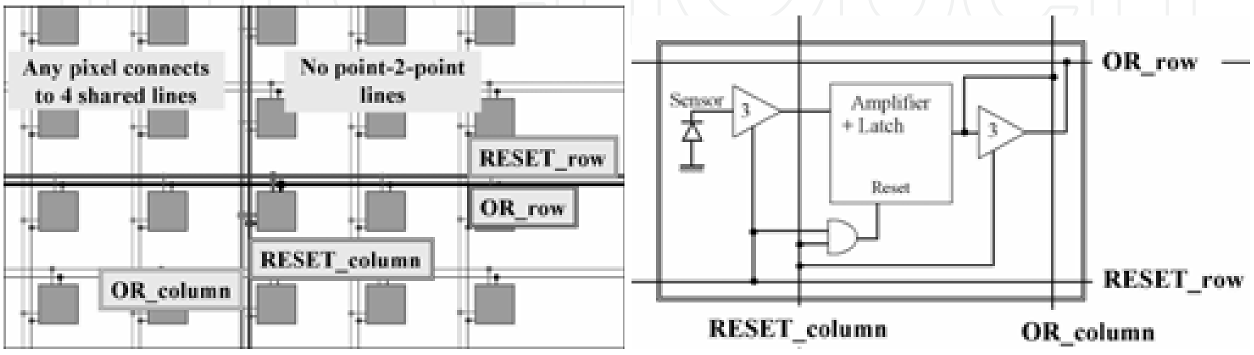


Fig. 5. Top schematic view of the peripheral readout and sensor matrix. Figure not in scale.

5. A sparsified readout matrix

The main goal of a sparsified readout architecture is the association of a spatial and temporal coordinate to each fired pixel. The term *sparsified* means that hit extraction and encoding is focussed on sparse randomly-accessible regions of the matrix, where it is known the presence of fired pixels. This method is in opposition to a full matrix sequential readout, and it is meant to achieve a faster readout and reset of fired pixels. In this architecture, these sparse and randomly accessible regions are the pixels themselves.

The idea is to incorporate few digital logic within pixels, exploiting for example a DNwell MAPS sensor technology, and realize in a dedicated portion of the chip area a complex digital readout system. The key concept is to use only inter-pixel global wires and not point-to-point wires from the border of the matrix to single pixels or groups of pixels. In Fig. 6.a is presented a pixel interconnection scheme exploiting global wires only. This approach allows to reduce wire density, that does not depend on the size of the matrix (number of pixels), in order to grant a higher scalability of the architecture.



(a) (b)

Fig. 6. In (a): The wired-or matrix layout. In (b): The 4 wire in-pixel logic.

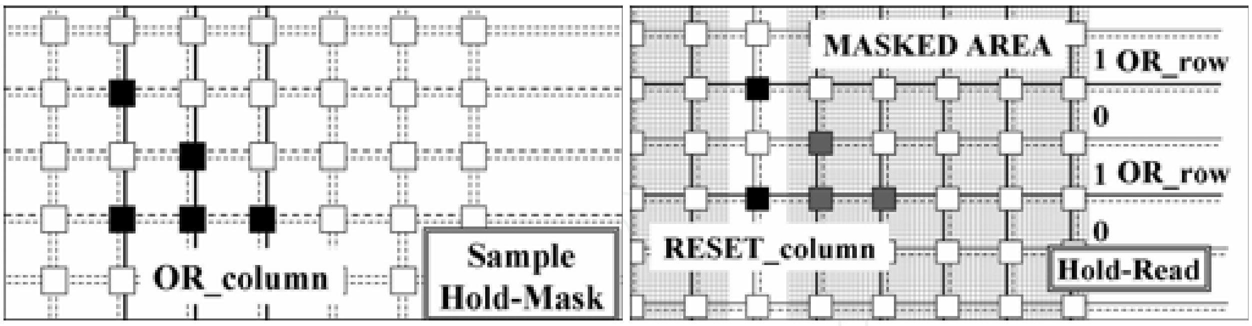
Let us now discuss in details the functions of each line:

- *OR_row* is a 3-state buffered horizontal output wire to read the pixel status. When the buffer is enabled through the RESET column vertical line, pixel output is read via the OR row wire. This line is shared with all pixels in the same row by creating a wired-or condition. As only one pixel at a time is allowed to be read, the OR row coincides with the pixel output value.
- *RESET_row* is a horizontal input wire to freeze the pixel by disconnecting it from the sensor. Moreover if RESET row is asserted along with the RESET column line, it resets the pixel. This line is shared with all pixels in the same row.
- *OR_column* is a vertical output line that is always connected to pixel output. This is shared with all pixels in the column by creating a wired-or condition. If at least one pixel of the column is fired, this global wire activates, independently of the number of hits and their location.
- *RESET_column* is a vertical input line to enable the connection to the sensor via a 3-state buffer. It is used to mask an entire column of pixels. Again, if used with the RESET row, it resets the pixel.

In Fig. 7 we present an example in the situation of a 5 hit cluster. The active wired-or conditions cause the activation of three OR column wires. This corresponds to the *Sample Phase* of Tab. 1.

Phase	RESET row	RESET column	OR row	OR column
Sample	1	0	Z	*pixel*
Hold-Mask	0	0	Z	*pixel*
Hold-Read	0	1	*pixel*	*pixel*
Reset	1	1	0	0

Table 1. Pixels readout phases.



(a) (b)

Fig. 7. In (a): Columns and rows of the hits. In (b): Readout starts for the first enabled column.

During *Hold-Mask phase* the matrix is frozen by de-asserting all the RESET row signals, no more hits can be accepted by the matrix. This determines the time granularity of the events. Pixels are then read out column by column during the *Hold-Read phase* by masking all matrix but the desired column with the RESET column signal. The pixel content is put on the OR row bus and can be read out. Afterwards, the column is reset by re-asserting the RESET row signal in conjunction with RESET column.

The readout process moves on to all the columns that presents an active OR column signal, and skipping the empty regions of the matrix. The two Hold-Read and Reset phases are the only two cycles needed to enable and read out an entire column of pixels, thus the entire readout phase takes twice as many clock periods as the number of activated columns.

During the readout process, the whole matrix is frozen in order to avoid event overlaps. This is done to individuate and delimit precise time windows to which hits belong. The time period is beaten synchronously in the whole detector, in order to allow the off-line reconstruction of tracks from the space-time coordinates of the associated hits.

6. The AREO readout architecture

6.1 APSEL3D

Next step of this chapter is the presentation of the peripheral readout logics that perform the hit extraction from the matrix, encode the space-time coordinates, and form the digital hit-stream to be sent out of the sensor chip. One of the first architectures that we have developed, has been realized on silicon within a sensor chip called APSEL that was realized by the SLIM5 collaboration [A. Gabrielli for the SLIM5 Collaboration (year 2008)]. The architecture involved took the name AREO because it is the APSEL chip REad Out. The IC is a planar MAPS sensor that exploits the triple well technology described in section 3 and provided by ST Microelectronics in a 130 nm process. The AREO architecture was developed to be coupled with a matrix that presents dedicated in pixel digital logic and global connection lines shared among regions of pixels. The sensor matrix is 256 pixels wide (32 columns by 8 rows) divided into 16 regions of 4 × 4 single pixels called *Macro Pixels* (MPs) (see Fig. 8). The pixel pitch is 50 microns.

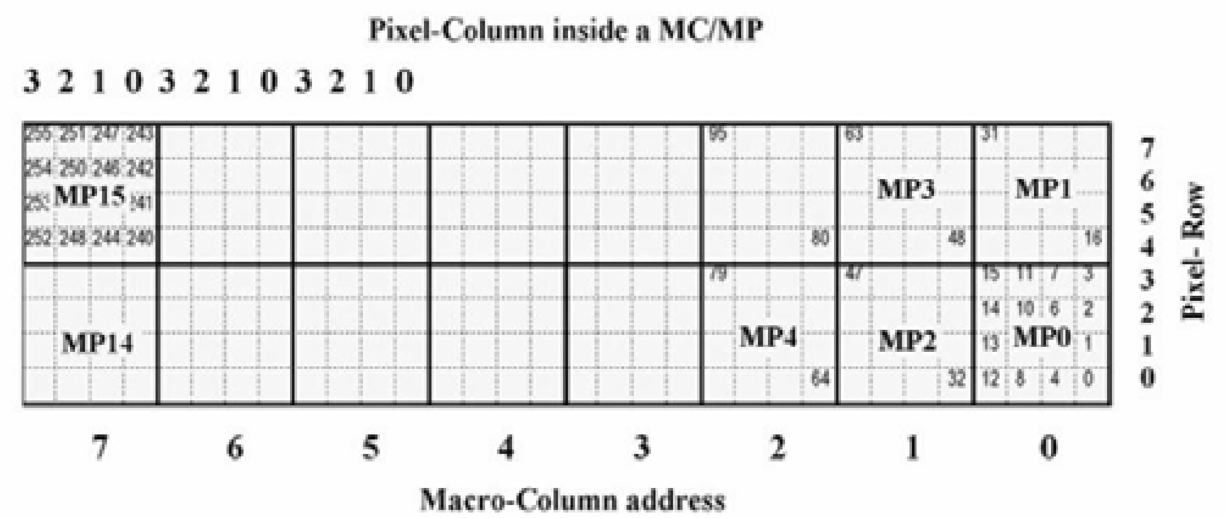


Fig. 8. The matrix divided into Macro Pixels

Each MP has two private lines that interconnect it to the peripheral readout: a *fast-or* and a *latch-enable* signal. When a pixel in a clear MP gets fired, the fast-or line get activated and, when the latch-enable is set to low, all the pixels within the MP are frozen and cannot accept new incoming hits any more.

Internally to the peripheral readout a time counter increments on the rising edge of a *bunch crossing* clock (BC). When the counter increments, all the new MPs that present an active fast or are frozen and they are associated to the precedent time counter value. In this way all the

fired pixels within a frozen MP are univocally associated to the common *time-stamp* (TS) stored in the peripheral readout.
The hit extraction takes place by means of an 8-bit wide *pixel data bus* shared among all the pixel rows. Each pixel is provided with a tri-state buffer activated by a *column enable* signal shared by the pixel column, as it is shown in Fig. 9.

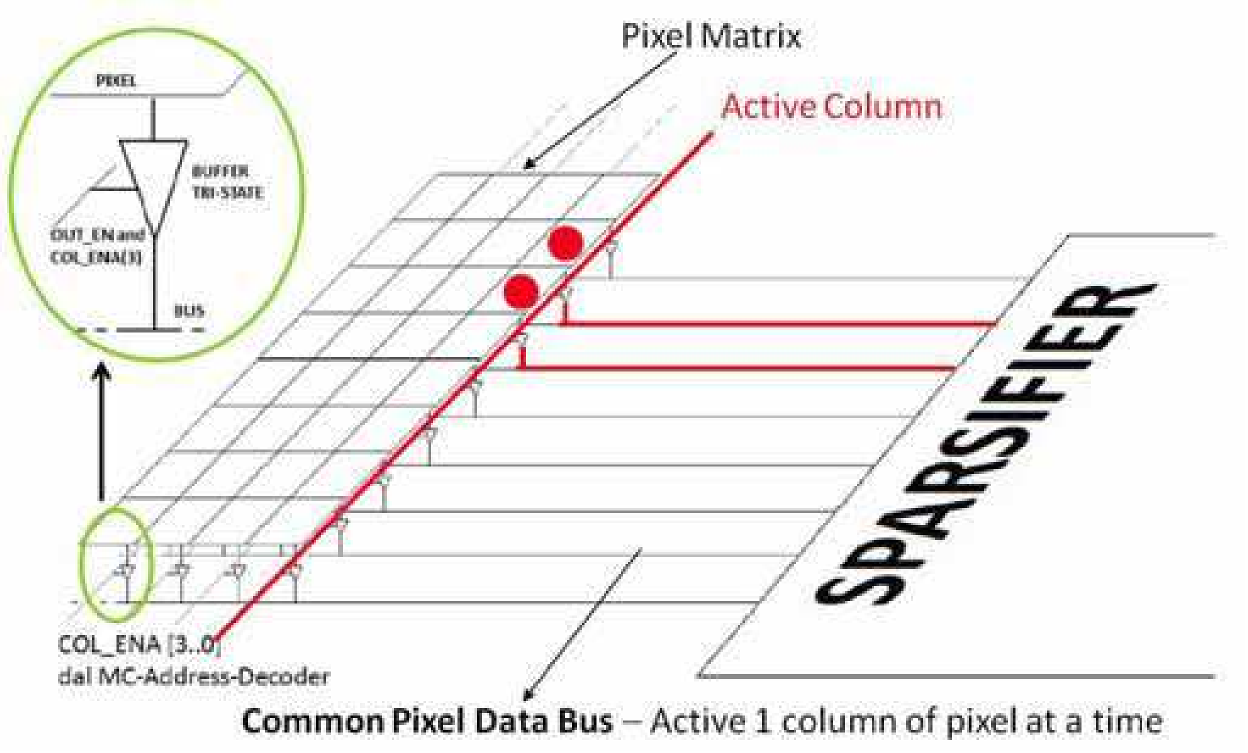


Fig. 9. Common data bus and pixel drivers

The vertical pile of 2 MPs is called *Macro Column* (MC). Only the MCs that present at least one frozen MP are scanned. If there are no frozen MPs in a MC, its four columns are skipped from the readout sweep in order to speed up the hit-extraction process.
To scan a MC means to activate in sequence its four columns since it is not know a-priori which is the one that contains the hit. Each pixel column is readout in one clock cycle, so the whole MC readout takes place in 4 read clock periods. After the readout phase of a MC, the reset condition is sent to the pixel logic by enabling contemporaneously the first and the last column of the MC (MC col. ena = 1001). Since the column enable signals are shared among all the pixels of a column, in order to prevent the resetting of a MP on that MC, which was not frozen, a *Macro Row* enable is routed to the matrix and taken into account during the output-enable and reset phase of the pixels. In this way only the desired MP of a MC can be read out and reset, while the other keeps collecting hits. The typical MP life-cycle is shown in Fig. 10.
All the hits found on the active column can be read out in one clock cycle, independently of the pixel occupancy, thanks to a component called *sparsifier*. This component is appointed to encode each hit with the corresponding x and y spatial coordinates and with the corresponding time stamp.
Next to the sparsifier there is a buffering element called *barrel*, which is basically an asymmetric FIFO memory with dynamic input width based on rolling read/write

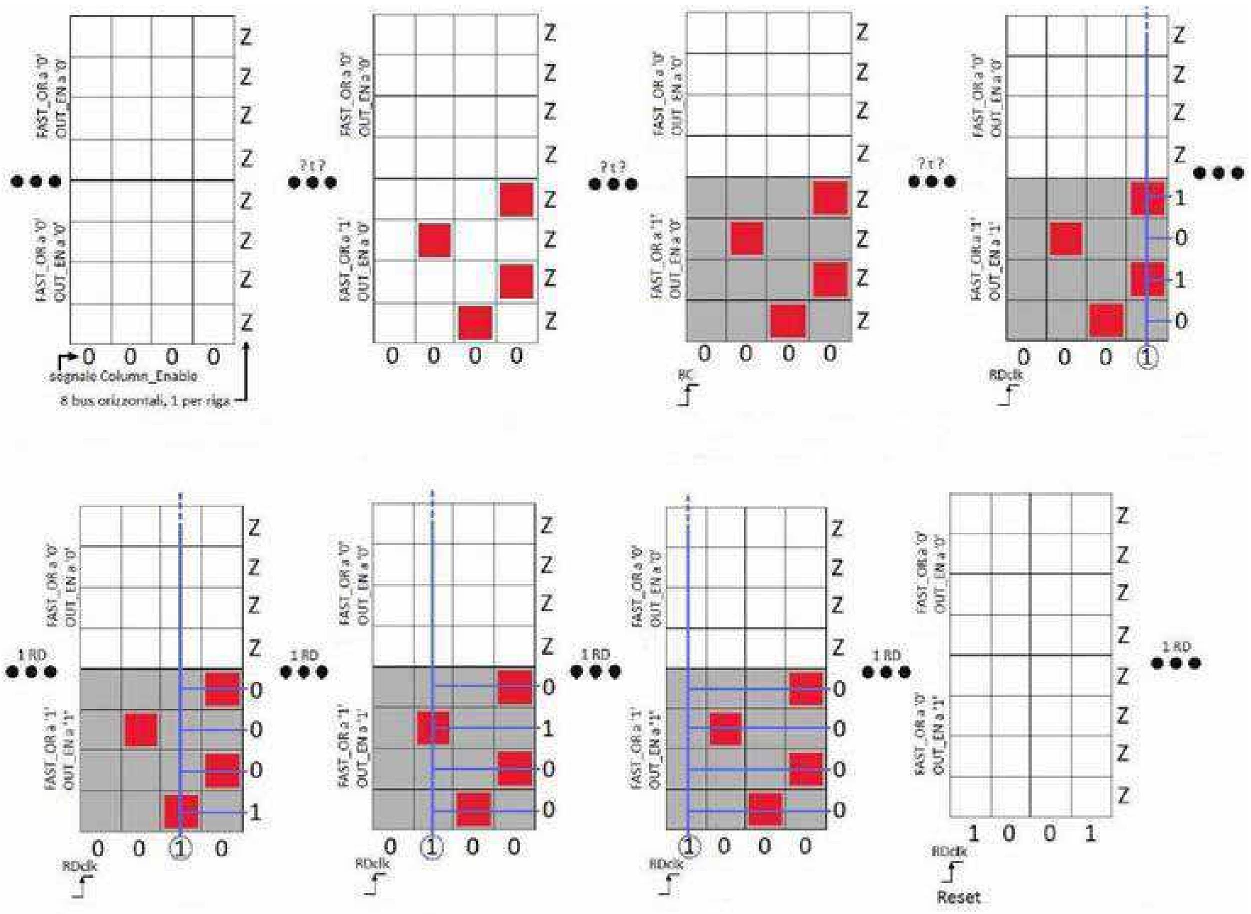


Fig. 10. MP life cycle. The hits populate the MP. A BC edge freezes the MP. The MP columns are read out one by one. A final reset condition is applied.

addresses. It can store up to 8 encoded hits per clock cycle which means that it has 8 independent write address pointers that can be enabled or not depending on how many hits are found on the current active column. Due to the reduced dimensions of the connected matrix, the barrel depth was of only 16 hit-words. The barrel output throughput is 1 hit per clock cycle. The hits are encoded with the format described in Tab. 2:

hit field	length	name	function
hit[11:9]	3 bits	pxRow	pixel row address
hit[8:7]	2 bits	pxCol	pixel column within MC
hit[6:4]	3 bits	MC	Macro Column address
hit[3:0]	4 bits	TS	time stamp field

Table 2. Hit encoding in APSEL3D readout. The global *x* address must be reconstructed by the MC and pxlCol fields. The algorithm is 4*MC* + *pxCol*. A *data valid* bit is added to the coded hits when they are sent on the output bus.

Since the developed architecture is data-push, which means that no external trigger is required, the hits are automatically popped out of the barrel and sent out on the synchronous *output data bus*. The readout architecture is synchronous on the external *read clock*. While a different clock is used to feed the slow control interface, for the chip control.

Slow control (SC) is based on a source synchronous bus of three *SC mode* bits and on 8 bits of *SC data*. Depending on the value of the SC mode bus sampled at the rising edge of the *SC clock*, different slow control operations can be performed. One of the main task of the slow control interface is to load the mask patterns that can exclude sets of MPs from the acquisition process.

The AREO architecture is also provided with a digital matrix, which is a copy of the full-custom sensors array but realized in standard-cells and residing in the chip periphery with the readout itself. It has been implemented for digital test purposes. With the slow control interface it is possible to select the operating mode from digital to custom: in *digital mode* the readout is connected to the register-based matrix, while in *custom mode* it is connected to the sensor matrix. Through SC it is possible to load a predetermined pattern on the digital matrix, in this way we can verify the correspondence between the loaded hits and those observed on the chip data bus.

The readout efficiencies will be presented in the next subsection, where the application of this architecture on a bigger matrix is described.

6.2 APSEL4D

Thanks to the fruitful SLIM5 collaboration, it was possible to implement the AREO architecture even on a wider 4096-pixel matrix, in the chip that was named APSEL4D. Scalability is one of the major issues when using non-global lines. The number of private connections scales with number of pixels and thus with area, which is a quadratic growth respect to linear matrix dimensions. The contact side between the matrix and the readout, where the routing signal shall pass through, increases linearly which means that whatever is the finite dimension of a wire, exists always an upper limit in matrix size. In our case the fast-or and latch-enable signals are non-global lines but they are shared among groups of pixels; this allows to push the limit further.

In this chip the readout is connected to a 128×32-pixel matrix with the same characteristics of the 3D parent. The subdivision into MPs follows the same rules of the APSEL3D version, a schematic view of the matrix of MPs is shown on Fig. 11.

Also the readout architecture kept the same original idea, but it has been scaled to the larger matrix with the replication of some basic components. Since the matrix readout takes place by columns, the enlarging in the horizontal direction led only to a longer column sweeping time and a longer x address field in data. The extension in the vertical direction was achieved by paralleling 4 couples sparsifier-barrel. A scheme of the AREO v.4D readout is presented in Fig. 12.

The parallel data coming out of the barrels are stored in the *barrel final* by the *sparsifier out*. In this way hits are sent one by one on the *formatted data out bus*. The barrels and the barrel final have a depth of 32 hit words. If a rate burst fills up the barrels, a feedback circuit stops the matrix readout in order to flush data out of the barrels. This increase the pixels dead-time but it grants that no data is lost. The hit format of the AREO v.4D architecture is reported in table Tab. 3.

Due to the higher number of channels, the encoded pixel address has increased in length. The time counter was raised from a modulo 16 to a modulo 256, thus the time stamp field is now 8-bit wide.

The implementation went through and the final layout of the readout is shown in Fig. 13.

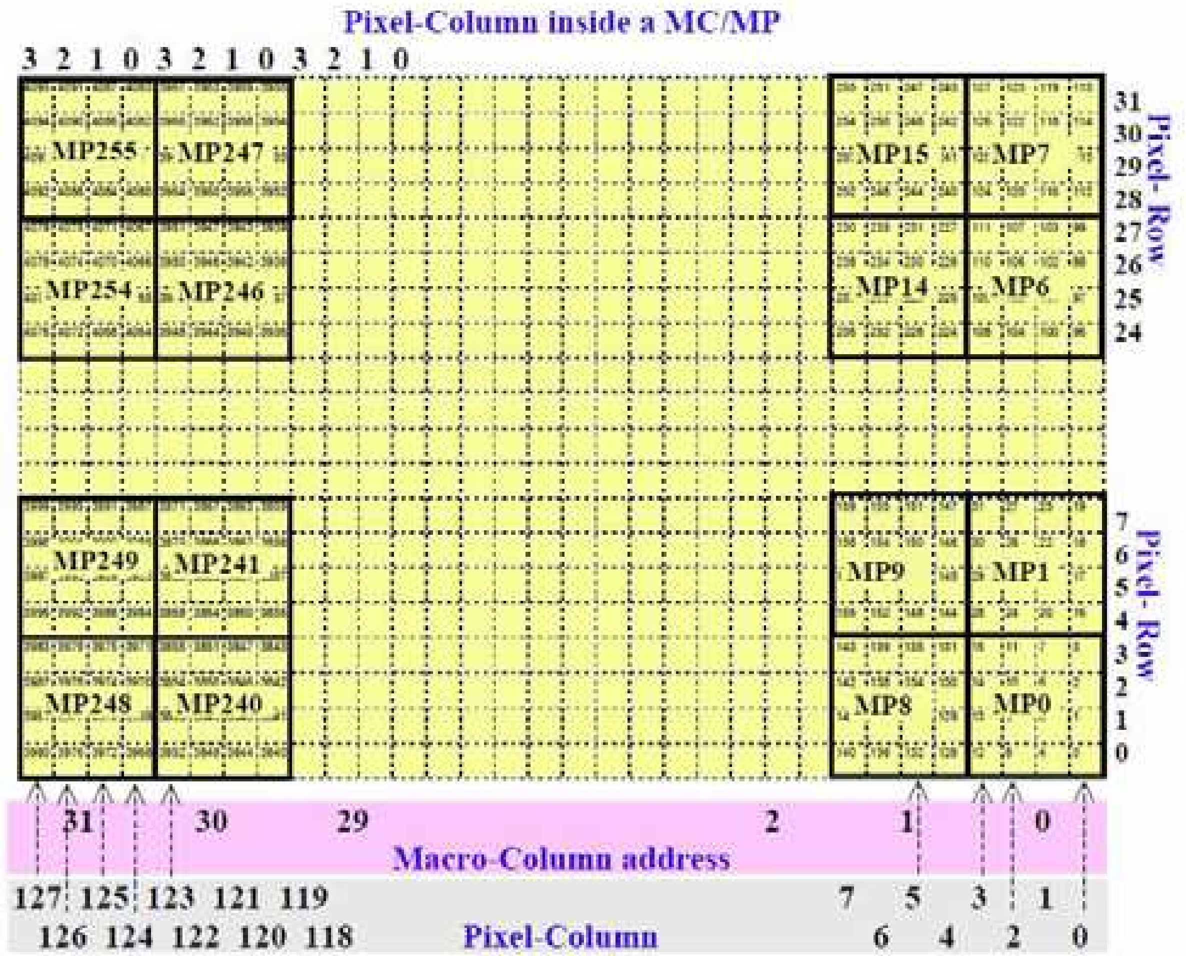


Fig. 11. APSEL4D matrix and MPs.

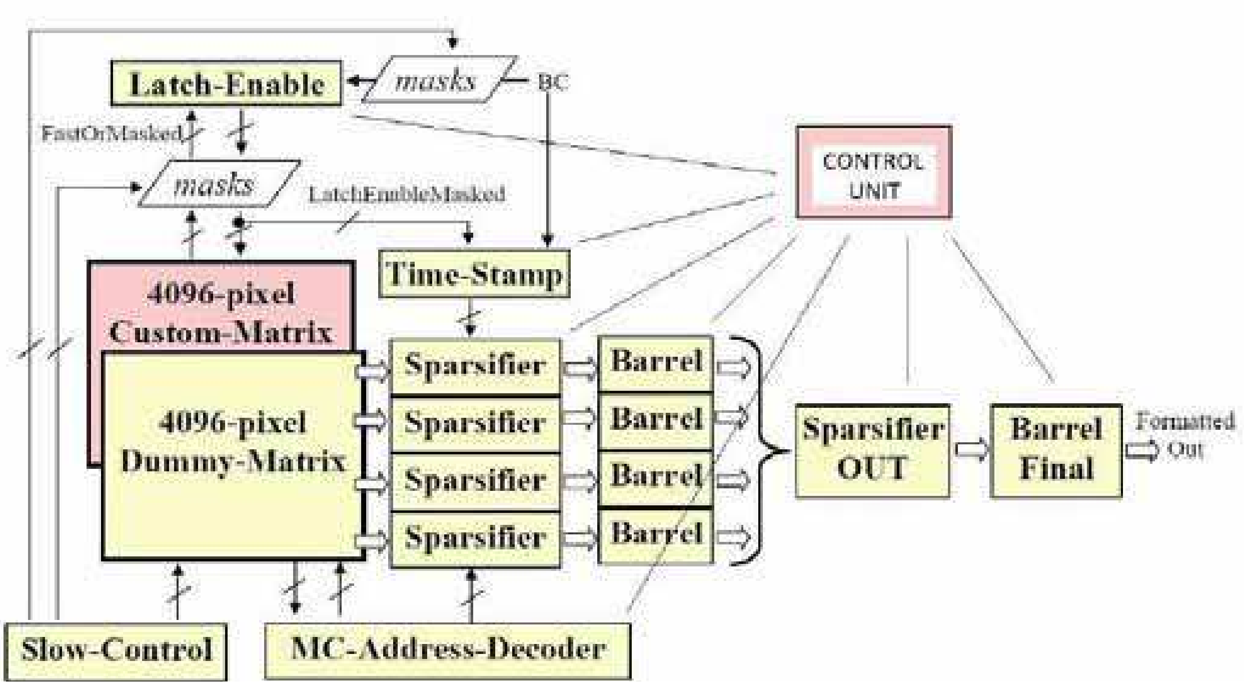


Fig. 12. APSEL4D schematic readout

hit field	length	name	function
hit[19:15]	5 bits	pxRow	pixel row address
hit[14:13]	2 bits	pxCol	pixel column within MC
hit[12:8]	5 bits	MC	Macro Column address
hit[7:0]	8 bits	TS	time stamp field

Table 3. Hit encoding in APSEL4D readout. The global *x* address must be reconstructed by the MC and pxlCol fields. The algorithm is 4*MC* + *pxCol*. A *data valid* bit is added to the coded hits when they are sent on the output bus.

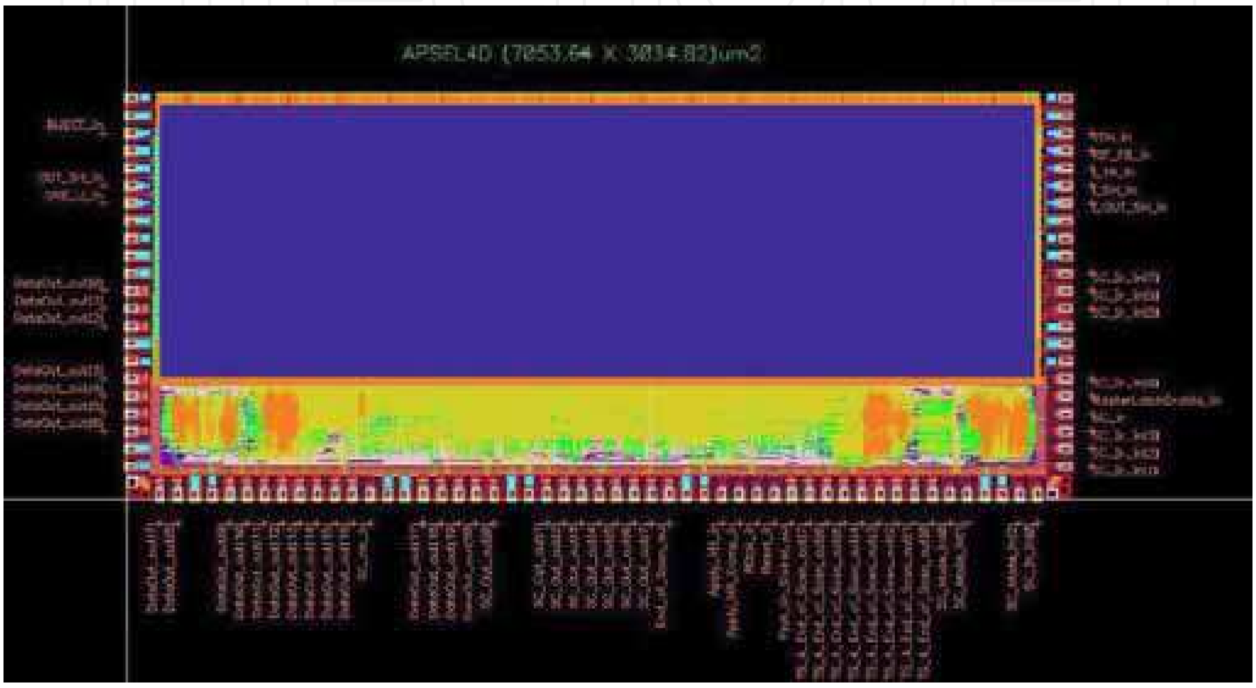


Fig. 13. APSEL4D layout

Several logical simulations were run with the source code of this architecture during the implementation phase. These simulations have generally two main objectives: formerly to verify the correct operation of the logic described and in second place to evaluate the efficiency of the architecture with a statistical sample of randomly generated hits. We present the results of the efficiency studies.

Several behaviours were observed by varying the flux of incoming particles and the readout clock speed. In Fig. 14 we plot the readout efficiency against the average hit rate. It is important to clarify what is the inefficiency, where it comes from and how we measure it. The inefficiency is the quantification of how much information we are loosing, being it of physical relevance or not. A part of it is proportional to the average pixel dead-time, being it due to front-end shaping time or to the readout hit-extraction speed. The longer the pixel is blind, the more information is lost. The readout scheme implemented and the readout clock, determine the hit extraction speed. Another origin of inefficiency is the hit congestion in the readout de-queuing system. For example, in this particular architecture, a hit congestion causes the hit extraction to stop, thus resulting in further increasing dead-time. Anyhow, it is important to understand that this origin of inefficiency is unrelated to previous one: if we could could count on an infinite output bandwidth, or a infinite buffer, then we would have no inefficiency due to hit congestion, but the same inefficiency due to hit-extraction algorithms.

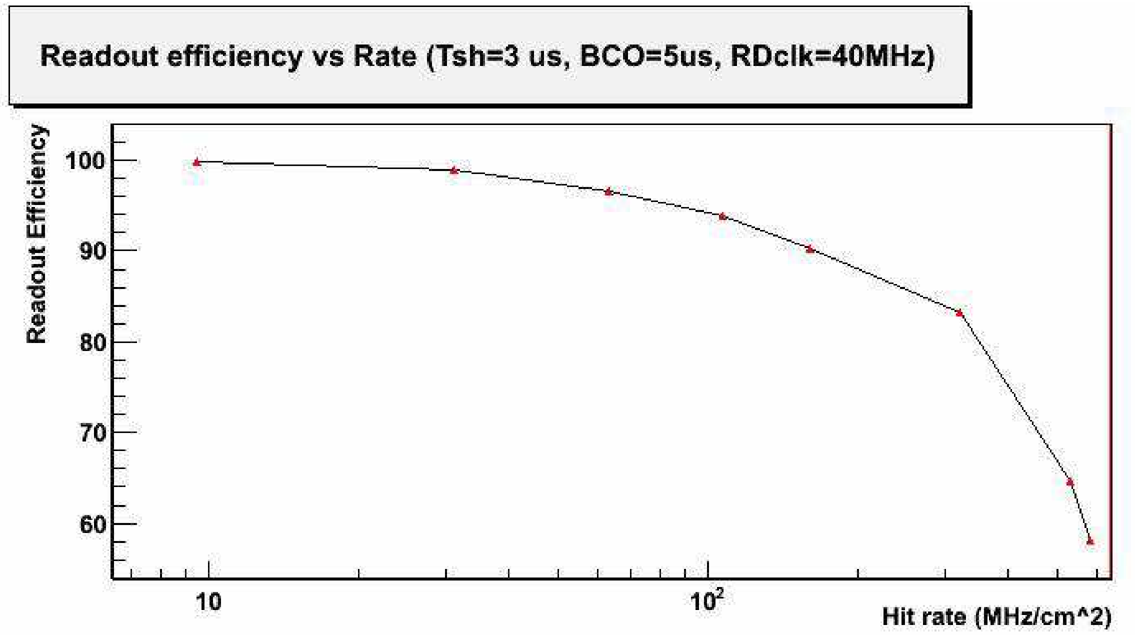


Fig. 14. Readout efficiency of the AREO v.4D architecture VS hit rate. 40 MHz of read clock and 5 μs of BC clock.

We measure the efficiency as

$$\epsilon = 1 - \frac{\nu_{blind}}{\nu_{TOT}} \quad (1)$$

where ν_{blind} is the number of hits generated on a blind pixel and ν_{TOT} is the total number of generated hits. In this case a pixel is considered blind if it is already latched or if it belongs to a frozen MP. For this particular architecture, this measure includes the hit-extraction and the hit-congestion inefficiencies.

For what concerns the results presented in Fig. 14, the inefficiency up to 300 MHz/cm^2 is dominated by the hit-extraction delay, thereafter, for higher rates, we start to observe hit-congestions that stop the matrix scan, with a resulting abrupt steepening of the curve.

In Fig. 15 we plot the efficiencies measured while varying the BC clock period. We recall that the BC clock increments the time counter and it makes start a new scan of the matrix.

In this case we see that there is a plateau extending up to about 3 microseconds, then a drastic fall in the efficiency occurs. This happens because it is more convenient to have a continuous sweeping of the matrix rather than long periods of scan inactivity. Remember that the readout is waiting for the next BC to start a new matrix scan. Thus, if the matrix scan is much faster than the BC period, then for the the most of the time hits accumulates in the matrix without being extracted. The points in the plateau (BC < 3 μs) correspond instead to a situation where the sweep is almost continuous, and then the efficiency is roughly constant. The average time that takes to the readout to perform a complete scan of the matrix is what we call the *Mean Sweeping Time* (MST). It depends on the architecture, the hit flux, the matrix dimensions and the read clock frequency. The point here is that a 5 μs-BC clock is for sure not the optimal working point for this configuration since the MST is much lower than BC period (MST $\ll$ BC).

For thoroughness we report also the readout efficiency plotted against the read clock frequency in Fig. 16.

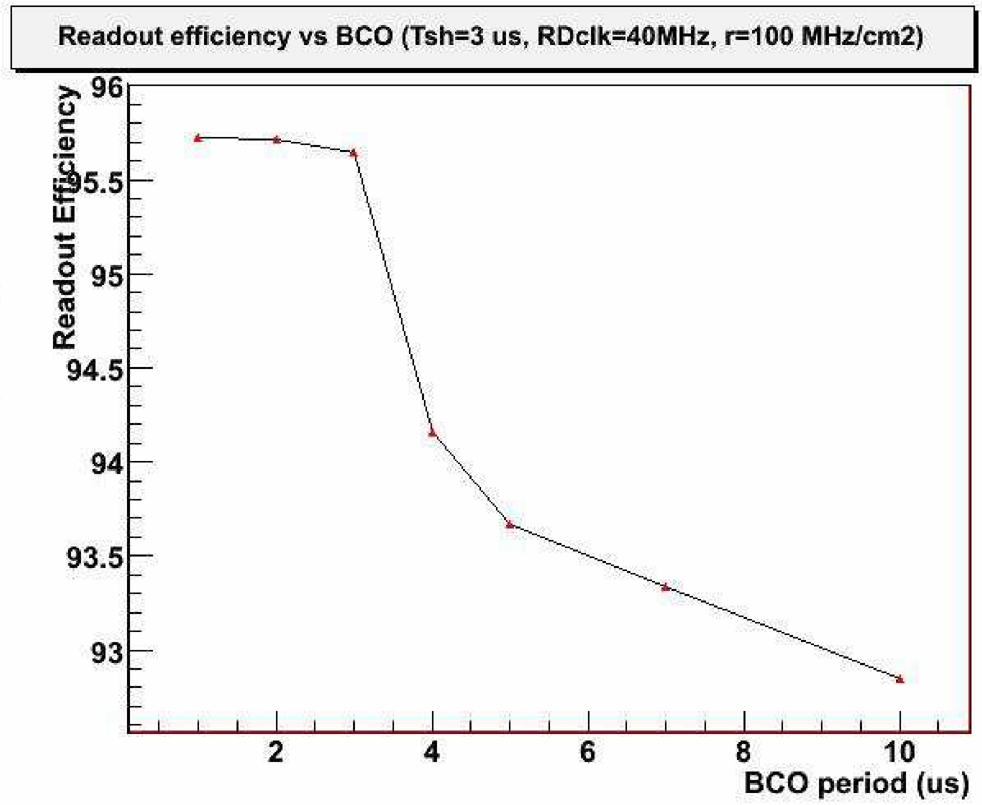


Fig. 15. Readout efficiency of the AREO v.4D architecture vs BC clock period. 40 MHz of read clock and 100 MHz/cm^2 of hit rate. The plateau within 3 μs is characterized by a continuous matrix scan operation. The efficiency drop as the mean sweeping time becomes negligible respect to the BC clock period.

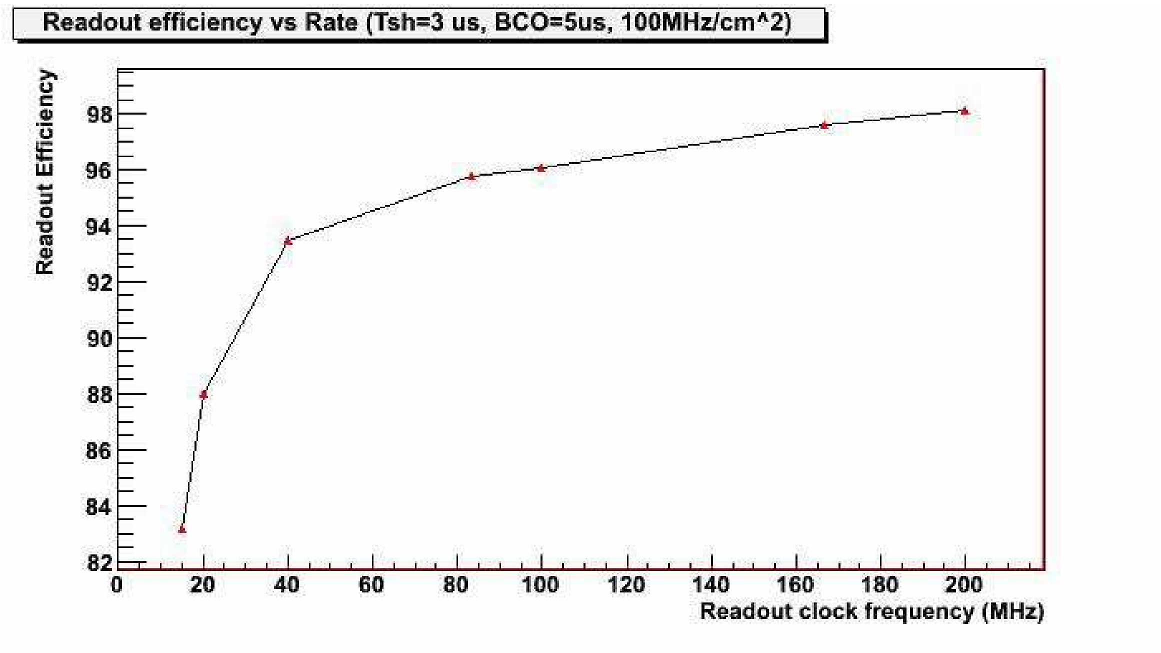


Fig. 16. Readout efficiency of the AREO v.4D architecture VS read clock frequency. 100 MHz/cm^2 of hit rate and 5 μs of BC period.

7. The SORTEX readout architecture

The experience matured during the AREO development and simulation, and then its integration in a DAQ chain (described in section 8), highlighted new possibilities of optimization.
In first place we developed a toy Monte Carlo in C++ that emulated the behavior of the matrix and readout. It was useful to run parametric scans, for example we could evaluate the dependency of the efficiency against the MP dimensions. The plot in Fig. 17 shows the efficiency against the MP x dimension (in pixels), the MP total area is preserved.

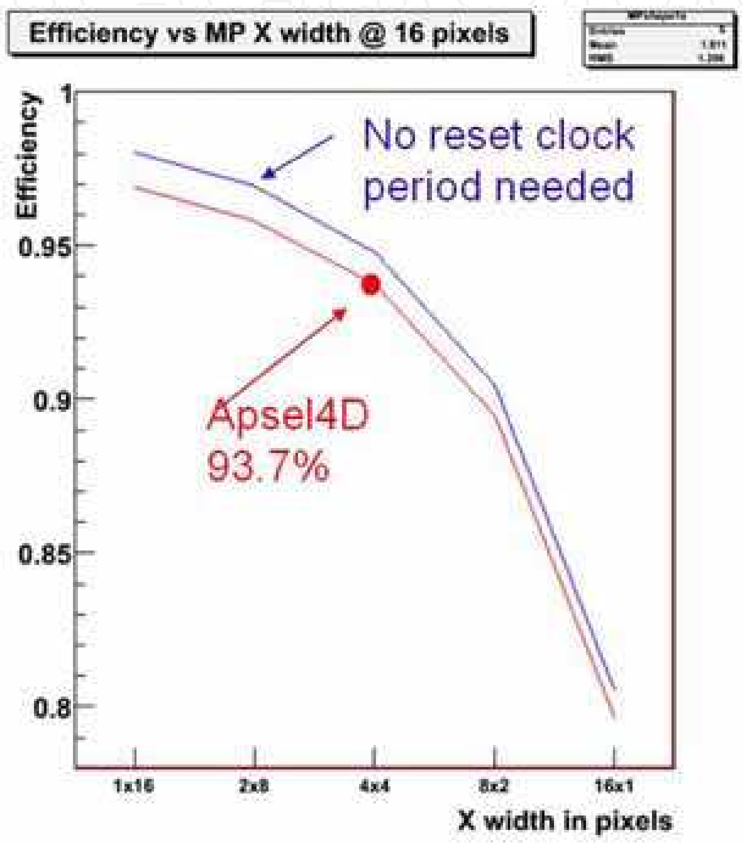


Fig. 17. Readout efficiency of the AREO v.4D architecture vs MP x dimension. 100 MHz/cm^2 of hit rate and 5 μs of BC period. The APSEL4D configuration is highlighted. The blue line doesn't take into account the clock period dedicated to the reset of the MP. Random and uniform pattern generated.

From this plot it is clear that a higher efficiency is obtained if we decrease the x dimension of the MP. This is due to the freezing of MPs and the high vertical parallelization of the architecture. Since the total MP dimension is conserved in these simulations, the dead-area induced by a single hit is the same, but it can be read out faster: A frozen 4×4 MP requires 4 clock cycles to be read out, a 2×8 MP only 2 cycles. Moreover, if we manage to remove the required reset clock cycle, we can further improve the readout efficiency.
These were the starting points for the development of a new architecture. Other consideration where done about adding extra parallelization to the architecture. We have yet a powerful vertical parallelization for the hit extraction then, foreseeing to scale towards bigger matrices, we decided to add also a horizontal parallelization.

The plot in Fig. 18 shows the efficiencies of 4 readouts running in parallel on different portions of the matrix area. The best results are achieved with the subdivision of the scanned area along the *x* dimension. It is clear that the shorter is the scan, the better is the efficiency. Thus we added the extra parallelization in this direction, by implementing 4 shorter readout scans, working on different vertical portions of the matrix area that we called *sub-matrices*. When we designed this architecture we had in mind to optimize it for a big matrix, suitable for the installation on a tracker module. The targeted final matrix dimension was 320×256 pixels with a pitch of 40 microns and a total area of about 1.3 cm^2. Each sub-matrix is then 80×256 pixel wide and covers an area of about 0.3 cm^2.

For what concern the optimization of the MP *x* size, since the expected hits on a tracker sensor can have a spatial correlation (which means that a single particle can fire two or more adjacent pixels) we decided for the 2×8 shape rather than 1×16. The Monte Carlo simulations, in fact, generated uncorrelated hits and then advantaged the thinnest configuration possible 1×16.

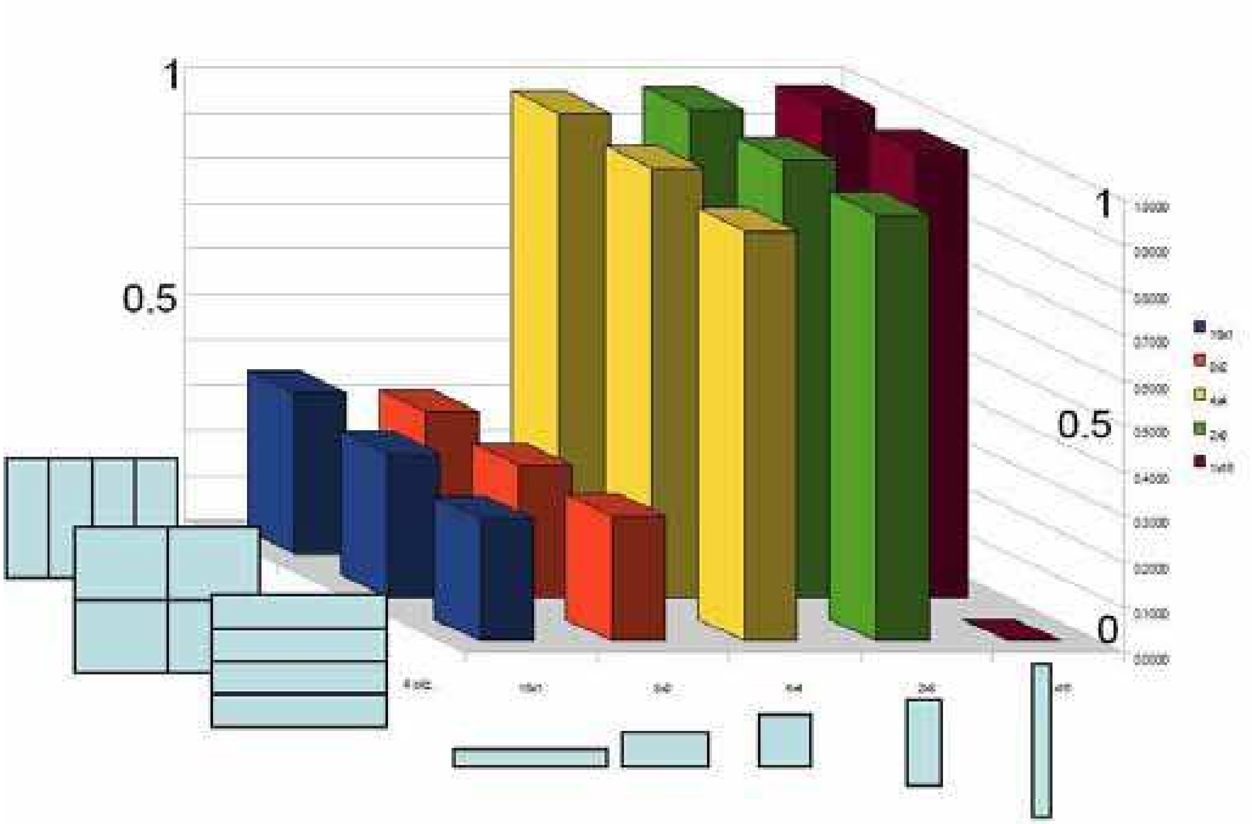


Fig. 18. Readout efficiency of 4 parallel AREO v.4D architectures VS MP *x* dimension and geometrical subdivision of the matrix area. 100 MHz/cm^2 of hit rate and 1 μs of BC period. Values are reported in Tab. 4

4×1	0.36	0.31	0.97	0.98	0.99
2×2	0.32	0.30	0.95	0.97	0.99
1×4	0.27	0.28	0.91	0.94	/
	16×1	8×2	4×4	2×8	1×16

Table 4. Table of values plotted in Fig. 18

During the integration of the AREO v.4D architecture in a DAQ system, we realized that was more difficult to re-order externally the flux of the time-unordered hits coming from the AREO architecture. This architecture in fact, doesn’t grant that hits are sent out ordered respect to their time stamp. The hit flux on output is time sorted only in the $MST \ll BC$ region, where to each time stamp (BC edge) corresponds an independent matrix scan, andwhere the hit queues have time to be fully emptied before the arrival of a new BC. Unfortunately this is the region where the architecture is less efficient, and therefore where we don’t want to make it work. We saw in fact that better efficiencies are achieved when we get close to the $MST \lesssim BC$ region. Since MST is a *mean* sweeping time, it means that its distribution can have tails above the BC period. In these tails we can have the mixing-up of time stamps in the AREO architecture. This situation is presented in Fig. 19.

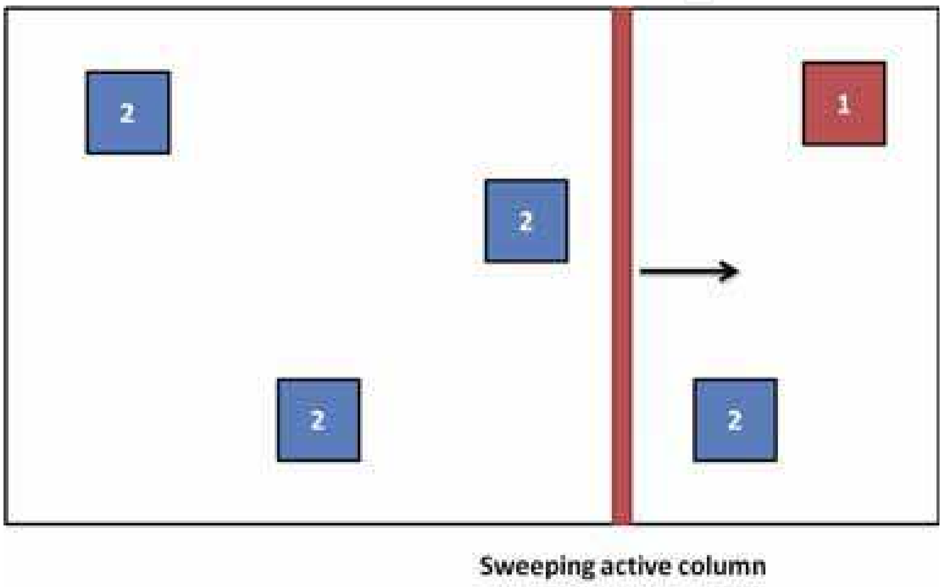


Fig. 19. Case of hit mixing in the AREO v.4D architecture. Tails of the *sweeping time distribution* allow a matrix scan to take longer than a BC period. Hence, before the end of the scan triggered by BC edge 1, a new set of MPs, tagged by BC edge 2, gets frozen and read out together.

The idea is to prevent tails to mix-up the hits in order to have a time sorted extraction of hits from the matrix. This concept inspired the name *SORTEX* (SORted hit EXtraction) of the new architecture developed.

For this purpose we introduced in this architecture the *scan buffer* element. It is a formatted FIFO memory that stores a new list of frozen MPs with its associated time stamp at each BC. The active column sweep logic pops a list out of this FIFO before starting a new scan. In this way the blue MPs drawn in Fig. 19 would not be considered during the scan number 1 since they belong to a different list.

These are tail effects and we will show through the results of simulations that the efficiency is not afflicted by this feature. It is worth to say that in this way we obtained a sorted hit extraction without any modification of the matrix logic.

Anyway, a sorted hit-extraction from the matrix is not sufficient to grant the time sorting on the output bus of the chip. It is important to avoid also the time stamp mixing in the hit dequeuing system. Till now, the *sparsifier out* component, that was shown in Fig. 12, did not consider the TS of the hits before writing them into the next *barrel out*. Therefore, in the

SORTEX architecture the *sparsifier out* is substituted by a new component called *concentrator* which takes care of preserving the time sorting of the hits before writing them into the next barrel.

Now that hits are ordered inside the barrels, we thought to exploit this feature in order to optimize the on-chip required memory and the total output bandwidth. The 8-bit TS field was removed from the hit word format, and it was replaced by a dedicated header word.

We wanted to improve the dynamic readout performances also by taking into account the spatial hit correlation that one would expect due to the physical nature of the signal. A cluster optimization algorithm was then introduced in the SORTEX architecture. The active column has been subdivided into 8-bit wide *zones*. Each sparsifier, that in the AREO architecture encoded a column of 8 pixels simultaneously, now encodes 8 zones per clock cycle, which means a column of 64 pixels. Thus 4 sparsifiers only can encode the whole of the active column.

The hit word now encodes the hit y coordinate by the zone address and the zone pattern itself. This technique is advantageous in case of clustered events. Respect to a classical direct xy encoding, the zone encoding technique increases the length of an encoded hit but in case of clustered events it reduces the total number of required hit words. In the AREO architecture the width of the y address is given by $\log_2 H$ where H is the height in pixels of the matrix (binary encoding). With a zone sparsification algorithm, the y pixel address is given by the zone pattern, which is W bit wide, and by the zone address which is only $\log_2 (H/W)$ bit wide. Then the hit word is incremented by the quantity Δ given by:

$$\Delta = \log_2 H / W + W - \log_2 H = W - \log_2 W \qquad (2)$$

The increment Δ is small for small W, and the hit word can transport the information of up to W hits. An example of the application of the zone encoding algorithm is shown in Fig. 20.

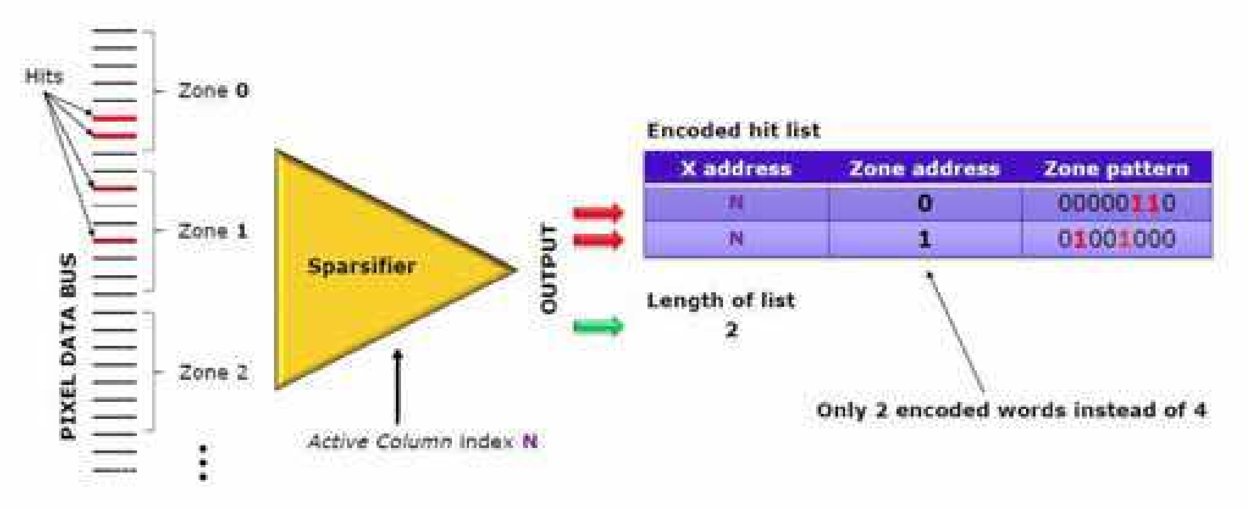


Fig. 20. Example of zone sparsification algorithm.

We want to show now, with the help a concrete example, the benefits brought by the time sorting and the zone sparsification algorithms. Let us apply to the 320×256 matrix the hit encoding scheme adopted in the AREO architecture. Then the hit-word length L is:

$$L = X_{addr} + Y_{addr} + TS = \log_2 320 + \log_2 256 + 8 = 9 + 8 + 8 = 25\text{bits} \qquad (3)$$

and the produced data rate *R* is:

$$R = L \cdot C_f \cdot \Phi \cdot A = 25\text{bits} \times 4 \times 25\text{Mtracks}^{-1}\text{cm}^{-2} \times 1.3\text{cm}^2 = 3.2\text{Gbps} \tag{4}$$

where C_f is an hypothetic cluster factor of 4, Φ is the particle flux and *A* is the sensor area. Now if we introduce the time sorting of the hits, and assuming that each leading TS word is followed by about 10 hits[3] we have

$$L = X_{addr} + Y_{addr} = 9 + 8 = 17\text{bits} \tag{5}$$

and

$$R = L \cdot C_f \cdot \Phi \cdot A \cdot 1.1 = 2.4\text{Gbps} \tag{6}$$

The 1.1 factor is the rate increment due to the presence of the TS words. Now let us introduce also the zone sparsification algorithm. In order to simplify calculations we assume that the cluster factor has a fixed shape of 2×2 pixels. The number of hit words that need to be sent depends on the overlapping of the cluster shape on the zones grid. There are 8 possible geometrical configurations, only in 1 of them the cluster overlaps 4 different zones. In the remainder 7 configurations only 2 hit-words are produced. It is possible then to evaluate the data rate *R* with a weighted average over the possible configurations.

$$R = [2(L+\Delta) \times \frac{7}{8} + 4(L+\Delta) \times \frac{1}{8}] \cdot \Phi \cdot A \cdot 1.1 = 1.8\text{Gbps} \tag{7}$$

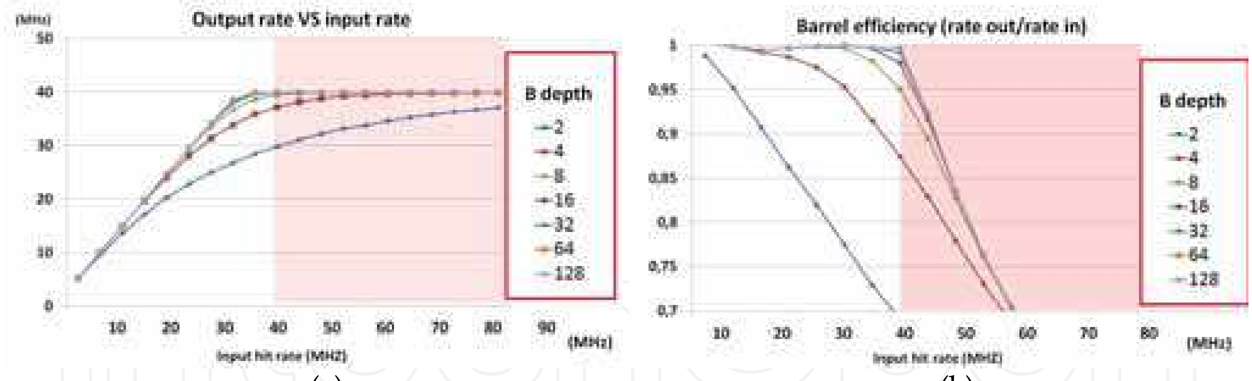
where Δ = 5 is the increase in word length due to the zone sparsification, considering that the SORTEX architecture adopted *W* = 8. (ref. to eq. 2).

1.8 Gbps vs 3.2 Gbps corresponds to a considerable 45% reduction of the output bandwidth.

The reduction of the produced data rate, can bring significant improvement also in the on-chip required memory. The SORTEX architecture, as well as the AREO one, is data-push which means that there is not an extensive memory that buffers the hits during the latency of an external trigger. Though buffering is useful in other situations, for example in case of rate-bursts. In these cases, that represent fluctuations over the average hit rate, for a short period of time the hits are produced at a higher rate respect to the output data bus bandwidth.

We realized a brief study on the optimal barrel depths. In Fig. 21.a we plotted the barrel output rate against the input rate. In ideal conditions, where no hits are lost, the two values should be equivalent until the maximum output rate of the barrel is reached. We recall that the barrel can store more than 1 hit-word per clock cycle. The saturation limit at 40 MHz corresponds to the output bandwidth of a barrel driven with a 40 MHz clock. In these non-ideal conditions instead, we observe hit losses even when the mean input rate is below the 40 MHz. This losses are due to the statistical fluctuations above the mean, it is possible to observe how they decrease as the barrel gets deeper.

[3] value not far from that expected with Φ=100 Mhit s^{-1} cm^{-2}, *A*=1.3 cm^2 and 1 μs of BC.



(a) (b)

Fig. 21. In (a): Output rate vs input rate of a barrel. In (b): barrel efficiency vs input rate. 40 MHz read clock, in the pink region the input rate is grater than the output bandwidth of the barrel.

We divided the barrels into 2 categories: the 2nd layer barrels or *B2*, which are connected to the spasrifiers and which receive data from a 16th of the matrix (4 barrels for each sub-matrix). And the 1st layer barrels that gathers data from the 4 B2s of a sub-matrix (refer to Fig. 22). A B2 must sustain a hit rate of 130 MHz/16 = 8.2 MHz, and a B1 8.2 MHz × 4 ≈ 32 MHz. Looking at those values on the *x* axis in Fig. 21.b, we opted for a depth of 8 for the B2s and a depth of 64 for the B1s. In this plot we expressed the efficiency as the ratio between the output and the input rate. A zoom in the knee region of the efficiency plot highlights the performance of different barrel depths.

Since we have now 4 independent readout units, and thus 4 *barrel out* equivalents (B1s), we introduced a new component called *final concentrator* that drives the output data bus. It performs a round robin cycle over the 4 B1s in order to extract all their data relative to a certain TS.

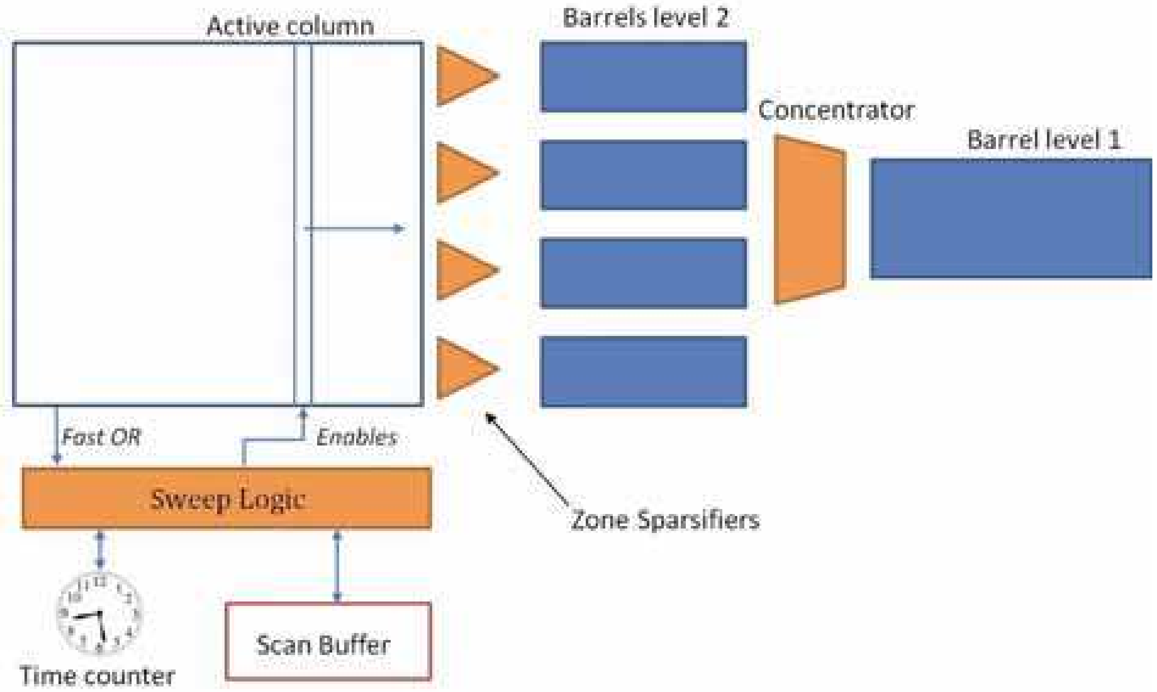


Fig. 22. SORTEX readout for a single sub-matrix

The final concentrator then, empties one B1 at a time, extracting first the leading header words containing the TS information. Before forwarding it to output bus, it adds a 2-bit formatted field that individuates the sub-matrix origin. The format of a header word on the output bus is shown in Tab. 5. The whole bunch of hits that follows the TS header in that B1 is forwarded to the output bus. When the following TS header-word is found, the final concentrator move to the next B1. The hit-word formats is presented in Tab. 6. We mention that the B1s and the final concentrator are fed by a dedicated clock-tree net, so that they could run at higher frequencies respect to all the rest of readout.

hit field	length	name	function
hit[21]	1 bit	DV	1 = data valid
hit[20]	1 bit	Header bit	1 = header word
hit[19:10]	10 bits	/	unused
hit[9:8]	2 bits	SM addr	sub-matrix address
hit[7:0]	8 bits	TS	time stamp

Table 5. Header word format. SORTEX architecture divided into 4 sub-matrices.

hit field	length	name	function
hit[21]	1 bit	DV	1 = data valid
hit[20]	1 bit	Header bit	0 = hit word
hit[19:18]	2 bits	Sp addr	sparsifier address
hit[17:15]	3 bits	Z addr	zone address
hit[14:8]	7 bits	X addr	x address (sub-mat. relative)
hit[7:0]	8 bits	TS	time stamp

Table 6. Hit word format. SORTEX architecture on 320×256 pixel matrix.

The SORTEX architecture is controlled by an I^2C-like interface that allows to read and write two sets of 16-bit registers, one can be accessed in read/write mode (RW) and the other set is read-only (RO). This interface was developed to reduce the footprint of slow control I/O pads foreseeing to mount several sensor chips on a module. In the AREO v.4D architecture, slow control required a dedicated clock, an SC mode 3-bit bus, then an 8-bit input data bus and an 8-bit output data bus. With an I^2C standard interface only 2 bidirectional pins are required. They can be connected to a bus shared by several sensor chips. The I^2C communication is based on a master that imparts clock and orders, and a slave that executes and answers. Only the master entity, which for us is the module controller, can start a communication by addressing a chip on the bus. Each chip is able to recognize its own address embedded in the transmission protocol. We implemented only few characteristic of the I^2C standard and that is why we use the "like" suffix. For example this communication protocol foresee a multi master initial negotiation, in case that several masters are connected on the same bus. In our case the master is always the module controller (or DAQ master) and the sensor chips are always the slave counterparts. A schematic representation of the foreseen I^2C interconnection scheme is shown in Fig. 23.

All the slow control operations are implemented through register R/W operations. Acquisition parameters and settings are mapped on the RWregisters, while acquisition monitors and flags can be read in the RO registers. A special RW register is the command register, a special slow control unit execute the instructions written in this registers. For example a testing routine that stimulates in sequence all the inputs of the sparsifiers has been implemented. It is possible to check if the outgoing pattern matches with that produced by the stimulus.

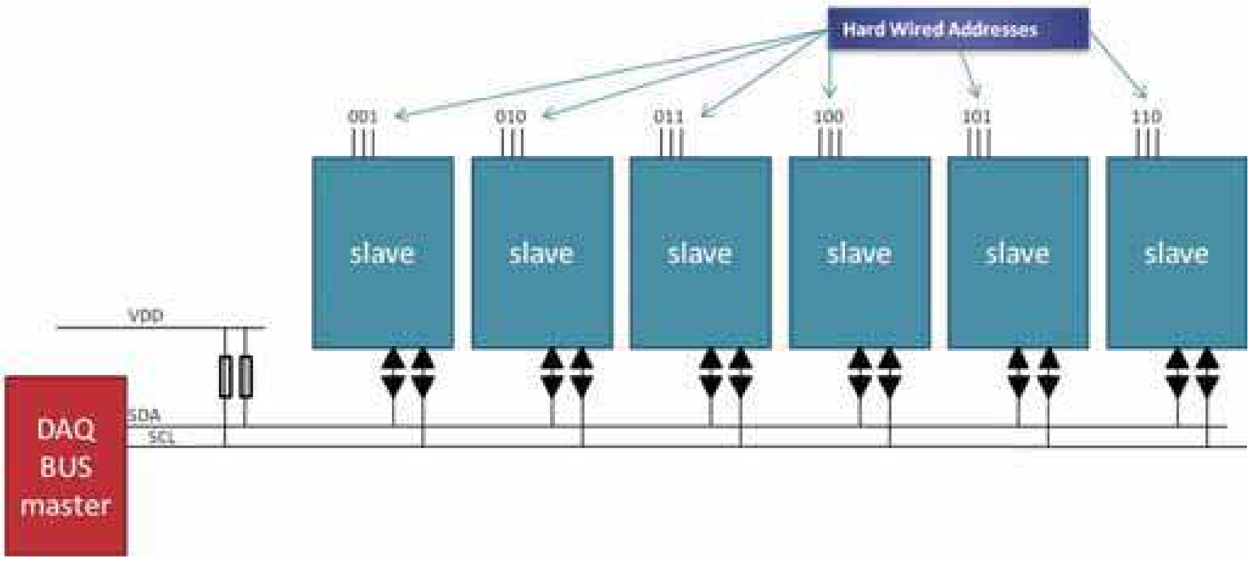


Fig. 23. I^{2}C bus scheme.

For SORTEX architecture verification and for efficiency evaluation purposes, we realized a full VHDL test bench with integrated Monte Carlo generator. We simulated several milliseconds of acquisition that corresponds to about 500k events. Three main reports are produced by these simulations. The first is the Monte Carlo generation report, that includes all the generated events. Then we produced a pixel latch report, reporting all the hits that were latched on the matrix. The third one is the output report, a list of the hits sent out on the data bus. The hits lost between the MC generation report and the pixel latch report are considered as hit-extraction inefficiency. They were lost due to a frozen MP or an already hit pixel. Hits lost by readout are considered as an acquisition fault. In Fig. 24 we report the hit extraction efficiency of 1 sub-matrix SORTEX readout plotted against read clock frequency and BC clock. In these simulations we never observed faulty hit losses deriving from a misbehaviour of the architecture.

The efficiency drop observed at point BC=250 ns and RDclk=60 MHz was due to repetitive scan buffer overflows. In that point the MST ≈ BC and the scan buffer depth is not sufficient to absorb the statistical fluctuations. It is important to state that a scan buffer overflow does not imply a loss of hits. When no space is available in the scan buffer, if there are new MPs that need to be frozen they are left active for the next BC period until an empty memory location is available. If the readout ”loose” one BC, the event tagged with the successive time stamp will include also the hits belonging to the previous time window. In this case no hit information is lost, the only inconvenient is a worse time resolution and a drop in the efficiency. The efficiency drop is due to slower scans since, in these situations, it happens often that a MP list refers to a longer acquisition time. Anyhow, the readout is not intended to work in this conditions, the results presented wanted to point out the performance limits of this architecture.

In Fig. 25 we plotted the efficiencies obtained with the full SORTEX architecture on the 4 sub-matrices. The same reverse trend under 200ns of BC is observed due to scan buffer overflows.

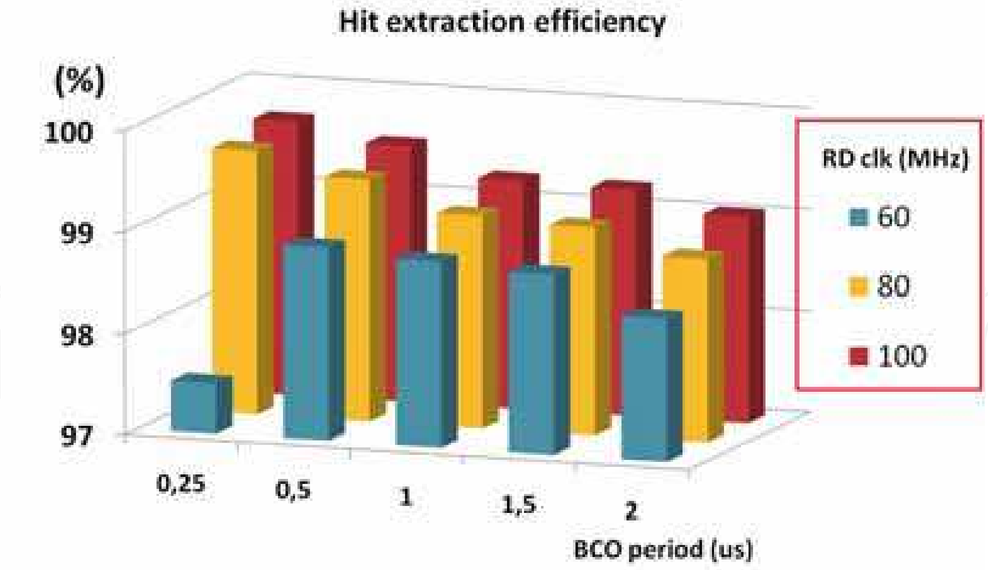


Fig. 24. Hit extraction efficiency of SORTEX architecture vs BC clock and read clock. 1 submatrix simulated only. Hit rate 100 MHz/cm².

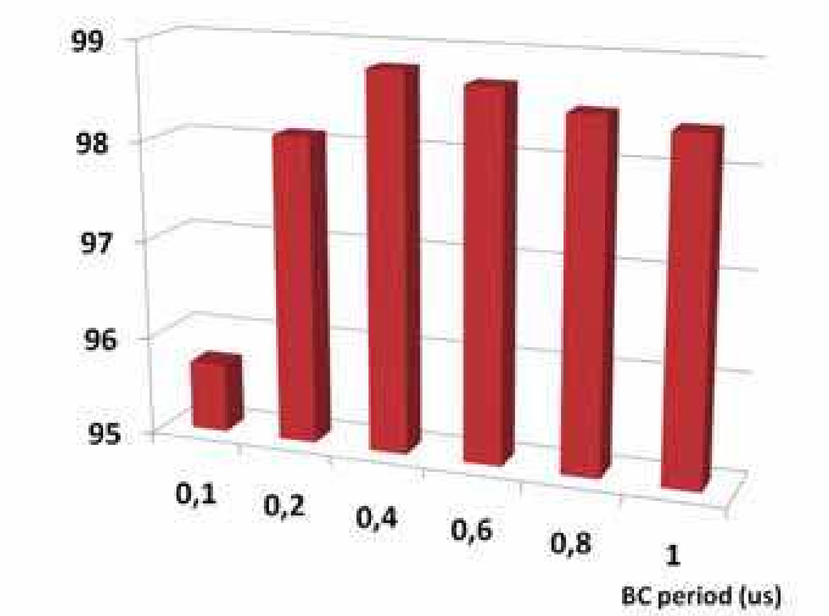


Fig. 25. Hit extraction efficiency of full SORTEX architecture vs BC clock. Read clock 67MHz, output stage clock 200 MHz. Hit rate 100 MHz/cm².

8. DAQ integration

The data-push architectures described, are designed to sustain intense hit fluxes, then producing high data rates (order of 2 Gbps per chip). A robust and powerful DAQ system then must be provided in order to handle the considerable amount of data received by the front end chips.

We present a high data rate acquisition system that was involved also for the beam tests of the APSEL4D chip [M. Villa for the SLIM5 Collaboration (year 2009)]. The data acquisition was done by means of two high bandwidth, fully programmable 9U VME board (EDRO) that have been designed to stand a 12 Gbit/s input rate, 1.2 Gbit/s output rate and have the possibility to perform different types of trigger strategies on data. The most important one was the on-line track identification performed with the help of an Associative Memory board [G. Batingani et Al. (year 2008)], which demonstrated the capability of the setup to trigger on identified tracks with a minimal latency ($< 1\mu s$).

The EDRO board is based on FPGAs (Field Programmable Gate Arrays), a picture is presented in Fig. 26. The acronym stands for Event Dispatch and Read Out. It is a 9U VME master board holding 5 mezzanine boards mounted on piggy-back. It is capable of an integrated input/output of 30 Gbps.

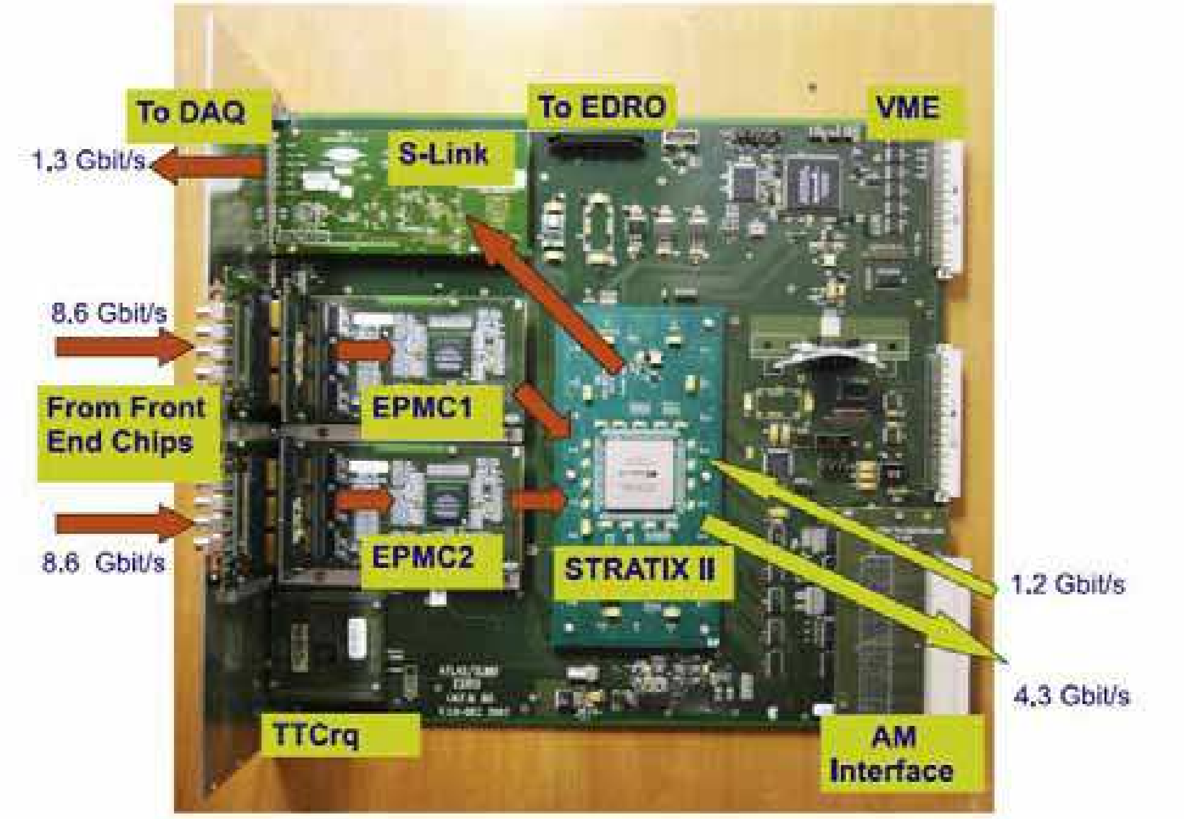


Fig. 26. EDRO board picture. (Event Dispatch and ReadOut).

A TTCrq mezzanine card [G.B. Taylor (2005)] developed for LHC experiments has been used as a 40 MHz clock source. Two Programmable Mezzanine Cards (EPMC) are responsible for the communication to/from the front-end chips. They host an Altera Cyclone II FPGA in a BGA package and several LVTTL/LVDS converters for the communication to/from the front-end chips.[4] Each EPMC can handle 2 Apsel4D chips. The limit is imposed by the high number of I/O required by the AREO architecture rather than the front-end data rate, since each EPMC can handle up to 8.6 Gbps.

Internal logic and most of the on-board data transfer run at 120 MHz clock, ensuring a data input/output of the order of 12.4 GBit/s. The hits collected from the EPMCs are forwarded

[4] Differential signaling is used on the 30 m cable that connect the EDRO board to the electronics in the experimental area. There, the signals are converted back to single ended CMOS to be connected to the APSEL4D digital I/O

to the main mezzanine of the EDRO board: an 18 layers board holding an Altera Stratix II FPGA with 1508 pins, developed for the CMS muon finder [J. Ero et Al. (year 2008)]. The large number of logical elements (> 100k) and memory (> 6Mbits) of the FPGA have been exploited to implement the event building and triggering process running at 120 MHz with minimal inefficiencies. Hits from the EPMCs can be forwarded to the external associative memory board by using LVDS serializer/deserializers. Triggered events are first stored on long local buffers and then forwarded to the last piggy back board, called S-Link LSC (Link Source Card) [H.C. van der Bij et Al. (year 1997)], developed at CERN for the data sending to the final DAQ PC. A set of connectors for EDRO-EDRO communication, EDROAM communication and input/outputs from LEMO signals completes the board.

These boards have been intensively used for the data acquisition from the chips featuring an AREO architecture. We are now developing few hardware and firmware upgrades to these boards, foreseeing to test other devices based on newer readout architectures like SORTEX.

9. Conclusions

The next generation of particle accelerators, that are being designed for new discoveries in the world of high energy physics, opened new challenges in the design of appropriate detectors. The high particle fluxes that are expected in proximity of the interaction region, require extensive researches in the field of silicon tracker detectors. We presented an introduction on silicon sensors and, in more details, on silicon pixel detectors. New opportunities are given by the technological improvement of the silicon industry, the 3D, or vertical integration for example is a promising process for hyper-integrated systems.

In this context we described our approach in the design of innovative digital readout architectures. We presented mainly two proven readout architectures, AREO and SORTEX. AREO architecture has been implemented on several chips thanks to the SLIM5 collaboration. We have mentioned the APSEL chip family, in particular the 3D and 4D versions characterized by different matrix dimensions (256 and 4096 pixels respectively) but same sensor technology (planar CMOS MAPS sensor). The AREO architecture has been adopted, within the same collaboration framework, also to build a vertically integrated MAPS sensor. At the moment, the 32×8 sensor matrix and the readout layers are still in production. An efficiency study on the 4D version of the architecture was presented, and it showed an efficiency of about 96% with a flux of 100 Mhit cm^{-2} s^{-1}. We have put this target since it seems to be the most probable value at 1÷2 cm of radii from the interaction point of the next generation, highest-luminosity B-factories like SuperB (10^{36} cm^{-2} s^{-1}).

We described then the SORTEX architecture designed for wider matrices (320×256 pixels) and born on the experience matured with the APSEL chip family based on AREO. The new architecture is characterized by several innovations suggested also by a more exhaustive simulation campaign. A higher parallelization and a higher optimization of the sparsification algorithms allowed to raise the efficiency over 98% with an increased area (read as global rate) increased of a factor greater than 10. The SORTEX architecture has been adopted on a reduced 32x128 matrix sensor realized as a hybrid pixel sensor by the VIPIX collaboration. The CMOS layer implementing the front-end electronics and digital readout has just come out of foundry.

We are planning the design of a new architecture based on a different matrix logic. Preliminary simulations showed a very promising hit-extraction efficiency above 99% and an improved time resolution capability, this architecture plans to take great advantage from the new vertical integration technology.

A powerful DAQ board has also been presented, a high bandwidth 9U VME board based on high-performance FPGAs for event building and triggering. It was developed to handle a front-end rate of more than 16 Gbps, and it was provided with a 4.3 Gbps associative-memory interface for a high-speed track identification support. A couple of this boards, called EDRO, have been integrated in the data acquisition system for the beam test set up of the APSEL4D chip. We intend to go further with the innovation of pixel readout circuits and with their integration in high performance DAQ systems, giving the scientific community our little contribution for a chance of new discoveries.

10. References

A. Gabrielli for the SLIM5 Collaboration (year 2008). Proposal of a sparsification circuit for mixed-mode MAPS detectors, *Nuc. Instr. and Meth. in Phys. Res. A* 596: 93–95.

A. Gabrielli for the SLIM5 Collaboration (year 2009). A 4096-pixel MAPS device with on-chip data sparsification, *Nuc. Instr. and Meth. in Phys. Res. A* 604: 408-411.

G. Batingani et Al. (year 2008). The associative memory for the self-triggered slim5 silicon telescope, *IEEE Nucl. Sci. Symp. Conf. Record 2008* pp. 2765 – 2769.

G. Rizzo for the SLIM5 collaboration (year 2007). Recent development on CMOS monolithic active pixel sensors, *Nuc. Instr. and Meth. in Phys. Res. A* 576: 103–108.

G.B. Taylor (2005). *Timing, Trigger and Control (TTC) System for the LHC Detectors,* http:// www.cern.ch/TTC/intro.html.

H.C. van der Bij et Al. (year 1997). S-link, a data link interface specification for the lhc era, *Nuc. Sci. IEEE Trans.* 44-3: 398–402.

J. Ero et Al. (year 2008). The CMS drift tube track finder, *J. Inst. 3 P08006* .

L. Gaioni et Al. (year 2009). A 3d deep n-well cmos maps for the ilc vertex detector, *Nuc. Instr. and Meth. in Phys. Res. A* doi:10.1016/j.nima.2009.09.041.

M. Villa for the SLIM5 Collaboration (year 2009). The l1 track trigger and high data rate acquisition system for the SLIM5 beam test, *IEEE Nucl. Sci. Symp. Conf. Record 2009.*

N. Neri et Al. (year 2010). Deep N-well MAPS in a 130 nm CMOS technology: beam test results, *Nuc. Instr. and Meth. in Phys. Res. A* doi:10.1016/j.nima.2010.02.193.

R. Klingenberg for the ATLAS pixel collaboration (year 2007). The ATLAS pixel detector, *Nuc. Instr. and Meth. in Phys. Res. A* 579: 664–668.

R. Lipton (year 2007). 3D-vertical integration of sensors and electronics, *Nuc. Instr. and Meth. in Phys. Res. A* 579: 690–694.

S. Bettarini et Al. (year 2007). Development of deep N-well monolithic active pixel sensors in a 0.13 μum CMOS technology, *Nuc. Instr. and Meth. in Phys. Res. A* 572: 277–280.

S. Schnetzer for the CMS Pixel Collaboration (year 2003). The CMS pixel detector, *Nuc. Instr. and Meth. in Phys. Res. A* 501: 100–105.

SuperB Collaboration (2007). *SuperB Conceptual Design Report,* http://arxiv.org/abs/ 0709.0451v2. V.

Re et Al. (year 2010). Vertically integrated deep N-well CMOS MAPS with sparsification and time stamping capabilities for thin charged particle trackers, *Nuc. Instr. And Meth. in Phys. Res. A* doi:10.1016/j.nima.2010.05.039.

W-M Yao et Al. (year 2006). *J. Phys G: Nucl. Part. Phys.* 33: 284–285.

Data Acquisition
Edited by Michele Vadursi

ISBN 978-953-307-193-0
Hard cover, 344 pages
Publisher Sciyo
Published online 28, September, 2010
Published in print edition September, 2010

The book is intended to be a collection of contributions providing a birdâ€™s eye view of some relevant multidisciplinary applications of data acquisition. While assuming that the reader is familiar with the basics of sampling theory and analog-to-digital conversion, the attention is focused on applied research and industrial applications of data acquisition. Even in the few cases when theoretical issues are investigated, the goal is making the theory comprehensible to a wide, application- oriented, audience.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821