Alma Mater Studiorum – Università di Bologna

DOTTORATO DI RICERCA IN

Ingegneria Elettronica, Telecomunicazioni e Tecnologie dell'Informazione

Ciclo XXVI

Settore Concorsuale di afferenza: 09/E3 - ELETTRONICA

Settore Scientifico disciplinare: ING-INF/01 - ELETTRONICA

TITOLO TESI

The APSEL4D Monolithic Active Pixel Sensor and its Usage in Single Electron Interference Experiment

Presentata da: Alberghi Gian Luigi

Coordinatore Dottorato

Prof. Alessandro Vanelli-Coralli

Relatori

Prof. Guido Masetti

Prof. Mauro Villa

Esame finale anno 2013

Contents

Introduction

Present and future high energy physics experiments, such as those currently taking place at the C.E.R.N. international laboratories, require the development of particle detectors with higher and higher capabilities. In particular, individual charged particles must be identified with very high demands on spatial resolution and timing. 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 that can be fitted 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. The innermost part of the high energy physics detectors, called vertex detectors, are currently based on silicon sensors with hybrid pixel matrices. A possible upgrade in the coming years requires the implementation of Monolithic Active Pixels, as this technology makes it possible to get rid of the delicate bump-bonding procedure needed for hybrid pixels, by integrating both sensor and readout on the same substrate, thus also reducing dramatically the material budget. These kind of devices are usually known as MAPS (Monolithic Active Pixel Sensors).

A common feature of pixel detectors is that when a pixel gets fired by a crossing particle, it is unable to detect any other impinging particle until it is read out and reset. This time lapse, during which the pixel is latched, is called dead time. In some specific cases the dead-time introduced by the reading of the collected charge is not affordable and consequently the readout is built to extract only the hit/no hit information from the pixels. The chip we will consider in the following presents this feature. The SLIM5 collaboration of I.N.F.N. has in fact developed a Monolithic Active Pixel Sensor, called APSEL4D, which is composed of a bidimensional matrix of 4096 Monolithic Active Pixels with a digital readout sparsification circuit, located outside the matrix area. The main goal of a sparsified readout architecture is the association of a spatial and time coordinate to each fired pixel, the term

sparsified meaning that the hit extraction and encoding is focused on sparse randomly-accessible regions of the matrix, where the presence of fired pixels is known. 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, the sparse and randomly accessible regions are the pixels themselves. The pixel dimension is 50×50 μm^2 and the readout logic is based on standard cells with the implementation of a parallel reading technique. The sensor cell is active, which means that the front end of the pixel is driven by active electronic components like a preamplifier, a shaper and a discriminator; in addition it is monolithic as it integrates a standard CMOS read-out logic. In order to minimize the digital lines crossing the active area, the matrix is organized in Macro Pixels (MP) i.e. square groups of 4×4 pixels. Each MP has only two private lines for a point-to-point connection to the sparsification logic. The readout logic performs the hit extraction from the matrix, encodes the space-time coordinates and finally produces the digital hit-stream that is sent out of the chip sensor.

The original work of this thesis was aimed at developing an online Data Acquisition framework both for characterizing APSEL4D, using a Transmission Electron Microscope and a Beam Test at CERN, and for employing the sensor, with its high space time sensitivity, in order to perform a single electron Young's interference experiment, aimed at showing in a very simple setup one of the most striking aspects of Quantum Mechanics: the wave-particle duality for single particles.

With this target in mind, we developed a transition board for signal enhancing and for voltage level translation and a firmware code to be implemented in the FPGA used for the data input/output management and for the chip control. An XEM3050 general purpose board provides the necessary connection with a personal computer, through an USB 2.0 port, and with the chip, through two high-density expansion connectors, bringing the necessary signals to and from APSEL4D. A Graphical Interface Unit, based on windows and widgets, was also developed by writing a C++ code extended by Qt and Qwt libraries. Its purpose is the online full chip control and data acquisition management. The data acquisition chain was finally used for the APSEL4D characterization with an electron beam inside a Transmission Electron Microscope and with a proton beam at CERN. After that, a single electron Young's experiment was realized, allowing for the first time to have the statistical distribution of the arrival time of the interfering electrons.

Let us now resume the content of the thesis. In chapter 1 the general principles of radiation detection are presented and in chapter 2 we specialize to silicon detectors and introduce the CMOS transistors. Chapter 3 is dedicated to pixel detectors for high energy physics, and in particular, at the end

of the chapter, we introduce the Monolithic Active Pixel Sensors. In chapter 4 we describe the chip APSEL4D and its characterization process. In chapter 5 we turn to the description of the data acquisition framework, and conclude in chapter 6 with the single electron interference experiment.

Chapter 1

Principles of radiation detection

The behavior of any system built for radiation detection (see [1]-[5]) depends on how the particular radiation which is intended to be detected interacts with the detector constituents. Understanding the fundamental mechanisms through which the particles interact, and in particular lose energy, on passing through matter, allows to determine which combination of detection material and readout electronics is best suited. This section is an overview of the interaction mechanisms mainly for four kinds of particles:

- heavy charged particles (α-particles, protons);
- electrons;
- photons (γ-rays, X-rays, UV and visible);
- neutrons;

The first three kinds can be directly revealed with a semiconductor detector by using a few different interaction mechanisms, whereas neutrons, and in general uncharged massive particle, need the use of some special conversion materials.

1.1 Heavy charged particles

Heavy charged particles interact with matter mainly through the Coulomb force acting on the orbital electrons bound to the nuclei. The charged particles traversing a detection medium transfer their energy to many electrons at the same time. This causes the electrons to gain energy and thus to get excited into a higher orbital shell or, for the highest energy transfer, to leave the atom and produce ionization. This way the incoming charged particle is

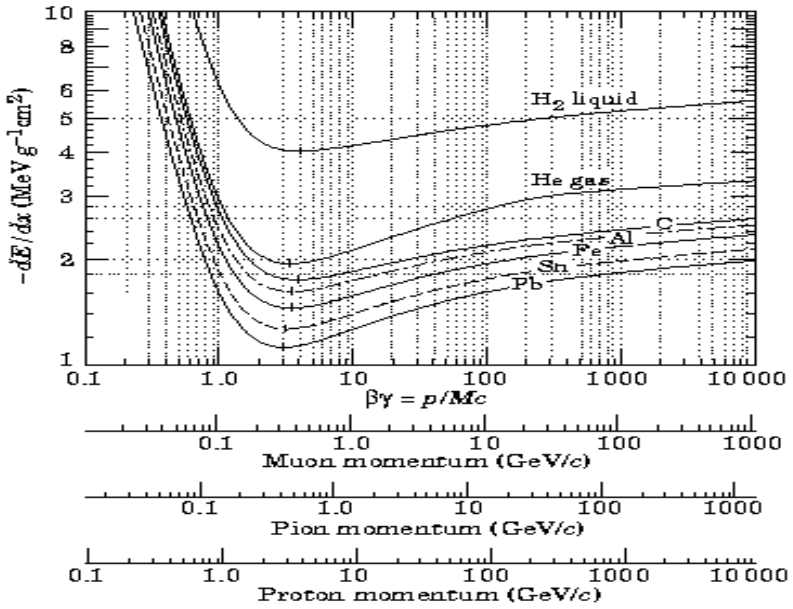


Figure 1.1: The Bethe Bloch Curve represents the specific energy loss of particles in a medium

gradually slowed down until it stops. The specific energy loss (or stopping power) of the particles in the medium, known also as "stopping power of the medium", is described by the Bethe equation, which is displayed in Fig. (1.1),

$$S = -\frac{dE}{dx} = \frac{4\pi e^4 z^2}{m_e v^2} N Z \left[\log\left(\frac{2m_e v^2}{I}\right) - \log\left(1 - \frac{v^2}{c^2}\right) - \frac{v^2}{c^2}\right] \qquad (1.1)$$

where

- v and $z\,e$ are the velocity and charge of the incoming particle,
- m_e is the electron rest mass,
- N is the number of atoms per unit volume,
- Z is the atomic number,
- I is the average ionization and excitation potential of the absorption material, which is an experimentally determined parameter.

A plot of the specific energy loss along the track of a charged particle is known as the Bragg curve. An example of this kind of energy loss for alpha

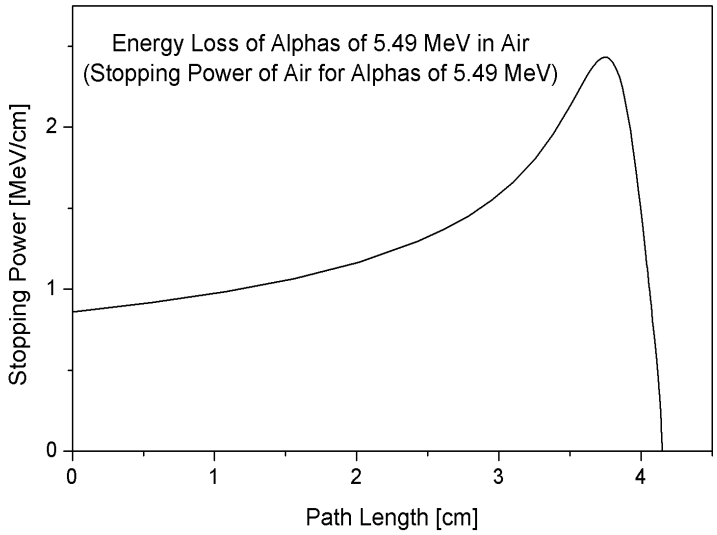


Figure 1.2: The Bragg Curve represents the specific energy loss along the track of a charged particle

particles in air is illustrated in Fig.(1.2). One can explain the displayed behavior: when the particle has high velocity ($\beta \simeq 1$) the specific energy loss is constant. Then, as the velocity decreases, the interaction time with the medium constituents increases and the energy transfer becomes more effective: this results in a peak. The charged particle thus starts losing energy at a higher and higher rate until it finally stops.

For the case of positively charged particles, as is the case displayed in fig.(1.2), the physical explanation for the observed behavior is, more specifically, this: at the beginning of the path a charged particle tends to capture electrons from atoms of the medium it passes through, then the particle gradually loses energy, with the consequence of an increase in the interaction cross section, and this results in a peak. Finally, near the end of the track, the particle charge is reduced through electron capture and the curve falls off.

1.2 Fast electrons

The specific electron energy loss for fast electrons is described by an expression similar to the Bethe equation. Contrary to heavy charged particles, electrons follow erratic paths and can either lose their energy in a single col-

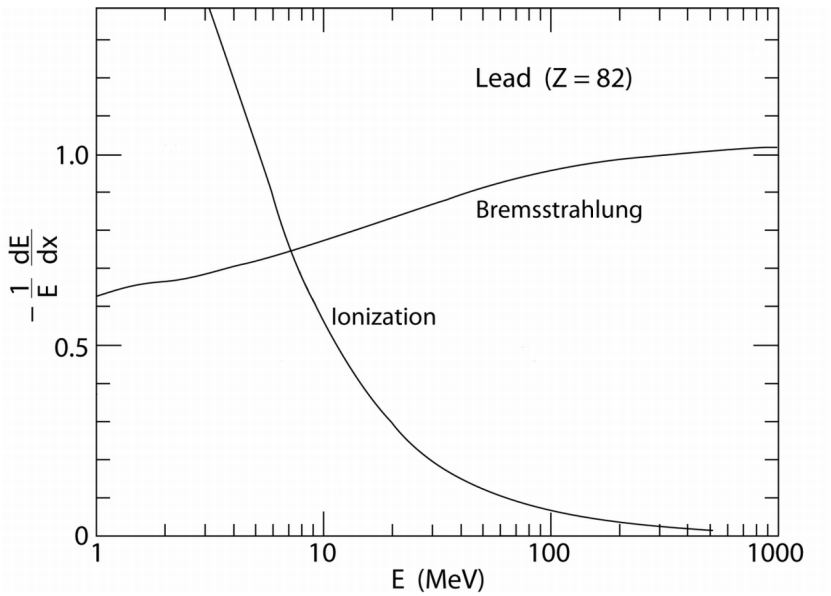


Figure 1.3: The radiation energy loss of electrons for ionization and bremsstrahlung processes (in units of cm^{-1}) as a function of the electron energy.

lision or through multiple scattering, with sharp changes of direction (this behavior is also known as "random walk"). The fast electrons not only lose their energy by Coulomb interactions: due to their small mass, a different energy loss mechanism comes into play: the emission of electromagnetic radiation arises from the electron scattering in the electric field of a nucleus in the form of bremsstrahlung. This can be understood as radiation arising from the acceleration of the electron in the electric field of the nucleus. Bremsstrahlung plays a significant role for high-energy electrons where in a single emission a significant fraction of the energy can be lost, and for absorption materials with large atomic number. The total linear stopping power for electrons is the sum of collisional and radiative losses. The specific energy loss for electrons of different energies, for the case of lead, is shown in Fig.(1.3). At low energies electrons lose energy primarily due to ionization. The value of dE/dx decreases and approaches a broad minimum at the energies of a few MeV. Such behavior is typical for a wide range of particles when they approach the speed of light. Fast electrons, as well as other relativistic particles, are often referred to as "Minimum Ionizing Particles" (MIPs). The bremsstrahlung energy loss becomes important only after a few MeV, it rises nearly linearly, and almost saturates at about 200 MeV.

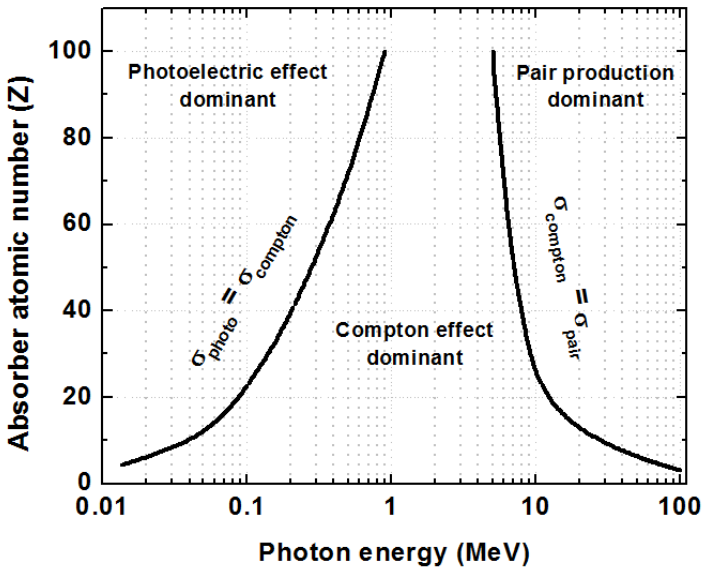


Figure 1.4: The photon energy loss for Photoelectric effect, Compton effect and pair production

1.3 Photons

Photons are quanta of electromagnetic radiation, so they are characterized by having a vanishing mass and charge and travel at the speed of light. Three main kind of interaction mechanisms play an important role in the radiation measurements. The photoelectric effect dominates in the energy range of up to 100 keV. The Compton effect contributes significantly at higher energies, and the electron-positron pair production becomes possible at energies above 1 MeV. The relative importance of these three processes, as a function of the photon energy, is shown in Fig.(1.4). The left and right lines represent the energy at which the Compton effect is equally probable with photoelectric effect and pair production respectively, as a function of the absorber atomic number.

The Photoelectric Effect

The photoelectric effect occurs when a photon is absorbed by an atomic electron with a subsequent ejection of an electron from one of the bound

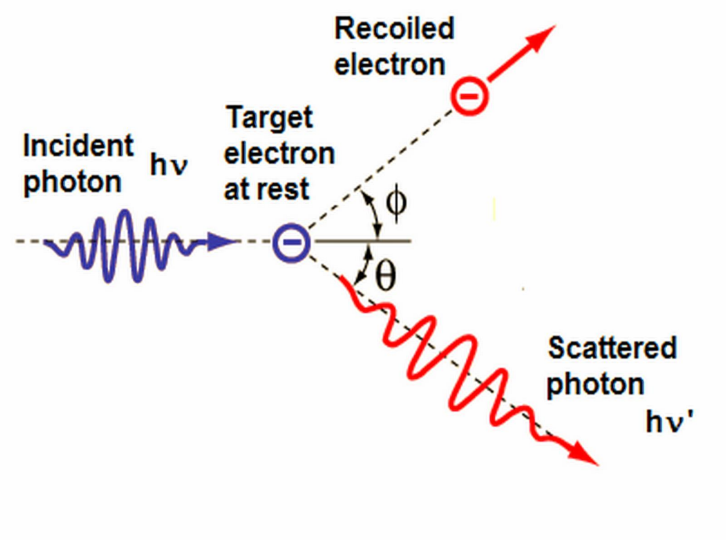


Figure 1.5: The Compton effect kinematics

state shells of the atom. The energy of the outgoing electron is given by

$$E = h\nu - E_{bin} \tag{1.2}$$

where E is the energy of the ejected electron, $h\nu$ is the energy of the incident photon and E_{bin} is the binding energy of the electron in the atom. The photoelectric effect occurs when the energy of the incident photon is higher than the binding energy of the electron it interacts with. The primary mechanism for the absorption of visible light photons is the sudden change of orbitals of single electrons, a process known as "excitation" of the atoms. In the case of X-ray photons, their energies are too high to be absorbed in electron transitions between states for most atoms, so they can interact with an electron only by knocking it completely out of the atom, causing ionization. If the energy of X-ray / γ-ray photon is much larger than the electron binding energies, another mechanism of energy loss takes place: Compton scattering.

Compton scattering

Compton scattering occurs as a result of a high-energy photon colliding with a target, which releases loosely bound electrons from the outer shell of the atom. The kinematics of the energy transfer is shown in Fig.(1.5) and is described by the following equation

$$h\nu' = \frac{h\nu}{1 + (h\nu/m_0c^2)(1 - \cos\theta)} \tag{1.3}$$

where m_0c^2 is the rest-mass energy of the electron (0.511 MeV), $h\nu$ and $h\nu'$ are the energies of the incident and of the recoil photon respectively. The incoming photon is detected to an angle θ with respect to its original direction. A fraction of the photon energy is transferred to the electron. This energy can vary from zero to a large fraction of the $\gamma-$ray energy, as all scattering angles are possible with different probabilities.

Electron-positron pair production

$\gamma-$ray photons with energy greater than 1.02 MeV may interact with a nucleus and convert into an electron-positron pair, which gives rise to a chain of processes ending in a complete energy transfer to the medium. Above 1.02 MeV there is enough energy to provide the rest masses of the electron and positron (0.511 MeV each). The excess energy goes into kinetic energy and is shared between these two particles, which produce ionization as they travel in the material. The positron eventually interacts with an electron and the annihilation of the two particles occurs. This results in the release of two photons each of 0.511 MeV known as annihilation radiation. These two photons then lose energy by Compton scattering or the photoelectric effect.

Photon attenuation

Photon interactions in the medium can be described globally with a parameter called "penetration depth" (δ) or the average attenuation coefficient (μ=1/δ). This quantity is related to the interaction probability and depends on both the photon energy and the material being traversed. The number of transmitted photons I is given by the following equation

$$I = I_0\, e^{-\mu t} \tag{1.4}$$

where I_0 is number of incident photons on the absorption material, μ is the sum of the probabilities of the Photoelectric effect, the Compton scattering and the pair production interactions and t is the absorber thickness.

1.4 Neutrons

Like the photon, the neutron lacks an electric charge, so that it is not subject to Coulomb interactions with the electrons and nuclei in matter. Instead, its principal means of interaction is through the strong force with nuclei. These reactions are, of course, much rarer in comparison because of the short range of this force. When the neutron does interact, however, it may undergo a

variety of nuclear processes depending on its energy, so neutrons are typically classified as

- thermal (or slow) for E $<$ 1 keV ;
- epithermal for 1 keV $<$ E $<$ 500 keV ;
- fast for 0.5 MeV $<$ E $<$ 100 MeV ;
- high energy for E$>$ 100 MeV .

The main interaction processes are:

- Elastic scattering from nuclei, A(n; n)A. If enough energy is transferred the recoiling nucleus ionizes the material surrounding the point of interaction. This mechanism is only efficient for neutrons interacting with light nuclei. In fact, only hydrogen and helium nuclei are light enough for practical detectors. This is the principal mechanism of energy loss for neutrons with an energy of the order of the MeV.

- Inelastic scattering, A(n; n') A^*. In such a reaction, the nucleus is left in an excited state which may later undergo a $\gamma-$decay. The neutron must have sufficient energy in order to excite the nucleus. This energy threshold is usually above 1 MeV.

- Other kinds of nuclear reaction in which the neutron is captured by the nucleus and a charged particle is emitted, (n; p), (n; α), (n; d), (n; t). These reactions are more relevant to slow neutrons in the energy range from the eV to the keV. There are two kinds of such reactions. The first is based on an immediate nuclear charged particle emission, similar to the reaction ${}^{6}_{3}Li(n;\alpha){}^{3}_{1}H$. The second kind of reactions is called activation and results in delayed radiation emission (β^+, β^- or $\gamma-$rays), which can be detected after the irradiated material is removed from the neutron field. The relative probability of different types of neutron interaction varies dramatically with neutron energy.

Chapter 2

Silicon Detectors

Silicon is an almost ideal material for building radiation detectors, as it allows to realize in the same substrate a region detecting the passage of a ionizing particle by collecting the produced charge, a component capable of translating it into an analog signal and, if needed, the signal digitization. These elements can be combined in a chip with a noticeable mechanical rigidity, which allows construction of self-supporting structures. In this chapter we will describe the main features of silicon detectors.

2.1 Silicon Properties

Silicon (see [6]-[10]) is a semiconductor with an energy gap of 1.1 eV which is low enough to produce large numbers of charge carriers when traversed by charged particles. Its intrinsic properties are shown in Table 2.1, extracted from [6]. On average about 80 electron-hole pairs per one micron of track length are produced for particle with an energy near the ionization minimum (these particles are also called Minimum Ionizing Particles or MIPs). It is thus possible to build thin detectors producing signals large enough to be measured and processed. On the other hand, this gap is high enough to avoid large dark currents at room temperature. The high mobility of electrons and holes in silicon at room temperature results in a very fast charge collection of the order of the nanosecond so that silicon detectors can be used in high-rate radiation environments. Silicon also exhibits an excellent mechanical rigidity, which allows construction of self-supporting structures. Since silicon is the basic material used in the integrated circuit industry, there is a worldwide experience in growing large silicon crystals of excellent purity, n-type and p-type doping, growing highly insulating layers like SiO_2 and building readout and on-chip signal processing microcircuits which can be integrated in the

Crystal structure	Diamond
Group of symmetry	Fd3m
Atomic Number	14
Number of atoms in 1 cm^3	$5 \cdot 10^{22}$
Density	2.33 g cm^{-3}
Energy gap	1.12 eV (300K)
Intrinsic carrier concentration	$1.45 \cdot 10^{10}$ cm $^{-3}$ (300K)
Intrinsic resistivity	$2.3 \cdot 10^5\,\Omega$ cm (300K)
Diffusion coefficient electrons	34 $cm^2 s^{-1}$ (300K)
Diffusion coefficient holes	13 $cm^2 s^{-1}$ (300 K)

Table 2.1: The main Silicon physical properties

same substrate as the detector. For all these characteristics silicon detectors are ideal sensor devices in several applications.

2.2 Conduction in pure and doped semiconductors

Silicon has an electrical resistivity in the range between that of a conductor (below $10^{-2}\,\Omega$ cm) and an insulator (above $10^5\,\Omega$ cm), so it is called a semiconductor. Due to its crystalline periodic structure, the electrons are not bound to the individual ions but rather they form a valence and a conduction band, separated by an energy gap E_g (see fig.2.1), and they are shared by the whole material. At zero absolute temperature all electrons are in the valence band; no current can flow if an electric field is applied. At higher temperatures, on the other hand, the electron behavior is described by the Fermi-Dirac statistics, so that the probability that a state of energy E is filled with an electron is given by the probability density function

$$f(E) = \frac{1}{1 + \exp\left(\frac{E - E_F}{k_B T}\right)} \tag{2.1}$$

where E_F is the Fermi energy, k_B is the Boltzmann constant and T is the absolute temperature. The density $N(E)$ of electron states with energy E in

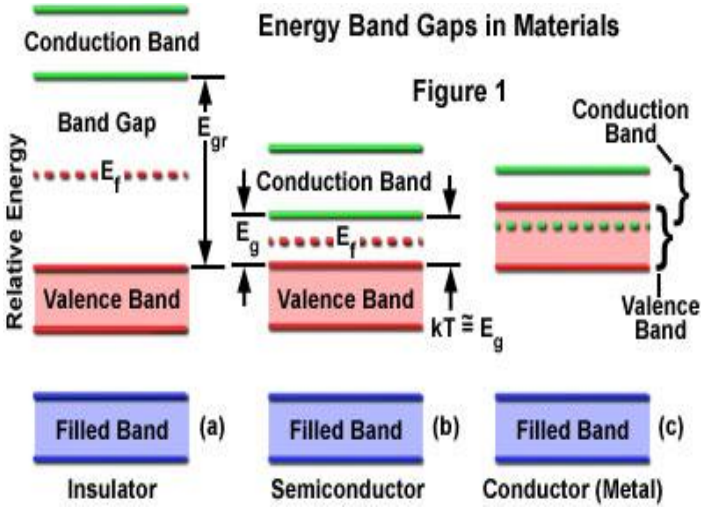


Figure 2.1: The band structure for (a) insulators, (b) semiconductors, (c) metals

a conduction band is proportional to $(E - E_C)^{1/2}$ with E_C is the energy of the conduction band. Thus the density of free electrons and holes (n and p, respectively) can be calculated by integrating the density of states per unity of energy $N(E)$ times the occupation probability $f(E)$, yielding:

$$\begin{aligned} n &= N_C \exp\left(-\frac{E_C - E_F}{k_B T}\right) \\ p &= N_V \exp\left(-\frac{E_F - E_V}{k_B T}\right) \end{aligned} \tag{2.2}$$

where N_C and N_V are the effective densities of states in the conduction and the valence band, respectively given by:

$$\begin{aligned} N_C &= 2\left(\frac{2\pi m_e^* k_B T}{h^2}\right)^{3/2} \\ N_V &= 2\left(\frac{2\pi m_p^* k_B T}{h^2}\right)^{3/2} \end{aligned} \tag{2.3}$$

where m_e^* and m_p^* are the effective mass of electrons and holes, and h is the Planck constant. In intrinsic semiconductors the thermal agitation excites electrons that leave the valence band and occupy the conduction one, leaving holes in the valence band. In this case $p = n = n_i$, where n_i is the intrinsic

carrier density. By assuming $n = p$ in (2.2) one is led to:

$$E_F = \frac{E_C + E_V}{2} + \frac{k_B T}{2} \log\left(\frac{N_V}{N_C}\right) \tag{2.4}$$

Thus the Fermi level of an intrinsic semiconductor lies very close to the middle of the energy gap. The intrinsic carrier density is given by the equation:

$$n_i = \sqrt{p\,n} = \sqrt{N_C\,N_V}\exp\left(-\frac{E_g}{2k_B T}\right) \simeq T^{3/2}\exp\left(-\frac{E_g}{2k_B T}\right) \tag{2.5}$$

where $E_g = E_C - E_V$ is the energy band gap (in ultrapure silicon the intrinsic carrier concentration is $1.5 \cdot 10^{10}\mathrm{cm}^{-3}$ with approximately 10^{22} atoms/cm^3 so that about 1 in 10^{12} silicon atoms is ionized). Electrical conductivity of semiconductor materials can be altered by several orders of magnitude by adding small quantities of other substances, called impurities. The process of replacing atoms in the semiconductor lattice with atoms of other elements is called doping. It leads to creation of additional energy levels within the energy gap. The doped semiconductor is usually called extrinsic. Pentavalent impurities such as phosphorus are called donors since they donate additional electrons to the conduction band. The four valence electrons of a donor are shared in the covalent bonding with neighboring silicon atoms, while its fifth electron is loosely bound. At room temperatures those electrons become free and hence available for conduction. Silicon doped with donors is called n-type and its Fermi energy E_F is close to the energy of the conduction band E_C. Alternatively, silicon may be doped with trivalent impurities such as boron. They are called acceptors since they accept electrons from the valence band leaving there a hole. In the case of acceptors, three strong covalent bonds are formed with adjacent silicon atoms but the fourth bond is incomplete. This vacancy can easily be filled with an electron from the valence band. This is called a p-type silicon and its Fermi energy E_F is close to the energy of the valance band E_V.

Carrier transport in semiconductors

This section will deal with two cases of non-equilibrium states. Diffusion occurs when charge carriers are trying to reach equilibrium conditions due to their non-uniform spatial distribution. Drift, on the other hand is an ordered movement of carriers due to an external electric field.

Diffusion

Diffusion takes place when the charge carriers are inhomogeneously distributed in the semiconductor. It arises due to the fact that carriers in

the higher concentration region are more likely to move into a region with lower concentration in order to reach equilibrium. This results in a diffusion current with density

$$J_n^{diff} = qD_n\nabla n \tag{2.6}$$
$$J_p^{diff} = -qD_p\nabla p \tag{2.7}$$

where D_p and D_n are the diffusion constants, proportional to the thermal velocity and the mean free path of the carrier in the material, (their typical values are shown in Table 2.1), and q = $1.602 \cdot 10^{-19}$ C is the carrier's unit charge. The total diffusion current density from holes and electrons is given by

$$J^{diff} = qD_n\nabla n - qD_p\nabla p \tag{2.8}$$

Drift

The charge carriers in the conduction band are in random thermal motion with zero average displacement, if a semiconductor is in an equilibrium state and no electric field is applied. An external electric field, E, will apply a force to each electron and hole. They are accelerated and scatter in the crystalline lattice losing momentum, eventually gaining an average drift velocity

$$\vec{v_n} = -\mu_n \vec{E} \tag{2.9}$$
$$\vec{v_p} = \mu_p \vec{E} \tag{2.10}$$

where μ_n and μ_p denote the carrier mobility for electrons and holes. The drift velocity of the carriers is small compared to the thermal velocity if the electric field is low enough. So the average velocity of electrons and holes increases linearly with the external field:

$$J_{drift} = (n\mu_n + p\mu_p)q\,E = \sigma\,E \tag{2.11}$$

σ is known as the material conductivity, which is the inverse of the resistivity. However, for higher electric fields a strong deviation from linearity can be observed. As the electric field strength increases, scattering occurs more frequently and the drift velocity eventually saturates, becoming independent of the electric field. Silicon sensors are usually kept in the linear regime. Combining current contributions from diffusion and drift gives total current density for electrons and holes

$$J_n = qD_n\nabla n + n\mu_n qE \tag{2.12}$$
$$J_p = -qD_p\nabla p + p\mu_p qE \tag{2.13}$$

Diffusion and mobility for both electrons and holes are related to each other by the Einstein equation

$$D_n = \frac{k_B T}{q}\mu_n \tag{2.14}$$

$$D_p = \frac{k_B T}{q}\mu_p \tag{2.15}$$

Since the mobility of the charge carriers is defined as

$$\mu_n = \frac{q\tau_c}{m_n} \tag{2.16}$$

$$\mu_p = \frac{q\tau_c}{m_p} \tag{2.17}$$

the diffusion constant of a material then depends upon the effective mass of the charge carriers, m_n, m_p, mean free time between collisions, τ_c , and the material temperature, T. The semiconductor properties described above are the critical parameters that have led to the development of solid state radiation detectors as well as the vast number of electronic devices.

2.3 Semiconductor structures

The p-n junction

The junction between p-type and n-type semiconductors exhibits interesting electrical properties that are of great importance for modern electronics as well as for ionization detectors. On contact, electrons diffuse from the n-type material into the p-type while holes diffuse in the opposite direction. Electrons leave exposed donor ions of N_D^+ concentration over a thickness x_n in the n-type semiconductor and holes leave exposed acceptor ions of N_A^- concentration over a thickness x_p in the p-type semiconductor. Thus at the p-n junction a fixed space charge of ionized donors and acceptors is created (the so-called depletion region), as illustrated in Fig.(2.2). This process creates an electric field that eventually balances the tendency of the current to flow due to the diffusion process. The Fermi levels in the p and n type doped materials become equal once the static condition is reached. The electrostatic potential V and electric field strength are related by the Poisson equation in one dimension:

$$-\frac{d^2V}{dx^2} = \frac{dE}{dx} = \frac{\rho(x)}{\varepsilon_{Si}\,\varepsilon_0} \tag{2.18}$$

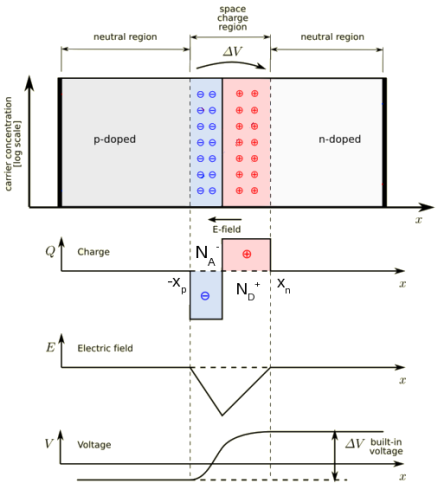


Figure 2.2: The P-N Junction

where ε_{Si} is the relative dielectric constant of Silicon, ε_0 is the dielectric constants of vacuum, and $\rho(x)$ is the charge density function given by:

$$\rho(x) = \begin{cases} qN_D^+ & \text{for } 0 < x < x_n \\ \\ -qN_A^- & \text{for } -x_p < x < 0 \end{cases}$$

Thus

$$E(x) = \begin{cases} \frac{-qN_D^+}{\varepsilon_{Si}\varepsilon_0}(x_n - x) & \text{for } 0 < x < x_n \\ \\ \frac{-qN_A^-}{\varepsilon_{Si}\varepsilon_0}(x + x_p) & \text{for } -x_p < x < 0 \end{cases}$$

and

$$V(x) = \begin{cases} V_n + \frac{qN_D^+}{\varepsilon_{Si}\varepsilon_0}(x_n - x)^2 & \text{for } 0 < x < x_n \\ \\ V_p - \frac{qN_A^-}{\varepsilon_{Si}\varepsilon_0}(x + x_p)^2 & \text{for } -x_p < x < 0 \end{cases}$$

where $V_n = V(x_n)$ and $V_p = V(-x_p)$ are the integration constants. Continuity of the field at x = 0 implies:

$$N_A^- \, x_p = N_D^+ \, x_n \tag{2.19}$$

which shows that depth of the depletion region is inversely proportional to the doping concentration on each side of a p-n junction. The potential step at the depletion region, the so-called build-in potential barrier $\Delta V = V_n - V_p$, can be calculated by imposing the continuity of the electrostatic potential at $x = 0$:

$$\Delta V = \frac{q}{2\varepsilon_{Si}\varepsilon_0}\left(N_A^- x_p^2 + N_D^+ x_n^2\right) \tag{2.20}$$

Exploiting equations (2.19) and (2.20) one can find a relation between the depth of the depletion region w and the build-in potential ΔV :

$$w(\Delta V) = x_n + x_p = \sqrt{\frac{2\varepsilon_{Si}\varepsilon_0}{q}\Delta V\left(\frac{1}{N_A^-} + \frac{1}{N_D^+}\right)} \tag{2.21}$$

In tracking detectors the doping concentration is usually much larger on one side of the junction than on the other. Assuming a higher doping concentration of the p-type material ($N_A >> N_D^+$), equation (2.21) reads:

$$w(\Delta V) = \sqrt{\frac{2\varepsilon_{Si}\varepsilon_0}{qN_D^+}\Delta V} \tag{2.22}$$

The depth of the depletion region can be increased by applying the bias voltage V_b with the same polarity as that of the built-in potential ΔV . The bias voltage needed to deplete the full detector thickness D (the so-called full depletion voltage) is given by:

$$V_{dep} = \frac{qN_D^+ D^2}{2\varepsilon_{Si}\varepsilon_0} - \Delta V \tag{2.23}$$

Junctions biased with a voltage of the same polarity as the built-in potential ΔV are called reverse biased, whereas junctions biased with a voltage with an opposite polarity compared to the built-in potential ΔV are called forward biased.

Leakage Current

The leakage or dark current is one of the principal sources of noise. It flows through the p-n junction even in the absence of ionization source, if a reverse bias is applied. The leakage current is due to the diffusion of free carriers from the undepleted volume into the sensitive space charge region and the thermal generation at the generation-recombination centers at the surface of

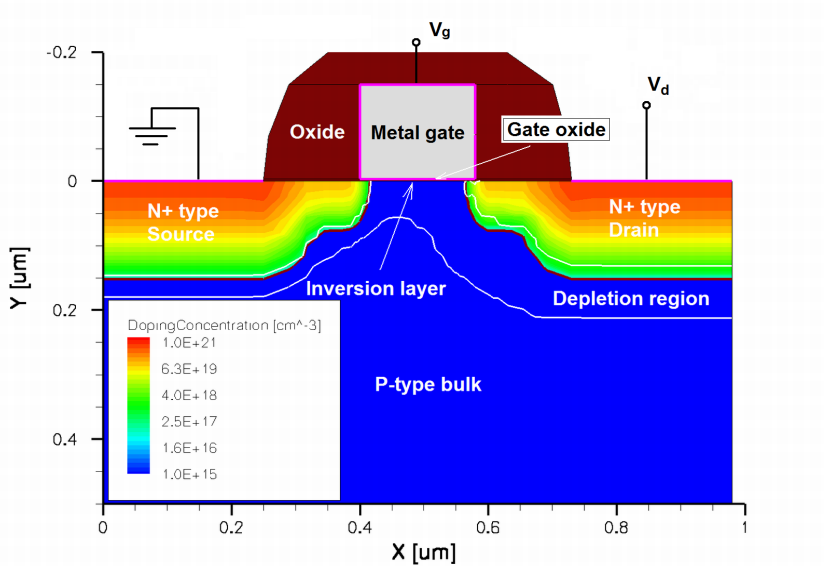


Figure 2.3: The typical nMos transistor

the device and in the depleted volume. The latter usually dominates the p-n junction leakage current and is proportional to the depleted volume

$$J_{vol} \simeq q\frac{n_i}{\tau_g}w \tag{2.24}$$

where J_{vol} is the volume generation current per unit area, τ_g is the carrier generation life time, n_i is the intrinsic carrier concentration and w is the depletion region depth of eq. (2.22). The τ_g and n_i temperature dependence imply

$$J_{vol} \simeq T^2 \exp\left(-\frac{E_g(T)}{2k_BT}\right) \tag{2.25}$$

One can thus note that the leakage current is strongly dependent on the temperature and can be significantly reduced by cooling the detector.

2.4 The MOS transistor

Nowadays the basic element of any electronic circuit is the transistor, and silicon radiation detectors are not an exception to this. The unipolar or Metal-Oxide-Semiconductor Field-Effect Transistor (MOSFET) will be described as it constitutes a basis for many radiation detector devices. Under

various operational conditions the transistor can be used as a current source, a simple switch, an amplifier, a variable resistor or a capacitor. The combination of several transistors provides the basis for the design of analog and digital circuits. Fig.(2.3) shows a model of a typical nMOS transistor (pMOS transistor may be obtained by exchanging p for n and reversing the polarity of the voltages). It has four terminals: the source, the drain, the gate and the substrate bias contact (usually kept at ground). These structures can be formed on a moderately doped p-type silicon substrate. The source and drain are heavily doped n-type regions (n-wells) implanted on the substrate with a separation distance (the transistor channel). Above the charge carriers depleted region, called "channel", there is a thin silicon dioxide layer (the gate oxide) that isolates the gate from the substrate. In a silicon integrated circuit, each MOSFET is surrounded by a thick field oxide to isolate it from adjacent transistors.

The MOS structure properties

The properties of the MOS structure suggest the operational application of the MOS transistor. A simple MOS structure is made of an oxide layer grown on top of a p-type semiconductor with a metal contact placed on the oxide. This structure is equivalent to a planar capacitor with one of the electrodes replaced by a semiconductor. When a positive voltage is applied across the MOS structure, the established electric field sweeps away positively charged holes from the oxide-semiconductor interface, creating a depletion layer by leaving electrons only. If the voltage is above some level know as the threshold voltage, a high concentration of electrons forms an inversion layer which is located on a thin layer next to the interface between the semiconductor and the insulator. In the transistor this layer is formed in the channel between the source and drain.

The nMOS transistor principle of operation

In equilibrium, when a voltage above threshold (V_{th}) is applied to the gate terminal (V_g), the source to drain channel acts as two back to back connected p-n junctions. If a small drain voltage (V_d) is applied, the silicon beneath the gate acts simply as a resistor, and the linear relationship between the current and the voltage is obtained as shown in Fig.(2.4). As the V_d is increased, the p-n junction at the drain becomes more reverse biased, and the depletion region spreads out into the bulk and the channel under the gate, which eventually becomes closed or "pinched off". Nevertheless, the characteristic curve does not become flat but still has a small positive slope. This small

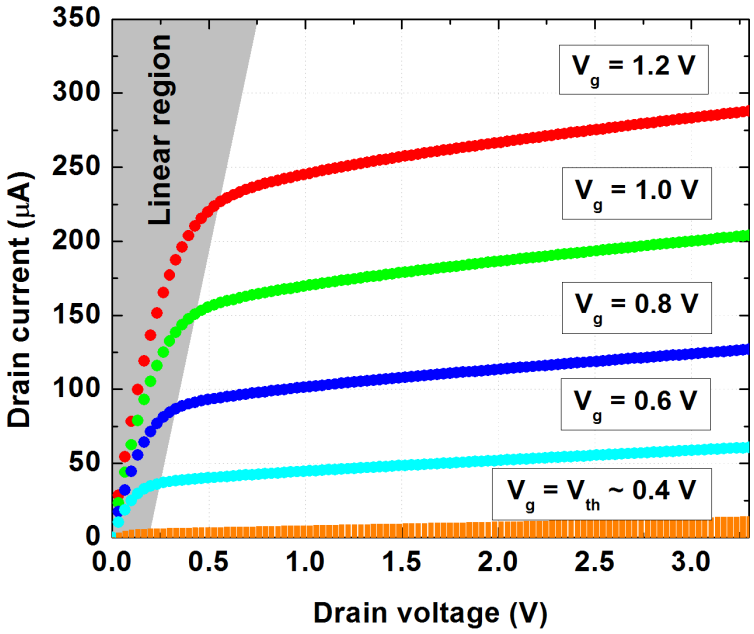


Figure 2.4: The transistor curve: the drain current as a function of the drain voltage

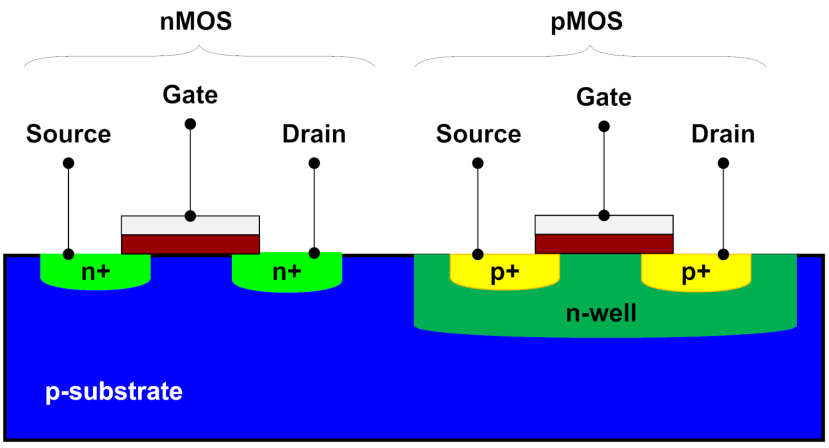


Figure 2.5: A simplified cross-section of a CMOS process

increase in current can be attributed to the shortening of the effective channel length due to the growth in the "pinched off" region. Fig.(2.4) shows a family of drain-to-source current curves as a function of drain-to-source voltage, for different applied gate voltages. For V_g below V_{th} the region under the gate oxide is poorly populated with free electrons and only a small sub-threshold leakage current flows between the source and the drain, and it is mainly due to the thermal excitation of carriers. If V_g is fixed at some small voltage and again V_d is gradually increased, then, as the depletion layer will initially be larger, the slope of the characteristic curve will be initially smaller and the "pinch off" will occur at lower current values.

2.5 The CMOS technology

The crucial reason for the success of the MOS transistors was the development of the Complementary MOS logic (CMOS). Modern CMOS technology is used in the production of microprocessors, memories, digital and analog circuits as well as imaging sensors. The technology allows the fabrication of nMOS and pMOS transistors on the same substrate, giving fundamental advantages for low power consumption and high noise immunity for the entire system. All devices have to be built on a single semiconductor substrate. Only capacitors and resistors are available as passive components,

while usually two transistor types can be integrated. The main advantage of CMOS over nMOS and bipolar technology is the much smaller power dissipation. Unlike nMOS or bipolar circuits, a CMOS digital circuit has almost no static power dissipation. In fact one of the two transistors always behaves as an open switch, so that power is dissipated only when the circuit actually switches states. This allows the integration of many more CMOS gates on an integrated circuit than in nMOS or bipolar technology, thus resulting in an improved performance and extended functionality. Besides the relative simplicity in fabrication and low-power consumption in digital applications of CMOS technology, the drawbacks are given by the limitation in speed and impossibility to drive large currents.

CMOS electronics uses only p- and n-channel transistors as active devices. The cross section of the typical CMOS active device is shown in Fig.(2.5). These complementary transistors are insulated from each other by placing one of them into a well with a doping which is opposite to the bulk material. The combination of such devices is extremely useful, as the same signal which turns on a transistor of one type is used to turn off a transistor of the other type, thus dramatically reducing the power consumption. Another main advantage is that such a combination allows the creation of paths to the output from either the voltage source or ground. These two concepts are the basis for developing the CMOS binary logic.

Chapter 3

Pixel Detectors for High Energy

The innermost part of the detectors for high energy physics experiments, namely those placed at the four interaction points of the Large Hadron Collider (LHC) at CERN, are called vertex detectors and are based on silicon sensors. They are currently based on hybrid pixel matrices and one of their possible upgrade in the coming years requires the implementation of Monolithic Active Pixels. In this chapter we will first introduce these technologies and then describe the evolution road that led to APSEL4D, the chip we used for our experiments. For completeness we conclude by describing the state of the art detectors used in the two main LHC experiments.

3.1 Pixel Sensors

Nowadays a large variety of silicon based electronic devices incorporate pixel sensors. The most common semiconductor pixel sensors are those employed in modern digital cameras and mobile phones. These kinds of silicon sensors detect visible-light photons and are designed to have a wide and optimized dynamic range in order to enhance, for example, the brightness and contrast of the subject. Photons are collected in the sensor array and converted in localized electron charge whose amount is proportional to the photons observed, making some pixels "brighter" than others. Since in each pixel there is some information, the whole matrix has to be read out in order to provide the final image. The requirements on semiconductor pixel aimed at detecting charged particle in high energy physics experiments can be very different from those made for imaging. In particle physics experiments, individual charged particles have to be identified with high demands on spatial resolution and timing. In imaging applications, on the opposite, the final image is obtained by an accumulation (integrating or counting) of quanta of the

impinging radiation. Silicon pixel detectors for high energy charged particle detection can assume typical signal charges collected at an electrode in the order of 5000-10000 electrons, even taking into account charge sharing between cells, and detector deterioration after irradiation to doses as high as 60 Mrad. Silicon is almost a perfect material for particle physics detectors, allowing the shaping of electric fields by tailored impurity doping, whereas the need of high photon absorption efficiency in radiological applications requires the study and use of semiconductor materials with high atomic charge, such as GaAs or CdTe. For such materials the charge collection properties are much less understood and mechanical issues, in particular those related to hybrid pixels, are abundant, most notably regarding the hybridization of detectors, when they are not available in wafer scale sizes.

In this chapter we focus on pixel detectors for high energy experiments (see [11]). Pixel sensors adopted at the Large Hadron Collider experiments must detect charged particles or photons and 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, tracking sensors are optimized in terms of readout speed. Moreover, in some cases, the readout phase is continuous, overlapped to the acquisition phase, and concentrated only on regions where a pixel was hit, as physicists are looking for the spatial position of a trace produced by a single particle. Each single hit pixel provides a spatial point on a particle trajectory and several layers of pixel detectors are stacked in order to reconstruct a particle trajectory. In some cases the information about the quantity of charge collected in a fired pixel might also be read out. It is very useful to enhance the sensor resolution in order to deal with clustered events, where the reconstructed crossing point of the particle can be evaluated with a center of mass algorithm (a spatial weighted average where the charge acts as a weight). This information is also useful for the reconstruction of 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. On the other hand, the extraction of this information does not come for free and it can be rather very time consuming, especially for pixels, as the density of channels is very high (400 channels/mm^2 with a 50-micron pitch). A common feature of pixel detectors is that when a pixel gets fired by a crossing particle, it is unable to detect any other impinging particle until it is read out and reset. This time lapse, during which the pixel is latched, is called dead time. In some specific cases, the dead-time introduced by the reading of the collected charge is not affordable, consequently the readout is built to extract only the hit/no hit information from the pixels. In the following we will mainly consider this kind of pixel sensors.

3.2 Vertex Detectors

In a high energy collider experiment, where high energy particles or ions collide head-on in a single point of space and produce lots of particles, the innermost detector is appointed to performing an accurate reconstruction of the particle tracks coming out of the interaction vertices. Several coaxial layers of sensors are displaced around the beam pipe, where the high energy particle beams collide in the interaction region, so that with the detected crossing points it is possible to trace a particle trajectory (see fig.3.1). Resolution and efficiency are the two main parameters to optimize, but typically one has to balance the two features. 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 used for the construction of a tracker, as 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 it 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 a low material budget for the entire tracking detector. 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, 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. In fact Silicon density and its small ionization energy, allow the production of 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 is formed in the silicon substrate, translating it into a signal (for minimum ionizing particles the most probable charge deposition in a 300 micron thick silicon detector is about 3.5 fC corresponding to 22000 electrons). 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

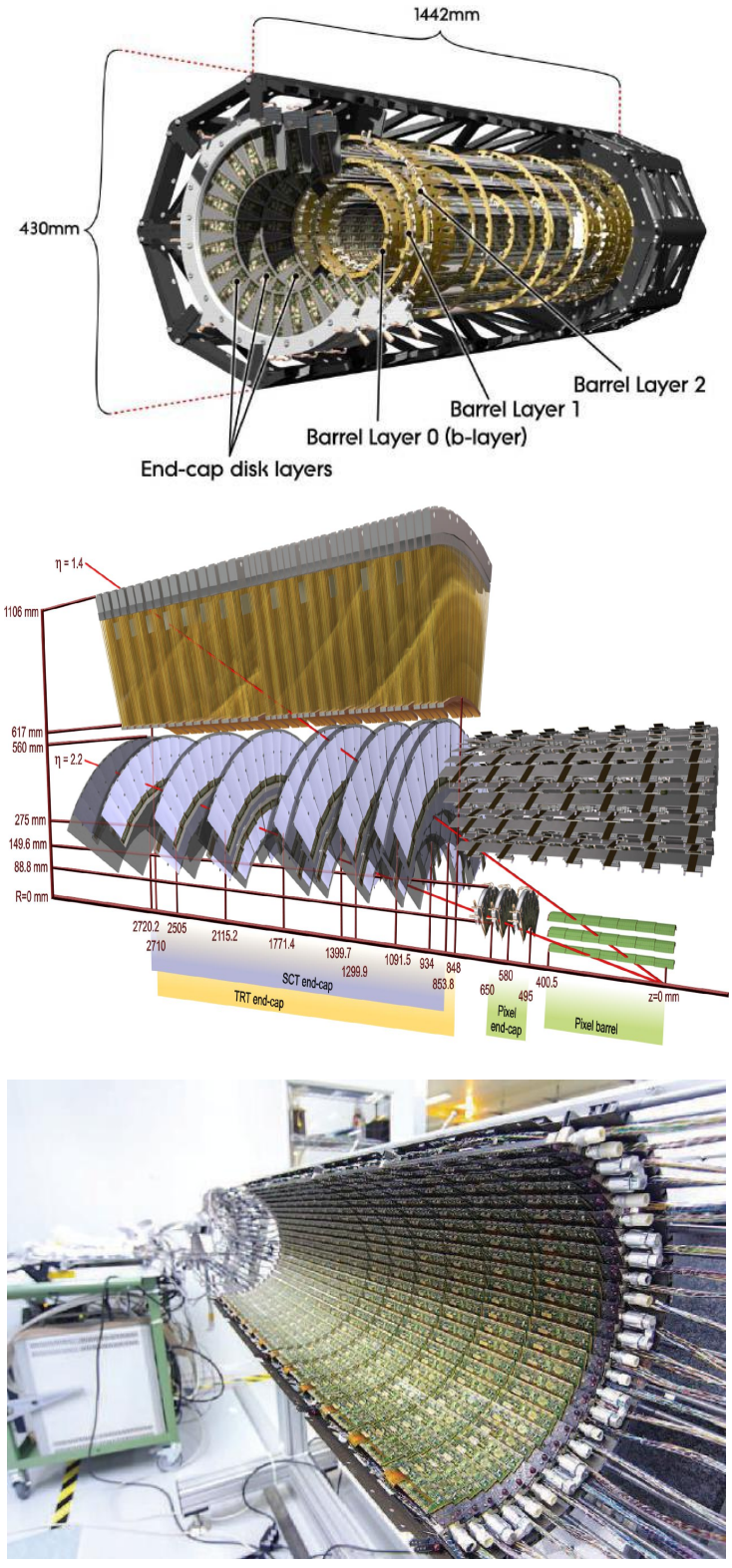


Figure 3.1: The scheme and picture of the Atlas Inner Detector

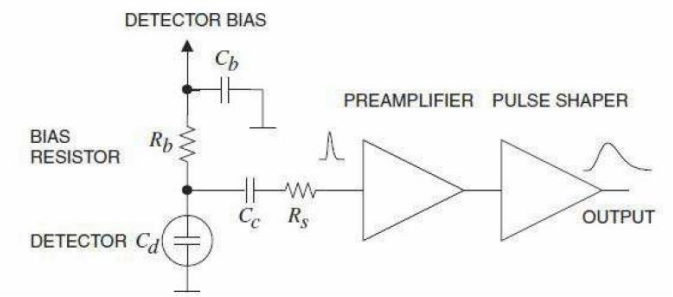


Figure 3.2: The binary readout scheme for a single pixel

signal with a voltage threshold. Keeping low the front-end noise is a critical issue, both for improving the energy resolution (which depends on the collected charge) and allowing a low detection threshold. A scheme of a typical front-end circuit is presented in Fig.(3.2).

- the digital component: it is the memory element that, if present, 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 pixel channel is "blind" to new incoming particles.

- the Readout: it is the electronic component appointed to the extraction the hit information from each pixel. It can be implemented in very different ways depending on the optimization targets. A possible implementation optimized for fast binary reading is shown in Fig.(3.3).

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, the higher the capacitive load, which means slower signals and higher power consumption. Pixel devices instead, are matrices of square-shaped sensors that improve granularity and provide faster signals. This way the same area is covered by a greater number of channels, allowing

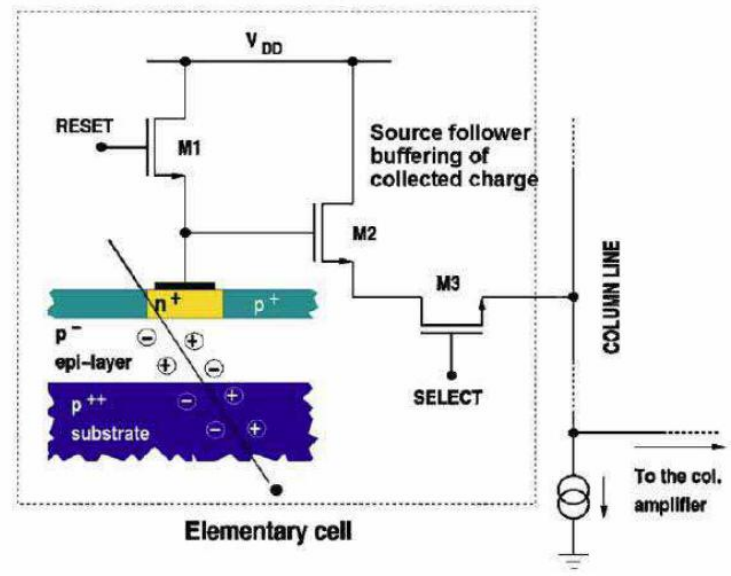


Figure 3.3: Schematic view of a CMOS MAPS device with typical 3T readout structure High-Efficiency Digital Readout Systems for Fast Pixel-Based Vertex Detectors

a more precise spatial information. In a particle tracker, the uncertainty on the reconstructed position of the collision (primary) vertex is dominated by the spatial resolution of the innermost layers, therefore they are typically instrumented with pixel sensors due to their higher resolution. Finally, since the area to be instrumented increases with radius and since pixels sensors present a higher cost-per-area, the outer layers of the tracker are typically instrumented with silicon strips.

As an example we show a very simple binary readout structure for a CMOS APS - Active Pixel Sensor - in Fig.(3.4), 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. At 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.

3.3 Hybrid Pixel Detectors

A pixel detector can be implemented with different fabrication technologies. The most common at the moment foresees the interconnection of a sensor

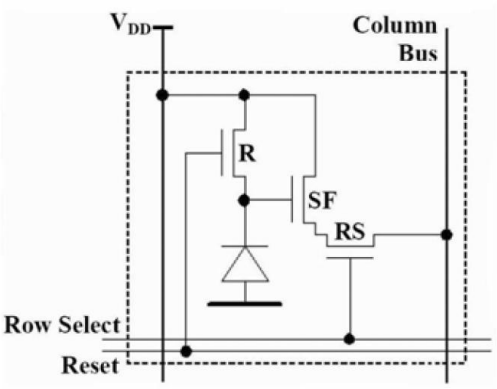


Figure 3.4: Three transistor readout for a matrix of pixels. Reset transistor R clears the pixel of integrated charge, Source Follow transistor SF amplifies/buffers the signal and Row Select transistor RS selects the row for readout.

silicon layer to a standard CMOS 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. A hybrid pixel detector technology comprises a detection medium and the readout electronics manufactured in different fabrication processes. Both parts of the detection system have a matching matrix of electrodes which are brought into contact through a standard flip-chip bonding process (Figs.3.5 and 3.6).

This technology allows independent design and optimization of the readout electronics and of the detection medium which have in general conflicting requirements, for example the silicon resistance. Due to advances in CMOS technology, the readout chip can fully exploit advanced signal processing techniques on-chip. This can include amplification, energy discrimination, digitization and post processing of the signal on the pixel level. Moreover, exotic detection materials, e.g. Ge, GaAs, CdZnTe, can be used, which makes hybrid technology very attractive for a wide range of applications despite its relatively high fabrication costs.

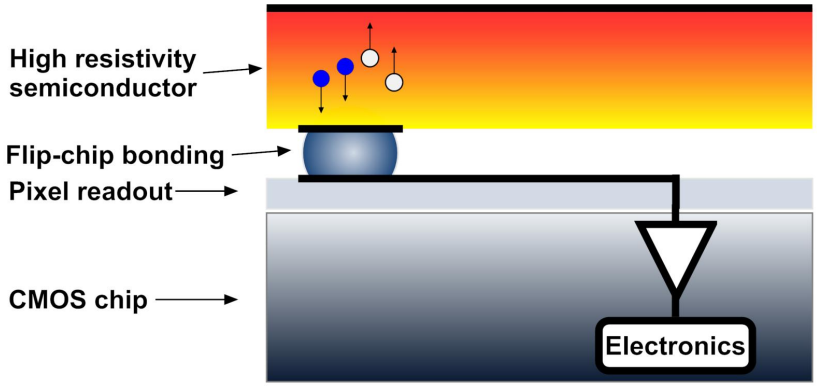


Figure 3.5: The hybrid pixel scheme

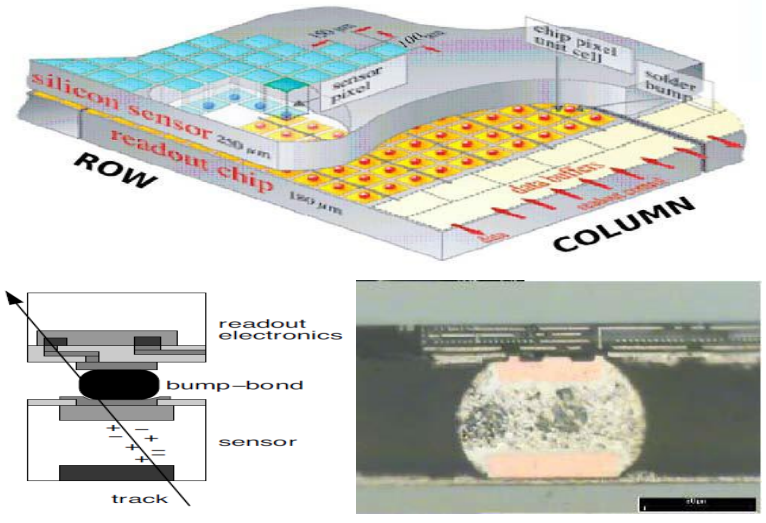


Figure 3.6: The scheme and image of bonded pixels

3.4 Monolithic Active Pixel Sensors

It is possible to get rid of the delicate bump-bonding procedure needed for hybrid pixels by integrating both sensor and readout on the same substrate, thus reducing dramatically the material budget. These kind of devices are known as MAPS (Monolithic Active Pixel Sensors). This technology can be used in High Energy Physics sensors and chips produced this way can fulfill very tight requirements on position resolution, readout speed, material budget and radiation tolerance. The next generation of vertex detectors need in particular a technological upgrade, as they are required to be closer and closer to the interaction point, where particle density can reach several tens of particles per cm^2. This means a higher radiation dose and a higher hit-rate, with the consequent need of fast sensors with high granularity. The main challenge for this kind of detectors is the upgrade of pixel sensors as they are exposed to the highest radiation dose and it is difficult to speed-up their read-out process for their extreme density. That's why new kinds of silicon pixel detectors are investigated at the moment.

Traditional pixel sensors integrate on a silicon chip a bi-dimensional array of cells that can be addressed typically in a pixel-by-pixel way or column-by-column by external read-out logic. The typical sensing technique is based on the collection of the charge that the impinging charged particle forms in the epitaxial layer: the electrons move simply by diffusion and are collected by the cathode of the N-well/P-epitaxial reverse-biased diode. Charge to voltage conversion is provided by the sensor capacitance, and thus collecting electrodes are kept as small as possible to increase the conversion factor. In NMOS technology, three NMOS transistors are typically used to address and reset the single pixel, for this reason this simple architecture is called 3T and it is illustrated in Fig.(3.7). This architecture presents several limitations: first of all the collecting area that, as we said, is limited by the voltage conversion factor. On the other hand, since it collects diffusing electrons, a limitation in its surface implies a limitation in the efficiency. In addition the pixel-by-pixel access makes the hit read-out process drastically slow.

A CMOS Monolithic Active Pixel Sensor (MAPS) on the other hand, is a pixellated imaging sensor, produced using a standard CMOS fabrication process. The wafers differ slightly from those used for electronic circuit fabrication. A high resistivity, high quality epitaxial layer of less than 20 μm is deposited on top of the wafer. This improves the charge collection and noise characteristics of the sensor. Such an approach can be applied not only for visible light imaging but also for charged particle detection. A basic sensor consists of an array of pixels each containing a photo sensitive diode, an amplifier, a reset transistor and a select transistor. The schematic in Fig.(3.8)

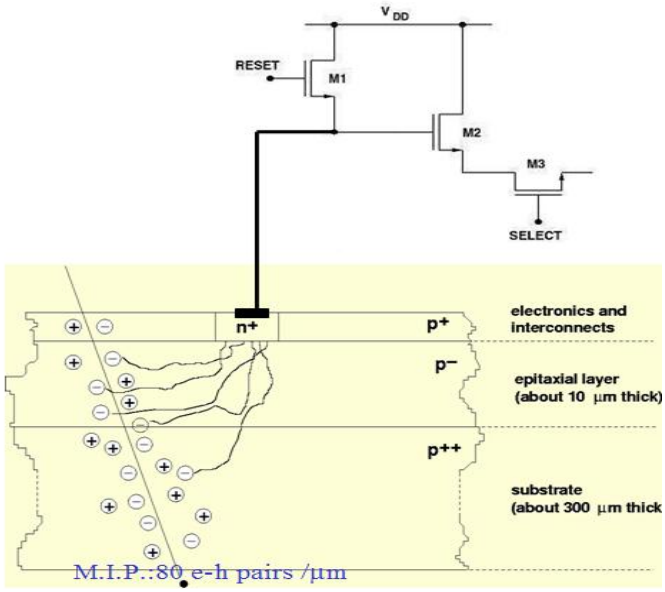


Figure 3.7: A 3T NMOS traditional binary pixel sensor. The diffusing charge is collected by the N-well electrode and it is read as a voltage tension exploiting the sensor intrinsic capacitance.

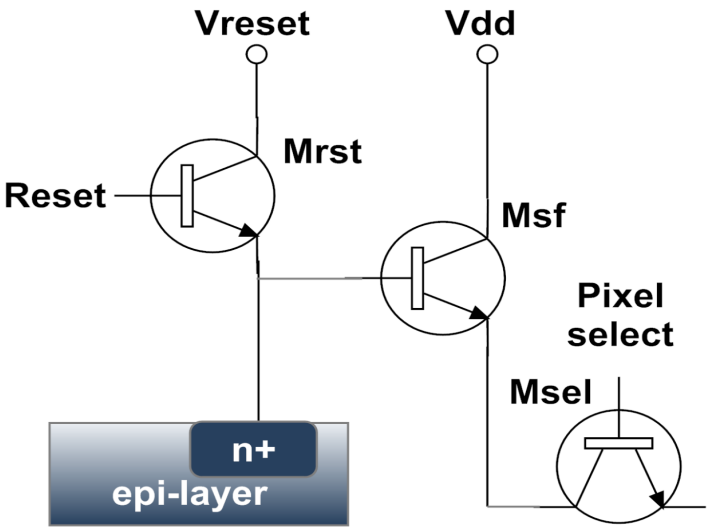


Figure 3.8: A scheme of the standard 3MOS pixel design of the MAPS.

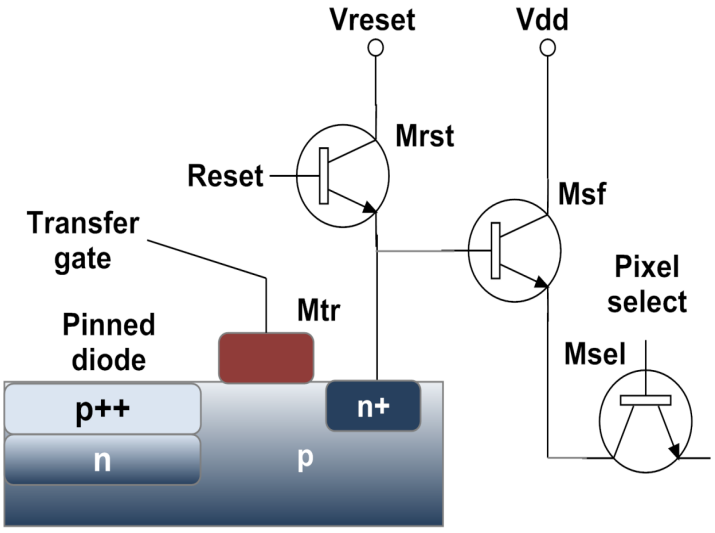


Figure 3.9: The scheme of the standard 4MOS pixel design of the MAPS.

shows the elements of such a pixel. This design is known as 3MOS pixel design. When a photon or a charged particle interacts with the silicon of the epitaxial layer, it generates a charge, which is then collected by the photo diode. The source follower transistor (M_{sf}) acts as buffer to convert charge to voltage within the pixel. The select transistor (M_{sel}) switches the pixel to the read-out electronics. After the signal is read out from the pixel the reset transistor (M_{rst}) clears the charge from the diode by pulling it to a reset voltage. The pixel array is usually readout using a "rolling shutter" method. The pixels of row n are individually selected by the select transistor using a column shift register and the digital information generated in each pixel in the row is sampled to a storage capacitor and then digitized off-chip. The 3MOS pixel has several significant shortcomings. The bulk of the noise in the pixel is introduced during the reset phase by the reset transistor and known as the kTC noise. The 3MOS pixel also has a high thermal dark current mainly due to fabrication damage in the n+ contact of the photo diode. Generally CMOS MAPS pixels show non-linear outputs, which are due to the fact that the diode capacitance changes as it charges up. The other most commonly used pixel architecture is referred to as a 4MOS pixel (Fig.3.9). It features an extra switch (M_{tr}) between the photo diode and the reset transistor. This design allows the reset level to be sampled just before integration, which means that the noise of the reset transistor can be subtracted. Such a technique is known as the correlated double sampling (CDS) . Despite tremendous noise reduction, this architecture suffers from

several disadvantages. The sensors featuring 4MOS design usually use pin-diode instead of pn-diode, complicating standard CMOS process. The charge capacity of the pixel is also dramatically reduced limiting the architecture to large pixel sizes.

3.5 The path to APSEL 4D

We already noted that the previously presented architectures present several limitations: first of all the collecting area that is limited by the voltage conversion factor. which implies a limitation in the efficiency. In addition the pixel-by-pixel access makes the hit read-out process drastically slow. To overcome these problems the SLIM5 collaboration has developed a Monolithic Active Pixel Sensor (MAPS) which features the following characteristics: the sensor cell is active, which means that the front end of the pixel is driven by active electronic components like a preamplifier, a shaper and a discriminator; in addition it is monolithic as it integrates a standard CMOS read-out logic. Modern foundry technologies allow to create a large deep N-well to separate the P epitaxial layer from another P region. Inside this region ordinary NMOS operational amplifiers can be implemented to realize the active circuitry of the pixel, and the large deep N-well can act as a large electron collecting anode. A cross-section view of a deep N-well (DNW) architecture is shown in Fig.(3.10). Once the signal has been amplified and discriminated, the CMOS latch, realized outside the DNW, stores the hit/non-hit binary information of that cell. The latch can thus be easily interfaced to the sparsification logic, realized with CMOS standard-cells. In addition, the voltage gain is determined by the feedback capacitance of the charge preamplifier and the size of the collecting electrode can be increased up to about 900 μm^2 in a pixel cell of $50\mu m$ pitch. It is thus possible to include within the pixel some small competitive n-well regions, crucial to develop the CMOS logic for the readout, yet keeping the sensor fill factor at the level of 90%.

During the activity of the SLIM5 collaboration several prototype chips of a series called APSEL were realized with the ST Microelectronics 130 nm triple well technology. The starting projects implemented single pixel and small matrix of pixels with simple sequential readout logic. The results on these prototypes proved that the new design, proposed for DNW MAPS, is viable and that it presents good sensitivity to photons from ^{55}Fe and electrons from ^{90}Sr. Due to the good results achieved along the first three series of chip APSEL, the collaboration went on in the characterization of a Deep N-Well (DNW) MAPS device with a greater granularity and a smarter readout logic.

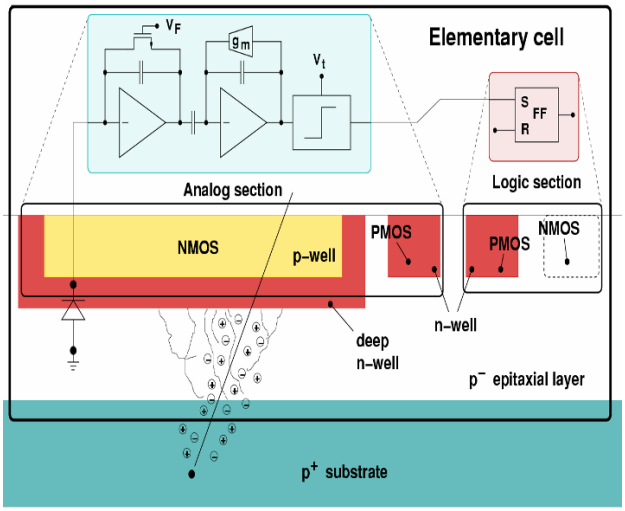


Figure 3.10: A Monolithic Active Pixel Sensor. The deep N-well acts as a charge collector and the inclosed NMOS electronics perform signal amplification, shaping and discrimination.

These studies led to the production of a third series of the APSEL chip. APSEL3 has been carried out simply after some rearrangement of the readout in order to lower the Signal-to-Noise ratio and power consumption. The total sensor capacitance has been reduced from 500 fF to 300 fF using for the new collecting electrode a combination of a DNW region and a standard N-well area. Power dissipation of the single pixel was reduced to 30 μ W. Particular attention has been given also to cross-talk problems induced by the digital logic signals. One of the six metal layer available in the used technology has been used as a shield between the sensor and the digital lines. The immediate following revision, APSEL4D, is a general improvement of the previous versions; the improvements deal mainly with the readout logic and with the pixel layout, trying to reduce cross-talk effects. A bigger matrix has also been integrated, now it covers an area of about 10 mm^2.

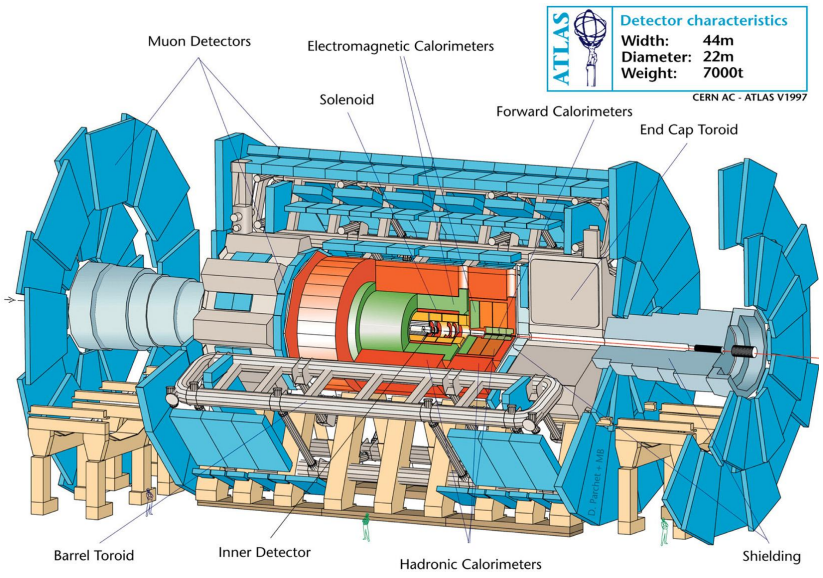


Figure 3.11: The Atlas detector scheme

3.6 The Large Hadron Collider vertex detectors

Two of the main experiments at the Large Hadron Collider at CERN, ATLAS and CMS, have a hybrid pixel silicon detector in their innermost cores and they can be regarded as a state of the art version of silicon detectors employed in particle physics high energy experiments. Let us briefly describe their main features.

The ATLAS Inner Detector

The ATLAS inner detector (see [12]), shown in Fig. (3.11), is composed of silicon devices aimed at recovering information on charged particles coming out of the collision point and traversing the detector, and its innermost part contains 80 million of electronic channels. The main purpose of this inner module is to identify the primary and secondary interaction vertices. In order to provide this information, there are three layers of pixel with a minimum distance of 5 cm from the interaction point. The pixel dimension is 50 $\mu m \times$ 400 μm (in the azimuth and longitudinal dimensions respectively). The detector is made of 46080 pixel modules with sixteen 250 nm CMOS front-end (FE) chips bonded to 18 $\times$ 160 pixel each. The front-end data are

sent to a Module Control Chip (MCC) with Low Voltage Differential Signals (LVDS), and the MCC forwards them to a Read Out Module (ROD), which is placed outside the detector, by using an optical fiber connection. The readout architecture is data-pull: all the incoming data are sent from the FE and MCC to the ROD, but only those triggered by an outside signal are extracted. Regarding the readout the matrix is divided in nine double columns of 160 pixels each. As the analog front-end is realized in such a way that the signal time length over the discriminator threshold (Time over Threshold, ToT) signal is proportional to the amplitude of the detector signal, each hit is provided with two 8 bit time stamps. One is the so called Leading Edge (LE) time stamp, corresponding to the beginning of the threshold crossing, the second, the Trailing Edge (TE) to the signal end. The hit extraction is concurrent on all the nine columns: starting with the higher row, all the hit pixels are extracted. The extracted hits are processes and the LE time stamp is subtracted from the TE providing the Time over Threshold information. LE and ToT, together with the hit address (8 bits for the row and one bit for the column) are saved in a buffer called End of Column (EoC), where they are stored during the latency time of the trigger corresponding to 3.2 μs). If, after the latency time, the trigger signal is not raised, the hits are delayed. If, on the other hand, the trigger is raised, the hits are sent to the front-end chip, where they are packed in the same trigger events and are serially sent with LVDS encoding to the MCC. The event is terminated by an End of Event word.

The CMS Inner Detector

The CMS detector (see [13]), shown in Fig. (3.12), was developed with the same purpose of ATLAS but with a different implementation. The vertex detector is made of 1500 sensor modules placed on 3 layers: 16000 total chips are mounted by bump-bonding, resulting in 45 million readout channels. In the central part of the detector, the modules host 16 Read-Out Chips (ROC) each and each chip reads out a 53×52 pixel matrix with a pixel dimension of $100\,\mu m \times 150 \mu m$. The maximum information rate for the innermost layer is 11 MHz/cm^2. As in the ATLAS detector, the readout has an external trigger and the matrix is divided in 26 double columns which are read in parallel, each of them having a single Column-OR signaling the presence of at least one hit pixel. The analog signal is converted by an Analog Digital Converter (ADC) and saved in a buffer (a procedure alternative to the the Time over Threshold determination of ATLAS). In each double column, the peripheral electronics begins the matrix sweep at each bunch crossing signal, looking for the hit pixels: the hits are written at a 40 MHz rate in a peripheral buffer

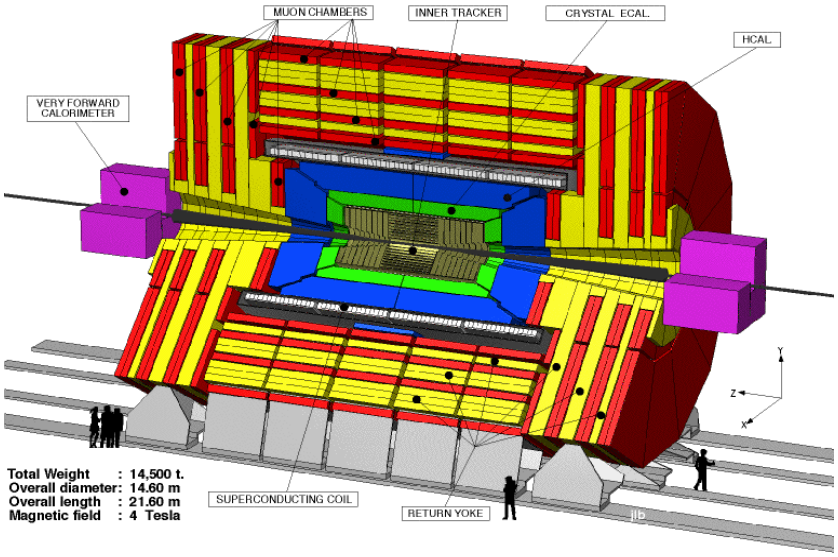


Figure 3.12: The CMS detector scheme

waiting for a trigger signal (as in ATLAS the latency for the trigger is 3.2 μm) in order to start the recording process. If the trigger signal is raised, all the recorded hits are sent to the Data Acquisition system through a 40 MHz analog optical link.

Chapter 4

APSEL4D

APSEL4D (see [14]-[17]) is composed of a bidimensional matrix of 32 $\times$ 128 Monolithic Active Pixels and a digital readout sparsification circuit, located outside the matrix area. The pixel dimension is $50 \times 50\,\mu m^2$ and the readout logic is based on standard cells with the implementation of a parallel technique, aimed at overcoming the readout speed limit of future large-matrix pixel detectors for particle tracking in high energy physics experiments. The ASIC can work in two different configurations: it can be connected to the actual full-custom matrix of active pixels or to a digital matrix emulator composed of standard cells, for testing purposes. For both operating modes, a slow-control phase is required in order to upload the chip configuration. The readout architecture is data driven and preliminary simulations and tests show that the readout system can cope with an average hit rate up to 100MHz/cm^2 when a master clock of 80MHz is used, while maintaining the overall efficiency above 99%. Let us now describe in more details its peculiar features.

4.1 The Chip

The main purpose of the APSEL4D realization were:

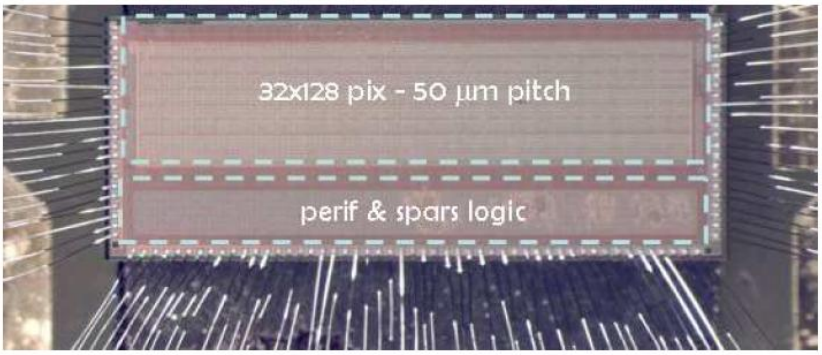
- minimizing the logical blocks realized with PMOS inside the pixel active area, in order to preserve the collection efficiency,
- minimizing the digital lines crossing the sensor area, allowing the readout scalability to larger matrices and reducing the residual cross-talk effects,
- minimizing the pixel dead-time by reading and resetting the fired pixels as soon as possible.

The sparsification and readout logic (see [18] - [22]) was realized outside the sensitive area of the matrix. A picture of the APSEL4D chip, bonded on the carrier module, is presented in Fig.(4.1). In order to minimize the digital lines crossing the active area, the matrix is organized in Macro Pixels (MP), i.e. square groups of 4×4 pixels. Each MP has only two private lines for a point-to-point connection to the sparsification logic. One line is used to signal that at least one pixel has been hit within the MP, and the second freezes the whole MP until the hits have been read out. Each pixel has been realized following the APSEL3T1 front-end flavor (transconductor in the shaper) and the layout is shown in Fig.(4.2). The pixel matrix has been realized with a full custom design and layout: during the CAD design of the chip a manual placement of the custom matrix was required in order to connect it to the digital logic block.

The readout logic

The readout logic performs the hit extraction from the matrix, encodes the space-time coordinates, and finally produces the digital hit-stream that is output of the chip sensor. The main goal of a sparsified readout architecture is the association of a spatial and time coordinate to each fired pixel. The term sparsified means that the hit extraction and encoding is focused on sparse randomly-accessible regions of the matrix, where the presence of fired pixels is known. 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, the sparse and randomly accessible regions are the pixels themselves. The basic idea is to incorporate some digital logic within pixels, exploiting, for example, a Deep N-well MAPS sensor technology, and realize a complex digital readout system in a dedicated portion of the chip area.

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 (see Fig4.4), the fast-or line gets 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 is incremented, all the new MPs that present an active fast-or are frozen, and they are associated to the precedent time counter value. This way all the fired pixels within a frozen MP are uniquely associated to the common time-stamp (TS) stored in the peripheral readout. In particular, during the Hold-Mask phase the matrix is frozen by deasserting all the RESET row signals, so that 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



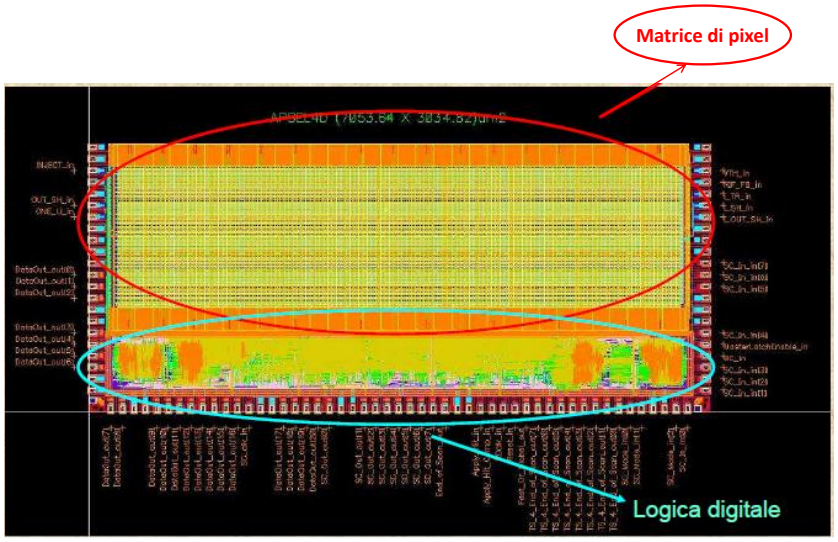


Figure 4.1: The APSEL4D chip, bonded on carrier. The 32×128 matrix is shown on top, while the read-out logic is located in the bottom part of the chip.

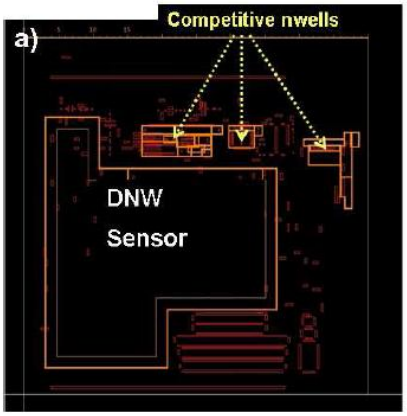


Figure 4.2: The single pixel layout. The collecting electrode is marked as Deep N-Well Sensor and the competitive N-wells are the CMOS N-type implantations.

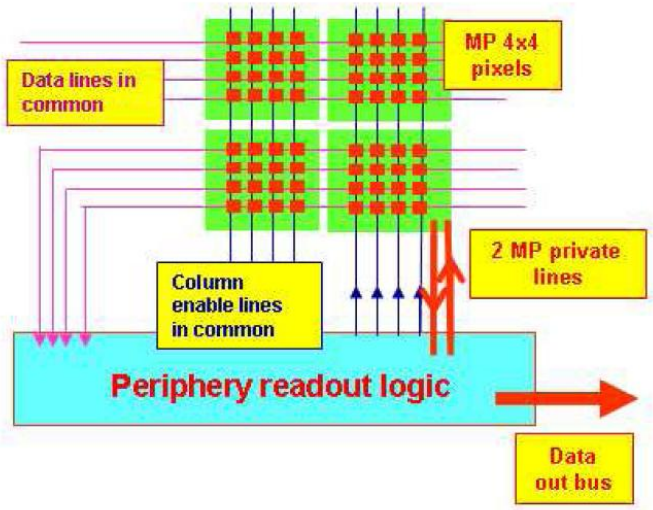


Figure 4.3: The APSEL4D matrix/logic interconnection signals. 32bit-wide row readout bus, 128 column enable signals and 2 private lines for each MP, one for the hit interrupt and one for the freeze command.

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. Subsequently 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 sparsification and the readout logic have been synthesized using a VHDL (Very high speed Hardware Description Language) code, in order to provide high-level smart procedures. The gate-level design, obtained after the project synthesis, has then been implemented using standard-cell libraries provided by the STM CMOS 0.13 μm technology. After the computer aided layout was performed, a manual routing operation allowed to connect the control lines of the digital logic to the full custom pixel matrix. The readout system works on three main clocks:

- the readout clock (RD-clk, designed to run at up to 100 MHz),
- the BC clock (BC-clk, typically runs at 5 MHz),
- the Slow Control Interface clock (SC-clk).

. Simulations show that the readout system can cope with an average hit rate up to 100 Mhit s^{-1} on a 1 cm^2 pixel matrix if a master clock of 80 MHz is used, while maintaining an overall efficiency above 99%.

The readout system uses the following signal infrastructure :
256 private hit-lines connect each MP to the digital logic in order to flag if one (or more) of their 16 pixels contains an hit flag, 256 more private lines allow to freeze independently each MP and a 32 bit column-wide common data bus is used for the readout operations. When a pixel crosses the threshold, the associated MP's hit-line gets fired. Other pixels within the same MP can still be fired before the arrival of a BC clock rising edge. When a BC edge arrives, the digital logic freezes the status of all the MPs that have been fired; this means that the hit/no hit status of their 16 pixels cannot change anymore, until the next readout and reset phase. The readout phase takes place in parallel, one pixel column per RD-clk cycle. A 32-bit row bus connects the desired pixel column to the readout logic that is meant to sparsify the hits and to associate a time label. Each column of pixels is provided with a global enable signal, decoded by the readout logic in order to have only one column at a time driving the common row bus as shown in Fig.(4.5). The readout logic enables sequentially the 4 columns of a macro-column if at least one MP of them is fired, reading one column per RD-clk cycle. During the fifth RD-clk cycle all the pixel latches of the macro-column (4 $\times$ 32 pixels) are

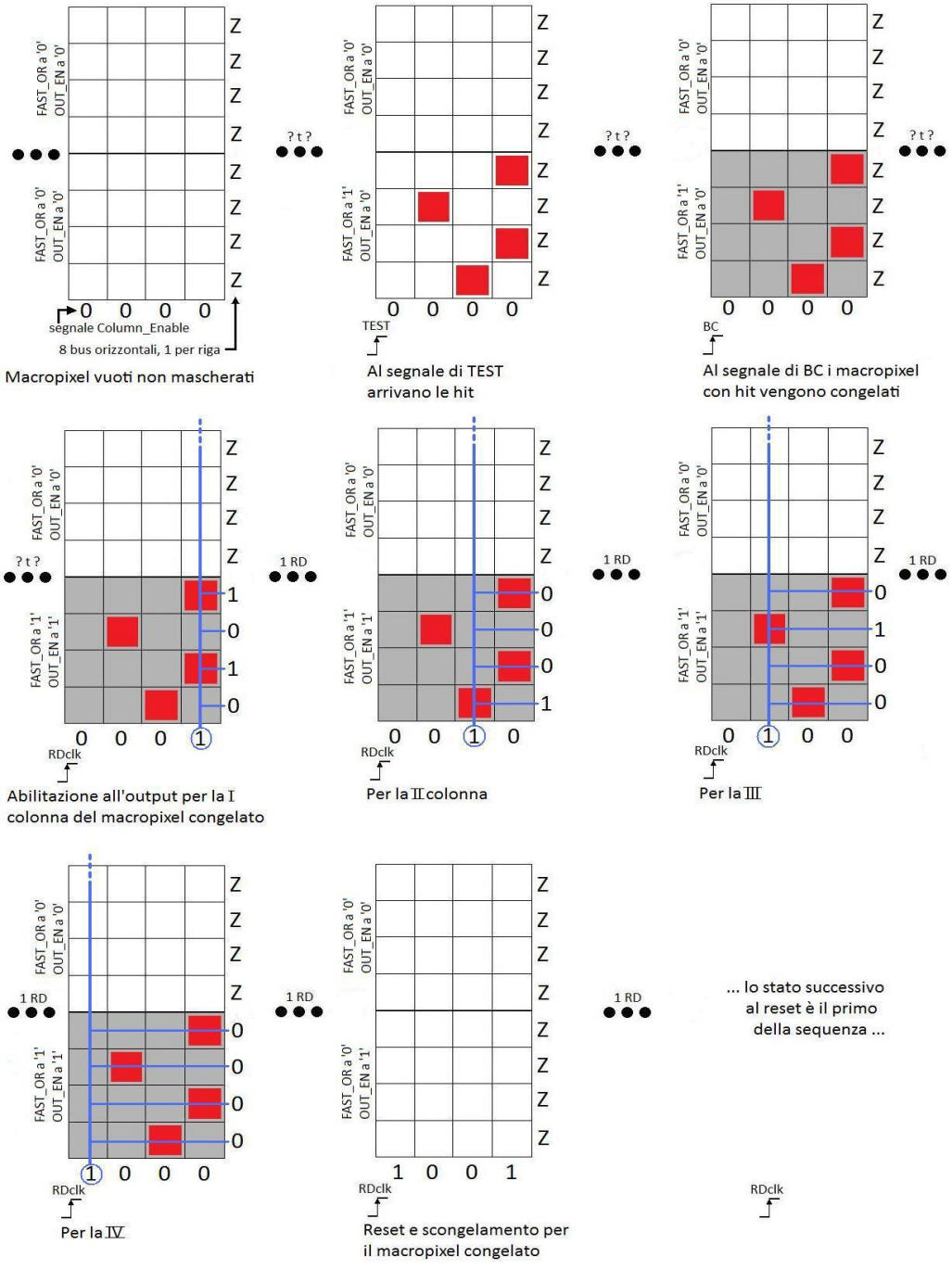


Figure 4.4: The pixel readout phases

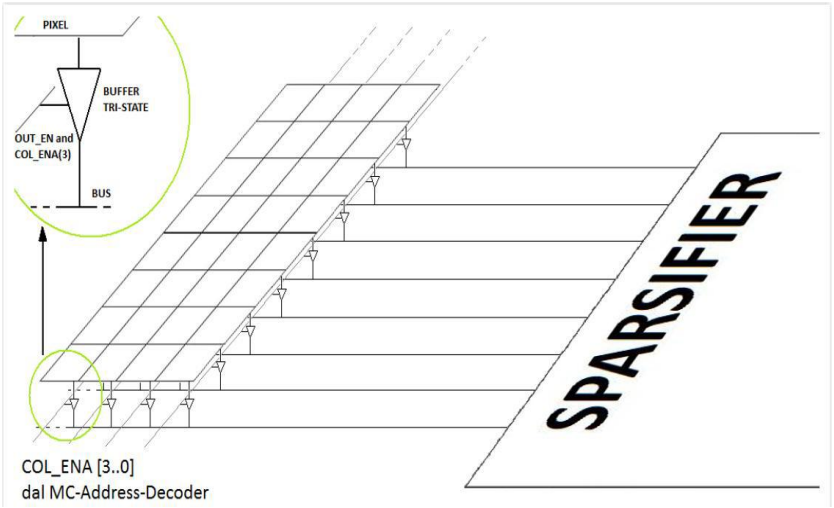


Figure 4.5: The row readout bus architecture. 32bit-wide bus for the readout of the latches of a pixel column.

reset. Thus, in principle, a fully fired matrix can be read in 128 (column read) + 32 (MP reset) RD-clk cycles.

The sparsifier logic is modular, which means that the 32-bit common row bus is analyzed by four separate sparsifiers, each one working on 8 rows only. Each sparsifier has its own private storage memory (called barrel in the following) as shown in Fig.(4.6) which is implemented inside the chip digital logic to retain a maximum of 160 hits while they are sent over the data out bus. The Sparsifier-OUT module is responsible then for the queuing of hits into the Barrel Final: this is the chip data output port which has a bandwidth of 1 hit per Readout clock cycle. The data-driven architecture pushes out the hits from the APSEL4D chip on a 21-bit parallel bus, in which the 21st bit is a data-valid flag. In fact, once a pixel column is read, it is sparsified and time-labeled; these spatial and time coordinates are then stored in 20-bit word whose structure is showed in Table (4.1). Now let's have a deeper look at the hit format: the output hit word is 20 bit long, the row address is unique, identified in the Pixel Row field by a number from 0 to 31 that follows the scheme of Fig.(4.7). The global column address, instead, must be calculated with the following formula
Global pixel column = Macro Column $\times$ 4 + MP column
Several additional features are present in the digital logic, included to provide

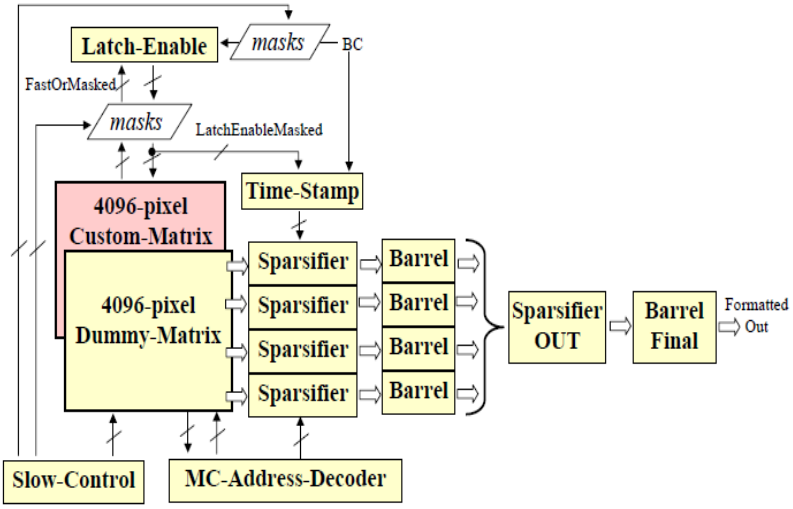


Figure 4.6: The readout hierarchy scheme

Fields	Bits	Information
hit[19:15]	5	Pixel row (0 - 31)
hit[14:13]	2	MP column (0 - 3)
hit[12:8]	5	Macro Column (0 - 31)
hit[7:0]	8	Time Stamp (0 - 255 BC edges)

Table 4.1: The APSEL4D hit format. MP column is the relative address of a pixel column within a MP; Macro Column is the global address of a MP column. Time stamp is a time label associated to a counter of BC-clock edges

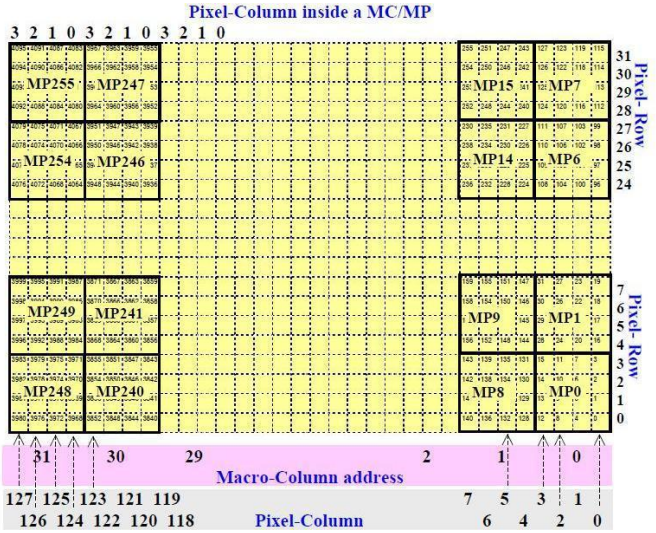


Figure 4.7: The APSEL4D matrix and Macro Pixels

on-chip debugging capability and slow control communication. A full size dummy matrix has been realized within the digital logic in the form of a 4096 array of latches remotely configurable via the slow control interface. This was meant to test the readout capability of the chip without the interaction with the real matrix. When the chip is configured, it can be started in real or dummy mode connecting the readout logic to the real sensor matrix, or to the dummy digital matrix. A pixel kill-mask feature is implemented in the chip as well, mask patterns can be loaded into the dedicated registers via slow control. In details, it is possible to mask entire rows or single Macro Pixels. This feature is meant to exclude noisy pixels that waste readout time and transmission bandwidth. The Slow Control Interface operates on the 8-bit wide SC data bus, it is synchronous with the SC-clk and receives commands on a 3-bit wide bus called SCmode. The instructions on the SCmode bus can address single registers or initialize sequential loads of bytes in an array of shift registers. The SCmode coding 001 individuates, for instance, the loading instruction of the 256+32 masking bits. In this case, on each SC-ck edge, 8 bits a time are loaded into the mask buffer, starting from the row masking and following this order: the first byte received corresponds to the mask of the first 8 rows if the whole mask pattern is successively loaded. Successive bytes individuate the mask patterns for the MPs as numbered in Fig.(4.7). In a similar way the hit pattern is loaded into the dummy

matrix with the SCmode code 000. In this case the whole operation is quite long and it takes 4096/8 = 512 SC-clk cycles. Once the pattern is loaded, the Dummy actual register must be set high by setting SCmode = 010. The external dedicated signal "Apply hit" is used in order to make the hits visible to the readout logic once the chip is in set to "run mode". Once the chip is configured, it can be put in run mode by setting SCmode = 111. If the logic is connected to the real matrix, there will be a certain data rate at the output, depending on the analog threshold and on the signals at the sensor cells. If the dummy matrix is set, instead, one should expect at the output the same hits that were previously loaded. The whole digital logic can be reset with an active-low dedicated pin, but a soft reset can also be sent with a SCmode = 100 in order to set the BC time counter to 0.

Matrix sweep

The matrix (Dummy or Custom) is swept as follows:

- the matrix is always swept from left to right and only the frozen MPs are considered,
- the 4 Sparsifiers work in parallel and cope with 8 rows of pixels each, out of the 32 rows,
- at any time one only column of pixels is considered, so it is not possible that different Sparsifiers read different columns at the same time,
- at any clock time, 8 hits per Sparsifier can be read at most, leading to a maximum of 32 hits reading at clock cycle.

Nevertheless, if that is the case, the successive Barrels force very soon the Stand-By condition as if a whole column is lit. This is a consequence of very high (local) occupancies which are not compatible with the fact that the port outputs only one hit at a time. The event of local high occupancy can be handled as long as the average hit-rate is smaller than the throughput.

The Signals

Figure (4.8) shows the I/O ports of the entire circuit plus SEED-VECTOR and TEST ports used only for design and simulation purposes: they are not present on the final ASIC. Here is a list of all the I/O ports:

- SC-In is an 8-bit port used to load the internal registers, depending on the SC-Mode value, at the rising edge of SC-clk,

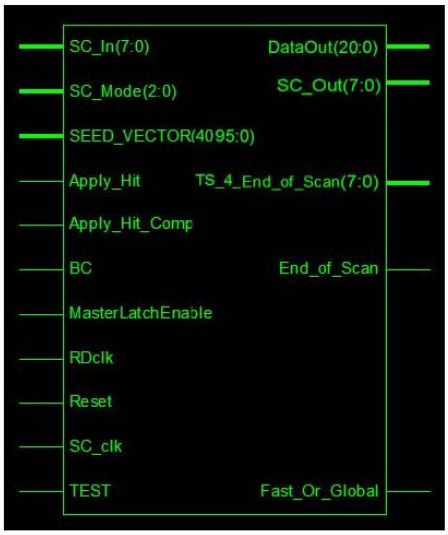


Figure 4.8: The input and output ports scheme

- SC-Mode is a 3-bit port that specifies the required slow-control operation. During a slow-control phase several operations may be requested:
  - a dummy-pixels load, in digital-mode, to configure the dummy matrix. It requires 512 SC-clk periods as it loads 8-bits at a time as internal hits,
  - a MP-mask load to select the MP that must be really seen from others which must be masked. It requires 32 SC-clk periods as it loads 8-bits at a time,
  - a ROW-mask load to select the ROW that must be really seen from others which must be masked. It requires 4 SC-clk periods as it loads 8-bits at a time,
  - an operating-mode selection to enable either the custom-mode or the digital-mode. It requires one SC-clk period as it loads just a 1-bit register period as it is a no-load operation,
  - a Soft-Reset to reset the time-stamp counter. It requires one SC-clk period as it is a no-load operation,
  - a time-stamp-read to see the time-stamp counter value. It requires one SC-clk period as it is a no-load operation.

  All these operations are synchronized on the rising edge of the slow-control clock, SC-clk, by following a proper scheme. Here is a summary

of the operations that can be selected via the SC-Mode slow-control port:

- SC-Mode = 000 : load 4096 Dummy-Pixels. First MP address ranges from 0 to 15, second from 16 to 31, and so on up to 256th MP that ranges from 4080 to 4095. At each SC-clk period a half MP is configured, 512 periods for 256 MPs. This is only valid in digital-mode,
- SC-Mode = 001 : load 256 + 32 Mask bits. When loading the masks the Row-masks must be placed first then follow the MP-masks. Altogether 4 periods are required to load the Row-masks and other 32 periods to load the 256 MPmasks. As all the masks are cascaded, if a number of SC-clk loading periods smaller than 36 is used, the masks are partially loaded. However, if only a subset of masks is required, a number of SC-clk loading periods smaller than 36 could be enough. For example if only Mask(2) and Mask(10) are required, only one SC-clk loading period is enough as they belong to the first 8-bit set of loaded masks. During the normal running mode, MP-mask(i) masks MP(i).
- SC-Mode = 010 : load the Actual-Dummy register. This enables the custom-mode when is low and the digital-mode when is high,
- SC-Mode = 011 : read the Time-Stamp counter that is posted on SC-Out port.
- SC-Mode = 100 : Soft Reset request to reset only the time-stamp counter inside the Time-Stamp,
- SC-Mode = 101 : 1-bit rolling shift on the Dummy-Pixels (Nth–> Nth+1, 255th –> 0th). This is valid only in digital-mode,
- SC-Mode = 110 : add 1 to each MP 16-bit configuration. Thus, if a MP is configured with 16 bits, then this configuration is added to 0001 to get a new configuration. This is valid only in digital-mode,
- SC-Mode = 111 : this is used as the normal running mode once either the dummy-pixels have been loaded in digital-mode or the matrix of MAPS is connected in the custom-mode,

The SC-Mode= 010 is crucial because it sets the operating mode. The SC-Mode= 111 is used as the normal run operation after the slow-control phase. SEED-VECTOR is a 4096-bit vector that provides the 256 MPs with a 256 different stimuli composed of 16 bits each. The stimuli are read from a text file, whose single lines correspond to a

coded single stimulus, and are applied whenever a TEST rising edge is provided. Thus, if n TEST pulses are provided, 256 × n lines of the file of stimuli are read, and 256 × 16-bit patterns are provided to the 256 MPs n times. These stimuli can be seen or not depending on the MPs Latch-Enable status. In fact, if this is low, it means that the MP is frozen and blind to new hits.

- Apply-Hit and Apply-Hit-Comp are signals, considered only for digital-mode, which force the stored dummy pixels to be copied into mirror registers (called Latches even if hardwired with Flip-Flop) to be read out as they were output latches of the pixel sensors. The process activates on the rising edge of SC-clk. Thus, when the Apply-Hit/Apply-Hit-Comp= 10 configuration is seen at the SC-clk rising edge, the Dummy-Pixels (in their 0/1 status) are copied to the 256 Latches, while the 01 configuration of Apply-Hit/Apply-Hit-Comp indicates that a complemented version (1..0 and 0..1) of the Dummy-Pixels must be stored on the Latches. Once the 10 or 01 Apply-Hit/Apply-Hit-Comp configurations are applied, the readout logic starts if it was standing or continues its matrix-sweep if it was on running. Once the whole matrix is swept, the read process stops. Then, a 00 configuration is required to let the column sweeping process start again from the left-most column. This allows avoiding immediate re-readout of the hits,

- BC is the Bunch-crossing signal, considered asynchronous, which forces an internal time-stamp register into being copied as a time-mark for any of the MPs that contain at least one hit (MPs FastOr is high). This is carried out after the BC is synchronized to the global readout clock RD-clk. The BC signal is masked by the SC-Mode=0,1,2,4; in other words, during slow-control phases, the BC external signal is not seen,

- MasterLatchEnable is an asynchronous signal inserted with an AND logic function along with the internal 256 MPs Latch-Enable signals and with the 256 Mask registers. In other words, eventually, the Latch-Enable signals that enter the MPs are given by: Latch-Enable(i) <− LatchEnableNMatrix(i) and Mask(i) and MasterLatchEnable where Mask(i) are the 16 masks loaded via slow-control and LatchEnableN-Matrix(i) are the 16 signals provided by the readout control unit for the 16 MPs. These latter signals are synchronized with the rising edge of SC-clk. Masked MPs have Mask(i) = 0. If MasterLatchEnable is low all the Latch-Enable(i) are forced to 0 and the MPs to which they

belong are frozen and kept blind until MasterLatchEnable goes high again,

- RD-clk is the 40 MHz global clock,
- Reset is a hard-reset that resets all the internal registers to a predefined state as soon as it is seen and synchronized via RD-clk: basically 0 for the registers and 1 for the masks, digital mode for operating mode. It must last high for more that one SC-clk and RD-clk period to take effect,
- SC-clk is an up to 40-MHz slow-control clock. It is asynchronous with respect to RD-clk when it is used to load the internal registers as the Dummy-Pixels in digital-mode or the masks Mask(i) in both modes. When a given configuration of hits has been loaded into the Dummy-Pixels, and they have to be copied into the Latches via Apply-Hit/ Apply-Hit-Comp couple, the SC-clk clock must be run with the same frequency of the RD-clk clock. This is why, in digital-mode, the dummy matrix is updated on the rising edge of SC-clk while it is frozen, read and reset on the rising edge of RD-clk. Using two different clock frequencies may lead to unpredicted states. Let's say that SC-clk frequency must be not lower than RD-clk during digital-mode matrix readout. In custom-mode instead, the SC-clk may be run at any frequency during configuration,
- TEST is an asynchronous non-real port used to specify when the SEED-VECTOR must be read and its values must generate the simulated hits, in digital-mode,
- SC-Out is an 8-bit port used to either shift out the dummy-pixels when SC-Mode = 000 or to readout the time-stamp internal value of the BC-counter when SC-Mode = 011, at the rising edge of SC-clk,
- DataOut is a 21-bit port. It is the output data bus with the format described in section 2.2,
- End-of-Scan is a single bit that indicates the end of sweeping phase. In other words, a pulse on this port confirms that a total sweep over the columns of pixels, from left to right, has finished,
- TS-4-End-of-scan is an 8-bit port used to output the time-stamp counter value related to the End-of-Scan. It should be noted that this port, along with End-of-Scan, can provide data before the internal queuing

system has been emptied. In more detail, these ports indicate that a given sweep with a given Time Stamp is finished and all the data related to earlier Time Stamps are for sure closed. However, the data can still be into the barrels. As the entire queuing system can hold up to 160 hits, the system can take up to 160 clock periods to output the data. Thus, if the occupancy is very high (not a great physics sense), these ports do not provide a significant contribute. Conversely, if the queuing system is not congested, the data exit very soon once the matrix is swept and the above ports make their sense,

- Fast-Or-Global is a single bit that indicates the status of the MPs FastOr output pins. In particular it is a global NOR logic function of the 256 FastOr signals after the masking operation.

Configuration Steps

For a proper operation the APSEL4D chip requires, after being powered, a couple of clock periods, both on SC-clk and RD-clk, with Reset= 1 (and SC-Mode $\leq$ 11). This is the way all the internal registers are initialized to their default values and the chip is ready to run in digital-mode with all the masks set to 1 and the pixels switched off. Then, depending on the required operating mode, other slow-control operations might be provided. During this phase it is important that SC-Mode does not move forth and back from the highest to the lowest values. In other words, if the Dummy-Pixels or the MP/ROW-Masks have to be loaded, this step must be done after reset phase. Then, when all the desired internal registers have been loaded, along with the operating mode register Dummy-Actual, the SC-Mode= 111, meaning running mode, can be provided once and together with at least one SC-clk rising edge. The SC-Mode= 0, SC-Mode= 1 and SC-Mode= 11 may be swapped to each other. In custom-mode, after the configuration the SC-clk can be frozen while, in digital-mode, it cannot. In digital-mode, besides 32 periods of SC-clk with SC-Mode= 1 and one period with SC-Mode= 10 (that could be omitted as it overwrites the default operating digital mode), 512 periods with SC-Mode= 0 are required to load the Dummy-Pixels.

4.2 The Matrix Characterization

The chip was designed to run with a readout clock frequency up to 100 MHz (20 MHz in test beam), a maximum matrix readout rate of 32 hit pixels/clock tick and a local buffer of maximum 160 hits in order to minimize the matrix

sweep time. Preliminary simulations show that the readout system can cope with an average hit-rate up to100MHz/ cm^2 if a master clock of 80MHz and a bunch time of 200 ns are used, while maintaining an overall efficiency above 99%. The basic functionality of the readout has been tested with a readout clock up to 50MHz, with the digital matrix implemented for this purpose at the chip periphery, and it works as expected.

The APSEL4D matrix has been fully characterized (see [23] - [25]) showing very uniform performance over the twenty chips measured. The basic functionality of the readout has been tested, with a readout clock up to 50 MHz, using a dummy digital matrix implemented for this purpose in the chip periphery. The readout circuit works as expected even at 100% occupancy.

Noise

The first part of tests was aimed at establishing the baseline and noise thresholds for the Matrix of pixels. From the values obtained on single pixels, it is possible to compute the average, in order to determine the baseline and noise values to be associated with the whole matrix. The noise measurement and evaluation of the threshold dispersion of the analog part of the pixel have been performed on the 32×128 pixel matrix measuring the background hit rate as a function of the discriminator threshold of the data acquisition chain inside the pixel (see Fig.4.9)

In fact, as the matrix has only a digital output, the pixel signal waveform is not available and one can only determine whether a signal is over a threshold or not. The best way to proceed is to make a threshold scan(we will call it DAC scan): the data are acquired at different step values for the threshold potential. The discriminator of the analog to digital conversion section (one bit: hit/no hit) compares the signal to the threshold voltage provided by a programmable DAC chip and produces a digital signal. The latch is sensitive only to 0-to-1 transitions, so the pixel is fired only if there is a threshold crossing. When the voltage provided by the DAC is much less than the baseline value, the noise fluctuations do not make any crossing and no hit is recorded. If the threshold approaches the maximum noise fluctuations around the baseline, the pulse height start to cross the threshold more and more frequently, until the pixel occupancy is saturated. Finally, for DAC values much higher than the baseline, the hit rate decreases and becomes zero as, again, no threshold crossings occur anymore. Figure (4.9) shows the actual results on some particular pixel. Statistical considerations on the noise (see [26]-[27]) bring to a theoretical expression for the occupancy

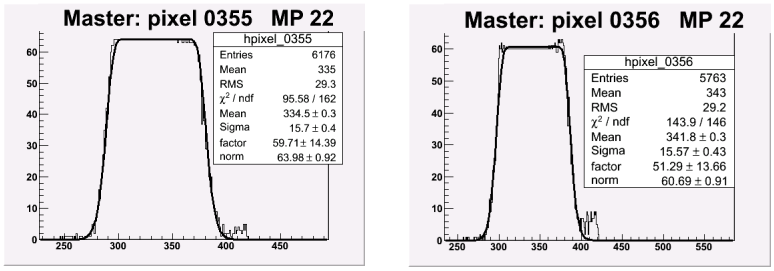


Figure 4.9: The typical occupancy as a function of the discriminator threshold for some individual pixels

as a function of the discriminator threshold of the form

$$\text{Occupancy } = 1 - \exp\left[-A\, e^{-\frac{(V_{th}-V_{bl})^2}{2\sigma^2}}\right] \tag{4.1}$$

where A is a constant related to the noise frequency at $V_{th} = V_{bl}$, V_{th} and V_{bl} are the threshold and baseline values and σ is the dispersion. By using a fit to the expected distribution we can deduce a pixel average Equivalent Noise Charge of about 75 e- (10.5 mV) with 20% dispersion inside the matrix, and a threshold dispersion of about 60 e- (8 mV). In order to determine this value one has to use the conversion equation

$$ENC(e^-) = \frac{\text{noise (mV)}}{\text{Gain (mV / fC)}\ \ 1.6\,10^{-4}\text{fC}} \tag{4.2}$$

where the gain is determined with a ^{55}Fe calibration which we will describe in the following. A typical noise distribution for the 4096 pixels of the matrix is shown in Fig.(4.10) and (4.11). Measurements performed on dedicated test structures indicate that the metal shield inserted in the new pixel layout design is effective in reducing at the level of the noise the cross-talk due to the capacitive coupling between the digital lines and the sensor. Nevertheless new effects of digital crosstalk have been observed in the APSEL4D, apparently correlated with the readout activity, and are still under investigation.

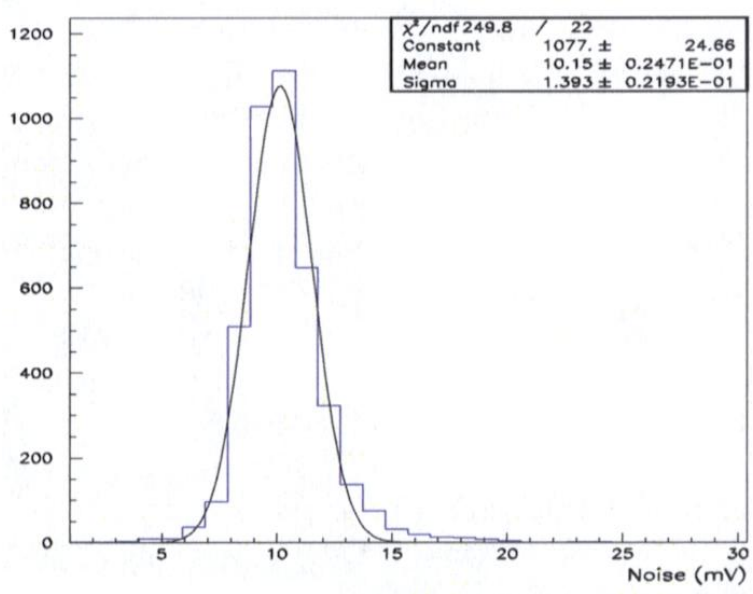


Figure 4.10: The noise distribution for the 4096 pixels of the APSEL4D matrix

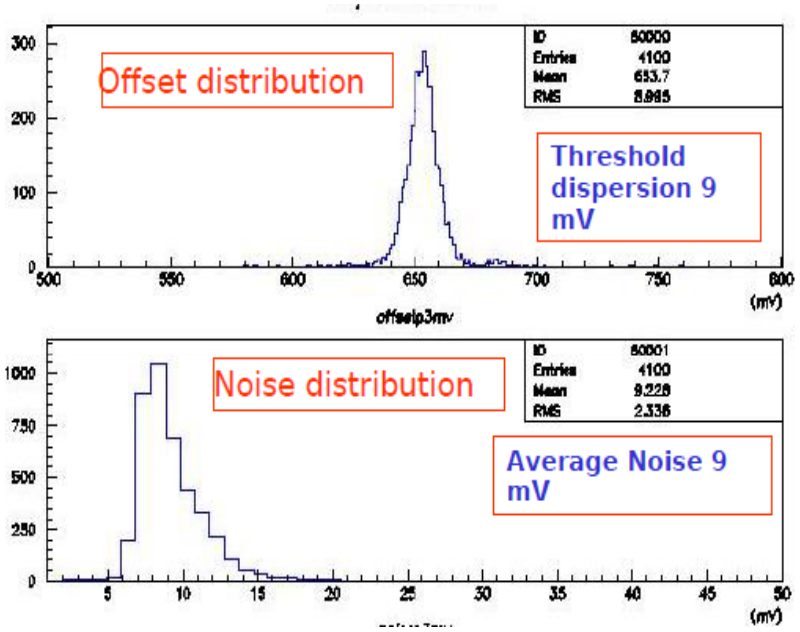


Figure 4.11: The offset and noise distribution for the 4096 pixels of the APSEL4D matrix

^{55}Fe and ^{90}Sr Calibration

The absolute gain calibration of the matrix has been performed using the 5.9 keV line from an ^{55}Fe source, which is a radioactive emitter of monochromatic X photons with an energy of 5.9 keV. The interaction of these photons with silicon produces photoelectron emission. The photoelectrons are completely collected in the active area of the pixels and this allows to determine the preamplifier and front end gain by using the following procedure. As we know the photoelectron energy and considering that 3.6 eV are needed in order to create an electron-hole couple, it is possible to determine the number of produced electrons

$$Q_{Fe} = \frac{5.9 \cdot 10^3 \text{eV}}{3.6\,\text{eV}} = 1638\,e^- = 0.26\,\text{fC} \tag{4.3}$$

which is the total charge released in the sensor. The curve of the energetic release thus obtained could be directly fitted with a Gaussian distribution, if the analog information of the single pulse height was available. But since no analog information is available, the photo-peak is reconstructed from the differential rate as a function of the discriminator threshold. With this technique an average gain of 890 mV/fC has been measured with a typical dispersion of about 6% inside the matrix. The ^{55}Fe calibration peak is shown in Fig.(4.12), summing up the contribution of all the pixels of the matrix.

The test with the ^{90}Sr source, on the other hand, is used in order to determine an estimate of the most probable value for the Landau curve describing the energy release, but this time an analog measurement was performed on a small pixel matrix. The fit of the expected Landau curve suggests an ENC of 40 electrons, a gain of 860 mV/fC and a signal to noise ratio S/N = 23. The results are displayed in Fig.(4.13)

Test Beam

Finally a beam test was performed at the T9 station of the CERN PS, where hadrons of momentum of 12 GeV/c are available. In order to minimize the effect of multiple scattering on the resolution, we have chosen the maximum available momentum of the impinging particles, that is protons of 12GeV/c, with spills of 480ms and typically $10^4 - 10^6$ particles for each spill. The beam profile had a width of about 0.5cm root mean square. For the reference telescope, we used four silicon strip detectors, 2cm $\times$ 2cm double-sided, AC coupled with 50 μm readout pitch, one pair upstream and one downstream the device under test (DUT). The modules (telescope and DUTs) were placed on a customized motorized table with remote control. The whole setup of

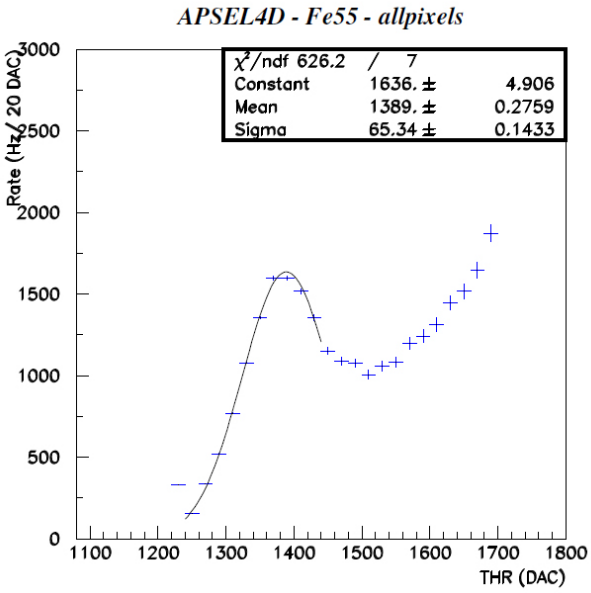


Figure 4.12: The APSEL4D absolute calibration with a ^{55}Fe source. The total contribution of all pixels has been summed up. The graph is a function of the discriminator threshold.

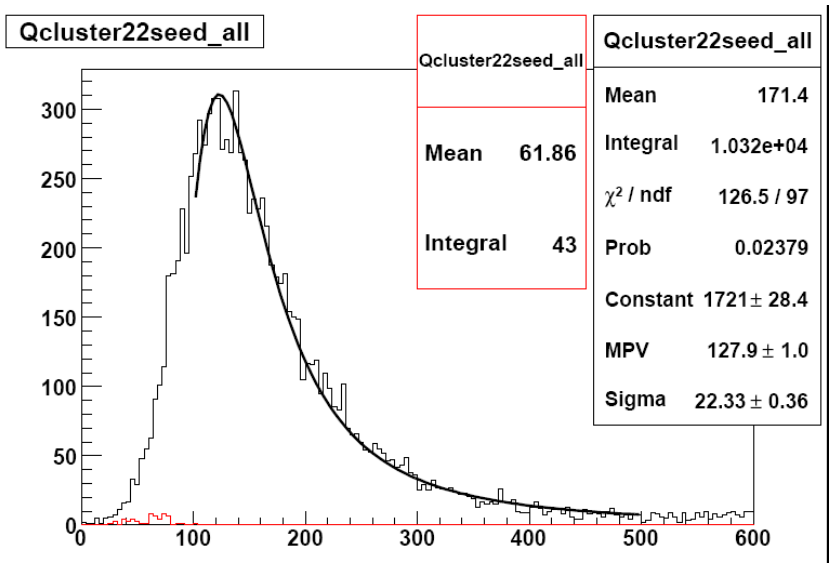


Figure 4.13: The APSEL4D calibration with a ^{90}Sr source. The total contribution of all pixels has been summed up. The graph is a function of the cluster signal and a Landau Fit is superimposed

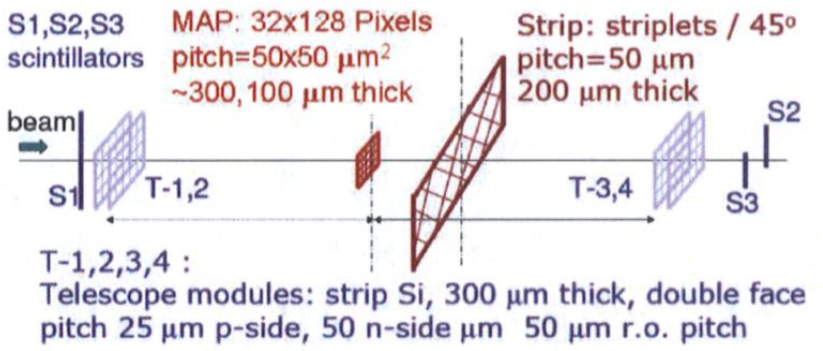


Figure 4.14: The Beam Test Setup

the beam test is schematically shown in Fig.(4.14). The main goal of this test was to measure the efficiency and the resolution of the DNW MAPS chip APSEL4D. During the beam test, with a DAQ rate of about 30 kHz, about 90 million events were written on the hard disk with different trigger configurations. The results on MAPS hit efficiency and resolution confirm that the Deep N-Well device is working well. Tracks reconstructed with the reference telescope were used to extract the APSEL4D hit efficiency with the following technique. For each track, whose extrapolated impact point is inside the MAPS fiducial region, the distance between the expected and the measured hit (residual) is calculated. In the residual distribution, for which an example is shown in Fig.(4.15), real hits tend to peak while fake noise hits are uniformly distributed. The hit efficiency is evaluated as the ratio between the number of real hits, extracted from a fit to the residual distribution, and the number of tracks extrapolated in the MAPS fiducial region. The results on APSEL4D hit efficiency as a function of the discriminator threshold are shown in Fig.(4.16). We have studied the efficiency of the MAPS as a function of threshold for two devices, specified as Chip 22 and Chip 23. The two devices are equivalent except for differing silicon thickness: Chip 22 is 300 μm thick, whereas Chip 23 is 100 μm thin. Fig.(4.16) shows the measured hit efficiency as a function of the threshold used to define a hit expressed in electron units (minimal charge collected).

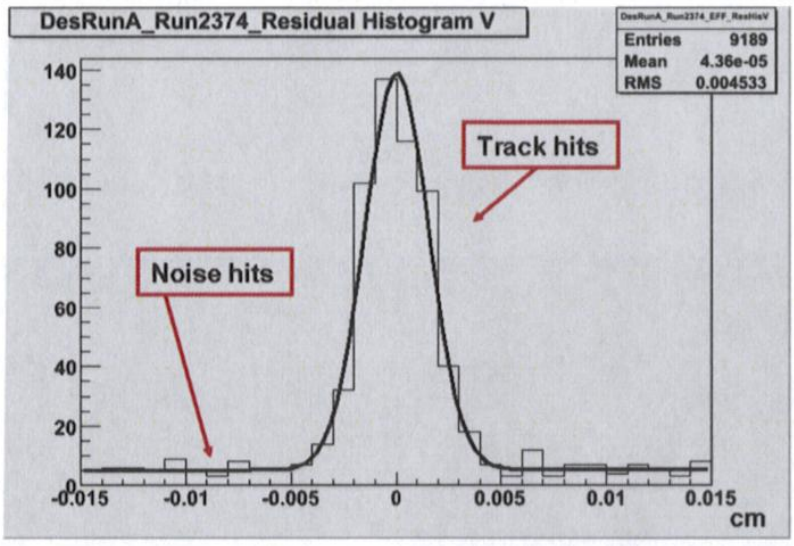


Figure 4.15: The residual distribution (y coordinate) after alignment. Real hits contribute to the peak while fake noise hits are uniformly distributed

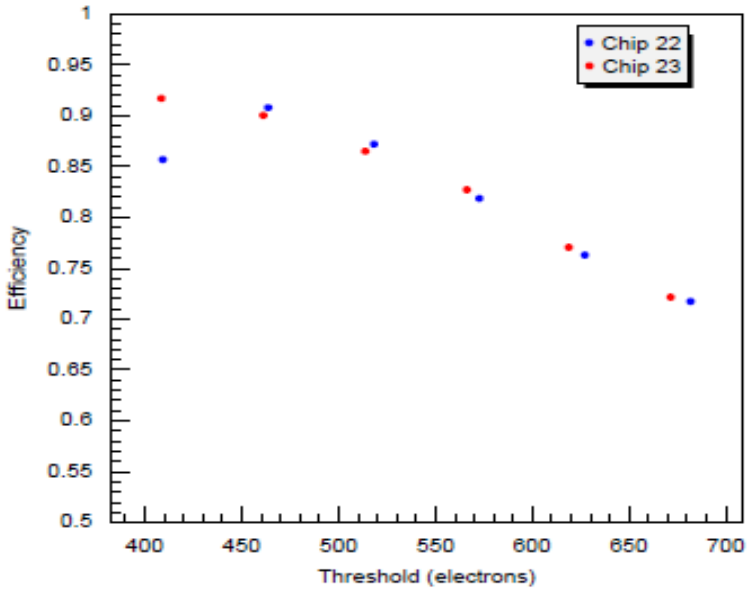


Figure 4.16: The efficiency results for two MAPS detectors, taken from a single threshold scan. The statistical uncertainty on each point is smaller than the size of the plotting symbol.

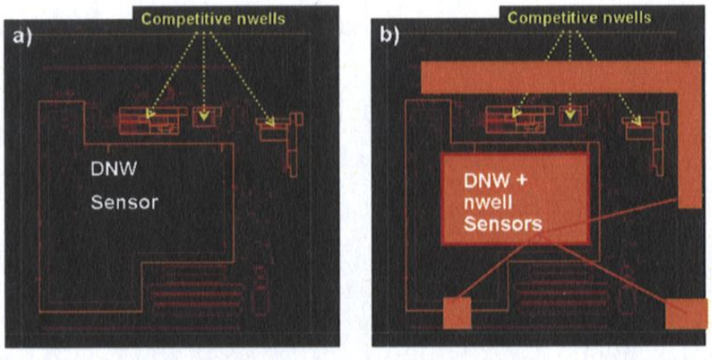


Figure 4.17: The schematic drawing of the APSEL4D cell layout : the central collecting electrode and the competitive n-wells for the standard cells, on the left, and for cells with improved sensor geometry (multiple n-wells collecting electrodes around the competitive n-wells connected to the main DNW central sensor), on the right

At the lowest thresholds we observe a maximum efficiency of approximately 92% and we see the expected general behavior of decreasing efficiency with increasing threshold. This is in agreement with what is expected taking into account the presence of competitive n-wells in the pixel layout and a cell fill factor of about 90% (shown in Fig.4.17, left). The noise occupancy for this range of thresholds varied from $2.5\,10^{-3}$ to 10^{-6}.

In the APSEL4D matrix, the sensor geometry has not yet been optimized, although there are indications from a fast device simulation, developed for this purpose, that it is possible to improve the hit efficiency while maintaining the same fill factor. Improvements in the efficiency are expected for example inserting in the cell multiple collecting electrodes around the competitive n-wells (Fig.4.17, right). Test structures with the improved cell design have been already implemented. The fast device simulation developed to optimize the sensor geometry takes into account the ionization process, charge diffusion, and the front-end response. The response of the simulation is in good agreement with the measured hit efficiency. The effect of competitive n-wells on the collection efficiency has been studied on data measuring the hit efficiency as a function of position within the pixel. We have in fact studied the efficiency for detecting hits as a function of the track extrapolation point

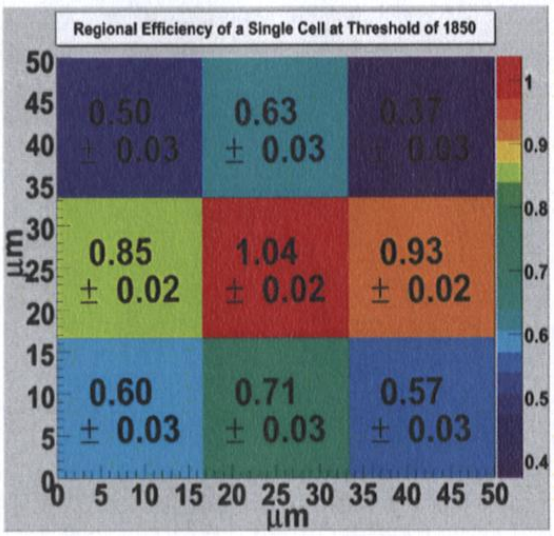


Figure 4.18: The hit efficiency inside the pixel cell (threshold 0.67 MIP).

within a single pixel. As the pixel has internal structure, with some areas less sensitive than others, we expect a varying efficiency as a function of position within the cell. The pixels are divided into nine subcells and the efficiency is measured in each subcell. The effect of cross feed among subcells, due to the uncertainty on the track impact point extrapolation (about 15 μm), has been taken into account. The results, shown in Fig.(4.18), point to a clear correlation with the cell layout (Fig.4.17). The central region, fully covered with the DNW sensor, reaches 100% efficiency, even with a quite high threshold applied of about 0.67 MIP, while the pixel periphery has a lower efficiency, especially in subcells where the competitive n-wells are located (top part of the pixel). The response uniformity across the chip has also been checked. The hit efficiency for the matrix subdivided in regions of 8 x 16 pixels is shown in Fig.(4.19). The observed spread of about 10% in the efficiency is consistent with the average gain and threshold dispersion of about 6%: since a single threshold is applied across the entire chip a similar dispersion in the effective threshold applied on each pixel is expected. We measured the intrinsic resolution σ_{hit} for the MAPS devices from the width of the residual distribution by subtracting in quadrature from the residuals σ_{res} the contributions from track extrapolation uncertainty σ_{track} and multiple scattering effects σ_{MS}.

The MAPS intrinsic resolution σ_{int} has been extracted from the width of the residual plot ($\sigma_{res} \simeq 5\mu$m, see Fig. 4.10), as the contribution from multiple scattering has been taken as $\sigma_{MS} \simeq 6\mu$m and the error on track extrapolation $\sigma_{trk} \simeq 5\mu$m, while using $\sigma^2_{int} = \sigma^2_{res} - \sigma^2_{trk} - \sigma^2_{MS}$. The results for the intrinsic resolution is shown in Fig.(4.20) for the y coordinate. We have found a resolution variable within the range $12 - 15\mu$m, depending entirely on the cluster size and on efficiency and in agreement with the single hit expectation of (pitch) $/\sqrt{12} = 50\ \mu$m $/\ 3.46 = 14.4\ \mu$m.

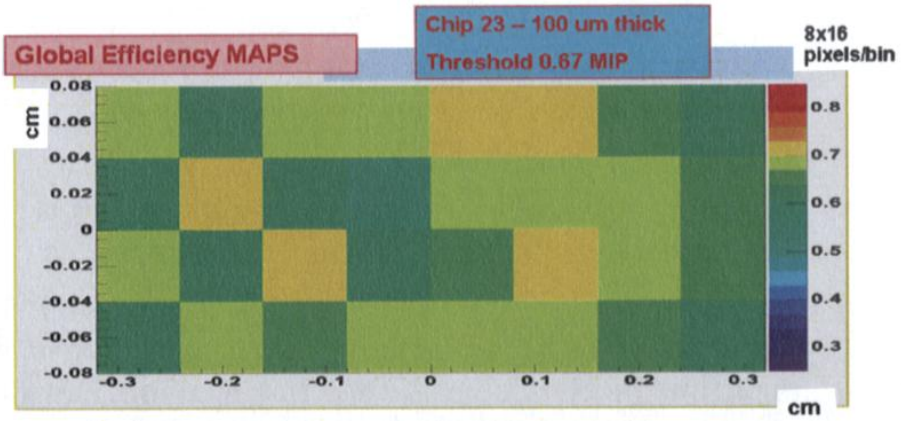


Figure 4.19: The APSEL4D hit efficiency across the matrix.

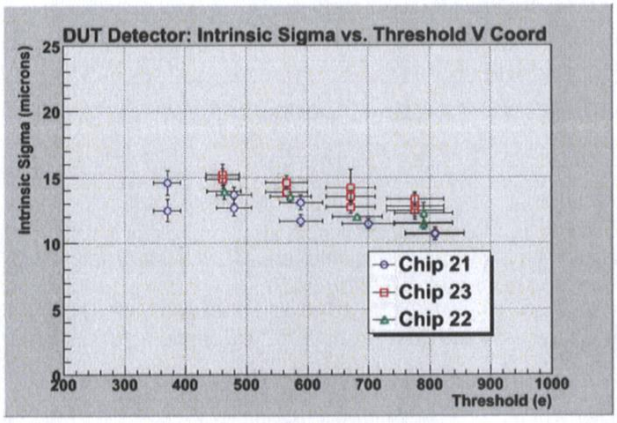


Figure 4.20: The APSEL4D intrinsic resolution as a function of the discriminator threshold

Chapter 5

The Data Acquisition System

The Data Acquisition System is devoted to the acquisition and online display of the data coming from APSEL4D and to the chip control and monitoring. The logical scheme and its actual hardware realization are shown in Figs (5.1, 5.2). It is composed of an electronic board, developed primarily for the conversion of the voltage levels between the APSEL4D chip and the FPGA mounted on the board, and for the signal quality enhancement. The FPGA is programmed with a VHDL firmware code that deals with the data flux coming from the chip and that controls and monitors the chip itself. Finally, a software framework, based on windows and widgets, was built in order to have an online data acquisition and monitoring system, featuring a real time hit display and a chip control panel. In the following we will describe in some details these components and their interconnections.

5.1 The Transition Board

The data signals coming from the chip, placed inside the microscope vacuum chamber, are carried outside the microscope by special connectors passing through a flange (see fig. 5.4) properly modified for this purpose, and arrive to a transition board (fig. 5.3), developed by the Electronic Group of the I.N.F.N. Bologna, with the task of enhancing the signal quality and converting voltage levels. This board makes it also possible to manage some manual settings on the chip working parameters and allows an interface with the chip through the board expansion connectors of an Opal Kelly XEM 3050, an FPGA integration module which we will now describe.

Figure 5.1: The Data Acquisition Logical Scheme and its implementation

5.2 The FPGA Board : XEM 3050

The XEM 3050 is a compact (the physical dimensions are 75mm x 50mm) general purpose board mounting a Xilinx Spartan-3 FPGA, and is shown in fig.(5.5). Designed as a full-featured integration system, the XEM 3050 provides access to over 110 I/O pins on a 676-pin Spartan-3 device, featuring 64-MByte of SDRAM, 1-MByte of SSRAM, and 1-MByte of Flash memory available to the FPGA, 8 Mb non-volatile flash, high-efficiency switching power supply, a Xilinx configuration PROM and two high-density 0.8-mm expansion connectors.

The functional block diagram is shown in Fig.(5.7) and the mechanical specifications are represented in Fig.(5.6). The XEM3050 uses a Cypress CY7C68013A FX2LP USB microcontroller to make the board a USB 2.0 peripheral. Regarding the clock capabilities, the featured Cypress CY22393 PLL is a multi-output, triple-PLL clock generator that can provide up to five clocks, three to the FPGA and another two to the expansion connectors JP2 and JP3. The PLL is driven by a 48-MHz signal output from the USB microcontroller and can output clocks up to 150-MHz. The clocks are configurable through the Front Panel software interface or the Front Panel API,

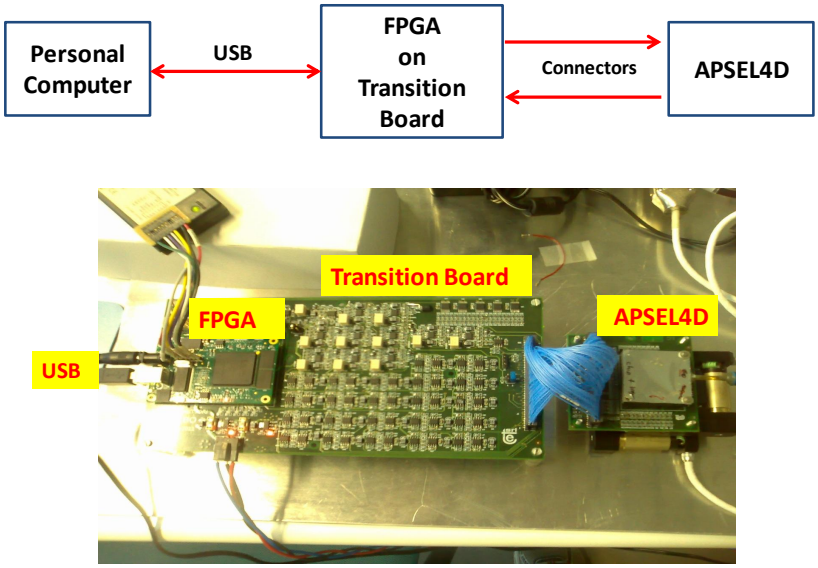


Figure 5.2: The Data Acquisition Logical Scheme and its implementation

which we will describe in the following.

There are two high-density, 80-pin expansion connectors, available on the bottom-side of the XEM3050 PCB. They provide the user the access to several power lines, two clock generator outputs, four FPGA clock inputs, the USB microcontroller I2C lines, the JTAG chain, and 116 non-shared I/O pins on the FPGA. In particular

- JP2 is an 80-pin high-density connector providing access to FPGA Banks 2 and 3. Pins 77 and 79 of this connector are wired to global clock inputs on the FPGA and can therefore be used as inputs to the global clock network. Pin 11 on this connector is SYSCLK4 and is directly connected to CLKD on the Cypress CY22393 PLL. For each JP2 pin, the corresponding board connection is listed in the documentation. For pins connected to the FPGA, the corresponding pin number is also shown. Finally, for pins routed to differential pair I/Os on the FPGA, the signal names and routed track lengths are provided, in order to equalize lengths on differential pairs.

- JP3 is an 80-pin high-density connector providing access to FPGA Banks 6 and 7. Pins 77 and 79 of this connector are wired to global

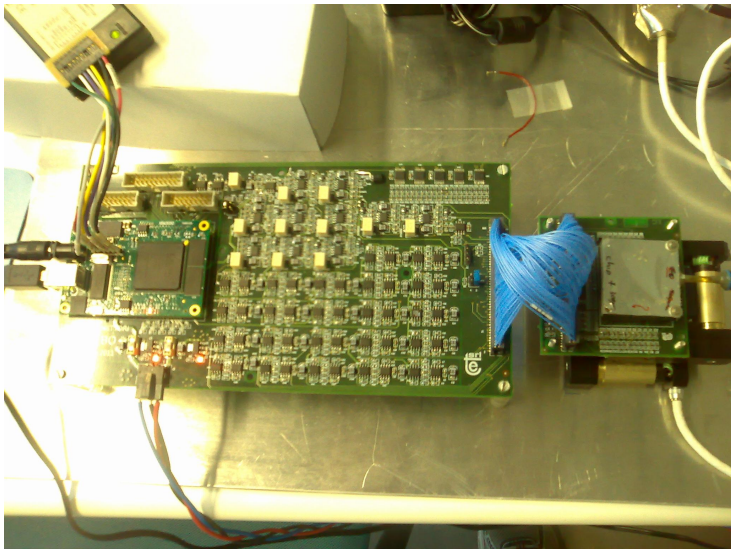


Figure 5.3: A picture of the Transition Board

clock inputs on the FPGA and can therefore be used as inputs to the global clock network. Pin 8 on this connector is SYSCLK5 and is directly connected to CLKE on the Cypress CY22393 PLL. Pin 7 is SYSCLK6 and is connected to XBUF on the PLL. For each JP3 pin, the corresponding board connection is listed in the documentation. For pins connected to the FPGA, the corresponding pin number is also shown. Finally, for pins routed to differential pair I/Os on the FPGA, the signal names and routed track lengths have been provided in order to equalize lengths on differential pairs.

The XEM3050 is fully supported by Opal Kelly Front Panel software. In addition to complete support within Front Panel, the XEM3050 is also fully supported by the Front Panel application programmer interface (API), a C++ class library available to Windows and Linux programmers allowing to interface the software to the XEM.

The host interface consists of 24 pins that connect the on-board USB microcontroller to the FPGA. These pins comprise the host interface on the FPGA and are used for configuration downloads. After configuration, these pins are used to allow Front Panel communication with the FPGA. If the Front Panel okHostInterface module is instantiated in the design, one must map the interface pins to specific pin locations using Xilinx LOC constraints. This may be done using the Xilinx constraints editor or specifying the constraints manually in a text file.

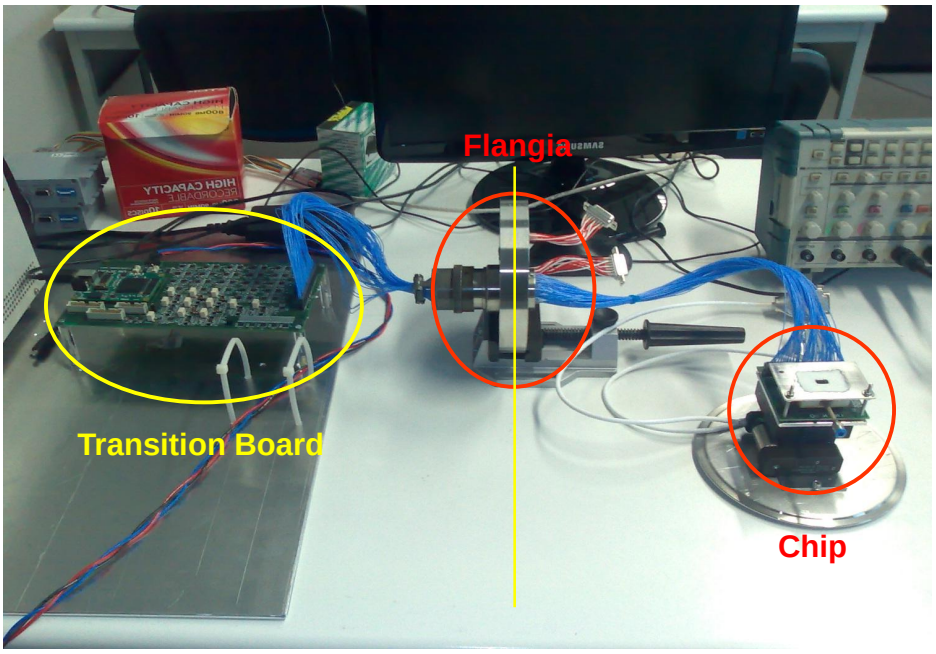


Figure 5.4: The path of the data through the flange

5.3 The Firmware

A custom hardware description code was developed for the communication between the FPGA and the PC through a USB 2.0 port, on one side, and for the communication between the FPGA and APSEL4D on the other. About 4000 lines of VHDL code were written for this purpose.

Constraint File

The first point to address in the firmware project is to match the signals and variables inside the FPGA with the outside signals through a constraint file. The FPGA must communicate with the USB port through the Host Interface Bus, as shown in Fig.(5.7): there are 24 pins that connect the on-board USB microcontroller to the FPGA. These pins comprise the host interface on the FPGA and are used for configuration downloads. After configuration, these pins are used to allow Front Panel communication with the FPGA. The following step is the mapping of the main clock signal from the PLL, and again one has to define the correspondence, for example
NET "clk1" LOC = "AE14";

Figure 5.5: The Opal Kelly XEM3050 model picture

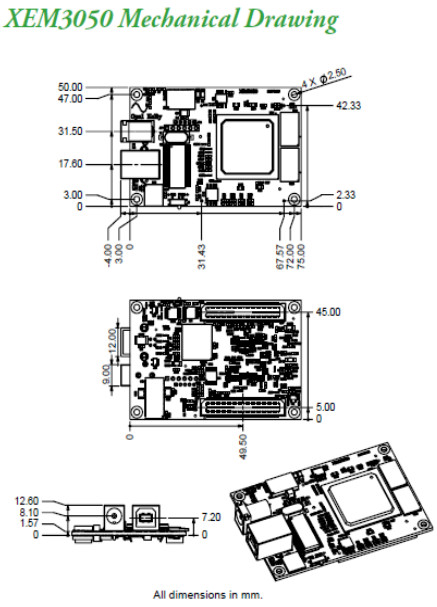


Figure 5.6: The Opal Kelly XEM3050 mechanical diagram

Host Interface Pin	FPGA Pin
HI-IN[0]	AF14
HI-IN[1]	AB5
HI-IN[2]	AC5
HI-IN[3]	AF12
HI-IN[4]	AE5
HI-IN[5]	AF5
HI-IN[6]	AF6
HI-IN[7]	AE6
HI-OUT[0]	AD14
HI-OUT[1]	AE12
HI-INOUT[0]	AA12
HI-INOUT[1]	AB12
HI-INOUT[2]	AB13
HI-INOUT[3]	AC13
HI-INOUT[4]	AA14
HI-INOUT[5]	Y14
HI-INOUT[6]	W14
HI-INOUT[7]	Y15
HI-INOUT[8]	AF11
HI-INOUT[9]	AE11
HI-INOUT[10]	AE10
HI-INOUT[11]	AE9
HI-INOUT[12]	AF8
HI-INOUT[13]	AE8
HI-INOUT[14]	AF7
HI-INOUT[15]	AE7
HI-MUXSEL	AF13

Table 5.1: The Host Interface bus constraint

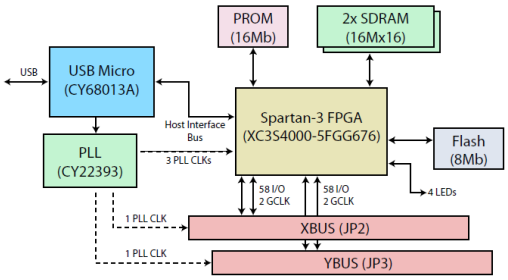


Figure 5.7: The XEM3050 functional scheme

For the clock signal one has to introduce some additional time specifications, for example we used
NET "clk1" TNM-NET = clk1;
TIMESPEC TS-clk1 = PERIOD "clk1" 20 ns
HIGH 50% INPUT-JITTER 500 ps;
One has then to deal with the input-output signals regarding the sensor, which is connected through the two high-density 0.8-mm expansion connectors JP2 and JP3 and mapped by using the schematic of Fig.(5.8) One has to note that the slow signals are single ended standard TTL signals, so that a simple map is sufficient in order to use them in the FPGA code, whereas data bits, which are fast signals, are LVDS differential signals. Beside mapping the signals one thus has to introduce a basic conversion entity IBUFDS, for example

```
Dato0 : IBUFDS generic map ( IOSTANDARD => "DEFAULT")
 port map (
 O    =>  ApselDatoIn (0) ,  -- Buffer output
 I    =>  DatoApselp  (0) ,  -- Diff-p  buffer input
 IB   =>  DatoApseln  (0)    -- Diff-n  buffer input    );
```

One must note that this can also be implemented at the level of the constraint file.

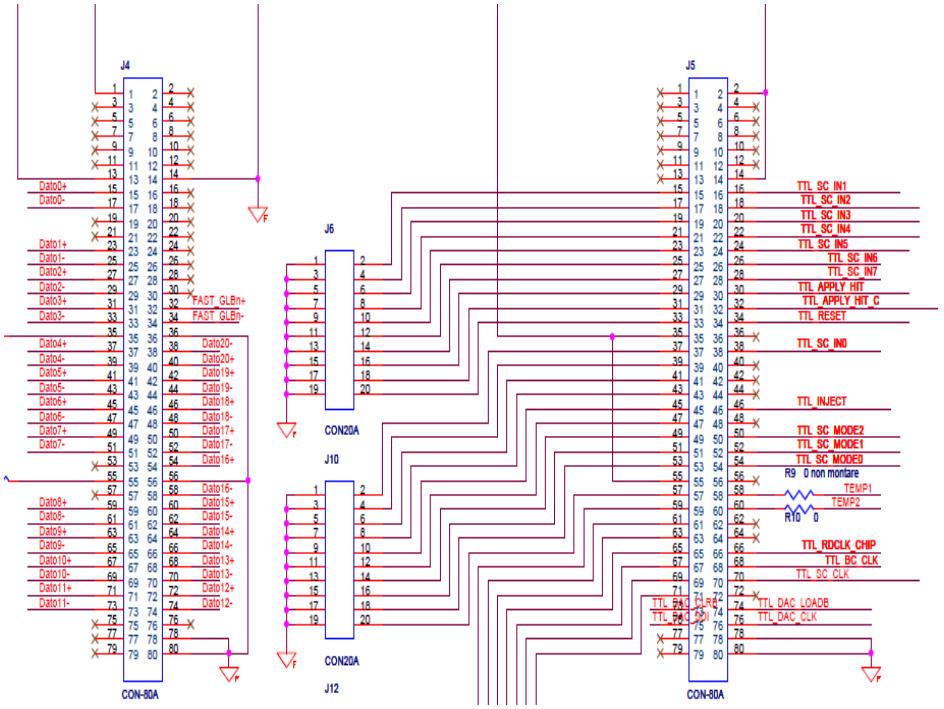


Figure 5.8: The expansion connector signal map

The Top Entity

In the behavioral description of the project we used about 70 input/output signals (standard logic vectors are counted as one signal) which we can group by their use. There are four basic sub-entities used for the data management and one further entity that takes care of the Sensor Chip control: they are

- Data FIFO
- Clock Manager
- Hit Formatter
- Apsel Interface
- Host Interface

Let us briefly describe their functions.

The Data FIFO is a First-In-First-Out buffer which takes the data from APSEL4D as an input and makes them available for the following logical units as a reading call is made. It also represents a bridge between the chip domain clock, which is the BC-clk, and the FPGA internal logic domain

Signal	Type
Clocks	
clk1	in std-logic
BC-clk	out std-logic;
SC-clk	out std-logic;
DAC-clk	out std-logic;
ApselClock	out std-logic;
Chip Interface	
SC-in	out std-logic-vector(7 downto 0);
SC-mode	out std-logic-vector(2 downto 0);
APSELreset	out std-logic;
DAC-nLOAD	out std-logic;
DAC-sdi	out std-logic;
DAC-nCLR	out std-logic ;
Host Interface	
hi-in	in std-logic-vector(7 downto 0);
hi-out	out std-logic-vector(1 downto 0);
hi-inout	inout std-logic-vector(15 downto 0) ;
hi-muxsel	out std-logic;
i2c-sda	out std-logic;
i2c-scl	out std-logic;
Data In and Out	
DatoApselp	in std-logic-vector (19 downto 0) ;
DatoApseln	in std-logic-vector (19 downto 0);
DatoApselValidp	in std-logic;
DatoApselValidn	in std-logic;
FAST-GLBnp	in std-logic;
FAST-GLBnn	in std-logic
Apply-hit	out std-logic;
Apply-hit-comp	out std-logic;

Table 5.2: The top entity project basic in and out ports

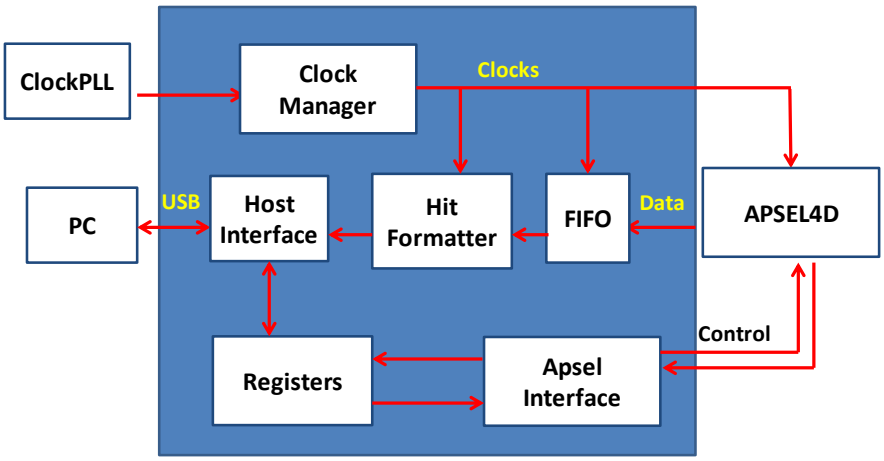


Figure 5.9: The main block of the VHDL code

clock, which is called clk1. The data coming out from the FIFO are formatted the same way as they are provided by APSEL4d, as a 20 bit vector, already described in the previous chapter, and a data-valid bit.

The Clock Manager is designed to provide a counter with a large period in order to associate univocally the hits with a global time stamp. As we have already seen, APSEL4D provides the hit coordinates with an 8 bit time stamp. This time stamp has a very short period, so one needs a longer period clock in order to univocally identify the timing identification of the hits. In the Clock Manager unit two 29 bit counters called TimeSinceStart and BCOCounter are created. They are basically two long period counters which, combined with the individual time stamp hit, give an absolute timing information. The BCO counter is a 29 bit counter which is incremented with the frequency of the BC-clk, and the TimeSinceStart vector is a counter of BCOCounter periods. They are provided as an input to the Hit Formatter unit.

The Hit Formatter unit takes as input the data from the Data FIFO as a 20 bit vector plus a data valid vector, and the clocks from the Clock Manager as a 29 bit vector. The hits are converted to a 32 bit format where the two most significant bits are set to 01, followed by a set of ten zero bits and by the

hit coming from APSEL4D as least significant bits. If no valid hit are coming from the sensor, the global timing information are sent as an output with an header set to 10 for the BCOCounter and to 11 for the TimeSinceStart counter. This way the data acquisition software can easily separate the hits from the timing information by looking at the 2 most significant bits of the 32 bit word, providing each hit with a global timing information.

The Apsel Interface is the unit that is dedicated to the control and monitoring of the chip. The connection with the data acquisition software if provided by a set of Read-Only and Read-Write registers which provide information from the chip and through which some chip settings can be made from the software user. We will describe the most important features in the software description. We here only mention the possibility to set the DAC threshold value, the BC clock period setting and the possibility to enable or disable the individual Macro Pixels by setting some Read-Write Registers.

Finally the Host Interface takes care of the communication between the FPGA and the PC through the USB port.

5.4 The Software

USB libraries for communication

The Front Panel application provides an easy method to make basic user interaction available to the FPGA hardware. But this is not suitable for all applications, particularly those which require further data processing on the PC side of the interface or when data transfer between the PC and FPGA is required. We were thus driven to develop a custom software application. To this end, Opal Kelly provides the Front Panel Application Programmer's Interface (API), a consistent interface to the underlying interface driver layer. The Front Panel API contains methods which communicate via the USB or PCIe bridge on the device, but the methods have been specifically designed to interface with FPGA hardware in a manner which is consistent with most hardware designs. The API provides methods to interface directly with the Front Panel HDL modules such as wires, triggers, and pipes. The library is written in C++ and is provided as a dynamically-linked library (DLL) that must be loaded into the application at runtime. The interface to the DLL is C, but a C++ wrapper is provided to make the entire DLL appear as if it were a native C++ class. The library contains a small number of classes which can be instantiated within the code. These classes are described below in further detail.

PLL Configuration

Each XEM product has a programmable Phase Locked Loop (PLL) that can be configured through the API. In many cases, the application will require a single pre-set PLL configuration which can be stored to the on-board EEPROM (not the PLL's EEPROM). With a preset PLL configuration, one can setup the PLL parameters using the Front Panel classes. One then stores these parameters to the on-board EEPROM for future recall. At the application startup (and typically before FPGA configuration), one can then configure the PLL from this stored preset using a single command. In other cases, one may want to configure the PLL from the software: this was our case. In this case one must use the PLL classes to configure PLL parameters and then load them into the PLL. This allows dynamic PLL configuration from the software. Software PLL configuration is a bit more complicated and requires more intimate knowledge of how the PLL parameters interoperate, needing to refer to the corresponding PLL datasheet. In particular we used the following lines of code to set the PLL for the master clock

```
\\ create a pointer to the PLL configuration device
okCPLL22393 *pll = new okCPLL22393 ;

\\ set the 48MHz reference clock
pll -> SetReference (48.0f)  ;

\\ define the clock base parameters
pll -> SetPLLParameters (0, 400, 48) ;

\\ define which of the three PLL to use
pll -> SetOutputSource (0, okCPLL22393::ClkSrc-PLL0-0) ;

\\ define the clock divider to be used to get the need output frequency
pll -> SetOutputDivide r(0, 40) ;

\\ enable the PLL 0
pll -> SetOutputEnable (0, true) ;

\\ load the PLL configuration to the interface device
device-> SetPLL22393Configuration(*pll) ;
```

API Communication

There are three endpoint types: Wires, Triggers and Pipes, providing a way for the PC and FPGA to communicate. Each type is suited to a specific type of data transfer and has its own associated usage and rules. All API methods are blocking commands. This means that the call completes before it returns to the caller. Therefore it is assured that if two consecutive API calls that update registers on the FPGA are made, the updates from the first call are completed prior to starting the second call. Let us now describe the endpoint types:

- Wires

  A Wire is used to communicate an asynchronous signal state between the host (PC) and the target (FPGA). The okHostInterface supports up to 32 Wire In endpoints and 32 Wire Out endpoints connected to it. To save bandwidth, all Wire In or Wire Out endpoints are updated at the same time and written or read by the host in one block. All Wire In (to FPGA) endpoints are updated by the host at the same time with the call UpdateWireIns(). Prior to this call, the application sets new Wire In values using the API method SetWireInValue(). The SetWireInValue() simply updates the wire values in a data structure internal to the API. UpdateWireIns() then transfers these values to the FPGA. All Wire Out (from FPGA) endpoints are likewise read by the host at the same time with a call to UpdateWireOuts(). This call reads all 32 Wire Out endpoints and stores their values in an internal data structure. The specific endpoint values can then be read out using GetWireOutValue(). UpdateWireIns() and UpdateWireOuts() also latch all wire endpoint data at the same time. Therefore, the data available on Wire Out endpoints are all captured synchronously (with the target interface clock). Similarly, the data available to Wire In endpoints are all latched synchronously with the target interface clock. Wires are 16-bits wide on USB 2.0 devices, 32-bits on USB 3.0 devices, and 32-bits on PCI Express devices. The API interface width is 32 bits. The upper bits are ignored if not supported.

- Triggers

  Triggers are used to communicate a singular event between the host and target. A Trigger In provides a way for the host to convey a "one-shot" on an arbitrary FPGA clock. A Trigger Out provides a way for the FPGA to signal the host with a "one-shot" or other single-event indicator. Triggers are read and updated in a manner similar to Wires.

All Trigger Ins are transferred to the FPGA at the same time and all Trigger Outs are transferred from the FPGA at the same time. Trigger Out information is read from the FPGA using the call UpdateTriggerOuts(). Subsequent calls to IsTriggered() then return 'true' if the trigger has been activated since the last call to UpdateTriggerOuts(). Triggers are 16-bits wide on USB 2.0 devices, 32-bits on USB 3.0 devices, and 32-bits on PCI Express devices. The API interface width is 32 bits. The upper bits are ignored if not supported.

- Pipes

  Pipe communication is the synchronous communication of one or more bytes of data. In both Pipe In and Pipe Out cases, the host is the master. Therefore, the FPGA must be able to accept (or provide) data on any time. Wires, Triggers, and FIFOs can make things a little more negotiable. When data is written by the host to a Pipe In endpoint using WriteToPipeIn(...), the device driver will transform the data into packets as necessary for the underlying protocol. Once the transfer has started, it will continue to completion, so the FPGA must be prepared to accept all of the data. When data is read by the host from a Pipe Out endpoint using ReadFromPipeOut(...), the device driver will again transform the data into packets as necessary. The transfer will proceed from start to completion, so the FPGA must be prepared to provide data to the Pipe Out as requested. Pipe data is transferred over the USB in 8-bit words but transferred to the FPGA in 16-bit words. Therefore, on the FPGA side (HDL), the Pipe interface has a 16-bit word width but on the PC side (API), the Pipe interface has an 8-bit word width. When writing to Pipe Ins, the first byte written is transferred over the lower order bits of the data bus (7:0). The second byte written is transferred over the higher order bits of the data bus (15:8). Similarly, when reading from Pipe Outs, the lower order bits are the first byte read and the higher order bits are the second byte read.

- Block-Throttled Pipes

  Block-Throttled Pipe communication is identical to Pipe communication with the additional specification of a block size. The FPGA sends (or receives) data in blocks sized 2, 4, 6, .., 1024 as specified by the arguments to the call. Block sizes are restricted to 64 bytes or less when using the device at full-speed. Because the FPGA has the opportunity to stall the transfer by deasserting EPREADY, the call may fail with a timeout.

HDL Modules

The use of Front Panel components to control and observe pieces of the FPGA design requires the instantiation of one or more modules in the toplevel HDL. These modules can quickly and easily be added into an existing or new design and take care of communicating with the Front Panel software. The host interface is the block which connects directly to pins on the FPGA which are connected to the USB microcontroller. This is the entry point for Front Panel into the design. The endpoints connect to a shared control bus on the host interface. This internal bus is used to shuttle the endpoint connections to and from the host interface. Several endpoints may be connected to this shared bus. Front Panel uses endpoint addresses to select which endpoint it is communicating with, so each endpoint must have its own unique address to work properly. As already said, Front Panel supports three basic types of endpoints: Wire, Trigger, and Pipe. Each can either be an input (from host to target) or output (from target to host). Each endpoint type has a certain address range which must be used for proper operation. The address is specified at the instantiation of the endpoint module in the design. To properly route signals between the host (PC) and target endpoints, each endpoint must be assigned a unique 8-bit address. For performance reasons (to minimize USB transactions), each endpoint type has been assigned an address range as indicated in the manual. The endpoint address is assigned in HDL through an additional 8-bit input port on the endpoint instance.

The HDL host interface is a slave interface from the host. It runs at a fixed clock rate that is dependent upon the interface type for the device. USB 2.0 interfaces run at 48 MHz (20.83 ns clock period), USB 3.0 interfaces run at 100.8 MHz (9,92 ns clock period) and PCI Express interfaces run at 50 MHz (20 ns clock period).

The Front Panel HDL Modules are provided as pre-synthesized files which get included in the design flow. Multiple endpoints are attached to the ok2 bus on the okHost by using a Wire-OR. Each endpoint is told when it can assert its data on the bus. At all other times, it drives 0. The Wire-OR component performs a bitwise OR operation on each bit of the bus and outputs the result. In this manner, multiple endpoints can share a bus without requiring the use of tri-states or a large multiplexer. The okWireOR is provided as a parameterized helper module in okLibrary.v and okLibrary.vhd.

The Host Interface

The host interface is the gateway for Front Panel to control and observe the design. It contains relatively simple logic that lets the USB microcontroller

on the device communicate with the various endpoints within the design. Exactly one host interface must be instantiated in any design which uses the Front Panel components. The Host Interface component is the only block which is synthesized with the design. It contains a Host Interface core component (provided as a pre-synthesized module) as well as the necessary IOB components to connect to the host interface pins of the FPGA.

- okHost

  This module must be instantiated in any design that makes use of Front Panel virtual interface components. The following signals need to be connected directly to pins on the FPGA which go to the USB microcontroller on the device:

  - HI-IN[7:0] : Host interface input signals.
  - HI-OUT[1:0] : Host interface output signals.
  - HI-INOUT[15:0] : Host interface bidirectional signals.

  The remaining ports of the okHost are connected to a shared bus inside the design. These signals are collectively referred to as the target interface bus. Each endpoint must connect to these signals for proper operation:

  - OK1[30:0] : Control signals to the target endpoints.
  - OK2[16:0] : Control signals from the target endpoints.
  - TI-CLK : Buffered copy of the host interface clock (48 MHz). This signal does not need to be connected to the target endpoints because it is replicated within OK1.

  Each endpoint is connected to 48 target interface pins on the okHost module. The direction is from the perspective of the endpoint module. These signals are present in every endpoint.

- okWireIn

  In addition to the target interface pins, the okWireIn adds a single 16-bit output bus called EPDATAOUT[15:0]. The pins of this bus are connected to the design as wires and act as asynchronous connections from Front Panel components to the HDL. When Front Panel updates the Wire Ins, it writes new values to the wires, then updates them all at the same time. Therefore, although the wires are asynchronous endpoints, they are all updated at the same time on the host interface clock.

- okWireOut

  An okWireOut module adds a single 16-bit input bus called EPDATAIN[15:0]. Signals on these pins are read whenever Front Panel updates the state of its wire values. In fact, all wires are captured simultaneously (synchronous to the host interface clock) and read out sequentially

- okTriggerIn

  The okTriggerIn provides EPCLK and EPTRIGGER[15:0] as interface signals. The Trigger In endpoint produces a single-cycle trigger pulse on any of EPTRIGGER[15:0] which is synchronized to the clock signal EPCLK. Therefore, the single-cycle does not necessarily have to be a single host interface cycle. Rather, the module takes care of crossing the clock boundary properly.

  - EPADDR[7:0] : Endpoint address.
  - EPCLK : Clock to which the trigger should synchronize.
  - EPTRIGGER[15:0] : Independent triggers from host.

- okTriggerOut

  The target may trigger the host using this module. EPTRIGGER[15:0] contains 16 independent trigger signals which are monitored with respect to EPCLK. If EPTRIGGER[x] is asserted for the rising edge of EPCLK, then that trigger will be set. The next time the host checks trigger values, the triggers will be cleared.

  - EPADDR[7:0] : Endpoint address.
  - EPCLK : Clock to which the trigger should synchronize.
  - EPTRIGGER[15:0] : Independent triggers from host.

- okBTPipeIn

  The Block-Throttled Pipe In module is similar to the okPipeIn module, but adds two signals, EPBLOCKSTROBE and EPREADY to handle block-level negotiation for data transfer. The host is still master, but the FPGA controls EPREADY. When EPREADY is asserted, the host is free to transmit a full block of data. When EPREADY is deasserted, the host will not transmit to the module. EPREADY could, for example, be tied to a level indicator on a FIFO. When the FIFO has a full block of space available, it will assert EPREADY signifying that it can accept a full block transfer

  - EPADDR[7:0] : Endpoint address.
  - EPDATAOUT[15:0] : Pipe data output.
  - EPWRITE : Active-high write signal. Data should be captured when this signal is asserted.
  - EPBLOCKSTROBE : Active-high block strobe. This is asserted for one cycle just before a block of data is written.
  - EPREADY : Active-high ready signal. Logic should assert this signal when it is prepared to receive a full block of data.

- okBTPipeOut

  The Block-Throttled Pipe Out module is similar to the okPipeOut module, but adds two signals, EPBLOCKSTROBE and EPREADY to handle block-level negotiation for data transfer. The host is still master, but the FPGA controls EPREADY. When EPREADY is asserted, the host is free to read a full block of data. When EPREADY is deasserted, the host will not read from the module. EPREADY could, for example, be tied to a level indicator on a FIFO. When the FIFO has a full block of data available, it will assert EPREADY signifying that a full block may be read from the FIFO.

  - EPADDR[7:0] : Endpoint address.
  - EPDATAIN[15:0] : Pipe data input.
  - EPREAD : Active-high read signal. Data must be provided in the cycle following as assertion of this signal.
  - EPBLOCKSTROBE : Active-high block strobe. This is asserted for one cycle just before a block of data is read.
  - EPREADY : Active-high ready signal. Logic should assert this signal when it is prepared to transmit a full block of data.

Qt libraries

Qt is a cross-platform application framework widely used for developing application software with a graphical user interface (GUI) and used for developing non-GUI programs such as command-line tools and consoles for servers. Qt uses standard C++ but makes extensive use of a special code generator (called the Meta Object Compiler, or MOC) together with several macros to enrich the language. Qt is a multi-platform C++ (with interfaces to other languages) GUI library that is built from modules with a common scheme

and built from the same API design ideas. The standard C++ object model provides very efficient runtime support for the object paradigm. But its static nature is not flexible in certain problem domains. Graphical user interface programming is a domain that requires both runtime efficiency and a high level of flexibility. Qt provides this, by combining the speed of C++ with the flexibility of the Qt Object Model. Qt adds features to C++ such as: a very powerful mechanism for seamless object communication called signals and slots, queryable and designable object properties, powerful events and event filters, sophisticated interval driven timers that make it possible to elegantly integrate many tasks in an event-driven GUI, hierarchical and queryable object trees that organize object ownership in a natural way, guarded pointers (QPointer) that are automatically set to 0 when the referenced object is destroyed, unlike normal C++ pointers which become dangling pointers when their objects are destroyed. Many of these Qt features are implemented with standard C++ techniques, based on inheritance from QObject. Others, like the object communication mechanism and the dynamic property system, require the Meta-Object System provided by Qt's own Meta-Object Compiler (MOC). The meta-object system is a C++ extension that makes the language better suited to true component GUI programming.

QObjects

The QObject class is the base class of all Qt objects and is the heart of the Qt Object Model. The central feature in this model is a very powerful mechanism for seamless object communication called signals and slots. One can connect a signal to a slot with connect() and destroy the connection with disconnect(). To avoid never ending notification loops one can temporarily block signals with blockSignals(). The protected functions connectNotify() and disconnectNotify() make it possible to track connections. QObjects organize themselves in object trees. When one creates a QObject with another object as parent, the object will automatically add itself to the parent's children() list. The parent takes ownership of the object; i.e., it will automatically delete its children in its destructor. One can look for an object by name and optionally type using findChild() or findChildren(). Every object has an objectName() and its class name can be found via the corresponding metaObject() (see QMetaObject::className()). One can determine whether the object's class inherits another class in the QObject inheritance hierarchy by using the inherits() function. When an object is deleted, it emits a destroyed() signal. One can catch this signal to avoid dangling references to QObjects. QObjects can receive events through event() and filter the events of other objects. Events are delivered in the thread in which the object was

created. Last but not least, QObject provides the basic timer support in Qt. The Q-OBJECT macro is mandatory for any object that implements signals, slots or properties.

Signals and Slots

Signals and slots are used for communication between objects. The signals and slots mechanism is a central feature of Qt and probably the part that differs most from the features provided by other frameworks. In GUI programming, when one changes a widget, one often wants another widget to be notified. More generally, one wants objects of any kind to be able to communicate with one another. For example, if a user clicks a Close button, the window's close() function should be called. Older toolkits achieve this kind of communication using callbacks. A callback is a pointer to a function, so if one wants a processing function to notify about some event one passes a pointer to another function (the callback) to the processing function. The processing function then calls the callback when appropriate. Callbacks have two fundamental flaws: firstly, they are not type-safe. We can never be certain that the processing function will call the callback with the correct arguments. Secondly, the callback is strongly coupled to the processing function since the processing function must know which callback to call. In Qt, we have an alternative to the callback technique: we use signals and slots. A signal is emitted when a particular event occurs. Qt's widgets have many predefined signals, but we can always subclass widgets to add our own signals to them. A slot is a function that is called in response to a particular signal. The signals and slots mechanism is type safe: the signature of a signal must match the signature of the receiving slot (in fact a slot may have a shorter signature than the signal it receives because it can ignore extra arguments.) Since the signatures are compatible, the compiler can help us detect type mismatches. Signals and slots are loosely coupled: A class which emits a signal neither knows nor cares which slots receive the signal. Qt's signals and slots mechanism ensures that if one connects a signal to a slot, the slot will be called with the signal's parameters at the right time. Signals and slots can take any number of arguments of any type. They are completely type safe.

Signals are emitted by an object when its internal state has changed in some way that might be interesting to the object's client or owner. Signals are public access functions and can be emitted from anywhere, but it is recommended that they be emitted from the class that defines the signal and its subclasses. When a signal is emitted, the slots connected to it are usually executed immediately, just like a normal function call. When this

happens, the signals and slots mechanism is totally independent of any GUI event loop. Execution of the code following the emit statement will occur once all slots have returned. The situation is slightly different when using queued connections; in such a case, the code following the emit keyword will continue immediately, and the slots will be executed later. If several slots are connected to a single signal, the slots will be executed one after the other, in the order they have been connected, when the signal is emitted. Signals are automatically generated by the MOC and must not be implemented in the .cpp file. They can never have return types (i.e. use void).

A slot is called when a signal connected to it is emitted. Slots are normal C++ functions and can be called normally; their only special feature is that signals can be connected to them. Since slots are normal member functions, they follow the normal C++ rules when called directly. However, as slots, they can be invoked by any component, regardless of its access level, via a signal-slot connection. This means that a signal emitted from an instance of an arbitrary class can cause a private slot to be invoked in an instance of an unrelated class. One can also define slots to be virtual, which we have found quite useful in practice.

Compared to callbacks, signals and slots are slightly slower because of the increased flexibility they provide, although the difference for real applications is insignificant. In general, emitting a signal that is connected to some slots, is approximately ten times slower than calling the receivers directly, with non-virtual function calls. This is the overhead required to locate the connection object, to safely iterate over all connections (i.e. checking that subsequent receivers have not been destroyed during the emission), and to manage any parameters in a generic fashion. While ten non-virtual function calls may sound like a lot, it's much less overhead than any new or delete operation, for example. As soon as one performs a string, vector or list operation that behind the scene requires new or delete, the signals and slots overhead is only responsible for a very small proportion of the complete function call costs. The same is true whenever one makes a system call in a slot; or indirectly call more than ten functions.

All classes that inherit from QObject or one of its subclasses (e.g., QWidget) can contain signals and slots. Signals are emitted by objects when they change their state in a way that may be interesting to other objects. This is all the object does to communicate. It does not know or care whether anything is receiving the signals it emits. This is true information encapsulation, and ensures that the object can be used as a software component.

Slots can be used for receiving signals, but they are also normal member functions. Just as an object does not know if anything receives its signals, a slot does not know if it has any signals connected to it. This ensures that

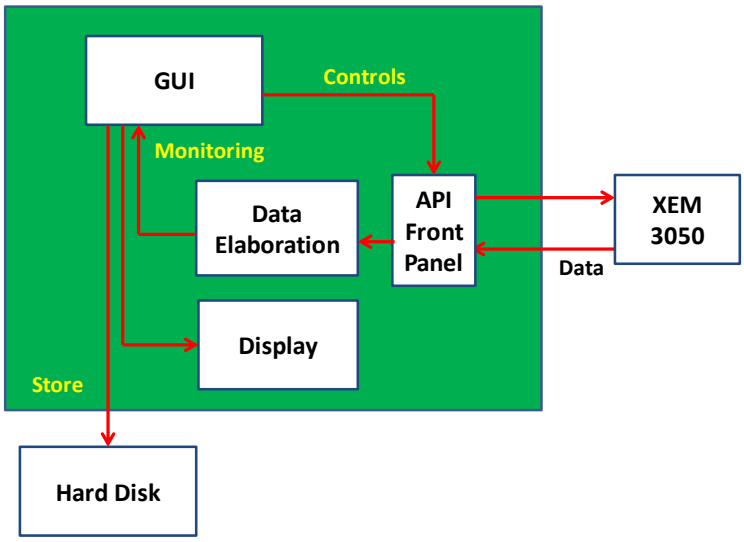


Figure 5.10: The Graphical User Interface framework logical blocks

truly independent components can be created with Qt. One can connect as many signals as one wants to a single slot, and a signal can be connected to as many slots as one needs. It is even possible to connect a signal directly to another signal. (This will emit the second signal immediately whenever the first is emitted.) Together, signals and slots make up a powerful component programming mechanism.

5.5 The Graphical User Interface

A custom Graphical User Interface (GUI) was built in order to have a framework for real time chip control and graphical display of the hits. In Fig. (5.10) one can see the logical scheme of the GUI framework. The code was developed in C++ programming language with the use of the Qt Libraries, as the basic elements of the interface are widgets, and of the Qwt libraries, for the online graphical display of the hits on the pixel sensor matrix. With the inclusion of the Qt libraries, the main file of the C++ application looks simply like

```
QApplication a(argc, argv);
```

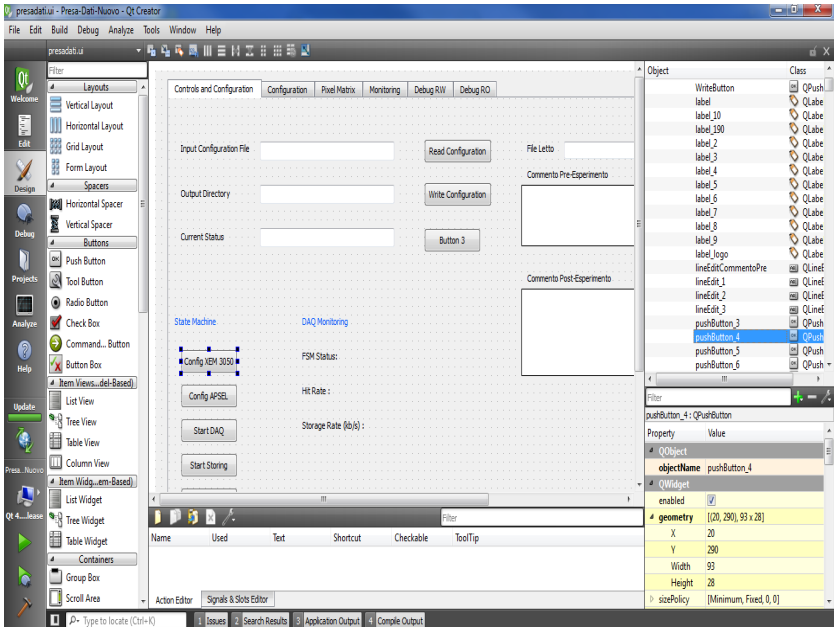


Figure 5.11: The QtEditor window

```
PresaDati w;
w.show();
```

where in the definition of the class "Presadati" one can implement all the needed features, from the graphical interface appearance to the connection between signals and slots through buttons or any other kind of graphical objects. In fact the Qt ISE, shown in Fig.(5.11), allows the programmer to insert inside the widget any kind of objects like buttons, text lines, or any other tool, in order to build the interface. Once these QtObjects are inserted in the GUI, their behavior and use can be completely defined by the programmer. Let us make an example with very simple objects like check-boxes. One can define a matrix of pointers
QCheckBox* MPmatrix[8][32];
corresponding to the MacroPixels of APSEL4D. Then one can define an unsigned integer corresponding to a register that can be passed to the chip in order to enable-disable some MacroPixels

```
unsigned int MatrixMaskLocal-21 =
MPmatrix [0][10] ->isChecked()            |
MPmatrix [1][10] ->isChecked() << 1    |
MPmatrix [2][10] ->isChecked() << 2    |
MPmatrix [3][10] ->isChecked() << 3    |
```

```
 MPmatrix [4][10] ->isChecked() << 4    |
 MPmatrix [5][10] ->isChecked() << 5    |
 MPmatrix [6][10] ->isChecked() << 6    |
 MPmatrix [7][10] ->isChecked() << 7    |
 MPmatrix [0][11] ->isChecked() << 8    |
 MPmatrix [1][11] ->isChecked() << 9    |
 MPmatrix [2][11] ->isChecked() << 10   |
 MPmatrix [3][11] ->isChecked() << 11   |
 MPmatrix [4][11] ->isChecked() << 12   |
 MPmatrix [5][11] ->isChecked() << 13   |
 MPmatrix [6][11] ->isChecked() << 14   |
 MPmatrix [7][11] ->isChecked() << 15   ;
```

Now that one has stored the information on the second row of Macro Pixels (that can be enabled or disabled), one is able to set the corresponding registers in the FPGA to the proper value

```
device -> SetWireInValue(0x07,MatrixMaskLocal-21);
device -> UpdateWireIns();
```

and once the registers are set one can activate a trigger to actually set the register in the chip.

```
device -> ActivateTriggerIn(0x40, 1);
```

If this whole procedure is put in the slot corresponding to the pressing of a specific button

```
void PresaDati::on-ButtonSendCommand-released()
```

we have implemented, in a few lines of code, the setting of some register in the sensor chip.

The building step that allow this are:

```
// include the Dynamic Library for the communication
// to the FPGA through USB port
#include "okFrontPanelDLL.h"

// define a pointer to the device in order to have access
// to all the classes for the communication
newokCUsbFrontPanel *device

//  associate the device pointer to the FPGA
```

```
 device = new newokCUsbFrontPanel()

 // the communication can be checked by getting the device ID
Idstring = device ->GetDeviceID()

 // one can load the VHDL project bitfile trough USB port
device ->ConfigureFPGA (bitfilename.bit")

 //one can configure the Main Clock Frequency
device->SetPLL22393Configuration(*pll)

 // then one can read the FPGA registers
device ->UpdateWireOuts()
int wireout20 = device -> GetWireOutValue(0x21)

 // or write the FPGA registers
device ->SetWireInValue(0x00,RWWrite0hex)
device ->UpdateWireIns()

 // activate triggers
device ->ActivateTriggerIn(0x40, 0)

 // read the data from the chip
pipe = device -> ReadFromBlockPipeOut(0xA0,block-size,length, data)
```

By using these basic commands the Graphical User Interface can communicate with the chip where the sensor is placed.

Register and Trigger Read and Write

In order to monitor the chip and to send commands and data for its setting, one needs an input-output protocol. In the VHDL project the corresponding read-only and read-write registers were created so that the GUI must give the possibility to use them. Two windows were created for this purpose, the one for the monitoring (read-only registers) is displayed in Fig.(5.12) and that for the register setting in Fig.(5.13). The use of the read-only window is very easy, as one has simply to push a button in order for the corresponding register to be displayed. The pushing of a button corresponds to the lines of code of the corresponding slot

```
 // updating the read-only registers
 device ->UpdateWireOuts()
```

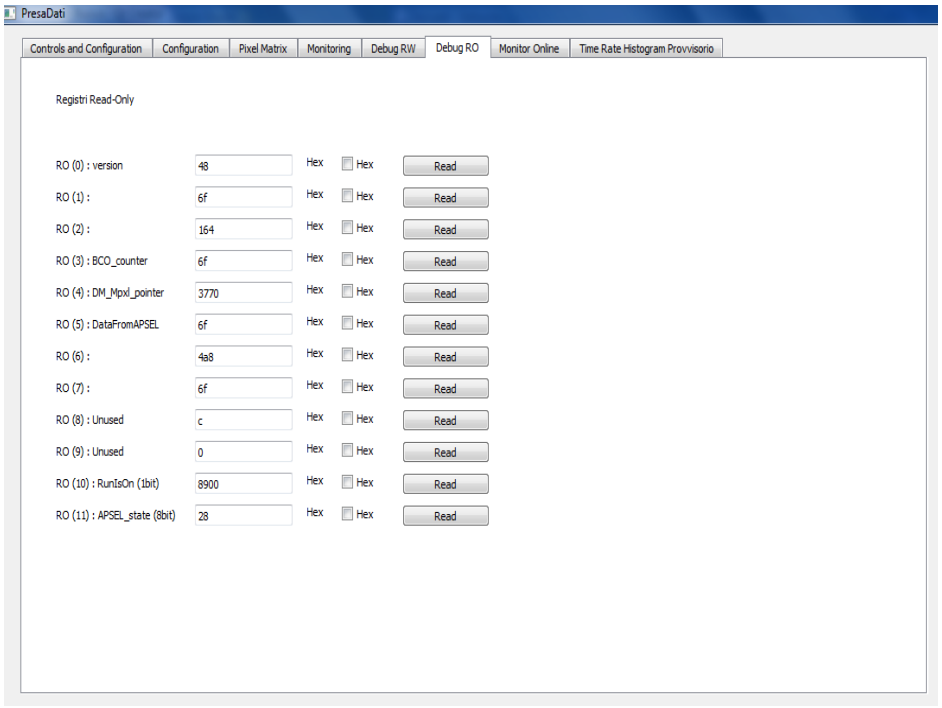


Figure 5.12: The Graphical User Interface for read-only registers

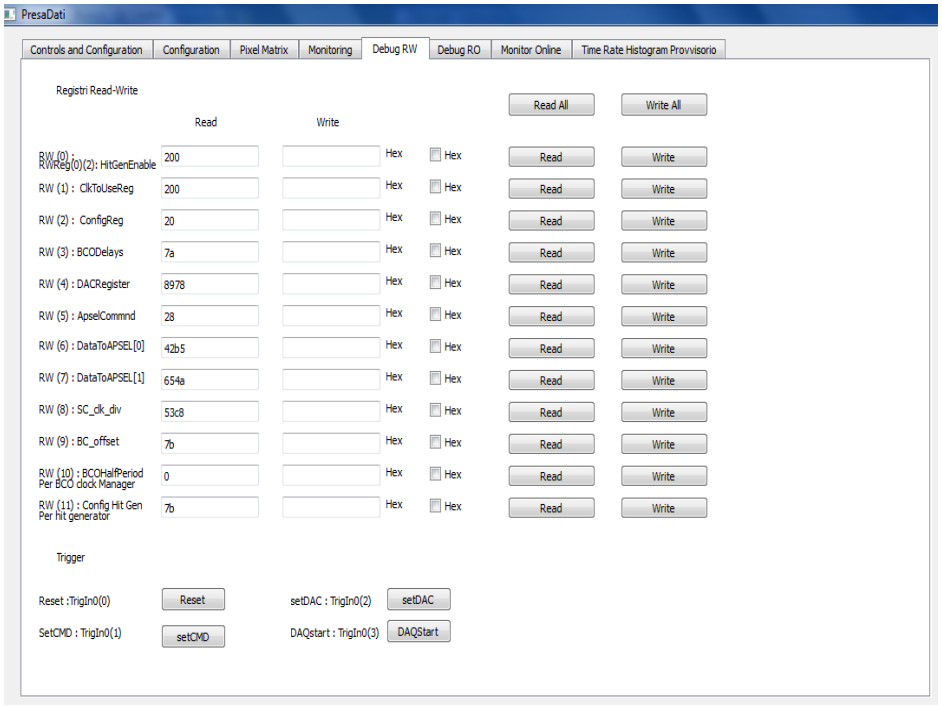


Figure 5.13: The Graphical User Interface for read-write registers

```
// read the register, in this case the 0x2C
int RORead0 = device ->GetWireOutValue(0x2C)

// set the corresponding variable
QString qRORead0 = QString::number(RORead0,16)

// display the value
ui -> lineEditRORead0 ->setText(qRORead0)
```

In particular during the data acquisition one can verify the state of the chip, for example if the flag Run-is-on, which means that the chip is properly set for data taking, is raised. One can also see if the BCO clock is running. Other quantities of interest are displayed in the read-write window. The read-write registers are used in order to give commands to the chip and to set values. In order to set some values, for example the Digital to Analog Converter (DAC) threshold, one simply has to set the corresponding read-write register through the corresponding code lines

```
//  read the values set by the user
QString qRWWrite4 = ui -> lineEditRWWrite4 ->text()

// set the corresponding register value
device ->SetWireInValue(0x04,RWWrite4hex)

// update the registers
device ->UpdateWireIns()
```

Now that the register is set in the FPGA on has to send it to the chip and this is done by raising a trigger, so that the slot corresponding to the "set DAC" button is composed of one simple line

```
device ->ActivateTriggerIn(0x40, 2);
```

The Control Interface

The control interface is shown in Fig.(5.14) and basically allows the user to interact with the chip. There are 14 commands that can be issued and they are encoded in the firmware as shown in Table (5.3) The expert user can give any of these commands by typing the corresponding code in the appropriate line and then pushing the button "Apsel Command Send", which activate the corresponding slot

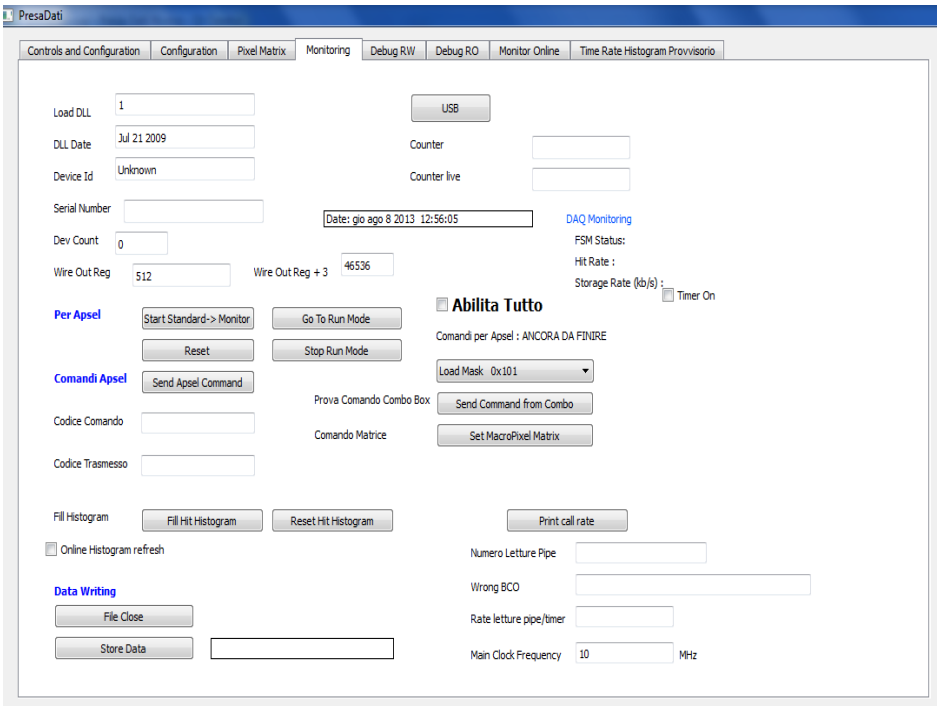


Figure 5.14: The Graphical User Interface for system monitoring

Command	Code 0x
load-mask	101
hard-reset	102
soft-reset	103
push-dummy	104
BCO-read	105
set-dummyReg	106
reset-dummyReg	107
DM-bit-rolling	108
App-hit	109
GotoRunMode	10A
stop-run	10B
gotoCalibMode	10C
gotoCalibMode2	10D
shift-Px-Mask	10E

Table 5.3: The command codification

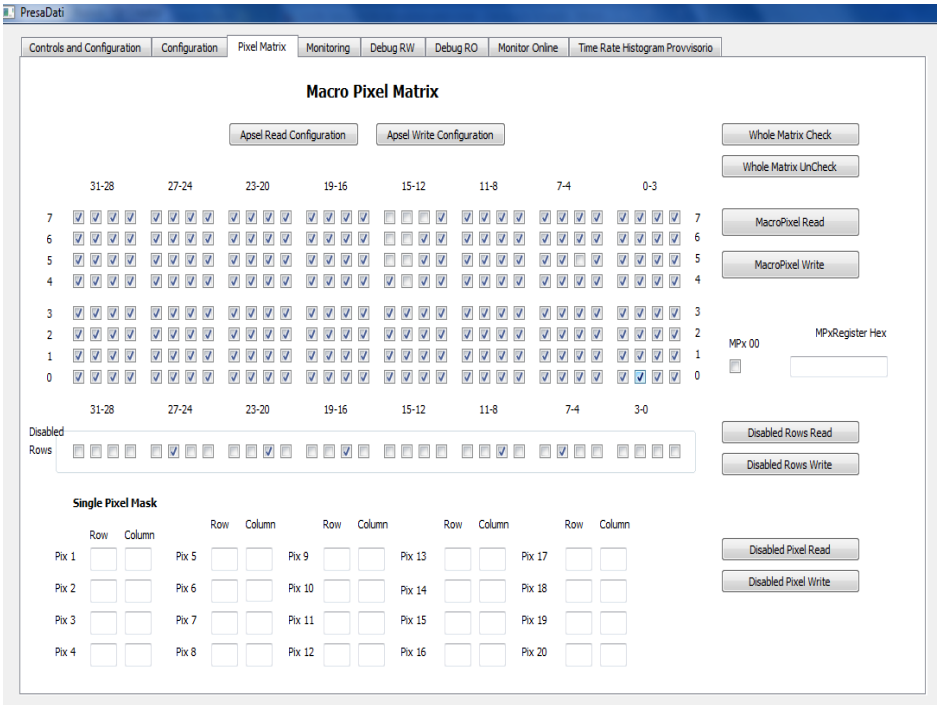


Figure 5.15: The Graphical User Interface for enabling and disabling Macro Pixels, pixel rows or single pixels

```
// read the command code
QString qCommandCode = ui -> ApselCommandCode ->text()

// set the proper register
int wirein = device ->SetWireInValue( 0x05 , CommandCode)

// raise the "command sent" trigger
device ->ActivateTriggerIn(0x40, 1)
```

The same result was made available to the final user by introducing a command from the list in the button "Send Command from Combo" and then pushing the button "Send Command from Combo". More involved procedures need the appropriate slots. This is the case of enabling and disabling Macro Pixels or lines of pixels: a widget tab was developed just for this purpose and is shown in Fig.(5.15). The full Macro Pixel Matrix is displayed through check-boxes so that the user can enable or disable any of the MPs by checking or unchecking the corresponding check-box. As the button "Set MacroPixel Matrix" is pushed, the corresponding slot takes care of writing in the proper order the corresponding registers and to give them as an input to

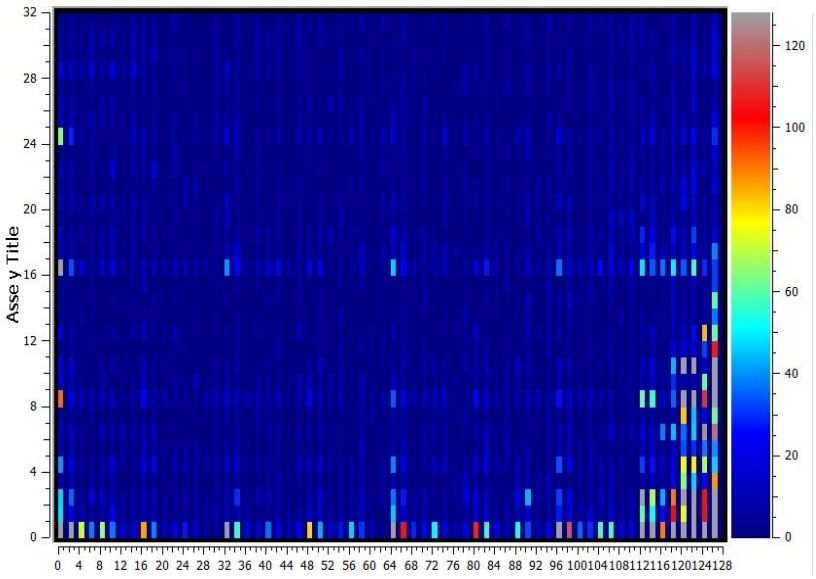


Figure 5.16: The Graphical User Interface for displaying the hits

Apsel in order to have some regions of pixels enabled or disabled at receiving the hits. By using buttons in this window the user is able to enable or disable the online hit display and the hit recording to a file.

The real time hit display

As Qt libraries are not well suited for real time histogram plotting, we used the qwt extension in order to feature an online plot of the hits. As one can see in Fig.(5.16) the whole 32x128 matrix is displayed in a window and every millisecond the plot is refreshed by using data of the hits as they come from the sensor using a color scale. A typical plot is shown in Fig.(5.17) This feature enables to set all the microscope parameters in order to have the best setting for the data taking.

The configuration and final user windows

Every time the system is configured, there is the possibility to save all the necessary parameters characterizing the session, so that one can recover them in a following one. Also in this case all this is automated and consists of a

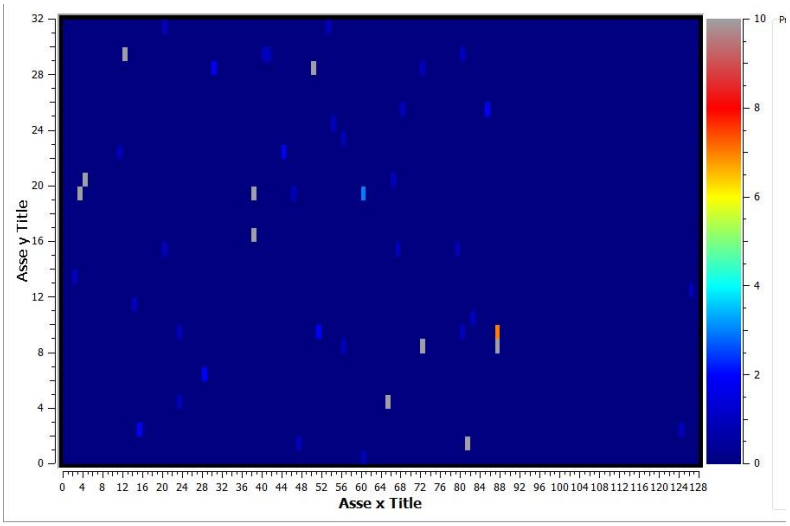


Figure 5.17: The Graphical User Interface for displaying the hits: in this data taking, the right part of the sensor was mechanically obstructed, in order to verify the consistence if the displayed hits

single button pushing. All the parameters that are needed for the system configuration are stored in a xml file. The configuration window is shown in Fig.(5.18). Going in the direction of having a final user which is not aware of all the system details, an easy interface window was built. A button programs the FPGA with the bitfile, another one configures the chip with the standard settings, a third button starts the data acquisition procedure and the fourth starts the data recording. The file containing the data features a header with many information. The file starts with the information on the date and time of the data taking, then all the chip configuration parameters are recorded, such as the BCO Clock Period, the DAC Threshold and all the chip configuration data, in order that in the off-line analysis one can reconstruct the whole chip configuration. After the header, the file consists of a list of 32-bit words containing the information on the hits and the timing, as discussed previously. The data analysis is made through the use of the Root package with some Macros developed for analyzing many files at the same time. At the same time 2 text files are stored, one with the information on the Transmission Electron Microscope working parameters, so that also the microscope configuration can be subsequently reproduced, and the other with comments on the data taking, prior and subsequent to the data taking,

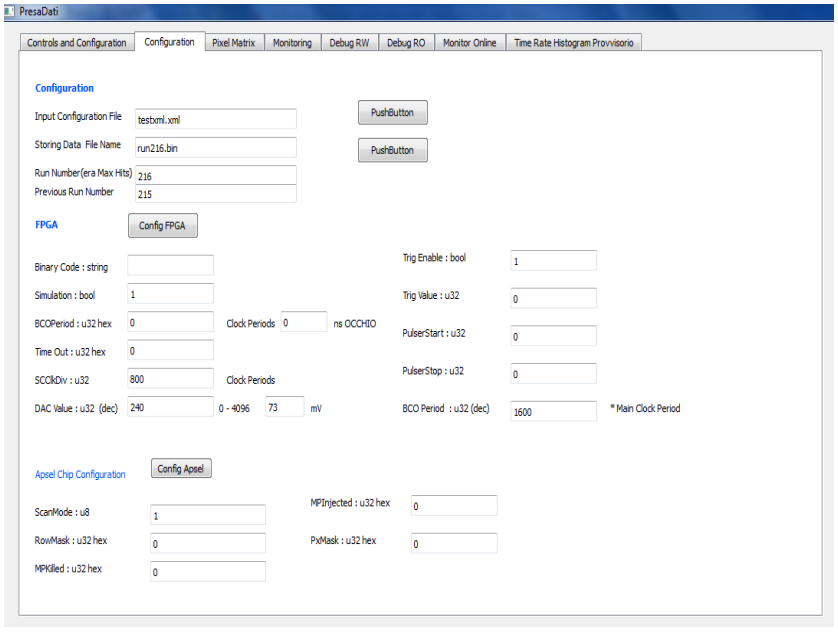


Figure 5.18: The Graphical User Interface: the chip configuration window

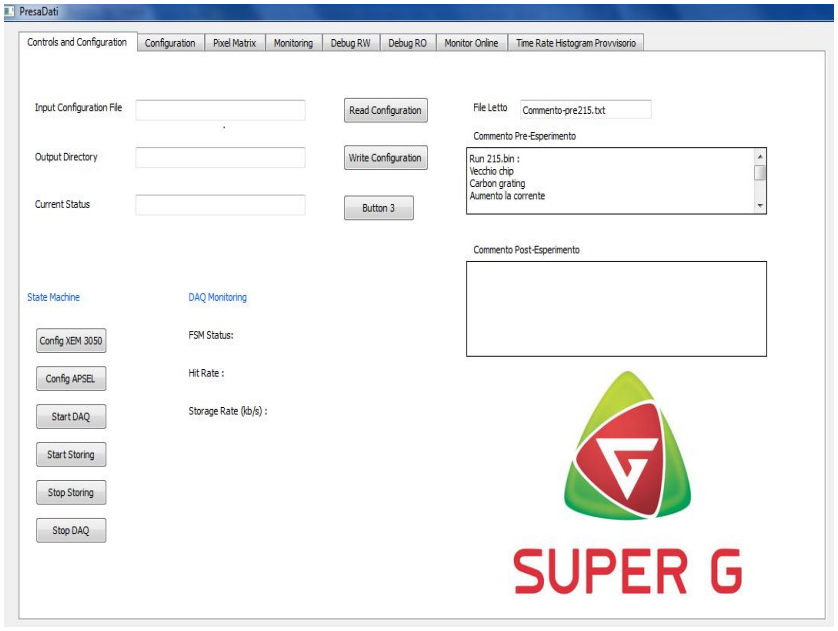


Figure 5.19: The Graphical User Interface: the control window

in order to be able to understand even after a long period of time, the targets and positive or negative results of the data taking procedure. Finally, two different buttons are used to stop the recording and the data taking. This interface is shown in Fig.(5.19).

Chapter 6

Single Electron Interference

One of the most effective experiments one can set up in order to investigate the wave behavior of material particles is the single electron two-slit interference. This arrangement, adopted by Bohr and Einstein to develop their considerations on quantum mechanics foundations, is widely quoted as Young-Feynman set-up. In fact Feynman, in his famous lectures, referred to this experiment as practically not realizable, given the very small de Broglie wavelength associated to the electrons. If this experiment would be possible it would unfold in its most striking evidence the wave behavior of particles and the wave-particle duality as single electrons interfere with themselves. Young's interference experiment is based on a few simple elements: first, a coherent monochromatic source of light or particles to which one can associate a wavelength. Second, two slits which behave as secondary wave sources. Third and final, an observation screen. The appearance of interference fringes on the screen is an undoubtable sign of the wave nature of the physical process of propagation. The appearance of the interference, in fact, implies that the two slits behave as secondary waves and the intensity of the signal on the observation screen is proportional to the square of the absolute value of the sum of these waves so the interference term must be not vanishing and time independent. One is thus compelled to thinking that a wavelike process is happening and that the single photon or the single electron is passing simultaneously through both the slits. In Feynman's own words, this is "A phenomenon which is impossible.... to explain in any classical way, and which has in it the heart of Quantum Mechanics. In reality, it contains the only mystery of Quantum Mechanics ".

Up to now, the superposition of electron waves has been demonstrated in a variety of arrangements: single slit, single hole, double slit, double hole, multiple slits and electrostatic bi-prism. (for a review on electron interferometry see [28]). Nevertheless, the most striking part of the experiment,

i.e. the build-up of the interference pattern by the single electrons arriving on the final screen, has been observed only by means of very poor statistic samples (see [28]-[33]) and without recording the single electron arrival time on the screen. The reason of this lies on the rate limitation of the available recording systems, able to collecting a relatively low number of pictures within a time interval compatible with a stable operation of the electron microscope. In this work, the time distribution and build-up of high statistic single electron interference pattern is observed for the first time with the Young-Feynman set-up. Nanometric slits were prepared by using modern nanotechnology tools. A conventional transmission electron microscope was used as a versatile optical bench and a recording system, sensitive to single electrons, replaces the final viewing screen of the microscope. The APSEL4D chip, described in the previous chapters, was used to record the impact position and time of single electrons. The fast read-out chain, which is able to manage up to 106 frames per second (fps) allowed us to collect high statistic samples of single electron events within a time interval where the stable operation and the coherence conditions of the microscope are guaranteed. Moreover, the large fraction of empty events made it possible to get the first measurement of the time distribution of electron arrivals. The results were published in a series of articles [34]-[37].

6.1 The Experimental Setup

The Transmission Electron Microscope

The experiments were carried out with a Philips EM400T Transmission Electron Microscope (TEM), equipped with a hair-pin filament source, operating mostly at an energy of 60 eV. In these conditions the electron De Broglie wavelength is

$$\lambda = \frac{\hbar}{p} = 4.9\,\mathrm{pm} \tag{6.1}$$

The 60 keV value for the electron energy was chosen as a balance between the minimum detectable energy and the features of the TEM. Due to the smallness of the wavelength value, the experiment required a dedicated set-up. Hence, exploiting the electron lenses, it was possible on one hand to demagnify the electron source in order to satisfy the necessary coherence conditions and, on the other hand, to magnify the image in such a way that its dimensions were compatible with the detector dimension and pixel size. In addition, due to the small electron diffraction angles (of the order of 10^{-5}

Figure 6.1: The Philips EM440T Transmission Electron Microscope

radiants) associated with the separation of the slits, the Fraunhofer pattern was observed in the so-called low-angle diffraction mode. In this electro-optical set-up the condenser lenses were excited at their maximum strength in order to reach the necessary lateral coherence of the illumination and the objective lens is weakly excited in order to project the image onto the selected area aperture plane. As a consequence, in these conditions, the microscope works as a diffraction camera having camera lengths extending up to several hundred meters.

The double slit

The slits, (shown in Fig.6.2) were fabricated by a Focused Ion Beam (FIB, see Fig.6.3) that mills a gold layer about 250 nm thick, deposited by flash evaporation on a commercial copper grid, coated with a carbon film. FIB milling was performed with a dual beam apparatus (FEI Strata DB235M) that combines a 30 keV Ga^+ FIB with a thermal field emission scanning electron microscope, having spatial resolutions of 6 nm and 2 nm respectively

In particular, the slits were obtained by using a 10 pA ion beam with a spot-size of about 10 nm. The passage through the gold-carbon bilayer was

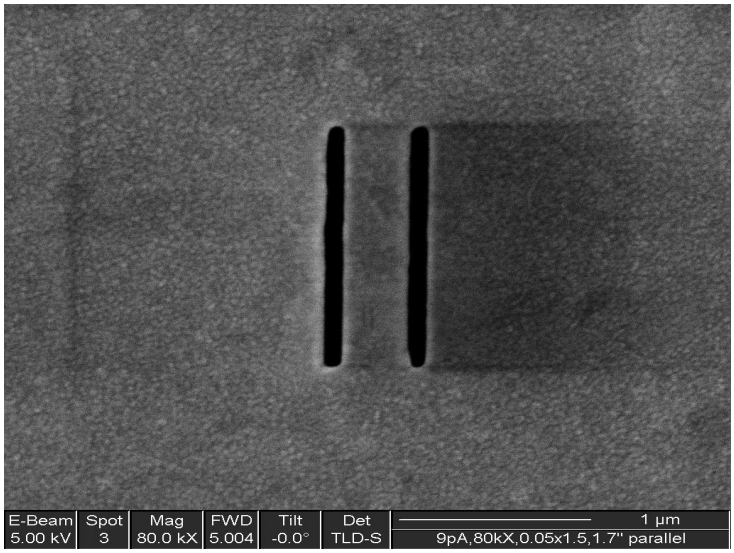


Figure 6.2: An image of the double slit obtained with a scanning electron microscope

monitored through the change in brightness of the ion-induced secondary electron emission. The slit width, length and spacing are 95 nm, 1550 nm and 450 nm respectively.

6.2 System Calibration

As we want to use a TEM to collect interference and diffraction patterns via a small matrix of pixels, the first step to take is the calibration of the system

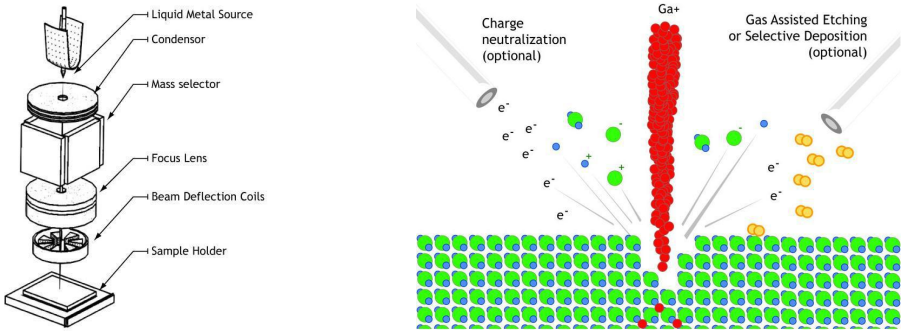


Figure 6.3: The scheme of the Focused Ion Beam milling technique

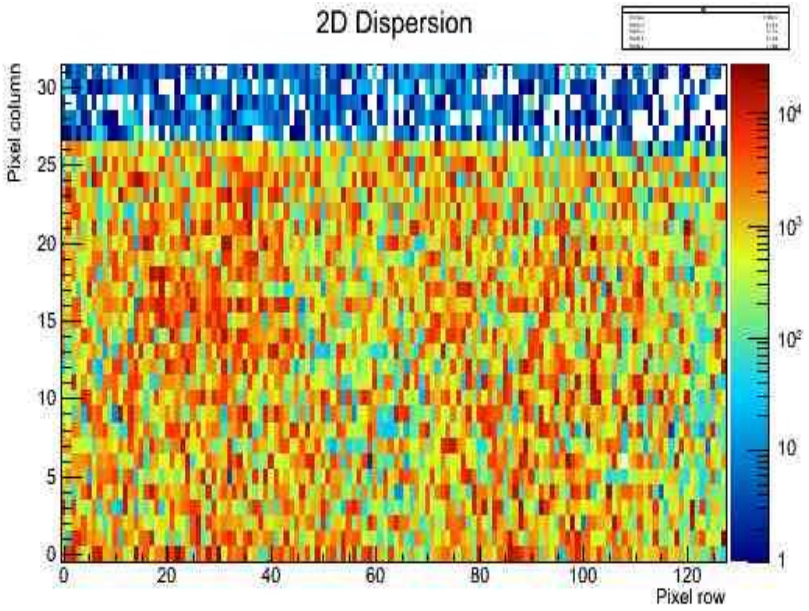


Figure 6.4: The hit distribution with part of the sensor obstructed

and of the data acquisition chain. As a first step we placed the sensor inside the microscope in a position where a part of the sensible area was obstructed. The image we recored is shown in Fig.(6.4) where one can note a part of the sensor is reached by the electronic beam and a part is masked. This was the first check that the DAQ chain was working properly and that the sensor was able to reveal electrons with an energy of 60 keV. Then a sample with a single slit and a triple one, with dimensions similar to those we were going to use for the Young experiment were placed inside the TEM. The electromagnetic lenses were set in order to project an image of the specimen to the plane were the chip is placed, so that an image of the slits was expected. The results are shown in Figs.(6.5) and (6.6), again showing the expected pattern. Finally, again setting the proper electromagnetic lenses conditions, we went to the diffraction regime using a carbon grating with a spacing of 463 nm, i.e., very similar to the slit spacing described earlier, as specimen. Fig.(6.7) shows a conventional image and a low-angle grating diffraction pattern. In order to make contact with the Young experiment and by analyzing the data we found that the average number of electrons per frame was about 8, meaning that it would be sufficient to increase the coherence and diminish the beam intensity in order to obtain the condition in which there is a single electron hit per frame. The right side of Fig.(6.7) shows a rectangular white box superimposed on the grating diffraction picture and the white box represents

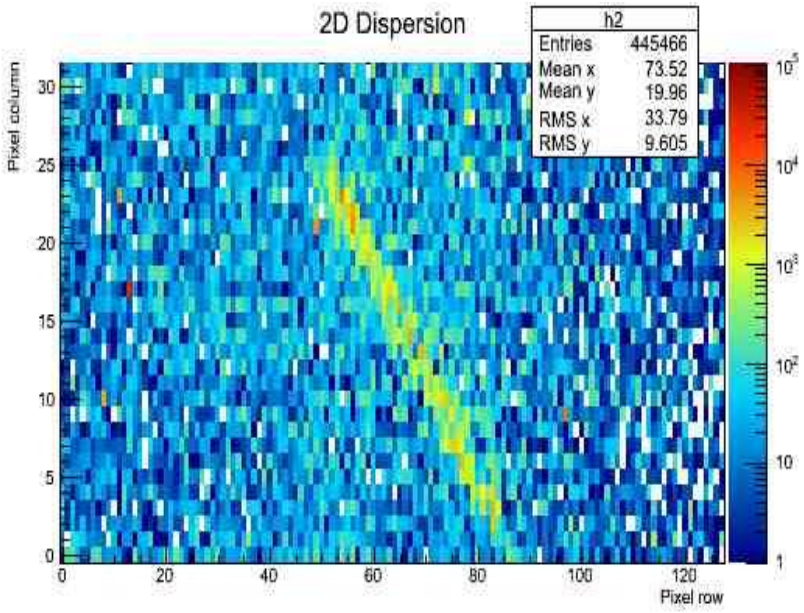


Figure 6.5: The hit distribution for a single slit specimen

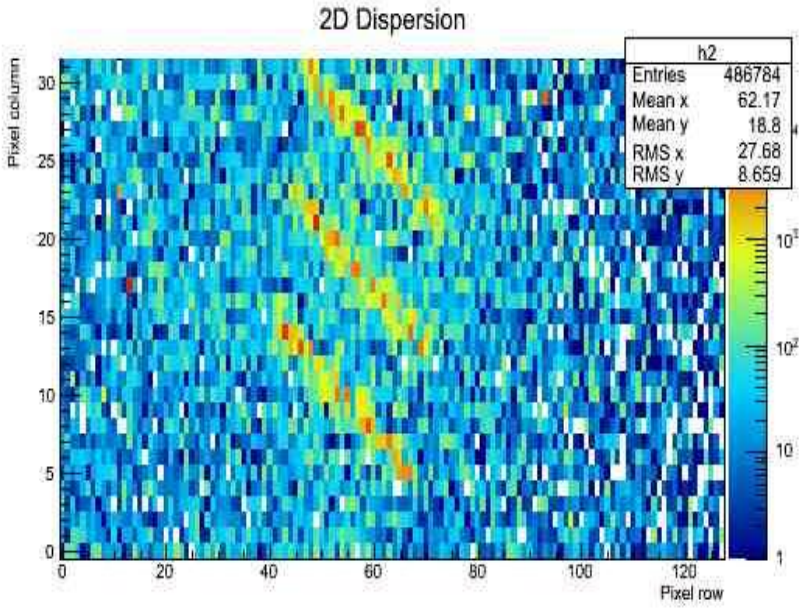


Figure 6.6: The hit distribution for a triple slit specimen

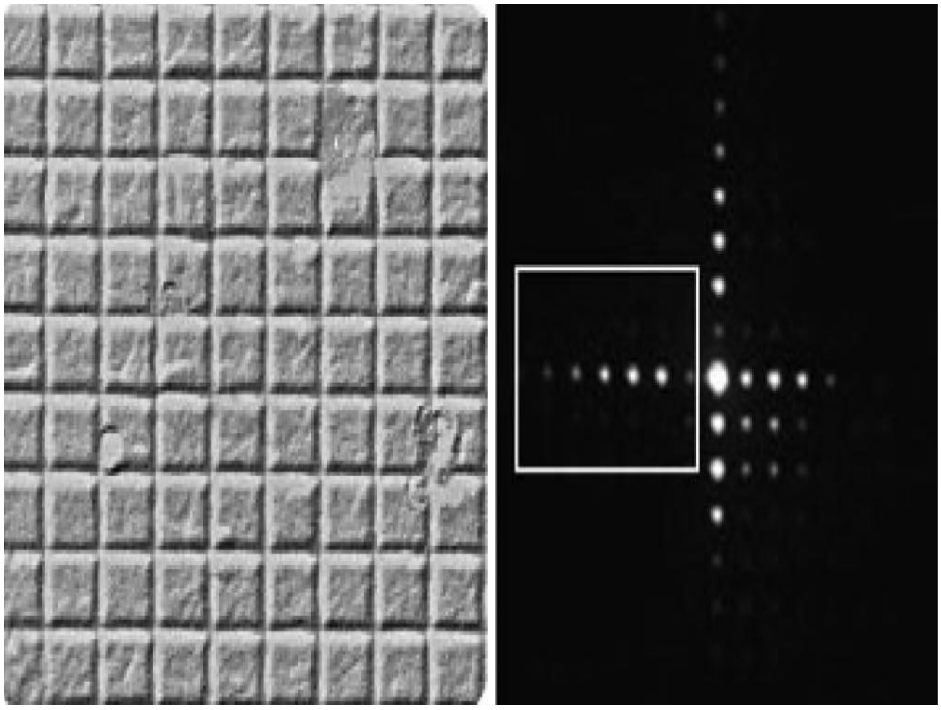


Figure 6.7: A typical diffraction grating image and the expected low-angle diffraction pattern: the white box enhances the portion of the pattern measured via the sensor

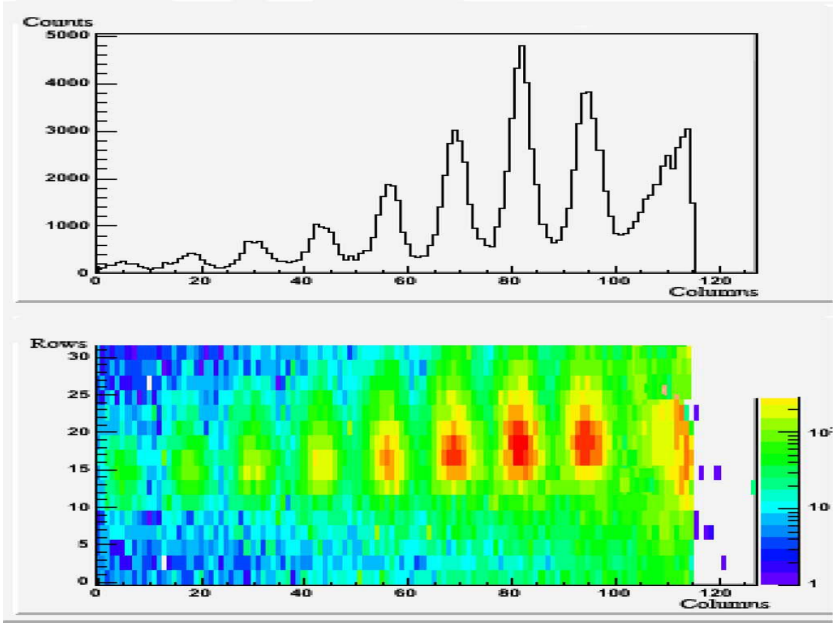


Figure 6.8: A carbon grating diffraction image with fringes on top

the portion of the (diffraction) pattern visible by our detection system. In fact, Fig.(6.8) shows one branch of fringes of the diffraction pattern from a carbon grating, using our rectangular matrix of pixels.

The single electron condition was finally reproduced by inserting a thin wire of 0.5 μm diameter as a specimen. Fig.(6.11) shows the image of the wire along with the hit distribution integrated in time. The picture shows the first lateral fringes of the diffraction pattern. Hence, through these first measurements, we were able to calibrate and tune the TEM in order to obtain diffraction patterns on the small pixel sensor, which measures only 6.4 $\times$ 1.6 mm^2 .

6.3 The Single Electron Young Experiment

The double slit used for the final experiment have already been described in (6.1). Regarding the TEM operating conditions, the current intensity must be high enough in order to collect a good statistics within a time interval that guarantees a stable operation of the microscope but, at the same time, the current should be as low as to guarantee the necessary coherence of the electrons associated with the poor brightness of the thermionic source. Eventually, a compromise condition was found so that it was possible to record the position and time of electron arrivals on the detector, in the single

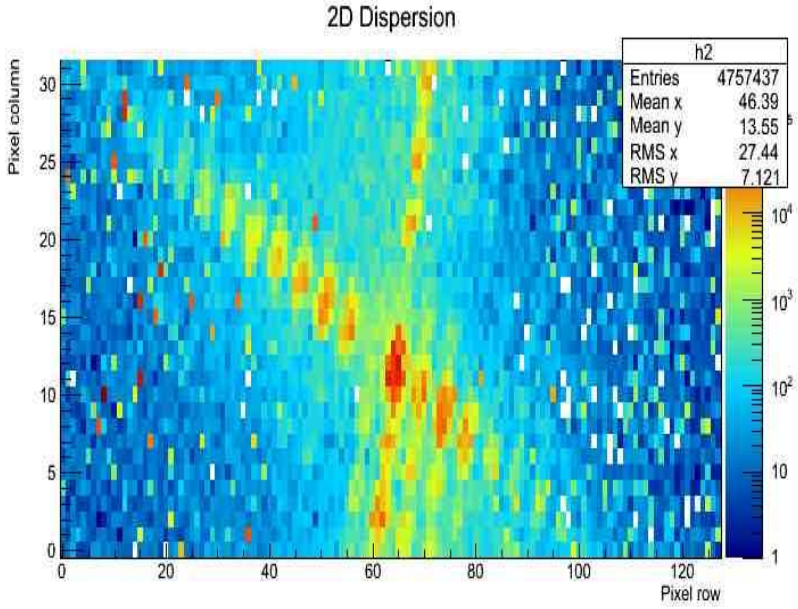


Figure 6.9: A carbon grating diffraction image with different TEM conditions

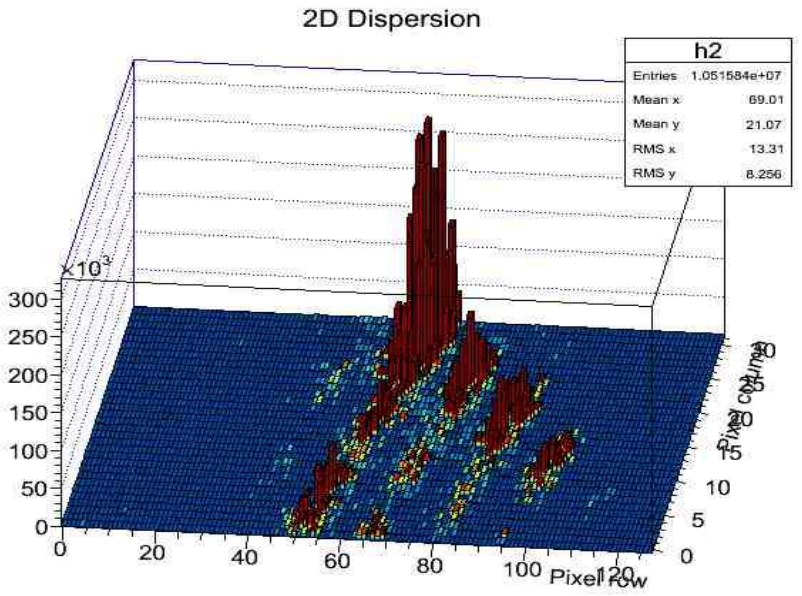


Figure 6.10: A three dimensional diffraction image for carbon grating diffraction

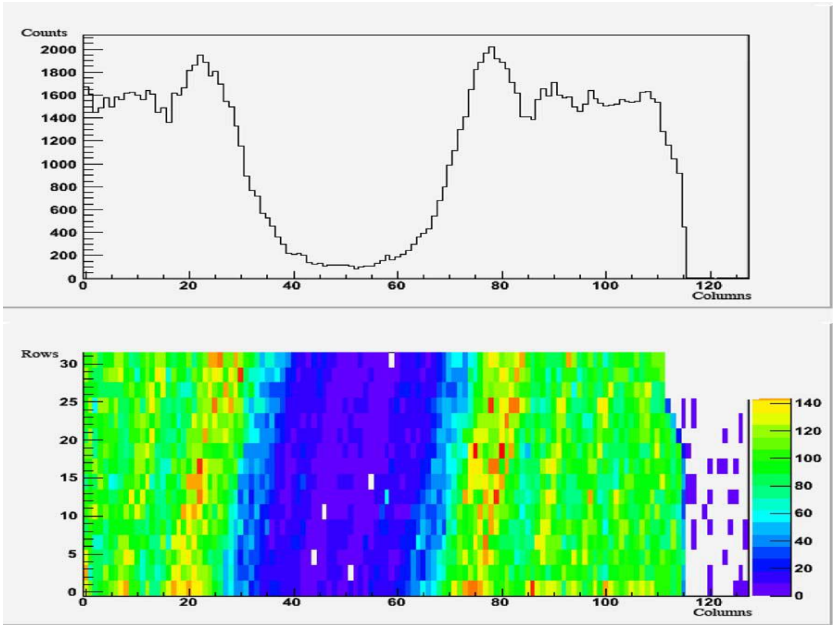


Figure 6.11: The hit distribution for a wire with diffraction fringes displayed on top.

particle regime, with the desired accuracy and resolution. A pictorial view of the stack of frames collected in a typical run is shown in Fig.(6.12). In particular, the frame rate was chosen in order to contain a fraction of frames with electron multiplicity higher than one at the percent level. In a typical run, 131 thousand non-empty frames were recorded during about 5 minutes, with a frame rate of 6.25 kfps. The plot in Fig.(6.13) reports a measurement with a BCO time of 160 μs, needed to sufficiently separate the electrons one from the other. The interference scatter plot appears as expected from the interference of the two slits modulated by the diffraction due to their widths (Fig.6.13(a)). More quantitative information, like the ratio between distance and width of the slits (which does not depend on the magnification of the interference figure) or the degree of coherence can be extracted by fitting the line scan obtained by averaging the data along the vertical direction (Fig.6.13(b) and 6.14). In fact, as the ratio between distance and width of the slits does not depend on the magnification of the interference figure, it can be extracted by fitting the projection of the figure on the axis passing through the maxima. The value of 3.8 $\pm$ 0.2 thus obtained is in good agreement with the direct measurement carried out by the Scanning Electron Microscope (4.0 $\pm$ 0.1). A coherence parameter was needed in order to describe the feature that the interference minima are not exactly at zero, this depending on the

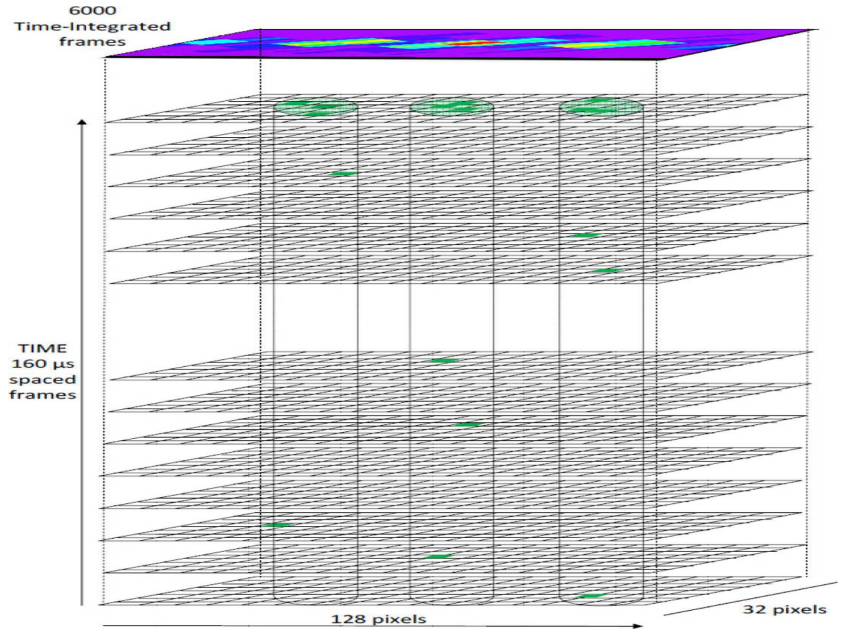


Figure 6.12: A pictorial view of the stack of frames collected in a typical two-slit interference run.

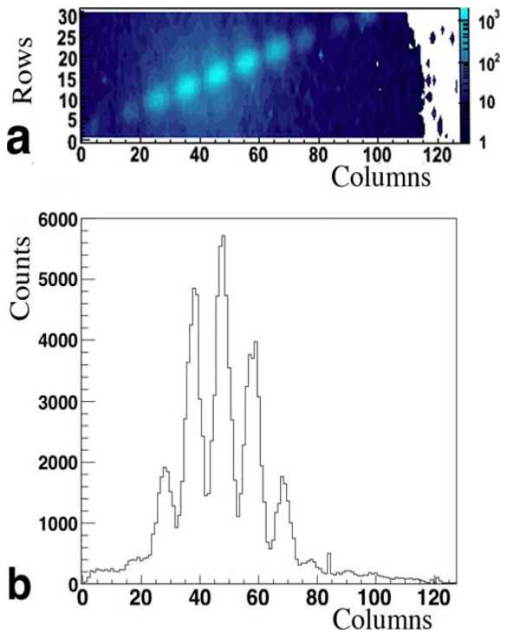


Figure 6.13: The interference scatter plot obtained by adding-up the stack of frames (a) and its projection along the vertical axis (b).

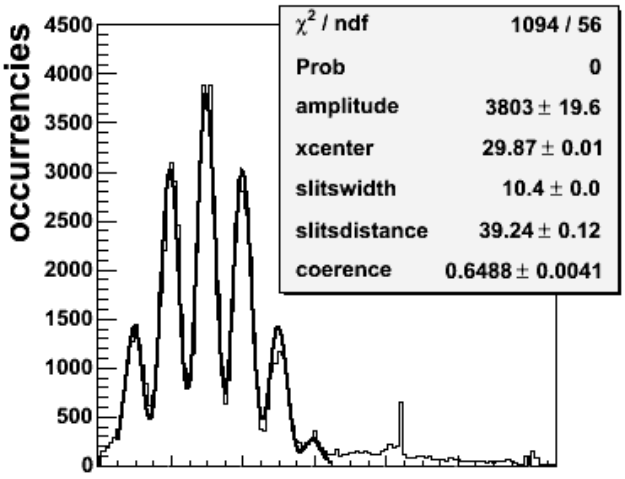


Figure 6.14: The interference scatter plot projection with a superimposed fit.

TEM electron beam experimental conditions.

Regarding the electron time arrival distribution, one can see in Fig.(6.15) that 95.2% of the collected frames are empty, 0.1% have multiple hits and 4.7% one-hit events. The zero bin of the histogram is not shown as we only represent the hit pixels. Moreover, by comparing the average time distance between the detected electrons, which amounts to 8.7 ms, to the time of flight between emission and absorption, which is of about 30 ns, we can conclude that the signal of one electron is read out before the next one is emitted: the single electrons are actually interfering with themselves. From the measurement of the time interval that separate two adjacent non-empty, frames we get the distribution of the arrival time of the electrons on the detector, and this is shown in Fig.(6.16). In more detail, the exponential behavior of uncorrelated electrons is observed as expected for a Poisson distribution of events as can be easily observed from the logarithmic scale plot.

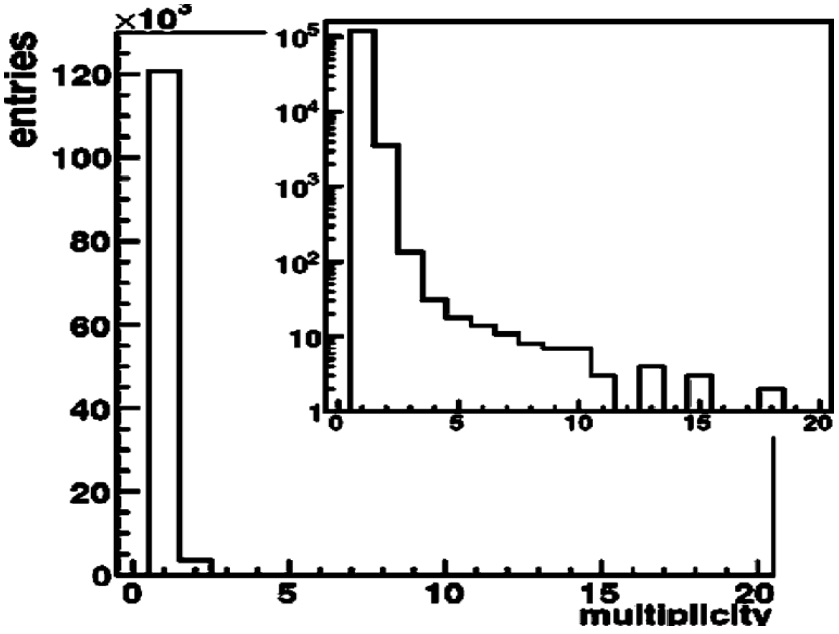


Figure 6.15: The multiplicity distribution of hit pixels within the frames.

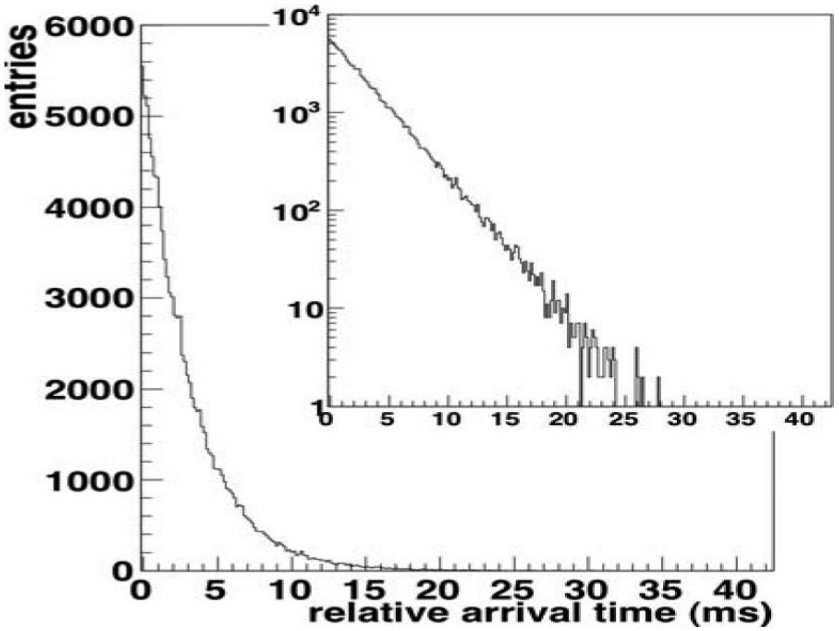


Figure 6.16: The distribution of time intervals between two consecutive non-empty frames in linear and logarithmic scale.

Conclusions

We have realized a Data Acquisition chain for the use and characterization of APSEL4D, a 32 $\times$ 128 Monolithic Active Pixel Sensor, developed as a prototype for frontier experiments in high energy particle physics.. In particular a transition board was realized for the conversion between the chip and the FPGA voltage levels and for the signal quality enhancing. A Xilinx Spartan-3 FPGA was used for real time data processing, for the chip control and the communication with a Personal Computer through a 2.0 USB port. For this purpose a firmware code, developed in VHDL language, was written. Finally a Graphical User Interface for the online system monitoring, hit display and chip control, based on windows and widgets, was realized developing a C++ code and using Qt and Qwt dedicated libraries. APSEL4D and the full acquisition chain were characterized for the first time with the electron beam of the transmission electron microscope and with ^{55}Fe and ^{90}Sr radioactive sources. In addition, a beam test was performed at the T9 station of the CERN PS, where hadrons of momentum of 12 GeV/c are available. The very high time resolution of APSEL4D (up to 2.5 Mfps, but used at 6 kfps) was fundamental in realizing a single electron Young experiment using nanometric double slits obtained by a FIB technique. On high statistical samples, it was possible to observe the interference and diffractions of single isolated electrons traveling inside a transmission electron microscope. For the first time, the information on the distribution of the arrival time of the single electrons has been extracted.

Bibliography

[1] W.R. Leo, *Techniques for Nuclear and Particle Physics Experiments*, Springer-Verlag, 1987

[2] F. Knoll, *Radiation detection and Measurement*, John Wisley and Sons, 1979.

[3] C. Amsler et al., *Review of Particle Physics*, Phys. Lett. B 667, 2008

[4] T. Del Prete, *Lectures on the Interaction of Particles and Radiation with Matter*, INFN-PISA, 2002

[5] M. Marucho et al., *The Landau Distribution for Charged Particles Traversing Thin Films*, Int. J. of Mod. Phys. C, 17, 2006very high

[6] S.M. Sze, *Physics of Semiconductor Devices*, Wiley, 1981.

[7] H. Spieler, *Semiconductor Detector Systems*, Oxford University Press, 2005

[8] N.E. Ashcroft and D.N. Mermin, *Solid State Physics*, Harcourt, Inc. 1976.

[9] G. Lutz, *Semiconductor Radiation Detectors*, Springer-Verlag, 1999

[10] Yu. K. Akimov, *Silicon radiation detectors*, Instruments and Experimental Techniques, 50, 2007

[11] L. Rossi, P. Fischer and T. Rohe, *Pixel Detectors: From Fundamentals to Applications*, Springer-Verlag, 2006

[12] ATLAS Detector and Physics Performance Technical Design Report, CERN-LHCC-99-015, 1999

[13] CMS Physics TDR: Volume I (PTDR1), CMS Physics TDR, Volume I: CERN-LHCC-2006-001, 2006

[14] A. Gabrielli, *Apsel 4D User Guide*, 2007

[15] G. Rizzo, *A novel monolithic active pixel detector in 0.13 μm triple well CMOS technology with pixel level analog processing*, Nucl. Inst. and Meth., A-565, 2006

[16] G. Rizzo, *Recent development on triple well 130 nm CMOS MAPS with in-pixel signal processing and data sparsification capability*, IEEE Nuclear Science Symposium, 2007

[17] G. Rizzo, *Recent development on CMOS monolithic active pixel sensors*, Nucl. Inst. and Meth., A-576, 2007

[18] A. Gabrielli et al., *Proposal of a data sparsification unit for a mixed-mode MAPS detector*, IEEE Nuclear Science Symposium, 2007

[19] A. Gabrielli et al., *Proposal of a sparsification circuit for mixed-mode MAPS detectors*, RD07 Int. Conf. on Large Scale Applications and Radiation Hardness of Semiconductor Detectors, 2007

[20] A. Gabrielli, F.M. Giorgi, F. Morsani and M. Villa, *Development and simulation results of a sparsification and readout circuit for wide pixel matrices*, Nucl. Phys. B, 2011, 215

[21] A. Gabrielli et al. *A 4096-pixel MAPS device with on-chip data sparsification*, Nucl. Inst. and Meth., A 604, 2009

[22] A. Gabrielli, F.M. Giorgi and M. Villa, *A high efficiency readout architecture for a large matrix of pixels*, Journ. of Instr., 2010, 5

[23] N. Neri et al. *Deep n-well MAPS in a 130 nm CMOS technology: Beam test results*, Nucl. Instr. and Meth., A 623, 2010

[24] M. Villa, et al., *Beam-test results of 4k pixel CMOS MAPS and high resistivity striplet detectors equipped with digital sparsified readout in the Slim5 low mass silicon demonstrator*, Nucl. Instr. and Meth., A 617, 2010

[25] A. Bagnari, *Test di efficienza del chip Apsel4D*, 2009

[26] T. Del Prete, *Methods of Statistical Data Analysis in High Energy Physics*, INFN-PISA, 2000.

[27] E. Gatti, and P.E. Manfredi, *Processing the signals from solid-state detectors in elementary particle physics*, Riv. Nuovo Cimento 9/1, 1986

[28] F. Hasselbach, Progress in electron and ion interferometry, Rep. Prog. Phys., 73, 2010.

[29] S. Frabboni, G.C. Gazzadi and G. Pozzi, *Interferenza di elettroni da due e tre fenditure*, Giornale di fisica, XLVIII, 2007

[30] G. Matteucci, G.F. Missiroli and G. Pozzi, *Interferometric and holographic techniques in transmission electron microscopy for the observation of magnetic domain structures*, IEEE Transactions on Magnetics, 20, 1984.

[31] P. Mazzoldi, M. Nigro and C. Voci, *fisica Vol.2 - Elettromagnetismo e Onde*. Edises, 1998

[32] P.G. Merli, G.F. Missiroli and G. Pozzi, *Diffrazione ed interferenza di elettroni - I : Diffrazione*, Giornale di fisica, XV, 1974

[33] P.G. Merli, G.F. Missiroli and G. Pozzi, *Diffrazione ed interferenza di elettroni - II : Interferenza*, Giornale di fisica, 17, 1976

[34] A. Gabrielli et al., *A 4096-pixel MAPS detector used to investigate the single electron distribution in a Young Feynman two slit interference experiment*, Nucl. Instr. and Meth., 699, 2013

[35] G. Matteucci et al., *Build-up of interference patterns with single electrons*, Eur. J. of Physics, 34, 2013

[36] G. Balbi et al., *Electron Interference via a 4096-Pixel MAPS Detector Designed for High-Energy Physics Experiments*, IEEE Trans. on Nucl. Sci., 60, 2013

[37] S. Frabboni et al., *The Young-Feynman two-slits experiment with single electrons: Build-up of the interference pattern and arrival-time distribution using a fast-readout pixel detector*, Ultramicroscopy, 116, 2012