Pixel detector group
Institute of Particle Physics
ETH Zurich

Qualification of the X-ray test setup for the Vcal calibration of the CMS pixel detector readout chip

Semesterarbeit

Daniel Narrias Villar
Supervisors: Prof. Rainer Wallny
Marco Rossini

July 6, 2012

Abstract

Keywords: CMS, pixel readout chip, Vcal calibration, X-123 X-ray spectrometer.

This semesterarbeit has two main aims: (1) to identify and to suppress the source of noise in the spectra of the targets Fe, Cu, Mo, Ag, Sn and Ba taken with the X-123 X-ray spectrometer of the Pixel Group at ETH Zurich; (2) to make the Vcal calibration for two CMS pixel readout chips and to obtain the number of electrons per Vcal unit. It was found that the noise was due to a wrong Pile-Up Rection (PUR). The thresholds of the Digital Pulse Processor fast and slow channels were appropriately tunned to suppress the noise. The Vcal calibrations for the chip 1 are 62.97 ± 4.18 $[electrons/Vcal]$ when the chip 1 is untrimmed, 61.20 ± 2.69 $[electrons/Vcal]$ when trimmed to Vcal60 and 55.86 ± 1.98 $[electrons/Vcal]$ when trimmed to Vcal110, and for the chip 2 is 60.75 ± 2.14 $[electrons/Vcal]$ when trimmed to Vcal110. A study of the pixel readout chip, the readout chain, the X-ray setup of Pixel Group, and Vcal calibration was made.

Contents

Chapter 1

Introduction: LHC and CMS

1.1 LHC

LHC is a multipurpose circular proton-proton collider of about 27 km of circumference, built and managed by Le Conseil Europen pour la Recherche Nuclaire, CERN. LHC has as aims the finding of the Higgs and the finding of physics beyond the Standard Model. The nominal center of mass energy of LHC is $\sqrt{s} = 14$ TeV, so each proton beam has a nominal energy of 7 TeV. Since a proton is a composite particle, only a fraction of this energy is available for the production of particles in hard interactions, so the masses of produced particle are up to some TeV. LHC has four experiments: ATLAS, ALICE, CMS and LHCb.

At the interaction points of the four experiments, the particle bunches cross with a nominal frequency of 40 MHz, or equivalently bunch crossing takes place every 25 ns, and the instantaneous luminosity $\mathcal{L}_{inst}$ of about $\mathcal{L}_{inst} \sim 10^{34} cm^{-2} s^{-1}$. At this luminosity, with nominal numbers of 2808 bunches per beam and about $1.15 \cdot 10^{11}$ particles per bunch, about 25 inelastic interactions take place per bunch crossing.

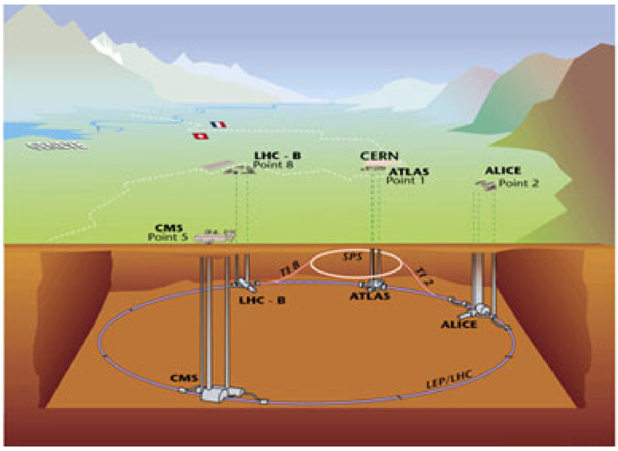


Figure 1.1: LHC and its experiments: ATLAS, ALICE, CMS and LHCb.

1.2 Compact Muon Solenoil

CMS is a multipurpose experiment, whose main purposes are the Higg boson search and the quest for supersymmetry. As shown in (here reference), the beam pipe in the very centre is surrounded by the silicon tracker which consists of two subdetectors: the **silicon pixel detector**, the innermost subsystem of CMS, and the **silicon strip detector**. Both consist, respectively, of 3 pixel and 10 strip layers in the barrel plus 2 pixel and 12 strip disks at each end of the detector. Then, the **electromagnetic** and **hadronic calorimeters** follow, in the outward direction. Silicon tracker and both calorimeters are surrounded by a **superconducting solenoid** which provides a 3.8 T magnetic field parallel to the beam axis. Finally, the **muon chamber detectors** follow. All subdetectors consist of a barrel and two forward parts to achieve an almost hermetic coverage.

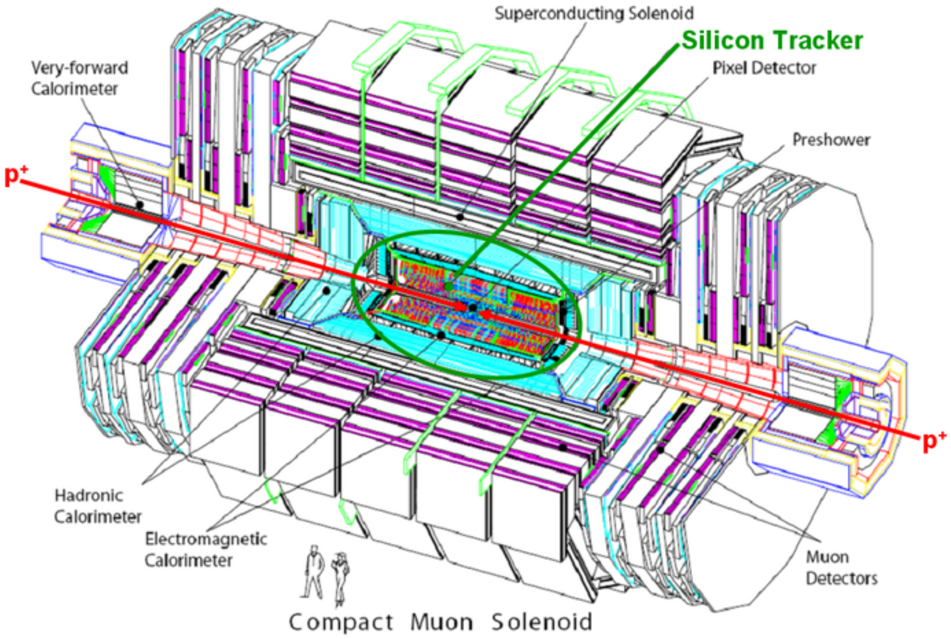


Figure 1.2: CMS detector layout: pixel detector, silicon tracker, electromagnetic and hadronic calorimeters, superconducting solenoid, muon detectors.

Chapter 2

CMS Barrel Pixel Detector

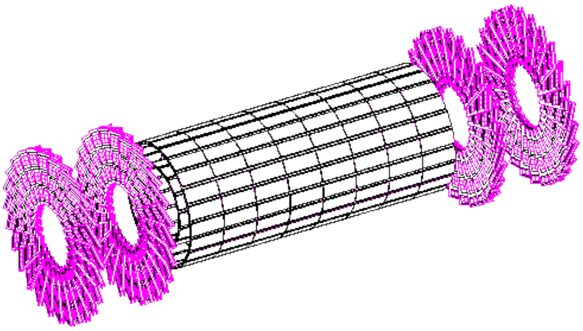


Figure 2.1: Layout of CMS pixel detector

The CMS pixel detector provides a *high tracking resolution* as close as possible to the interaction point. A precise hit reconstruction allows for a precise tracking and vertexing, a good impact parameter resolution and a good separation between primary and secondary vertices, which is very important for many physics channels. The high hit resolution is thanks to the absence of so called ghost hits in a pixel system, unlike the silicon strip detector which presents ghost hits due to wrong combination of hit strips.

As shown in fig. 2.1, the layout of the pixel detector consists of a 53 cm-long barrel part with 3 layers at radii 4.4 cm, 7.3 cm and 10.2 cm, and of 2 forward disks placed on each side at $z = \pm 34.5$ cm and $z = \pm 46.5$ cm with an acceptance up to $|\eta| = 2.4$. The entire detector is constructed in a modular way and contains $66 \cdot 10^6$ channels.

Lorentz drift is used to improve the hit resolution. If the electric field in the sensor is not parallel to the magnetic field in the beam axis, the moving electron hole pairs in the silicon sensor undergo Lorentz drift which distributes the charge among several pixels, which then allows for un improved hit resolution by knowing the direction of the Loretnz force. In the barrel part the drift direction of the electron hole pairs is perpendicular to the magnetic field, so they are forced to the neightboring pixels. In the forward part, the modules are slightly rotated by 20°, which make the electron hole pairs drift away.

2.1 The Pixel Barrel Modules

The whole pixel detector contains 672 modules distributed in its three layers along with 96 half modules to put two half shells of the detector together. The barrel part has $48 \cdot 10^6$ channels and the whole pixel detector including the forward disks has $66 \cdot 10^6$ channels. A single barrel module has a size of 66.6 mm·26 mm, weights 3.5 g, has 16 readout chips (ROCs) and comprises 66560 ($= 4160 \cdot 16$) pixels. The sensor has a thickness of 285 μm and a pixel has a size of 100 μm ·150 μm (rϕ·z).

The components of a barrel model are shown in the fig 2.2. The silicon sensor is connected to 16 readout chips (ROCs) via indium bumps, which connect each sensor pixel with a pixel unit cell (PUC) on the ROC. A ROC has 4160 ($= 52 \cdot 80$) PUCs. On top of the sensor a High Density Interconnect (HDI) serves as an interface to the front end electronics. The connection betweem the HDI and the front end electronics is established through the power cable which supplies voltages, and the signal cable which controls signals and the chip analog readout. The readout of all the ROCs is organized by the Token Bit Manager (TBM) on top of the HDI. Finally, the base stripes provide mechanical rigidity and are used to mount the module onto the support and cooling structure.

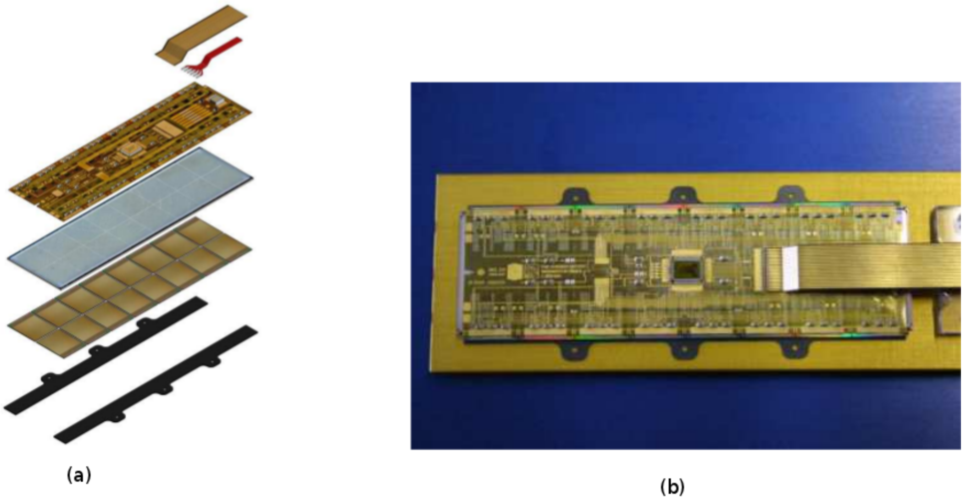


Figure 2.2: (a) Components of pixel barrel module, from top to bottom: the signal cable, the power cable, the HDI, the silicon sensor, the 16 ROCs and the base strips. (b) Photograph of a pixel barrel module.

2.2 The Readout Chip

This section is based mostly on papers [1] and [2].

The ROC consists of three main blocks: (1) 4160 pixel unit cells (PUC) organized in 26 double columns of $2 \cdot 80$ pixels; (2) 26 double column peripheries which have 12 8-bit time stamp buffers and 32 data buffers each; (3) and one control and interface block.

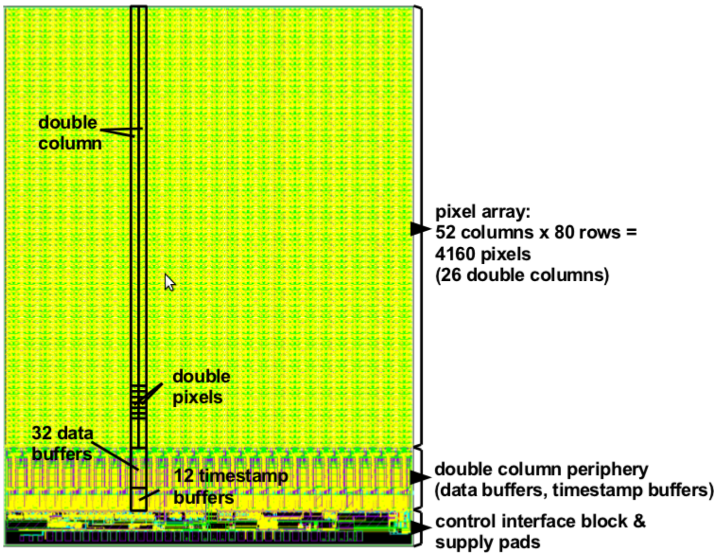


Figure 2.3: Double columns with their 32 data buffers and 12 time stamp buffers each.

For data recording, triggering and readout purposes each ROC has four counters, the contents of which are available in each double column perifery:

- *Bunch crossing counter* **RBC**, 8 bits. Its content is written as time stamp into a time stamp buffer. It is incremented every clock cycle.
- *Bunch crossing counter with trigger delay* **WBC**, 8 bits. It runs typically 128 counts (3.2 μs) behind the bunch crossing counter and is used to validate hits belonging to a trigger. It is incremented every clock cycle.
- *Trigger counter*, 4 bits.
- *Token counter*, 4 bits. Both of them ensure that for each readout token the correct double columns are read out.

A PUC measures the amount of charge produced in the sensor, amplifies it and shapes it, comparing it to a threshold and sending it out along with the address of the hit pixel.

The goal of the ROC is to measure how much ionization charge was produced in which pixel. The liberated charge in the sensor will generate a voltage in the PUC via a capacitance. The analog signal goes then through the preamplifier and the shaper. After the shaper, the signal takes two paths, one into the comparator, and one into the sample and hold mechanism. When the rising edge of the signal has passed the threshold in the comparator, the signal height is sampled after some delay (*VhldDel*), necessary to reach the peak, and stored in the sample and hold capacitor until the first readout mechanism is started via the first TBM token. The threshold can be set by a global value for the whole ROC (*VthrComp*) and by four trim bits for each single pixel (applied via *Vtrim*).

When the signal in the comparator passes the threshold, the pixel sends a current signal to the periphery (adjustable by *VIColOr*), which determines the clock cycle at which the pixel was hit. This timing information concerns the whole double column, so the timing information is the same whether or not more than one pixel are hit. Thus, in the periphery a time stamp (bunch crossing number) is created and written into the time stamp buffer presently pointed at.

The first readout of all pixels has two path processing, first a fast one, corresponding to the signal through the comparator, to store the time of a hit, and second a slow one, corresponding to the signal through the sample and hold mechanism, to read the signal height and the pixel adress.

The readout in the low path is organized using the column drain mechanism, which consists of one first TBM token that arranges the readout of two neighboring columns (double column): the readout starts in the left column with the pixel closest to the periphery, goes up the left column, comes back down the right column and ends with the last pixel in the right double column. The adress and signal height stored in the sample and hold capacitance are read, assigning them to the corresponding time stamp, when the first token goes through the pixel. The pixel adress is sent from the PUC to the periphery as digital current levels which are then converted there into voltage levels (*VIbias bus*). The analog signals and the pixel adresses of all hit pixels are written into the next free data buffers (one data buffer per hit pixel). This hit-recording into the time stamp and data buffers of the first readout runs autonomously and asynchronously in each double column of the ROC, independently of the bunch crossing clock. The recorded hit information is kept in the buffers during the latency time of the level 1 trigger (3.2 μs or 128 bunch crossings).

The time stamp corresponding to the hit pixels in a column is compared with the bunch crossing counter with trigger delay. If the hits in a column are not validated by an external level 1 trigger, i.e if the time stamp does not correspond to the WBC, the hits are cleared. Otherwise, the value of a 4-bit trigger counter is latched into the column periphery. The column is frozen and cannot record further hits until reset after readout of the triggered hits. Untriggered columns remain of course active.

All frozen columns with the latched value of the trigger counter equal to the present value of the token counter are set to the readout mode. Directly afterwards the trigger counter is incremented. If the trigger counter of the validated column is not equal to the token counter, the column waits for next token out. The second readout process starts when the second token bit from the TBM enters the ROC. After a three bit TBM header is sent, each consecutive double column which is in the readout mode is read out and reset. Just before the token leaves the chip, the token counter is incremented. It is worth emphasizing that only data belonging to a trigger along with its corresponding token is read out, so triggers without tokens go stand-by. When the token leaves a ROC, it enters the following ROC and all this process is repeated. Then, the readout is terminated by the TBM trailer and the token goes back to the TBM which then sends another token if necessary.

Finally, the data of all the columns that were read out is sent out from the control and interface block to the data bus to be processed.

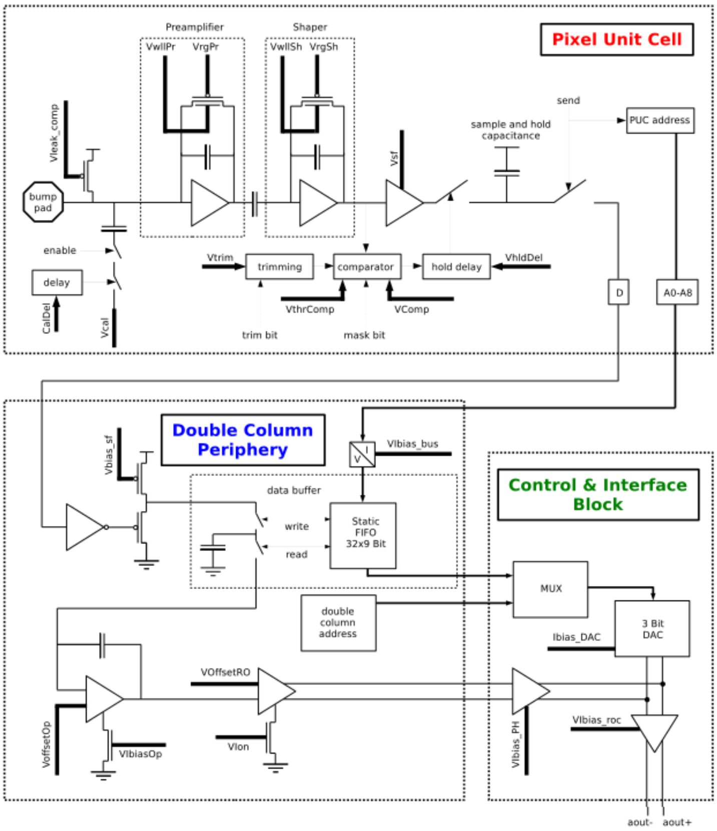


Figure 2.4: Squematic view of the readout chain.

The output of a module after being read out is shown in fig 2.5. The TBM trailer consists of 8 clock cycles. It starts with 3 Ultra Black Levels (UBLs), which mark the lower bound of the analog signal. Then, a black level follows, which defines the zero level of the differential analog signal. The 4 remaining clock cycles encode an 8-bit event number. The minimal readout if each ROC in absence of hit goes as an UBL, a black level and a ”last DAC”, the latter representing the value of the most recently programmed DAC. Each hits adds a block of 6 clock cycles to the analog readout. The first 5 clock cycles encode the adress of the hit pixel (digital data) and the last one the pulse height (analog data). The readout is terminated by the TBM trailer consisting of 8 clock cycles: 2 UBLs, 2 black levels and 4 clock cycles with the TBM status information.

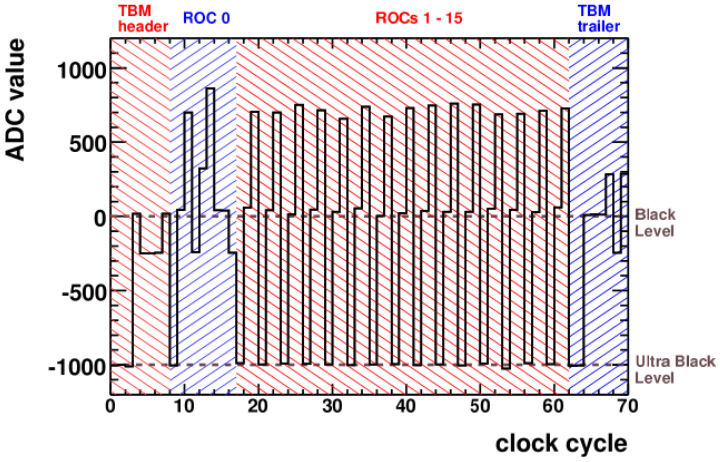


Figure 2.5: Readout output of a module with one pixel activated on ROC 0.

2.2.1 Thresholds

We can measure the threshold of a pixel in two different ways. We define the *Vcal-threshold* as the threshold measured by fixing the VcThr DAC and varying the Vcal value until finding the Vcal at which pixels start responding. We define the *VcThr-threshold* as the threshold measured by injecting a fixed Vcal DAC pulse and varying VcThr until finding the VcThr at which the signal is above threshold.

If the measurement of these thresholds is done by reading out the hits in a fixed bunch crossing, we call these thresholds *in-time thresholds*. If a pixel has a Vcal-threshold of 50, it does not mean that the pixel does not respond for Vcal values lower than 50 but that in that bunch crossing no hits with lower Vcal were registered. If we measure thresholds in different bunch crossing, the minimum of this thresholds is called the *absolute threshold* which is timing independent.

2.2.2 Calibration

To simulate an amount of change inside the sensor, a ROC internal signal can be used. The voltage signal is applied through *Vcal* and delayed by *CalDel*.

Vcal is an 8-bit DAC that generates a pulse which is injected to the PUC first through its preamplifier.

CalDel is an 8-bit DAC that delays the Vcal pulse in order that pulses peak in the clock cycle considered in the Vcal Threshold Map (see *Vcal-CalDel* optimisation subsection).

The aim of the Vcal calibration is to find the number of electrons to which a Vcal DAC unit corresponds.To find the actual number of electrons to which the Vcal input corresponds, we use a flourescent target to radiate the chip. The emission lines of the target liberate a known number of electrons in the Silicon sensor, which is collected in the bump pad and then sent to the pixel PUC as a voltage pulse. Then, a threshold is set in the PUC at which the pixel starts responding to the emission lines. A Vcal input then is used to inject a pulse in the PUC, using the same threshold, until finding the Vcal at which the pixel starts responding. And finally, as we know how many electrons this pixel threshold corresponds to, we can associated the number of electrons to the Vcal input, obtaining the desired calibration.

2.3 Chip Setup

Before making Vcal calibration, we need to run a few tests and settings to prepare the chip and to make sure that the chip is working well. Mainly, these processes are the Pretest and Trimming.

2.3.1 Pretest

The Pretest adjusts several DAC parameters to values for which the chip is working well. Among them, those which influence directly the fuctionality of the ROC are *Vana*, *VcThr*, *Voffset OP*, *Ibias DAC* and *CalDel*.

Vana DAC controls the voltage regulator that every ROC has. By changing the voltage of this voltage regulator, that is to say, by changing the Vana, we can control the amount of analog current that the ROC draws. We need to inject a current of about 3μA to the Preamplifier and Shaper of every PUC. As a ROC has 4160 pixels, we need to inject a current of about 24 mA to the ROC. Starting from the default value of Vana DAC, Vana is increased (decreased) as long as the analog current is below (above) 24 mA.

To use the internal calibration signal for further tests, its timing has to be tuned. The calibration signal of the ROC can be delayed with respect to the 40 MHz clock in steps of 0.32 ns with the CalDel DAC. The signal threshold has to be tuned along with the calibration signal, which we do by varying the VcThr DAC. The aim here is to minimize the timewalk, i.e. minimize the difference time between two signal when crossing the threshold. Both of them, the timing and the threshold, are tuned together in one step by running a DacDac test of VcThr versus CalDel (fig 2.5). The response of one pixel is scanned for over the whole VcThr-CalDel space and two values in the working region for both parameters are chosen arbitrarily.

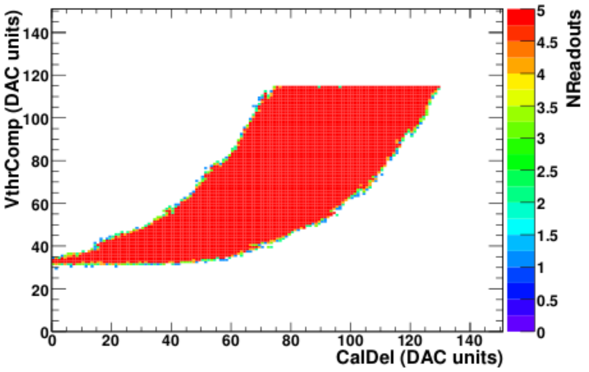


Figure 2.6: VcThr-CalDel space. The red area is the working region of the pixel.

The Voffset OP ADC in the double column periphery adds a DC offset to the analog pulse height.

The Ibias DAC in the control and interface block shifts the digital adress levels before they are joined together with the analog signal.

2.3.2 Trimming

Due to variations in the electronics components, each pixel has a slightly different threshold. That is why we need the trim bits and the trim algorithm to achieve an uniform threshold for all pixels.

The threshold (in Vcal units) to which the response of all pixels should be unified is the only input parameter to the trim algorithm. The three degrees of freedom to adjust are the *VthrComp* DAC, the *Vtrim* DAC, and the *trim bits* value of each pixel. The VthrComp sets a global threshold for the ROC, Vtrim determines the global step size of a trim bit and how much the trim bits lower this threshold, and the trim bits value is set independently for every single pixel to shift further the threshold to the target threshold from the value it reached with the VthrComp. The lower VthrComp and Vtrim, the higher the threshold is; the higher the trim bits value, the higher the threshold is.

The first step is to find the value of the VthrComp DAC which corresponds to the chosen threshold in Vcal units. A fixed charge signal defined by the Vcal DAC input is injected in every pixel and then a VthrComp Scan is done, that is to say, the VthrComp value of the chip varies and the respond of the pixels to the Vcal injection is measured, extracting then the VthrComp value from the measured *VthrComp-threshold* above which pixels start to respond. At this point all pixels have the same threshold but different VthrComp's. Since the thresholds can only be lowered afterwards, the minimum value of the VthrComp distribution determines the global VthrComp value. This value is used during the rest of the algorithm. It is worth mentioning that this threshold is a timing independent absolute threshold, as all other threshold measurements of the trim algorithm. We do this by looking at two bunch crossing, the one of the measurement and the following one.

The second step is to determine an appropriate Vtrim value. After the first step, all pixels have a given threshold defined by the VthrComp DAC. Here we fix the VthrComp DAC and make a Vcal scan, that is to say, a variable Vcal signal is injected until the pixel starts to respond, thus measuring the *Vcal-threshold*. The pixel which has the highest threshold is used to determine the necessary Vtrim value. For this pixel the trim bits value is set to zero and the Vtrim is increased, until the threshold of the pixel is at the same level as the target threshold.

The third step consists in setting the trim bits for all pixel. The trim bit values fill a range from 0 (maximally trimmed) up to 15 (not trimmed). The pixel with the highest threshold in the second step is then maximally trimmed. Then, for other pixels, by using a binary search, we look for the trim value which gives a threshold as close as possible to the target threshold (trim bits only takes non-negative integers values). At the end, all thresholds are measured once again to validate the procedure.

Chapter 3

X-ray setup

The X-ray setup consists of an X-ray tube, flourescent targets, a spectrometer, and a readout chip (ROC) with a testboard, as it can be seen in fig. 3.1.

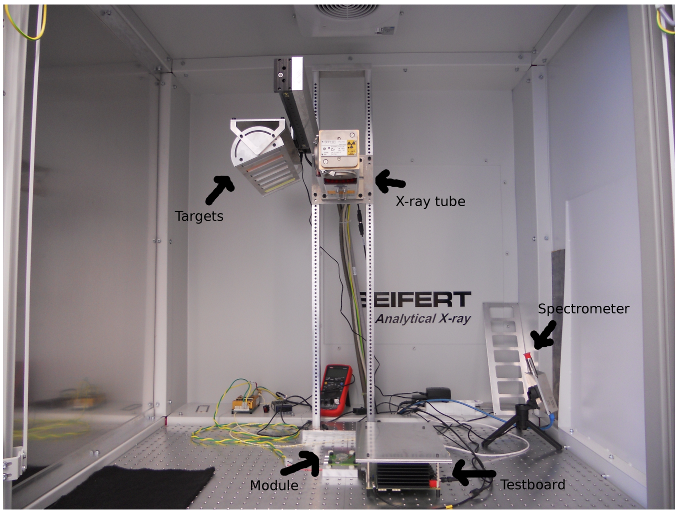


Figure 3.1: Experiment setup: X-ray tube, flourescent targets, spectrometer and ROC with testboard.

In general terms, the X-ray tube radiates bremsstrahlung X-ray to the targets. The targerts in turn emit fluorescence radiation in certain characteristic emission lines. This fluorescence can radiate either the ROC or the spectrometer, depending on which one we are using.

3.1 X-ray tube and flourescent targets

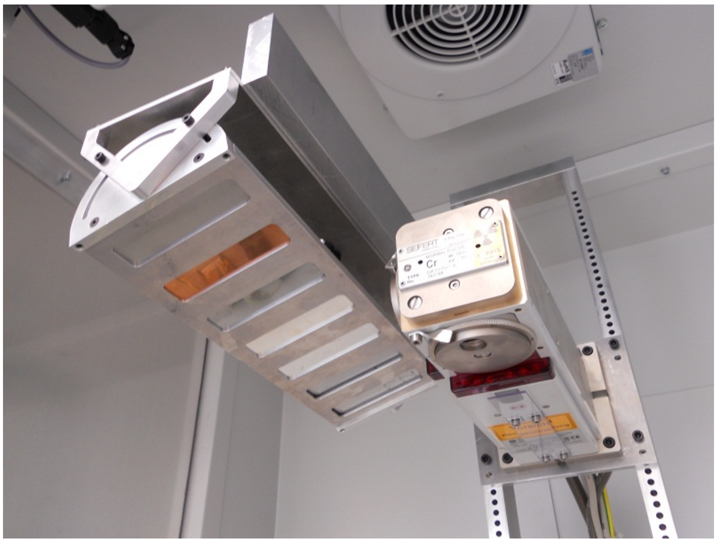


Figure 3.2: Left: Target Array. Right: X-ray tube.

An X-ray tube is a vacuum tube that produces X-ray radiation. The X-ray tube we use is an Analytical X-ray tube, SEIFERT DX-Cr10x1-S. In the X-ray tube, there is a cathode which emits electrons into the vacuum and an anode which collects the electrons, establishing an electrical current in the tube. In this case, by applying a DC voltage between cathode and anode up to 60 kV, electrons are accelerated. The X-ray spectrum depends on the anode material, which in this case is Chromium, and on the accelerating voltage. As the voltage can be up to 60 kV, electrons can have an energy up to 60 keV. The maximum power that the X-ray tube can sustain is 1800 W.

Targets have an inclination angle of 15°.

The electrons fired at the Chromium anode are decelerated, generating Bremsstrahlung continuum radiation. We use this Bremsstrahlung radiation to radiate the targets. It may happen that some electrons eject electrons from the anode, leaving energy vacancies. When electrons at higher energies fill these vacancies, energy is emitted from the anode in characteristic lines determined by the anode material.

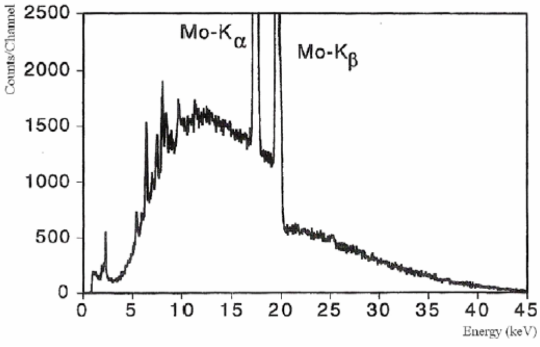


Figure 3.3: As an example, we can see the K-α and K-β lines of Mo superimposed on Bremsstrahlung continuum. Source: $www.cnstn.rnrt.tn/afra-ict/NAT/xrf/html/instrumenta_sec3.htm$

When materials are radiated, the radiation can be energetic enough to expel tightly held electrons from the inner orbitals of the atom. The electron removal rendes the electronic structure of the atom unstable, so electrons in higher orbitals fall into the lower orbital to minimize the energy of the electronic configuration. In falling, energy is released as photons whose energy is equal to the energy difference of the two orbitals involved. Thus, materials radiate flourescent emission lines in certain wavelengths characteristic of the atoms.

The most prominent emission lines to which the ROC is sensitive are the K-α and K-β lines of the targets. We use a few targets to radiate the ROC with flourescent lines of different energies and then find the VthrComp associated with the threshold at which the K- lines start to activate the pixels.

We use photons of Copper, Molybdenum, Silver, Tin and Baryum. In table 3.1, the K-α and K-β line energies for all the targets are given.

	Cu	Mo	Ag	Sn	Ba
K-α_1 [keV]	8.047	17.479	22.162	25.271	32.193
K-α_2 [keV]	8.027	17.374	21.990	25.044	31.817
K-β_1 [keV]	8.905	19.608	24.942	28.486	36.378
K-β_2 [keV]	—	19.965	25.456	29.109	37.257
K-β_3 [keV]	8.905	19.590	24.911	28.444	36.304

Table 3.1: K-α and K-β energies for several targets. Source: X-Ray Data Booklet Table $1-2$.

3.2 Readout Chip and Testboard

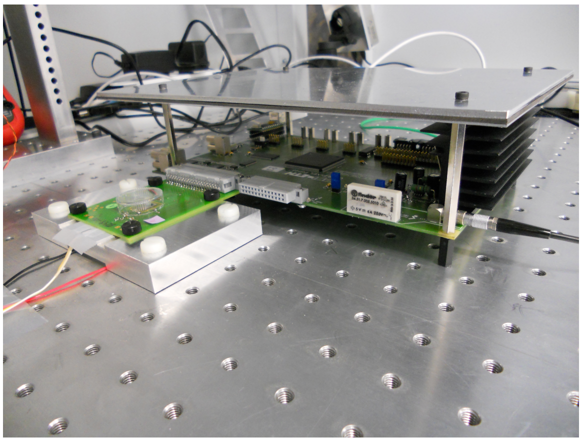


Figure 3.4: ROC in the left connected to testboard in the right. The ROC is on a cooling plate.

The description of the ROC can be found in section 2.2 *The Readout Chip*.

The ROC is mounted on a cooler structure which has a peltier element, which makes sure to keep the ROC in an working and stable temperature of 17 C.

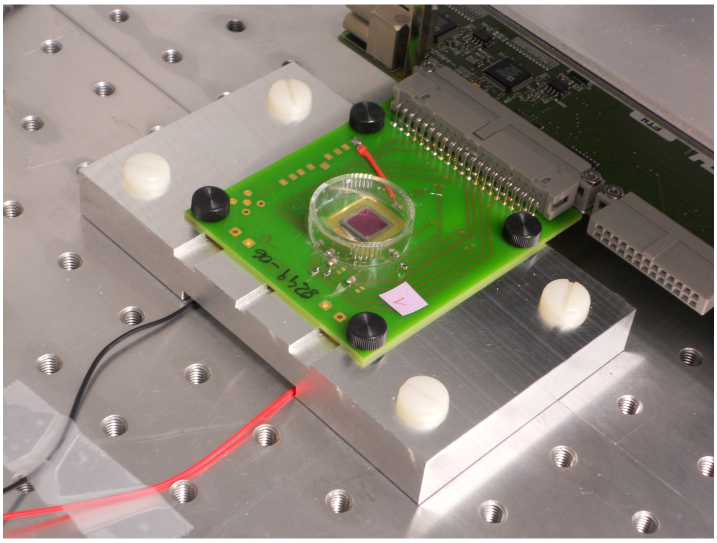


Figure 3.5: ROC connected to testboard and mounted on a peltier element.

We use the testboard to communicate with the ROC. The testboard is connected to a computer via USB, and the ROC in turn is connected to the testboard. We use a custom software called *psi46expert* to control the ROC via the testboard. By using psi46expert, we can run several tests such as Pretest, trimming, X-ray test, Vcal Threshold Map, DacDac, among others, and set the DAC parameters of the ROC to a chosen value.

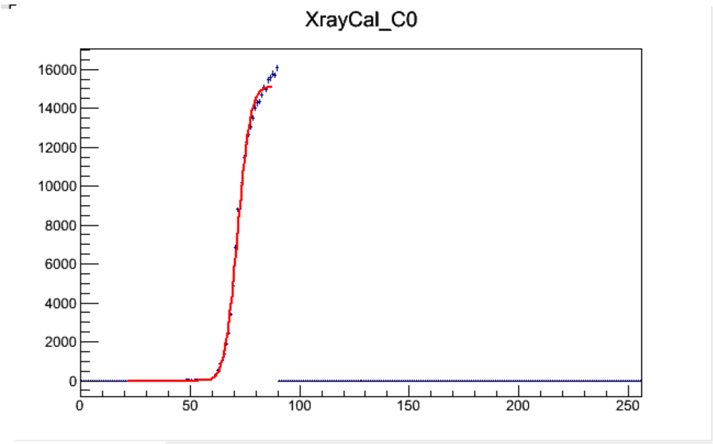


Figure 3.6: *Psi46expert* interface running a X-ray test for Molybdenum.

3.3 Spectrometer

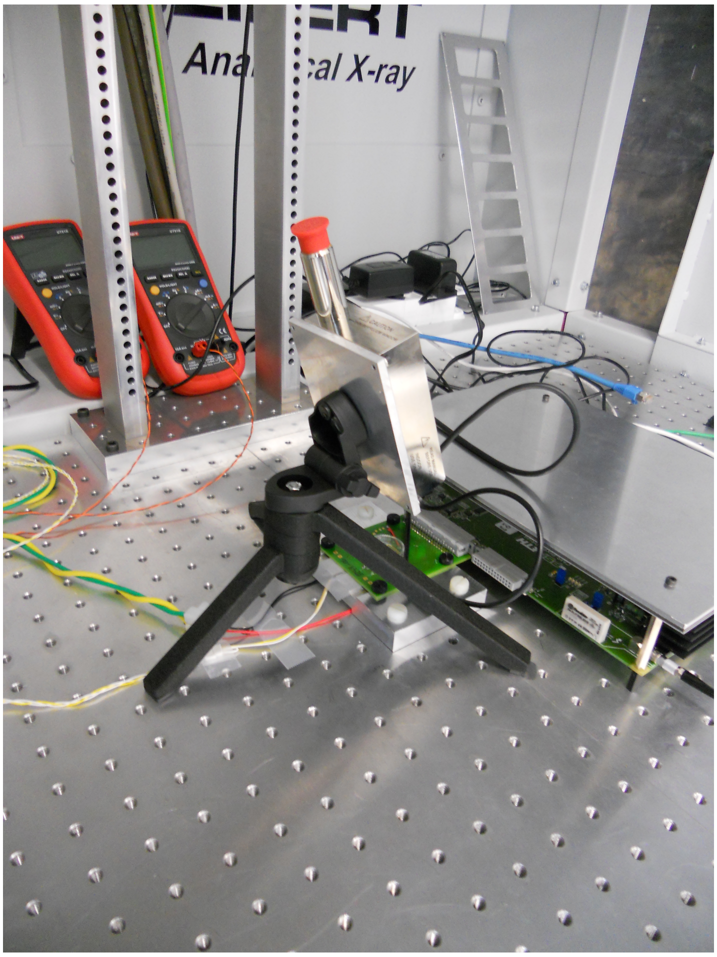


Figure 3.7: Spectrometer in laboratory setup pointed toward the targets.

The purpose of the spectrometer is to analyse the spectra from the targets and make sure that the incoming radiation that we are going to use for the ROC is free from noise and any strage behaviour.

The X-123 X-ray spectrometer consists of several devices which accomplish different functionalities. Such components are the XR-100CR, consisting of a sensor, the x-ray detector, a preamplifier and a cooler, and the Digital Pulse Processor DP5 which analyses the signals.

3.3.1 Devices

XR-100CR

The XR-100CR is a **x-ray detector**, **preamplifier** and **cooler** system which uses a thermoelectrically cooled Si-PIN photodiode as X-ray detector.

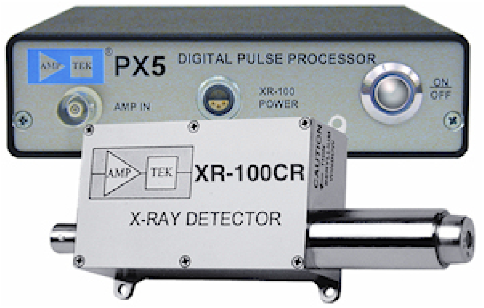


Figure 3.8: X-123 X-ray detector system

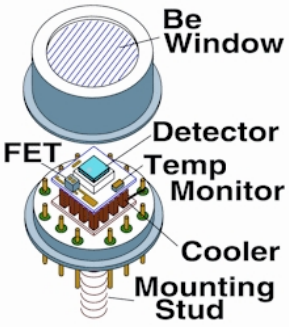


Figure 3.9: Si-PIN photodiode detector, cooler and Be window

X-rays interact with silicon atoms creating an average of one electron-hole pair for every 3.62 eV of energy deposited into the silicon detector. The energy deposition is dominated by either Photoelectric effect or Compton scattering depending on the energy of the incoming X-ray radiation.

In order to lower the system noise of the detector, a high bias voltage is applied across the silicon to decrease the capacitance of the detector. A high bias voltage (of 100 – 200V depending on the detector thickness) facilitates as well the collection of electron-hole pairs. However, this bias voltage range at room temperature causes excessive leakage and eventually breaks down. That is why a cooling system is necessary, bringing the detector to a working temperature of about 17 C.

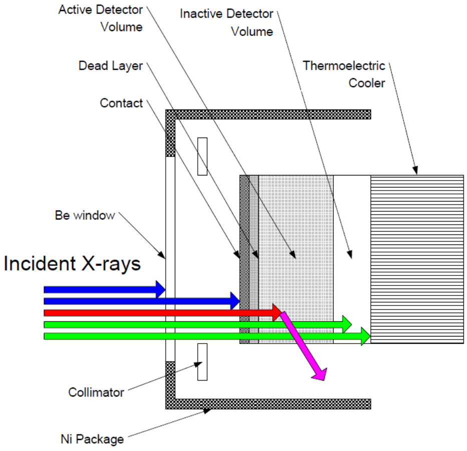


Figure 3.10: Squeme of X-ray interaction

For a photon incident on the window of the detector, heading towards the detector's active volumen, it can happen:

- The photon may interact before reaching the detector (blue arrows in fig. 3.10), either in the Be window, in a contact material or in a dead layer at the top of the detector. Interactions in these layers are responsible for the loss of efficiency at low energies.
- The photon may interact in the detector's active volume (red arrows), producing a signal. The photoelectric effect is dominant at low energies of about 100eV - 10keV. For these photons, photoelectric interaction results in total energy deposition, contributing to the photopeak. For higher energies, 10kev - 100keV, Compton scattering is dominant, so here X-ray may scatter from the active volume or a secondary particle may escape the active volume (pink arrows), depositing less than the full energy en the detector.
- The photon may pass through the active volume without interacting (green arrows). This leads to the loss of efficiency at high energies.

Figure 3.11 shows the efficiency, combining the effects of transmission through the Be window (including the protective coating anti reflection), and the probability of a photon undergoing photoelectric interaction or Compton scattering.

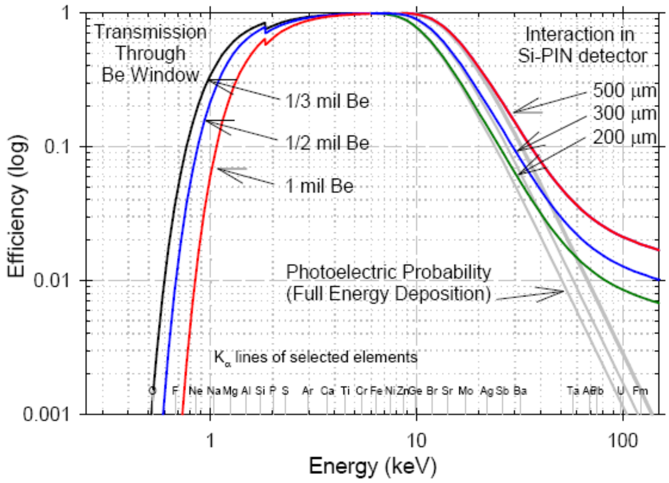


Figure 3.11: Efficiency

Be window The beryllium (Be) window has two main functions: (1) vacuum isolation; (2) absorption of a significant fraction of the unused low-energy radiation, which reduces the overall power delivered to downstream components.

The thickness of the Be window is of 25μm. The low energy range in fig. 3.11 is dominated by the thickness of the Be window. For thickness of 12.5-25μm, around 90% of the incident photons reach the detector at energies of $2-3$KeV.

Si-PIN photodiode detector A Silicon PIN photodiode is a device consisting of a thin p-doped Si transparent contact layer, an undoped Si absorbing layer (with a width larger than the absorption length of X-rays in Si) and then a n-doped Si contact layer. X-rays are absorbed in the depletion region of the diode, then by the action of the built in electric field electron-hole pairs are separated contributing to the photocurrent. A very small voltage is necessary to deplete the undoped region, ensuring a short transit time.

The detector area size is 25 mm^2 (collimated to 21.5 mm^2, the silicon thickness is 500μm, its peaking time is 32 μs and its Peak to Background (P/B) ratio is 2000/1.

The detector has an internal **MultiLayer Collimator** made by progressively using lower Z materials, so that each layer absorbs the fluorescence peaks of the previous layer. The final layer has a low enough Z, so its fluorescence peaks are energetically outside of the X-ray detection range. The ML collimator consists of a base metal of 100μm of tungsten (W), a first layer of 35μm of chromium (Cr), a second layer of 15μm of titanium (Ti) and a last layer of 75μm of aluminium (Al).

The ML collimator is used to improve the spectral quality of X-rays. X-rays interacting near the edges of the active volume of the detector may produce small secondary pulses due to partial charge collection, pulses that may obscure the signal of interest. The collimator restricts X-rays to the active volume, producing clear signals. The ML collimator improves the P/B ratio, eliminates edge effects and eliminates falses peaks.

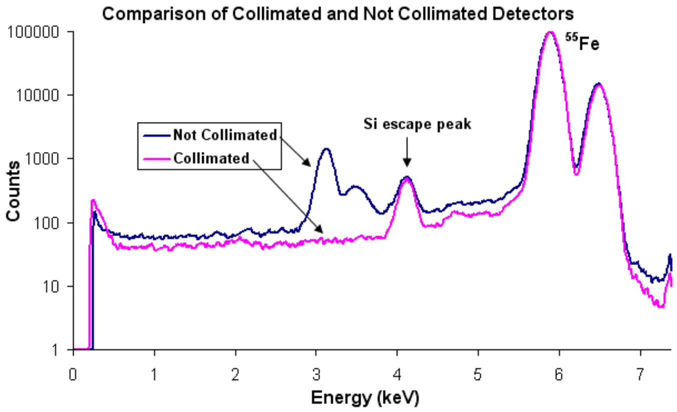


Figure 3.12: Collimating effects

2-Stage thermoelectric cooler The cooler can generate a maximum temperature differential on the detector of $75 - 85$C. At a room temperature of 25C, the lowest possible temperature would be 220K. Temperature stability is important, since if the detector temperature varies (which is often the case when set to full cooling), then the gain, noise and offset will vary, affecting the spectrum integrity.

Digital Pulse Processor DP5

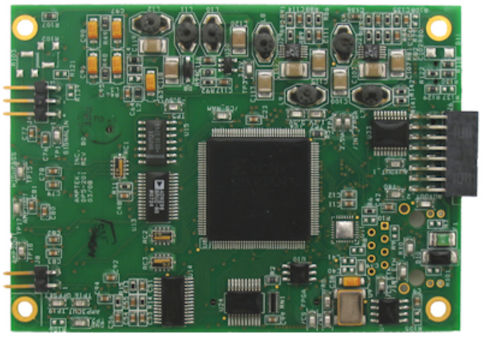


Figure 3.13: Digital Pulse Processor DP5

The charge liberated by a X-ray photon appears at the output of the preamplifier as a voltage step (see 1) of fig.3.14) on a linearly increasing voltage ramp. The role of the digital pulse processor (which shapes digitaly the pulse) is to accurately measure the energy of the incoming X-ray, and give it a digital number which then adds a count to the corresponding channel in the computer software.

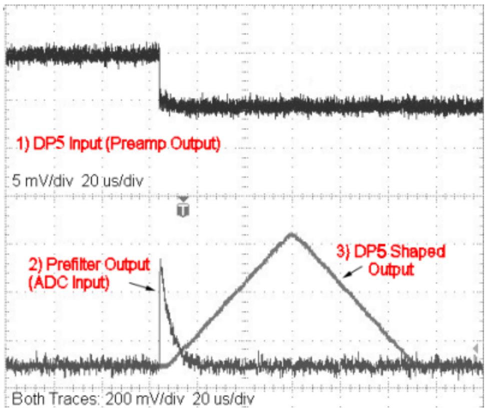


Figure 3.14: Preamp Output, Prefilter Output and DP5 Shaped Output

DP5 digitizes the preamplifier output signals. The process is as follows (see fig. 3.14, fig. 3.16): the preamplifier output is passed as a step voltage to an analog prefilter, which is a high-pass-filter. The prefilter output is continuously sampled by an ADC (Analog to Digital Converter) and then processed digitally by a digital pulse shaper, producing a DP5 shaped output, which is handed to the computer multichannel analizer (MCA).

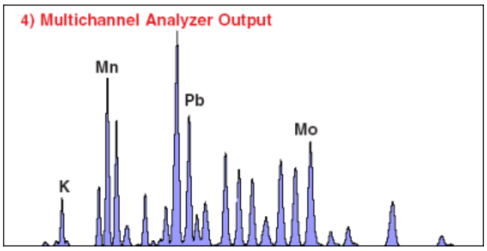


Figure 3.15: MCA Output

DP5 arquitecture

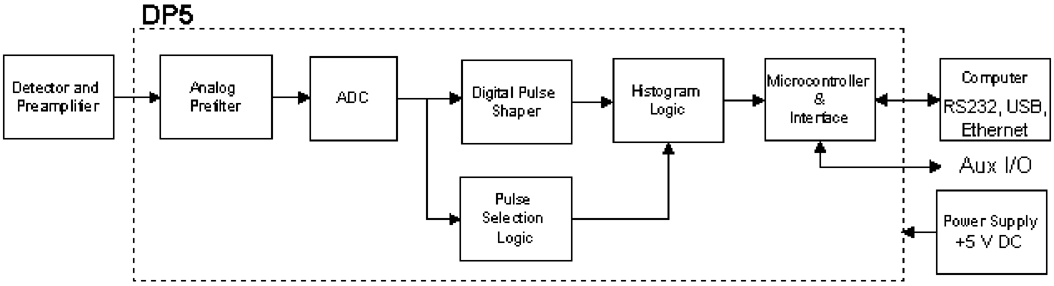


Figure 3.16: DP5 arquitecture

As we said before, the input to the DP5 is the preamplifier output. The DP5 digitizes the preamplifier output, aplies real-time digital processing to the signal, detects digitally the peak amplitude, and bins this value in its histogramming memory generating an energy spectrum. The spectrum is then transmitted over the DP5's serial interface to the computer.

Analog Prefilter The input to the DP5 is the outpu of a charge sensitive preamplifier. The analog prefilter circuit: (1) applies appropriate *gain* and *offset* to use the dynamic range of the ADC; (2) performs *filtering* and *pulse shaping* functions to optimize the digitization.

ADC The 12 bit ADC digitizes the output of the analog prefilter at a 20 or 80 MHz (software selectable). The digitized values are sent, in real time, into the digital pulse shaper.

Digital Pulse Shaper The ADC output is processed continuously generating a real time shaped pulse (the shaped pulse is a purely digital entity). There are two parallel signal processing paths inside the DPS (and shaping amplifiers in general): the **fast** and **slow channels**. The slow channel, which has a long shaping time constant (or peaking time), is optimized to minimize electronic noise and ballistic deficit, in order to obtain an accurate pulse height: the peak value for each pulse in the slow channel is the primary output of the pulse shaper. The fast channel, which has a short shaping time constant, is optimized to obtain timing information as detecting pulses which overlap in the slow channel (pile-up) to then reject them, measuring the **input count rate** $\mathcal{R}_{in}$, measuring pulse risetimes, etc. The DP5 uses trapezoidel pulse shaping, which presents high energy resolution, reduces ballistic deficit and provides excellent baseline stability at high count rates.

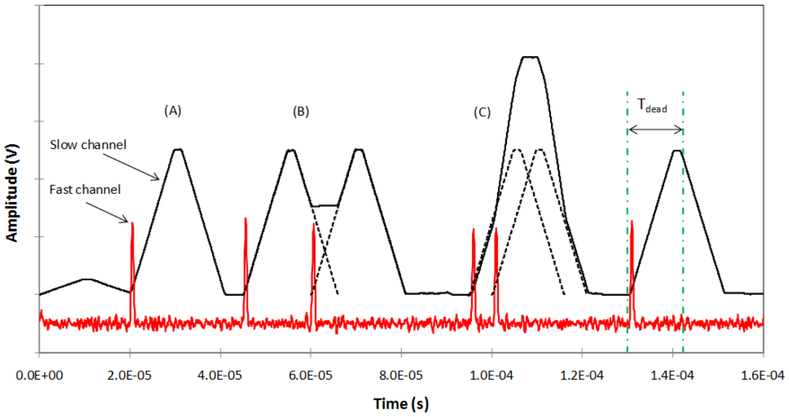


Figure 3.17: Fast and Slow channels

Since radiation interacts in the detector at random invervals, it is possible to have two interactions occurring within the processing time of the slow channel. When pulses overlap in time: (1) only one pulse is recorded instead of two; (2) the detected peak has an incorrect amplitude. The fig. 3.17 shows: *(A)* a single non-overlapping pulse counted correctly and with the correct amplitude; *(B)* two pulses overlapping somewhat but with non-overlapping peaks, so two pulses are counted with accurate peaks; *(C)* two pulses overlapping: in the slow channel only a single event is recorded with an incorrect pulse height. This problem is solved with the fast channel of much shorter peaking time (with minimum possible time of 50ns). Thanks to its short peaking time, many pulses that overlap in the slow channel can be distinghuished, allowing to measure the input count rate, where far fewer pulses are rejected due to overlap, and allowing as well to pile-up reject pulses in the slow channel (PUR, Pile-Up Rejection). Thus, with PUR enabled, (C) event is rejected and not recorded. Neither event is recorded in the slow channel (**total counts**) but both are recorded in the fast channel (**input counts**).

For most detectors electronic noise is minimized at long peaking times. That is why the fast channel has more rms noise than the slow one. Separate thresholds are used in both channels, setting them above the respective noises. For the XR100 Si-PIN detector the minimum noise is found with a peaking time of 25μs.

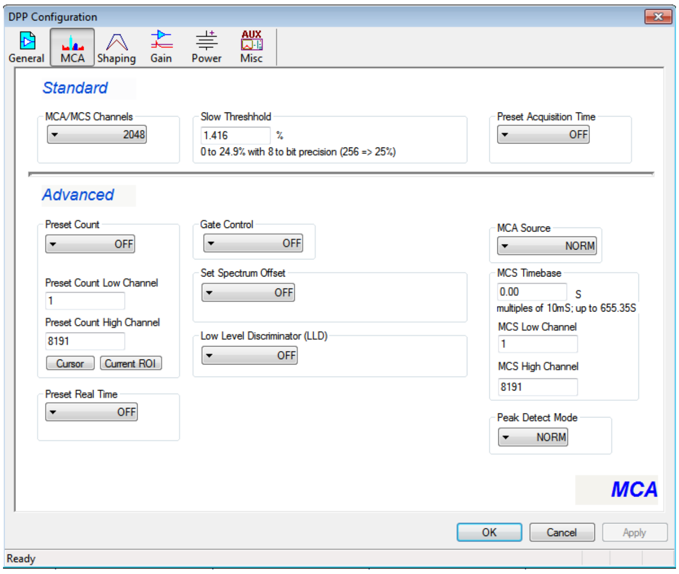
Pulse Selection Logic It rejects pulses for which an accurate measurement cannot be made, by using pile-up rejection, risetime discrimination, etc.

Histogramming Memory When a pulse occurs with a particular peak value, a counter in a corresponding memory location is incremented, resulting in a histogram which contains the number of events with the corresponding peak value. This is the energy spectrum and is the primary output of the DP5.

Interface The DP5 includes hardware and software to interface between these various functions and the user's computer. The interface transmits the spectrum to the user, controls data acquisiton by starting and stopping the processing and by clearing the histogram memory, and controls certain aspects of the analog and digital shaping.

3.3.2 Digital Pulse Processor parameters

DPP MCA configuration



Standard

MCA/MCS Channels: 2048

Slow Threshhold: 1.416 %
0 to 24.9% with 8 to bit precision (256 => 25%)

Preset Acquisition Time: OFF

Advanced

Preset Count: OFF

Preset Count Low Channel: 1

Preset Count High Channel: 8191

Cursor | Current ROI

Preset Real Time: OFF

Gate Control: OFF

Set Spectrum Offset: OFF

Low Level Discriminator (LLD): OFF

MCA Source: NORM

MCS Timebase: 0.00 S
multiples of 10mS; up to 655.35S

MCS Low Channel: 1

MCS High Channel: 8191

Peak Detect Mode: NORM

Figure 3.18: DPP MCA parameters

- **MCA/MCS Channels** determines the number of channels in the spectrum, ie, the number of bins in the histogram. It can be set to 256, 512, 1024, 2048, 4096 or 8192 channels.
- **Slow Threshold** sets the noise threshold for the slow channel. It's used to distinguish real pulses from noise fluctuations in peak detection logic.
- **Preset acquisition time** Data acquisition stops when the *acquisition time* (see below) reaches this value. If OFF, acquisiton is stopped manually.
- **Preset Count Low/High Channel** set the upper and lower channels in a region of interest. Only events occurred in $PRCL < channels < PRCH$ are counted.
- **Preset Count** Data acquisition stops when the number of counts between the low and high channels reaches this value.
- **Preset real time** Data acquisition stops when the *real time* (see below) reaches this value.
- **Gate control** permits to enable an external gate, to synchronize data acquisition with external logic. If the Gate signal is active ((A gate is a logic signal, what else to say??)) at the time the peak is detected, the pulse is rejected or accepted, depending on setting (polarity of the gate input). ((What this 'external logic' is necessary for??)) Ask Marco.
- **Set spectrum offset** shifts the zero channel of the pulse height spectrum. A positive offset shifts the spectrum up (i.e. higher in energy). It can be used to set channel zero equal to zero energy or to shift zero energy high enough to observe the noise Gaussian.

- **Low Level Discriminator** Since the fast threshold is higher than the slow threshold (see below), small pulses may be measured in the slow channel but not in the fast channel, giving a slow channel count rate exceeding the fast channel count rate. The LLD is an independent threshold in the slow channel higher than the slow threshold, which rejects low amplitude pulses that are not measured in the fast channel. Shaped pulses must have peak amplitude above both the LLD and the slow threshold to be recorded in the spectrum (slow channel).
- **MCA source** *NORM* provides normal MCA operation, giving an energy spectrum using the shaped channel. *MCS* provides multichannel scaler operation, yielding counts versus time, so each bin in the histogram represents the number of counts in a time interval.
- **MCS timebase** sets the timebase used by the Multi-Channel scaler (MCS). This is the duration of each 'channel' in the MCS histogram.
- **MCS low/high channel** sets the upper and lower channels in a region of interest counted in the MCS mode.

Real time is the actual time between starting and stopping a measurement. As the digital processor stops acquiring data (in the slow channel) for brief intervals in which no counts can be measured, dividing counts by the real time does not give the correct count rate. Data acquisition is paused: (1) for about 113μs to 2.5ms (depending on how many MCA channels are used) when the spectrum is transferred from the FPGA memory ((What the hell is this?)) into the microcontroller ((?)); (2) for about $2 - 20$ms (depending on MCA channels) when the spectrum is sent to the computer; (3) following each preamplifier reset (duration of pause set by the reset lockout parameter. **Acquisition time** (or **accumulation time**) is the time during which counts can be measured, excluding the previous time pauses but not correcting for overlapping pulses, i.e. the dead time per pulse.

Dead time per pulse is the minimum time that must separate two interactions to allow them to be distinguished from each other and then recorded as two separate events (this time has as lower limit the shortest possible fast peaking time): even the fast channel has count losses (dead time losses). In digital processors, the dead time per pulse is not well defined, but fortunately their fast shaping channels have in general low dead time losses.

Total count is the total number of pulses accepted in the spectrum using the slow channel. The pulse must (1) have a peak height exceeding both the slow threshold and the LLD; (2) be separated in time from other pulses by more than the dead time per pulse; (3) not have been rejected by PUR, RTD or GATE logic. **Total count rate**, or **output count rate** $\mathcal{R}_{out}$, is the total counts divided by the acquisition time.

Input count is the total number of counts measured using the fast channel. **Input rate** $\mathcal{R}_{in}$ is an estimate of the rate of radiation interaction and is equal to the fast channel count rate $\mathcal{R}_{fast}$ which is the input counts divided by the acquisition time.

Dead time is the time in the experiment during which data acquisition in the slow channel is paused and no counts can be measured. To get rid of the count errors introduced by the dead time per pulse (even the fast channel cannot resolve two pulses separated by a shorter time than the dead time per pulse), we defined the dead time as the following estimate

$$DT = \frac{\mathcal{R}_{fast} - \mathcal{R}_{slow}}{\mathcal{R}_{fast}}$$

which includes count losses from all sources (pulse overlap and data acquisition pauses).

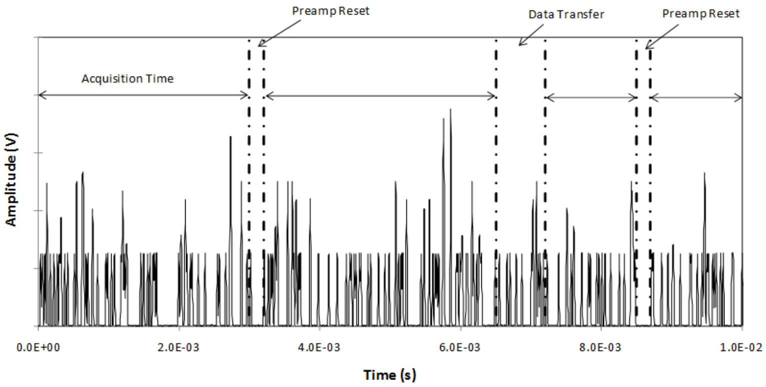
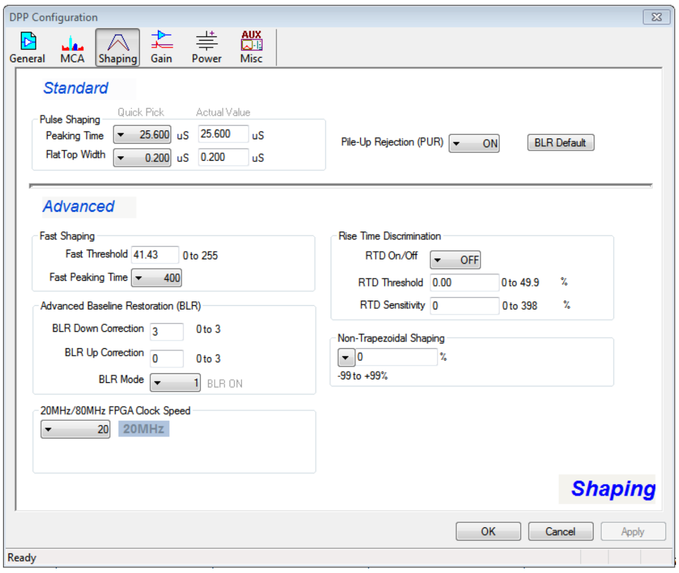


Figure 3.19: Acquisition time

DPP shaping configuration



Pulse Shaping

	Quick Pick		Actual Value	
Peaking Time	25.600	uS	25.600	uS
FlatTop Width	0.200	uS	0.200	uS

Pile-Up Rejection (PUR) ON

Fast Shaping

Fast Threshold 41.43 0 to 255

Fast Peaking Time 400

Advanced Baseline Restoration (BLR)

BLR Down Correction 3 0 to 3

BLR Up Correction 0 0 to 3

BLR Mode 1 BLR ON

20MHz/80MHz FPGA Clock Speed

20 20MHz

Rise Time Discrimination

RTD On/Off OFF

RTD Threshold 0.00 0 to 49.9 %

RTD Sensitivity 0 0 to 398 %

Non-Trapezoidal Shaping

0 %

-99 to +99%

Figure 3.20: DPP shaping parameters

- **Peaking time** sets the time to peak of pulses in the slow channel, i.e. time in which a shaped pulse goes from the **baseline** to the peak. This parameter is used for pulse height analysis in the main spectrum and can be set to a high precision typically of about 2%.

- **Flat top width** $\mathcal{T}_{flat}$ sets the flat top duration for the trapezoid (form of shaped pulse) in the slow channel, which should be longer than the maximun charge collection time in the detector. For our XR-100CR, a value of $> 0.2\mu$s is suggested to eliminate ballistic deficit in the fast channel.

- **PUR** Pile-Up Rejection is used to reject from the stored spectrum pulses in which the peaks are distorted due to pulses overlapping in time.

- **Fast peaking time** sets the time to peak of pulses in the fast channel, which is used to detect overlapping pulses and to measure $\mathcal{R}_{in}$. Setting this parameter short allows detection of events closer in time but increases the electronic noise, which requires a higher threshold in the fast channel (usually it is higher than the slow threshold). Fast channel has minimal flat top, so ballistic deficit is significant.

- **Fast threshold** sets the noise threshold for the fast channel, which should be set just above the noise to distinguish pulses from noise fluctuations. If the fast threshold is too low and so triggers on noise, every real pulse will be followed by a false noise pulse, overlapping in the slow channel. If besides PUR on, this will lead to rejection of every signal.

- **BaseLine Restoration Mode** MCA measures and analizes the input pulse amplitudes with respect to an internal reference voltage or baseline voltage to which signals at the output of the shaper amplifier (or equivalently at the input of the MCA) are superimposed. The baseline is not stable due to **shot noise** (statistical fluctuations of current through junctions), **thermal noise** in devices as preamplifier (in its feedback resistor) or input FET, variation of radiation rate (a count rate too high could make trouble with pile-up), etc. Fluctuations of baseline affect the energy resolution of hight-rate spectroscopy experiments, because pulse amplitudes are measured by substracting the baseline amplitude of the signal from the signal pulse itself. The BLR circuit in the DPP keeps stabilized the baseline of a shaped pulse. BLR typically uses active feedback, where an error amplifier samples only the baseline (the shaped output between valid signals) and produces the correction signal necessary to hold it constant.

  Set this parameter as "1" (valid under most conditions).

- **BLR down correction** sets the slew rate of the downward correction in the BLR feedback.

- **BLR up correction** sets the slew rate of the upward correction in the BLR feedback.

- **FPGA clock speed** What does FPGA stand for? what the hell is it????? Ask Marco.

- **RiseTime Discrimination on/off**

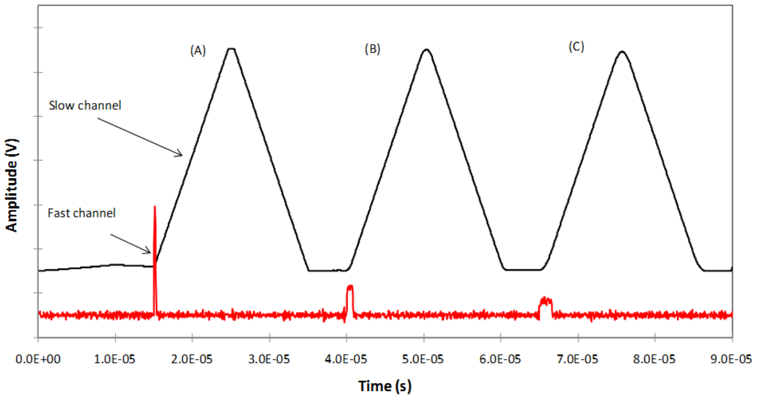


Figure 3.21: Risetime and ballistic deficit in fast channel

The risetime of a pulse (out of the preamplifier) is the time during which the charge is collected in the detector electrodes (charge collection time, *cct*). Figure 14 shows the simulation of the response to three different pulses whose deposited energies are the same but charge collection times are 0.2μs, 0.8μs and 1.6μs in (A), (B) and (C) respectively. The top of the trapezoid pulse goes as $\mathcal{T}_{flat} - cct$. As $\mathcal{T}_{flat}$ is set to 0.8μs in the slow channel, the peak amplitudes for (A) and (B) are the same but slightly lower for (C), underestimating its amplitude. In the fast channel, however, the peak amplitude depends strongly on the charge collection time, so the deposited energy is strongly underestimated for pulses with long charge collection time (here, $\mathcal{T}_{fast} = 0.1\mu$s). The dependence of the peak amplitude on charge collection time is called **ballistic deficit**, only ocurring when $cct > \mathcal{T}_{flat}$ for trapezoidal shaper.

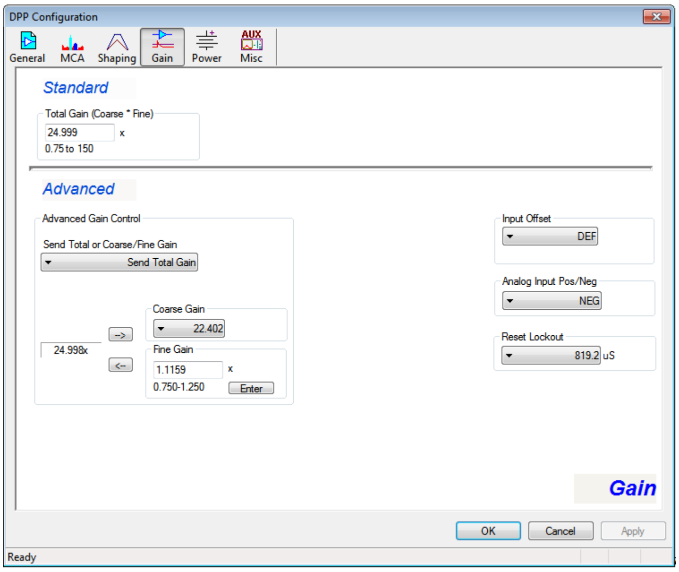
This is the basis of the RTD in the DPP: RTD ratio is the ratio of the peak slow channel amplitude to the peak fast channel amplitude for a given pulse. As the risetime gets slower, the fast channel response falls off faster than the slow channel, so the ratio increases.

- **RTD sensitivity** An event is rejected if the ratio is above the sensitivity setting.
- **RTD threshold** If the peak slow channel amplitude is below the RTD threshold, the event will be accepted regardless of the sensitivity setting.

Baseline of a pulse is the instantaneous value that the voltage would have had at the time of the pulse peak in the absence of that pulse. Pulse heights are measured with respect to the baseline, which is in general not zero. In general, zero energy (no pulse) in the spectrum does not correspond to channel zero, but corresponds to a channel number comparable to the rms value of the electronic noise which determines the baseline. This channel number is the **offset** of the spectrum. Notice that the thresholds (in channels) must be placed above both the noise and the offset (noise rms).

Shot noise is electronic noise arising from statistical fluctuations in the current through a juction, forming a current or parallel noise source. Devices with this noise are, for example, Si PIN photodiode, input FET, etc. **Thermal noise** is electronic noise arising from the random thermal vibrations of charge carriers in a conductor. Thermal noise takes place in the channel of the input FET as a voltage or series noise source, and in the feedback resistor of the preamplifier as a current or parallel noise source.

DPP gain configuration



DPP Configuration

Standard

Total Gain (Coarse * Fine): 24.999 x
0.75 to 150

Advanced

Advanced Gain Control

Send Total or Coarse/Fine Gain: Send Total Gain

24.998x --> <--

Coarse Gain: 22.402

Fine Gain: 1.1159 x
0.750-1.250 Enter

Input Offset: DEF

Analog Input Pos/Neg: NEG

Reset Lockout: 819.2 uS

Gain

Ready

Figure 3.22: DPP gain parameters

- **Total gain** Gain is the ratio of the output of a system to its input. There are two distinct gains: the **system conversion gain** and the **digital processor gain**. The system conversion gain relates the system input, the energy deposited by a particle, to the system output, the corresponding channel number in the output histogram. So, it has eV/channel units. This gain is used when performing an energy calibration. Given a deposited energy E in the detector, the output channel C is

  $$C = \left(\frac{E}{\epsilon}\right)\left(\frac{1}{q \cdot C_f}\right)(\mathcal{G}_{shape}\mathcal{G}_{DPP})\left(\frac{N_{chan}}{V_{max}}\right)$$

  where ϵ is the energy required to produce an electron-hole pair, q is the electron charge, C_f is the feedback capacitance of the charge sensitive preamplifier, $\mathcal{G}_{shape}$ is related to the pulse shaping network (influenced by peaking time, flat top time, etc) and to the time profile of the detector current, N_{chan} is the number of channels for the MCA, V_{max} is the voltage corresponding to the highest MCA channel, and $\mathcal{G}_{DPP}$ is the voltage gain of the digital processor or digital processor gain.

  $\mathcal{G}_{DPP}$ is the product of the **coarse gain** and the **fine gain**.

- **Coarse gain** is an analog gain set by resistors and operational amplifiers (op-amps). There is a list of 16 fixed coarse gain values for our DP5 processor.

- **Fine gain** is a digitally controlled gain. The fine gain is combined with a normalization for fine gain variations due tu peaking time variations, since fine gain is depedent on peaking time. Gains depends on the temperature of the detector, of op-amps, etc.

When the total gain is set, first the nearest available coarse gain for the analog front end is chosen, then the necessary fine gain to achieve the commanded total gain is calculated while also normalizing for gain differences caused by peaking time variations.

- **Sent Total or Coarse/Fine gain** It is possible to send either coarse, fine or total gain.

- **Input offset** sets a DAC which controls the DC offset at the input to the ADC, shifting it so that it is kept within the ADC input range (0-2V). Generally, the default setting can be used (the offset set at the high-pass-filter before the DAC), since the DP5 has an AC-coupled signal input which removes DC voltage signals from the preamplifier output (high-pass-filter).

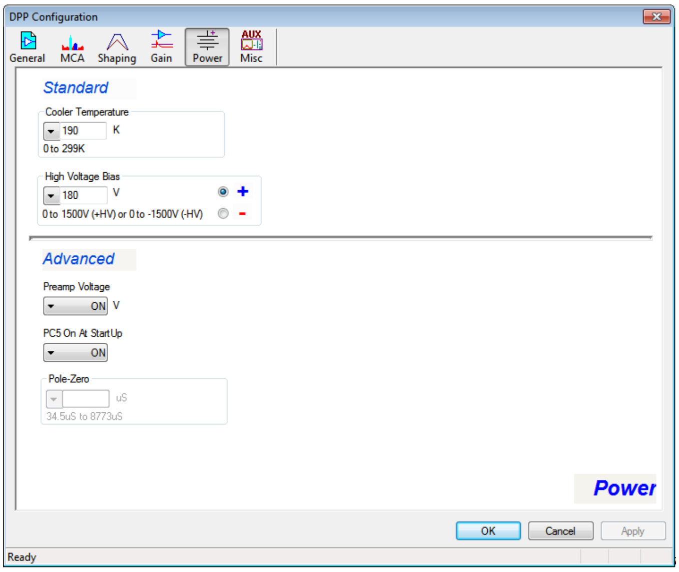
- **Analog input pos/neg** selects the polarity of the pulses at the DP5 signal input. The charge sensitive preamplifier (associated with our Si-PIN detector) produces negative steps for each x-ray interaction (step polarity depends on how the detector is biased and coupled to the preamplifier), so the input polarity of the Si-PIN detector is negative. Pulses of the wrong polarity (or with an incorrect input offset) will be outside the range of the ADC and therefore will not be measured. The Si-PIN detector is constructed in such way that its Hight Voltage polarity (the polarity of the bias) and its input polarity are opposite. So, the HV polarity of the Si-PIN detector is positive (HV polarity is set in hardware, since a wrong polarity in the detector damages it).

- **Reset lockout** When the preamplifier resets (in a reset style preamp), the analog electronics may be saturated and may take some time to recover fully. Reset lockout sets a time during which data acquisition is gated off, after each reset pulse. For preamplifiers with continuous feedback, the lockout should be turned off ((Why??)). For our Amptek Si-PIN detector a setting of 409.6μs is recommended. As said before, during the lockout no counts are recorded neither in the fast channel nor in the slow channel, so the acquisition time clock is stopped. The SCAs still produce output pulses ((WHY??)). If the lockout interval is set too short then some pulses with a distorted amplitude may be measured (often causing a non-Gaussian distortion to a photopeak). If it is too long and count rate is high, then data acquision may be very slow because it is stopped during reset lockout after each reset pulse, which follows in turn a pulse.

DPP power configuration

- **Cooler temperature** Temperature to which detector is cooled. The lowest possible temperature (full cooling) will give the lowest noise and the best resolution. With a maximum temperature differential of about $75 - 85$C, at a room temperature of 25C, the lowest possible temperature would be 220K.

- **High Voltage bias** is the bias voltage applied by the HV power supply to the Si-PIN detector to deplet the I-type region of the PIN photodiode and produce an electric field to sweep out the signal charge. The bias voltage must be above the full depletion voltage of the photodiode and lower than the breakdown voltage. Detector operation and performance change slowly with bias within this range. The Si-PIN detector typically operates between +100 and +250V.



Standard

Cooler Temperature
190 K
0 to 299K

High Voltage Bias
180 V ● + ○ -
0 to 1500V (+HV) or 0 to -1500V (-HV)

Advanced

Preamp Voltage
ON V

PC5 On At StartUp
ON

Pole-Zero
uS
34.5uS to 8773uS

Power

Figure 3.23: DPP power parameters

- **High Voltage polarity** The polarity of the HV bias is set in hardware by jumpers, which are soldered in the PC5. The DPP compares the hardware polarity configuration to this software parameter. If they do not match, the HV supply is disabled to prevent damage to the detector.

- **Preamp voltage**turns on or off the PC5 preamp power supplies. For the PC5, the preamp voltage is fixed in hardware.

- **PC5 on at startup** The PC5 power supply oparates on the preamplifier, DPP, thermoelectric cooler and the detector. It has a nominal input of +5VDC with a current of 500mA (2.5W), which depends on temperature varying between 300 – 800mA. The range of the input is 4 – 6VDC. The PC5 uses switching supplies to produce all the low voltages required for the DPP and the preamplifier. It also includes a HV multiplier to produce the detector bias, up to 400V, and gives supply for the thermoelectric cooler. This parameter determines whether the PC5 power supplies for the preamplifier, detector (HV bias) and thermoelectric cooler will be turned on automatically when the PC5 is turned on. If it is on, as soon as PC5 is turned on, power is applied to the preamp, HV and TEC.

3.3.3 Tuning settings

Description of the measurement problem

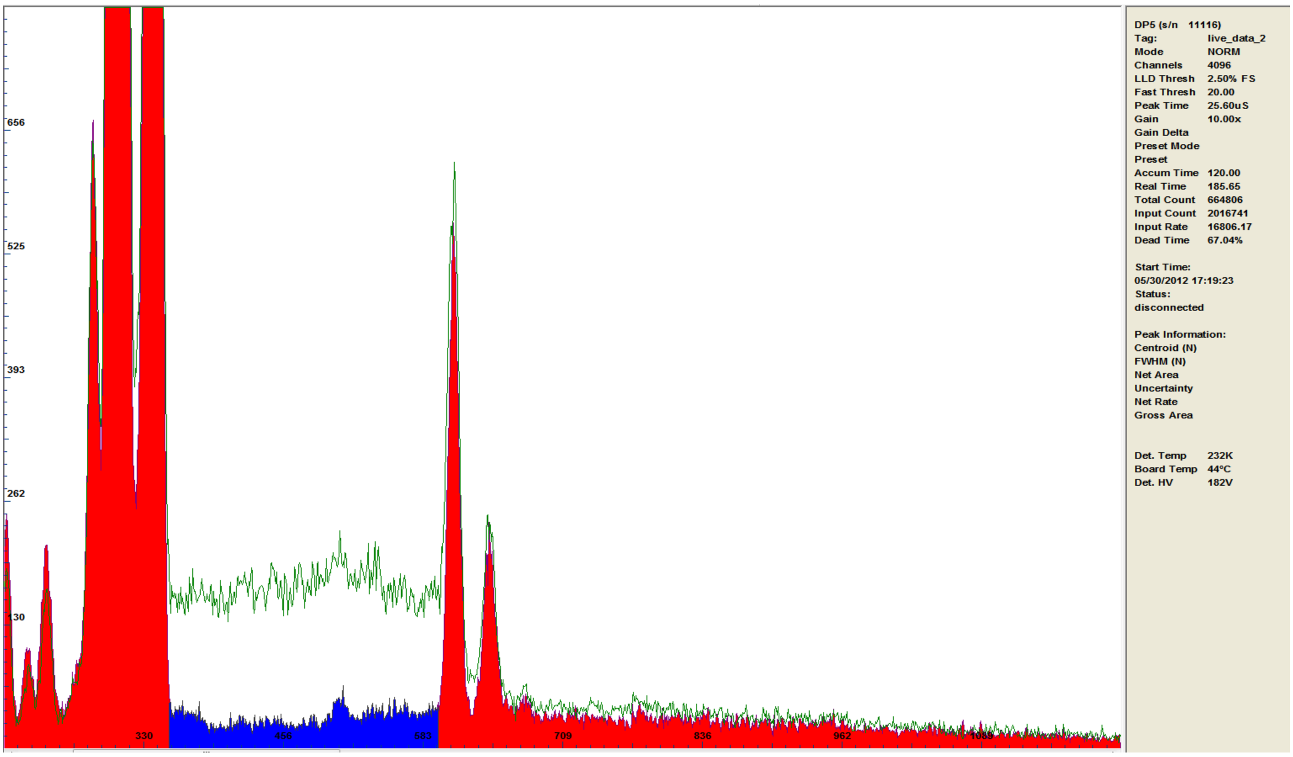
It was observed in the spectra of all the targets an unexpected step noise. This step noise was independent of the target, since its area was quite the same for all of them as well as the declining point. This step noise was, however, setting dependent, since it was observed that the declining point shifted as parameters changed (gain, thresholds, peaking times, etc). So, this step noise was originated by an inappropriate setting configuration, which means that by selecting an appropriate

DPP configuration, we may get rid of the step noise. The first complication to face was the enormous number of parameters to play with and the fact that many of them are not independent, that is why the identification of the relevant parameters whose setting was originating the step noise was very important.

Solution

After trying a few possible solutions, the origin of the step noise turned out to be a pile-up rejection problem. The step shape of the noise and the presence of the noise only before a certain energy tells us that the noise is related to a threshold: noise pulses above certain energy are not counted and noise pulses below that energy are counted. *Counted* means that the noise peak is higher than the LLD threshold and the slow channel threshold. So, the problem in principle has nothing to do with the thresholds of the slow channel, since once a pulse is measured in the slow channel, all the pulses above that one should be measured. In fact, in this case it occurs otherwise: the noise is not recorded when it is energetic enough. That means actually that the noise is identified as such only when it goes above a certain energy, that is to say, above a certain threshold. That threshold is the fast channel threshold. The fact that sometimes noise goes above the slow channel thresholds but not above the fast one is expectable since in principle the fast threshold is higher than the slow threshold. As explained before, one of the functionalities of the fast channel is to resolve pulses overlapping in the slow channel to then reject them (PUR).

The problem was the following. Several noise pulses were close enough to overlap in the slow channel and then produced another pulse which went above the LLD threshold and the slow threshold, being then recorded as a single pulse with a wrong peak. Pile-up rejection failed to reject them because even though they were resolved by the fast channel (otherwise they would have overlapped in it and been probably measured by the fast channel), they did not go above the fast threshold and then were not measured by it. So, to solve this PUR problem, we lowered the fast threshold (in most cases to a half was good enough), leaving the LLD threshold unchanged in order to conserve the actual signal. Thus, the fast channel was able to measure as a single pulse pulses overlapping in the slow channel, and then reject them, getting rid of the noise. It is important not to lower the fast threshold excessively, since this may reject even signal by PUR. This was applied to all tatgets, obtaining spectra without step noises. In fig. 3.24 we plot the spectrum of Ba before tunning settings and after doing it. It can be observed how the noise is suppressed.



```
DP5 (s/n   11116)
Tag:            live_data_2
Mode            NORM
Channels        4096
LLD Thresh      2.50% FS
Fast Thresh     20.00
Peak Time       25.60uS
Gain            10.00x
Gain Delta
Preset Mode
Preset
Accum Time      120.00
Real Time       185.65
Total Count     664806
Input Count     2016741
Input Rate      16806.17
Dead Time       67.04%

Start Time:
05/30/2012 17:19:23
Status:
disconnected

Peak Information:
Centroid (N)
FWHM (N)
Net Area
Uncertainty
Net Rate
Gross Area


Det. Temp       232K
Board Temp      44°C
Det. HV         182V
```

Figure 3.24: Ba spectrum before and after tunning settings.

3.4 Spectrometer data converter

The efficiency of the spectrometer is not perfect, which means that the probability of a photon undergoing a interaction in the sensor and depositing all its energy is not 1, considering besides the effect of the Be window that may stop some photons before these reach the sensor. This efficiency can be seen in fig. 3.11. By using the efficiency of the spectrometer, we can correct the spectrum of all our targets and obtain spectra as if the spectrometer had perfect efficiency. To do so, we wrote in C++ a spectrometer data converter which takes the spectrum of the targets as input and gives the corrected spectrum as output. The code is in the Appendix.

In fig. 3.25 we plot the corrected spectra of all the targets available in the laboratory that we obtained by using the spectrometer-data-converter. The targets used are Iron, Copper, Molybdenum, Silver, Tin and Baryum. We used the same settings for all the targets, including an acquisition time of 180 s.

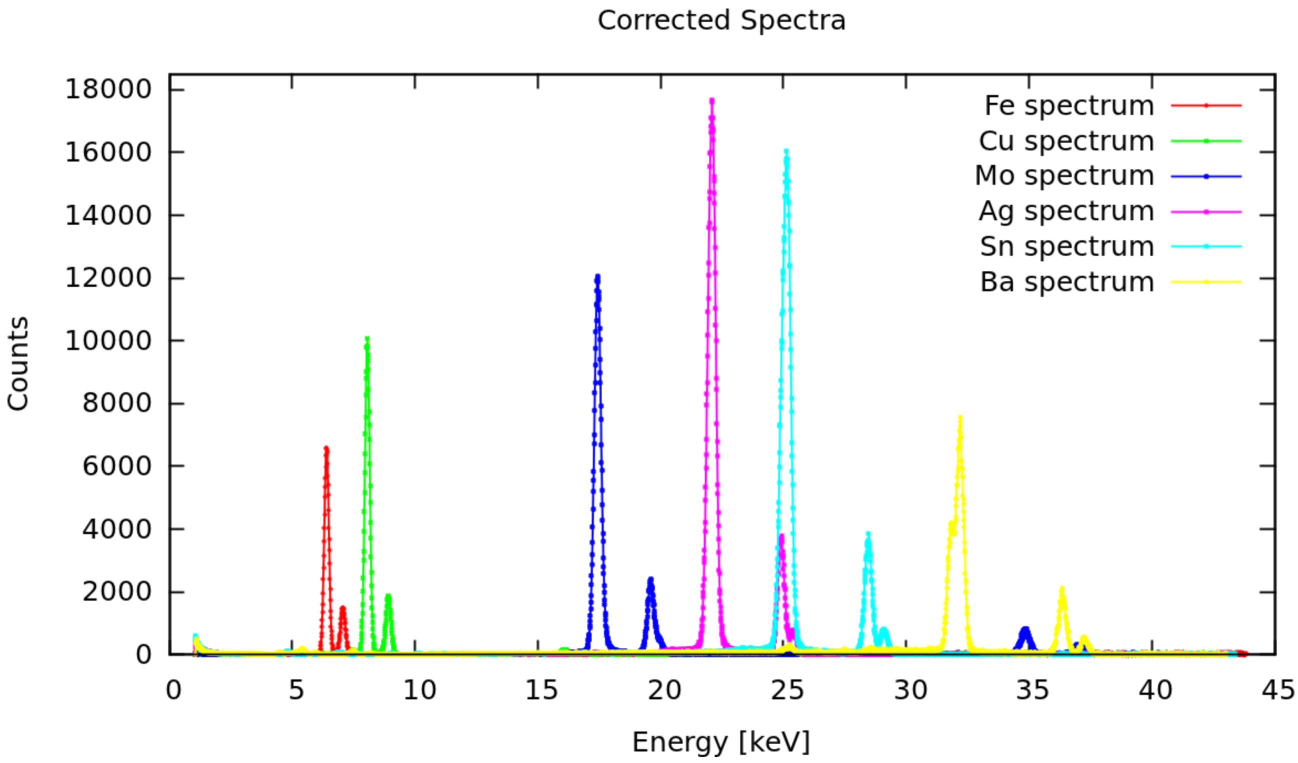


Figure 3.25: Corrected spectra of Fe, Cu, Mo, Ag, Sn and Ba, with acquisition time of 180 s.

Chapter 4

VCal Calibration

We made a Vcal calibration for two chips, chip 1 and chip 2, one after another. First, we ran a Pretest. Then, for chip 1, we made an untrimmed Vcal calibration and Vcal calibrations trimmed to Vcal60 and Vcal110. For chip 2, we only made a Vcal calibratrion trimmed to Vcal110.

4.1 X-ray test and Xcurves

Using the *psi46expert*, we make a X-ray test which is a VthrComp Scan in presence of a flourescent target. For every target we change the range of the VthrComp Scan according to the energy of the K- lines to save time. If there were no noise and trimming were perfect, all pixel would have exactly the same threshold, so they would never respond to photons whose signal is below the threshold and always respond to photons whose signal is at or above the threshold, in which case the Xcurve would be a Heaviside step function. In this case, when running the X-ray test, we would expect the step point of the Heaviside Xcurve to be equal to the VcThr corresponding to the signal of the K-line. However, because of the gaussian noise and an always imperfect trimming, the Xcurve in the histogram *hits vs VthrComp* is an *error function*, whose 50 % point is equal to the VcThr value at which half of the pixels start to respond to the K- line. This histogram is saved in a root file.

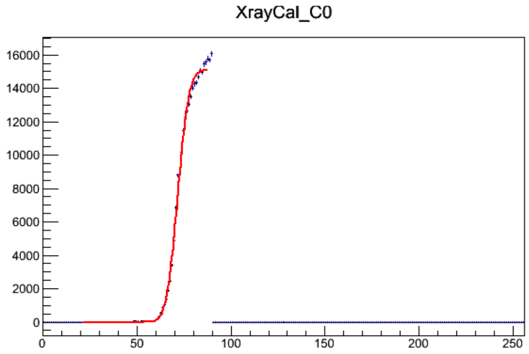


Figure 4.1: Xcurve of Silver.

4.1.1 Fitting of Xcurves

To find the actual 50% points of the Xcurves we need to fit the data, because it is necessary to separate the contributions of the K-α and K-β lines to the spectra. Since the spectra is dominated by the K-*alpha* line because of its much higher intensity with respect to the K-*beta* line, we are going to use only the 50% point of the K-α line. It was observed that approximately for VthrComp values higher than the 50% point of the K-*alpha* line, the curve behaves linearly, that is to say, the curve has an asymptotic linear behaviour.

One K- line is fitted by $\left(1+Erf\left(\frac{x-A}{B}\right)\right)$. To combine the contribution of the K-α and K-β lines we have to scale them by the relative intensities and then add them. To take into account the asymptotic linear behaviour, we finally multiply all the expresion by $(G(x-A-2B)+H)$, obtaining as fitting fuction

$$F(\cdot)=\left\{0.5\cdot C\left(1+Erf\left(\frac{x-A}{B}\right)\right)+0.5\cdot(1-C)\cdot\left(1+Erf\left(\frac{x-E}{F}\right)\right)\right\}\cdot(G(x-A-2B)+H)+D$$

where

- A, E are the 50% points of the K-α and K-β lines of a given target, respectively.
- C is the ratio of the K-α intensity to the sum of the K-α and K-β intensities. Its value is approximately 0.8 for all the targets.
- B, F are parameters to scale the VthrComp ADC units.
- D is an overall offset parameter.
- G is the slope of the asymptotic linear behaviour of the spectra.
- H is the offset of the asymptotic linear behaviour of the spectra.

For some spectra, the K-β line is not intensive enough to be distinguishable and appear in the data. In that case, only the K-α line contributes to the spectra, so that we only find one associated error function. To take this into account, we set the ratio value C to 1, getting rid of the second error function of the fitting function which would correspond to the K-β line.

We made X-ray tests and obtained Xcurves for two chip, chip 1 untrimmed and trimmed to Vcal60 and to Vcal110, and chip 2 trimmed to Vcal110. For each trimming, we used as targets Cu, Mo, Ag, Sn and Ba. Here we only show the Xcurves for chip 1 trimmed to Vcal 110. We can observer for most of the targets the K-α and K-β lines along with their 50% points.

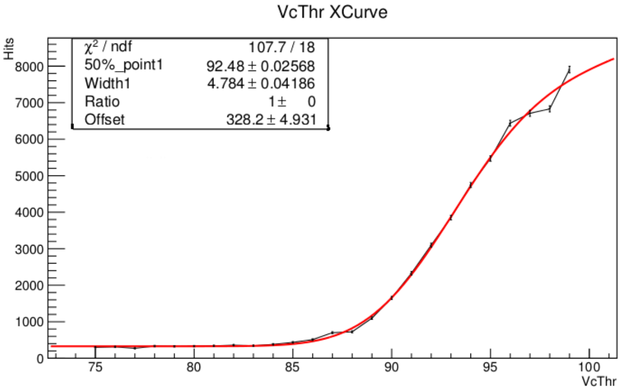


Figure 4.2: Target: Copper. K-α: 8.047KeV; K-β: 8.905KeV

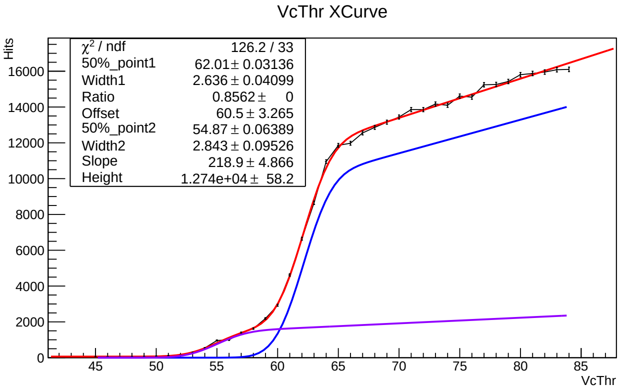


Figure 4.3: Target: Molybdenum. K-α: 17.479KeV; K-β: 19.608KeV

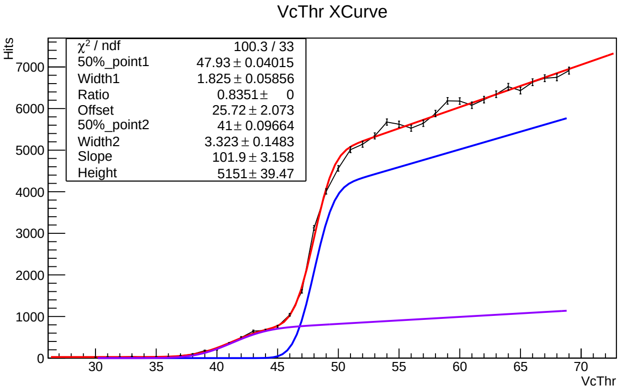


Figure 4.4: Target: Silver. K-α: 22.162KeV; K-β: 24.942KeV

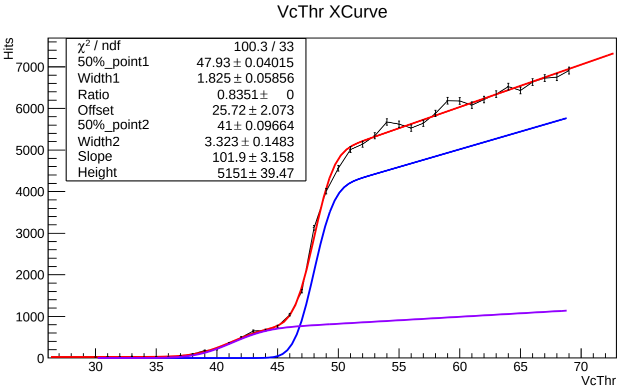


Figure 4.5: Target: Tin. K-α: 25.271KeV; K-β: 28.486KeV

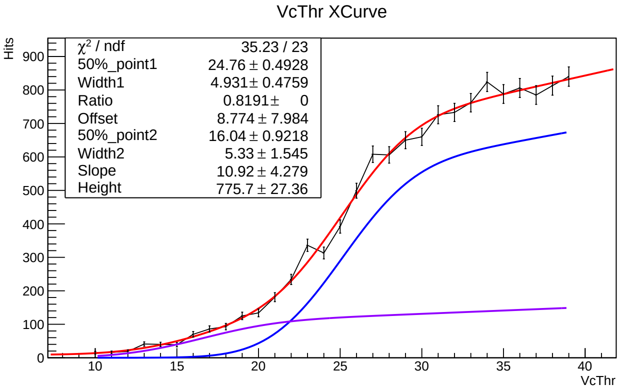


Figure 4.6: Target: Baryum. K-α: 32.193KeV; K-β: 36.378KeV

For chip 1 untrimmed and trimmed to Vcal60 and Vcal110 and for chip 2 trimmed to Vcal110, the VcThr values of the K-α lines of different targets along with their energies and numbers of electrons generated in the Silicon sensor are in Table 1.

	K-α [keV]	elects	VcThr (untrimmed)	ΔVcThr (untrimmed)	VcThr (trim Vcal60)	ΔVcThr (trim Vcal60)	VcThr (trim Vcal110)	ΔVcThr (trim Vcal110)	VcThr (trim Vcal110)	ΔVcThr (trim Vcal110)
Cu	8.05	2223	99.06	0.2801	92.66	0.1828	92.48	0.02568	98.63	0.0260
Mo	17.48	4828	73.22	0.2297	61.11	0.04209	62.01	0.03136	70.02	0.0603
Ag	22.16	6121	59.78	0.09866	46.65	0.07088	47.93	0.04015	55.09	0.1013
Sn	25.27	6980	51.29	0.08195	37.17	0.2098	40.1	0.08438	47.25	0.0525
Ba	32.19	8892	36.57	0.08092	17.28	0.6312	24.76	0.4928	30.18	0.142

Table 4.1: Energies, electrons and VthrComp values of K-α lines of several targets. Columns 4-9 correspond to chip 1 and columns 9-10 to chip 2.

We plotted as well the number of electrons vs VthrComp for chip 1 and chip 2. p_0 and p_1 are the offset and the slope, respectively, of the linear fit we used.

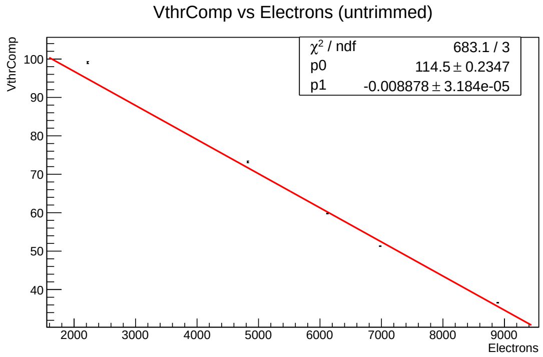


Figure 4.7: Chip 1 untrimmed.

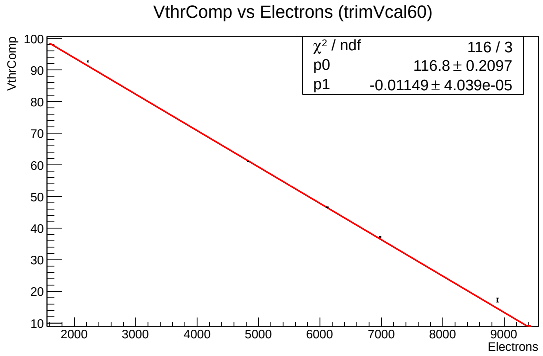


Figure 4.8: Chip 1 trimmed to Vcal60.

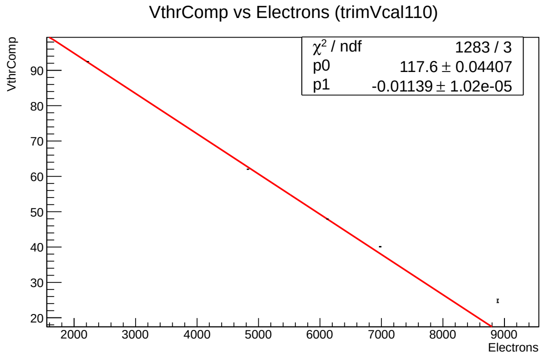


Figure 4.9: Chip 1 trimmed to Vcal110.

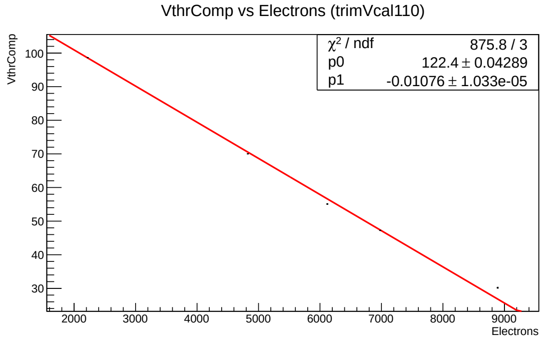


Figure 4.10: Chip 2 trimmed to Vcal110.

4.2 Vcal Threshold Map and Vcal calibration

The aim of the Vcal calibration is to find the number of electrons to which a Vcal unit corresponds. In step 2 we found the number of electrons per VthrComp unit. With the Vcal Threshold Map, we can associate the number of electrons with the Vcal by finding the mean Vcal to which a certain VthrComp corresponds.

The *Vcal Threshold Map* is a Vcal scan for all the pixels. Given an input VthrComp for the chip, a Vcal pulse is injected to each pixel varying gradually until finding the Vcal at which the pixel starts responding. Then, a map of the chip is plotted with the activating Vcals of each pixel. As shown in Fig 3.13, the pixels are labelled by columns and rows and their colour represents the activating Vcal value.

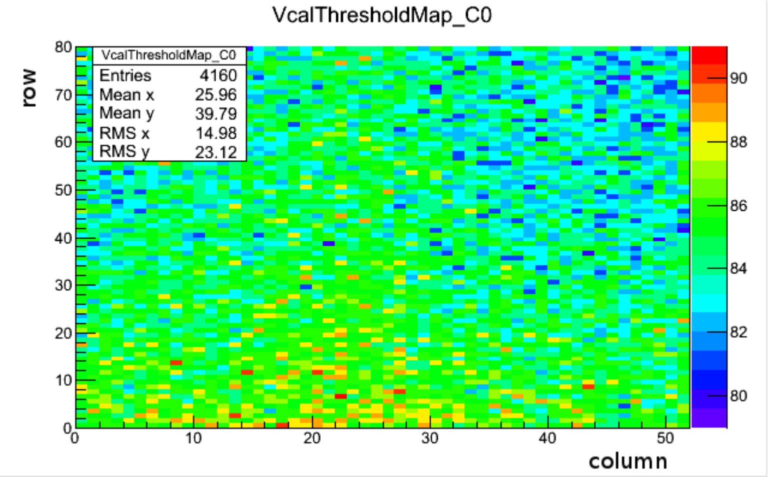


Figure 4.11: Vcal Threshold Map.

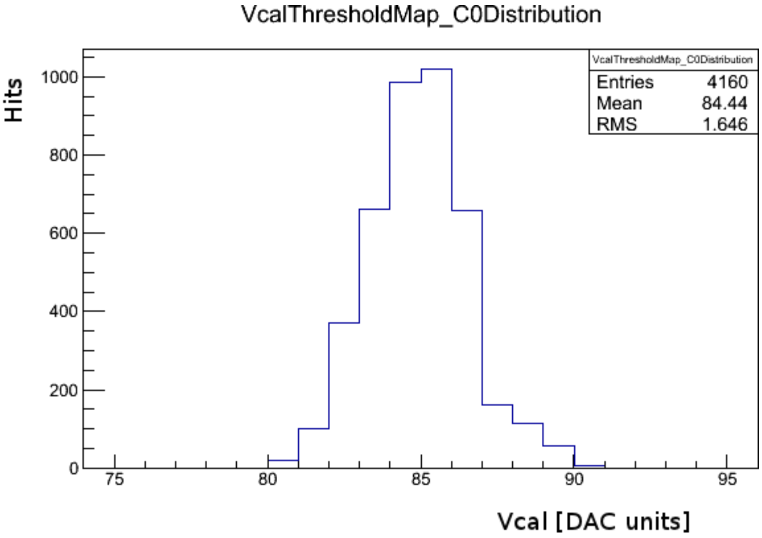


Figure 4.12: Vcal-threshold Map distribution for Molybdenum.

4.2.1 *Vcal-CalDel* optimization

For a chip trimmed at Vcal 110, for instance, we expect to find a Vcal equal to 110 when making a Vcal Threshold Map for the VthrComp of the trimming. However, what we find is a higher Vcal. The reason is the following. Since in the Vcal Scan the variable pulses are injected only in one clock cycle, the Vcal-threshold we are dealing with is a in-time threshold. This means that as pulses are measured only in one clock cycle, many of them do not have enough time to peak in the considered clock cycle and then fail to be measured. So higher pulses are more likely to go above the threshold even if not peaking, which overestimates the Vcal-threshold and the value of the mean Vcal in the pixel.

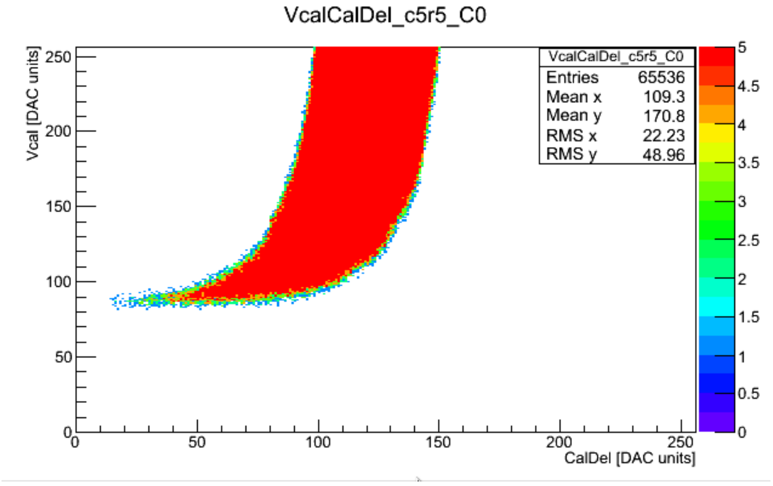


Figure 4.13: Vcal-CalDel working region.

To solve this, we run a *DacDac* test, which plots (under appropriate settings) the Vcal-CalDel range in which the ROC works. To measure this region VcThr is set to a fixed value in low range DAC units, five calibrate signals are sent for each pair of Vcal and CalDel, and the number of readouts is counted. This procedure is done for one single pixel since the working Vcal-CalDel is large enough to accomodate all pixels. As we are looking for the minimum Vcal at which pixels respond, we choose the CalDel value associated with the minimum working Vcal in the readout area. This CalDel is defined as working value for calibration purpose. At this CalDel, pulses have enough time to peak in one clock cycle, obtaining a timing-independent Vcal-threshold.

We run the Vcal-threshold Map for all the VthrComp values we obtained in step 2. In *psi46expert* we set the VthrComp parameter to a given value. Then, we run the Vcal-CalDel optimization, choosing a CalDel value in the readout area for the lowest possible Vcal and we set the CalDel parameter to this value. Then, we run the Vcal Threshold Map, getting as well a Vcal-threshold Map distribution, from which we read the mean Vcal value of the ROC along with its root mean square for the corresponding VthrComp input.

4.2.2 Vcal calibration

We can only insert integer VthrComp values as input for the Vcal-threshold Map. Therefore, the mean Vcal values we obtained before are not associated with the VthrComp (50% points) nor with the electrons liberated by the K-α lines of the targets. To find the right number of electrons with which the mean Vcal values are associated, we use the linear relations on figures 4.7, 4.8, 4.9 and 4.10 and the input integer VthrComp values for the Vcal-threshold Map.

In tables 4.2 and 4.3 we can see the number of electrons along with the Vcal for chip 1 untrimmed, trimmed at Vcal60 and Vcal110, and for chip 2 trimmed at Vcal110.

	electrons	Δelectrons	Vcal (untrimmed)	ΔVcal (untrimmed)	electrons	Δelectrons	Vcal (trim Vcal60)	ΔVcal (trim Vcal60)
Cu	1746	31.547	44.44	4.675	2223	13.46	44.26	1.263
Mo	4674	25.871	85.28	5.222	4828	3.10	85.5	2.752
Ag	6139	11.112	109	5.468	6121	5.22	108.4	4.189
Sn	7153	9.230	125.3	6.55	6980	15.45	121.9	6.396
Ba	8729	9.114	160.7	7.228	8892	46.48	160.8	13.45

Table 4.2: Number of electrons and its error along with Vcal and its error for chip 1 untrimed (columns 2-5) and trimmed at Vcal60 (columns 6-9).

	electrons	Δelectrons	Vcal (trim Vcal110)	ΔVcal (trim Vcal110)	electrons	Δelects	Vcal (trim Vcal110)	ΔVcal (trim Vcal110)
Cu	2248	2.254	45.51	2.716	2175	2.42	44.44	2.981
Mo	4881	2.753	87.88	1.998	4870	5.61	84.44	1.646
Ag	6111	3.524	110.2	1.143	6264	9.41	106.4	1.184
Sn	6813	7.407	125.7	2.071	7007	4.88	121.2	1.69
Ba	8130	43.262	157.4	4.529	8587	13.20	153.7	3.83

Table 4.3: Number of electrons and its error along with Vcal and its error for chip 1 (columns 2-5) and chip 2 (columns 6-9) trimmed at Vcal110.

We can see the Vcal calibration for the chip 1 untrimmed and trimmed at Vcal60 and Vcal110 in figures 3.15, 3.16 and 3.17, and for the chip 2 trimmed at Vcal110 in figures 3.18 and 3.19. We used five targets to make the Vcal calibration, Copper, Molybdenum, Silver, Tin and Baryum, so we have five data points (electrons,Vcal).

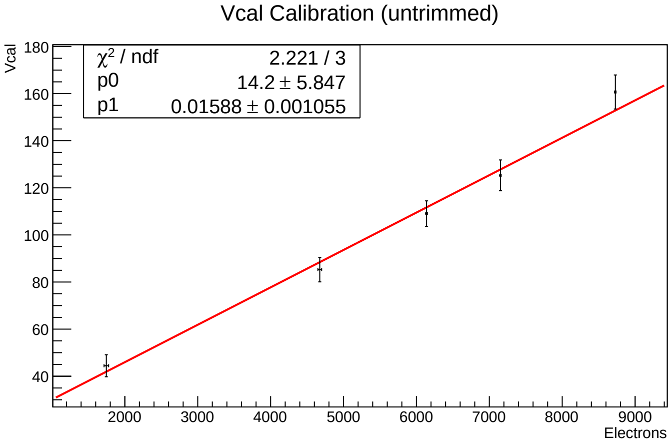


Figure 4.14: Vcal calibration for chip 1 untrimmed.

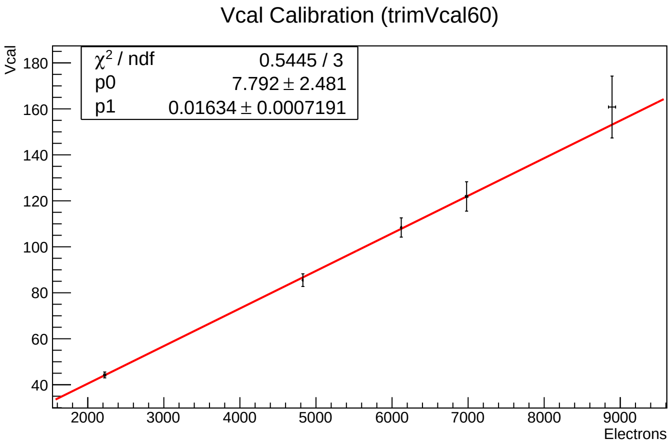


Figure 4.15: Vcal calibration for chip 1 trimmed to Vcal60.

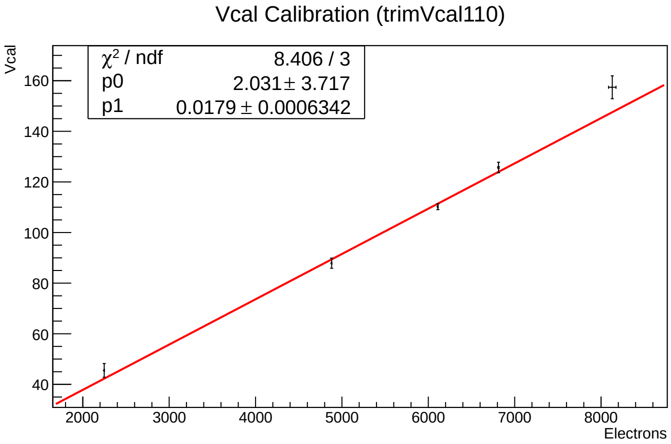


Figure 4.16: Vcal calibration for chip 1 trimmed to Vcal110.

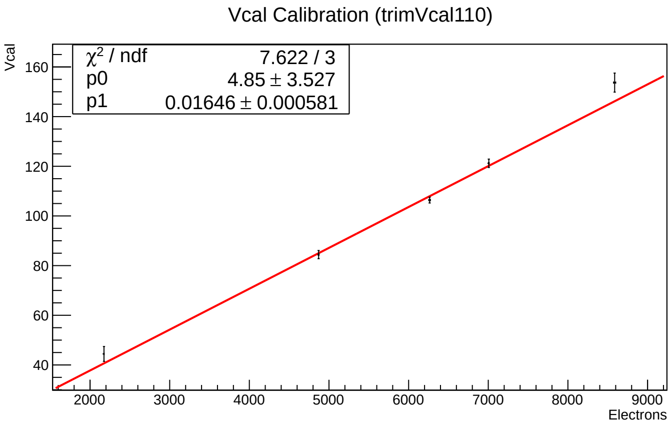


Figure 4.17: Vcal calibration for chip 2 trimmed to Vcal110.

As we can see in the figures, the offset of the Vcal calibration is non vanishing. In principle, we would have expected it to be 0, which means that for non calibrate signal, i.e, for $Vcal = 0$, there are no electrons in the ROC. Since we want to know how many electrons a Vcal unit corresponds to, we are interested in $m(p_0) = \frac{1}{p_0}$, $\Delta m = \frac{\Delta p_0}{(p_0)^2}$. So, the $[electrons/Vcal]$ values for the different Vcal calibrations are the following:

	m(p_0)	Δ m
Chip 1 - untrimmed	62.97	4.18
Chip 1 - trimmed to Vcal60	61.20	2.69
Chip 1 - trimmed to Vcal110	55.86	1.98
Chip 2 - trimmed to Vcal110	60.75	2.14

Table 4.4: Number of electrons and its error along with Vcal and its error for chip 1 (columns 2-5) and chip 2 (columns 6-9) trimmed at Vcal110.

Chapter 5

Summary and Conclusion

This semesterarbeit has two main aims. The first one is to identify the source of noise in the spectra observed systematically for all the targets and to tune the settings of the X-123 X-ray spectrometer in order to suppress this noise. The second one is to make the Vcal calibration for two CMS pixel readout chips and obtain the number of electrons per Vcal unit.

We studied the pixel barrel modules, the readout chip along with its readout chain and the most relevant DAC parameters for our purposes, Chip setups as Pretest and Trimming and the Vcal calibration. We presented the X-ray setup of the Pixel Group at ETH Zurich and explained briefly the functionality and experiment setup of the X-ray tube, targets, spectrometer and ROC. We studied as well the functionality and different devices of the spectrometer along with its Digital Pulse Processor (DPP) parameters.

We found that the source of the noise was a PUR problem due to a wrong tunning of the thresholds. To solve it, we lowered the threshold of the DPP fast channel while keeping the LLD threshold of the DPP slow channel unchanged. Thus, the fast channel was able to measure as a single pulse pulses overlapping in the slow channel to then reject them, suppressing the noise.

The Vcal calibrations for chip 1 and 2 are in the following table.

	m	Δ m
Chip 1 - untrimmed	62.97	4.18
Chip 1 - trimmed to Vcal60	61.20	2.69
Chip 1 - trimmed to Vcal110	55.86	1.98
Chip 2 - trimmed to Vcal110	60.75	2.14

Table 5.1: Number of electrons and its error along with Vcal and its error for chip 1 (columns 2-5) and chip 2 (columns 6-9) trimmed at Vcal110.

where m is the number of electrons per Vcal unit and Δm its error.

Chapter 6

Appendix

6.1 Spectrometer Data Converter

```
/*
*       Processes a spectrum file generated by the AmpTek spectrometer
*       software and corrects for detector efficiency.
*
*                                               Written by Daniel Narrias
*                                               07. June 2012
*/

#include <iostream>
#include <fstream>
#include <vector>
#include <cmath>

using namespace std;

const int nlookup = 194;

double lookup_energy [nlookup] = {
1.40E-02,  1.49E-02,  1.60E-02,  1.71E-02,  1.82E-02,  1.91E-02,  1.94E-02,
1.95E-02,
1.95E-02,  1.96E-02,  1.99E-02,  2.08E-02,  2.23E-02,  2.38E-02,  2.55E-02,
2.58E-02,
2.62E-02,  2.63E-02,  2.64E-02,  2.68E-02,  2.72E-02,  2.91E-02,  3.11E-02,
3.32E-02,
3.55E-02,  3.80E-02,  4.06E-02,  4.34E-02,  4.64E-02,  4.96E-02,  5.30E-02,
5.67E-02,
6.06E-02,  6.48E-02,  6.92E-02,  7.40E-02,  7.91E-02,  8.46E-02,  9.04E-02,
9.67E-02,
9.72E-02,  9.87E-02,  9.91E-02,  9.97E-02,  1.01E-01,  1.03E-01,  1.09E-01,
1.10E-01,
1.11E-01,  1.12E-01,  1.13E-01,  1.18E-01,  1.26E-01,  1.35E-01,  1.44E-01,
1.46E-01,
1.48E-01,  1.49E-01,  1.49E-01,  1.52E-01,  1.54E-01,  1.65E-01,  1.76E-01,
1.88E-01,
2.01E-01,  2.15E-01,  2.30E-01,  2.46E-01,  2.63E-01,  2.78E-01,  2.81E-01,
2.82E-01,
2.84E-01,  2.85E-01,  2.89E-01,  3.01E-01,  3.21E-01,  3.43E-01,  3.67E-01,
3.92E-01,
3.94E-01,  4.00E-01,  4.01E-01,  4.04E-01,  4.10E-01,  4.20E-01,  4.48E-01,
4.79E-01,
5.12E-01,  5.48E-01,  5.86E-01,  6.26E-01,  6.69E-01,  7.15E-01,  7.65E-01,
8.18E-01,
8.74E-01,  9.34E-01,  9.99E-01,  1.07E+00,  1.14E+00,  1.22E+00,  1.30E+00,
1.39E+00,
1.49E+00,  1.59E+00,  1.70E+00,  1.80E+00,  1.82E+00,  1.83E+00,  1.84E+00,
1.85E+00,
1.88E+00,  1.95E+00,  2.08E+00,  2.22E+00,  2.38E+00,  2.54E+00,  2.72E+00,
2.90E+00,
3.11E+00,  3.32E+00,  3.55E+00,  3.79E+00,  4.06E+00,  4.33E+00,  4.63E+00,
4.95E+00,
5.30E+00,  5.66E+00,  6.05E+00,  6.47E+00,  6.92E+00,  7.39E+00,  7.90E+00,
8.45E+00,
9.03E+00,  9.65E+00,  1.03E+01,  1.10E+01,  1.18E+01,  1.26E+01,  1.35E+01,
1.44E+01,
1.54E+01,  1.65E+01,  1.76E+01,  1.88E+01,  2.01E+01,  2.15E+01,  2.30E+01,
2.46E+01,
2.63E+01,  2.81E+01,  3.00E+01,  3.21E+01,  3.43E+01,  3.67E+01,  3.92E+01,
4.19E+01,
4.48E+01,  4.79E+01,  5.12E+01,  5.47E+01,  5.85E+01,  6.25E+01,  6.68E+01,
7.15E+01,
7.64E+01,  8.17E+01,  8.73E+01,  9.33E+01,  9.98E+01,  1.07E+02,  1.14E+02,
1.22E+02,
1.30E+02,  1.39E+02,  1.49E+02,  1.59E+02,  1.70E+02,  1.82E+02,  1.94E+02,
2.08E+02,
2.22E+02,  2.37E+02,  2.54E+02,  2.71E+02,  2.90E+02,  3.10E+02,  3.32E+02,
3.54E+02,
3.79E+02,  4.05E+02};

double lookup_attenuation [nlookup] = {
5.99E-253, 1.49E-246, 4.71E-237, 3.36E-225, 1.04E-211, 1.90E-201, 4.33E-198, 3.2
3.39E-197, 4.01E-197, 5.60E-194, 1.12E-183, 5.59E-169, 1.28E-154, 7.77E-141, 2.3
2.03E-135, 1.23E-134, 1.83E-133, 1.57E-130, 8.85E-128, 1.51E-115, 3.43E-104,
1.02E-93,
4.14E-84,  2.61E-75,  2.71E-67,  5.10E-60,  1.93E-53,  1.64E-47,  3.56E-42,
2.18E-37,
4.20E-33,  2.84E-29,  7.37E-26,  8.11E-23,  4.09E-20,  1.02E-17,  1.36E-15,
1.02E-13,
1.41E-13,  3.30E-13,  4.14E-13,  1.63E-14,  3.73E-14,  1.23E-13,  1.57E-12,
3.20E-12,
3.83E-12,  0.00E+00,  0.00E+00, 5.86E-307, 7.53E-267, 3.55E-231, 2.05E-202, 7.57
5.90E-191, 4.06E-189, 1.64E-186, 1.29E-179, 8.27E-172, 2.53E-147, 5.98E-126, 2.1
2.33E-91,  1.51E-77,  1.06E-65,  1.45E-55,  6.34E-47,  1.41E-39,  3.34E-35,
4.42E-35,
5.68E-35,  7.97E-37,  2.35E-36,  3.75E-35,  1.21E-29,  5.63E-25,  5.00E-21,
1.07E-17,
1.40E-17,  5.90E-17,  8.66E-17,  1.54E-16,  6.47E-16,  6.85E-15,  1.56E-12,
1.49E-10,
6.77E-09,  1.65E-07,  2.38E-06,  2.20E-05,  1.40E-04,  6.49E-04,  2.32E-03,
6.68E-03,
1.60E-02,  3.31E-02,  6.02E-02,  1.00E-01,  1.52E-01,  2.14E-01,  2.83E-01,
3.56E-01,
4.29E-01,  5.00E-01,  5.67E-01,  6.19E-01,  6.29E-01,  6.32E-01,  6.36E-01,
5.76E-01,
5.88E-01,  6.21E-01,  6.75E-01,  7.25E-01,  7.68E-01,  8.05E-01,  8.37E-01,
8.64E-01,
8.87E-01,  9.07E-01,  9.23E-01,  9.37E-01,  9.48E-01,  9.57E-01,  9.65E-01,
9.71E-01,
9.76E-01,  9.80E-01,  9.84E-01,  9.87E-01,  9.89E-01,  9.91E-01,  9.92E-01,
9.92E-01,
9.89E-01,  9.81E-01,  9.65E-01,  9.39E-01,  9.01E-01,  8.51E-01,  7.91E-01,
7.24E-01,
6.54E-01,  5.83E-01,  5.14E-01,  4.49E-01,  3.89E-01,  3.35E-01,  2.88E-01,
2.46E-01,
2.10E-01,  1.79E-01,  1.52E-01,  1.29E-01,  1.10E-01,  9.44E-02,  8.13E-02,
7.02E-02,
6.11E-02,  5.35E-02,  4.73E-02,  4.21E-02,  3.78E-02,  3.43E-02,  3.13E-02,
2.89E-02,
2.68E-02,  2.51E-02,  2.36E-02,  2.23E-02,  2.12E-02,  2.03E-02,  1.95E-02,
1.87E-02,
1.81E-02,  1.75E-02,  1.69E-02,  1.64E-02,  1.59E-02,  1.55E-02,  1.51E-02,
1.47E-02,
1.43E-02,  1.39E-02,  1.35E-02,  1.32E-02,  1.28E-02,  1.25E-02,  1.21E-02,
1.18E-02,
1.15E-02,  1.11E-02};

// Gives the index of the lower limit of the energy, since int (a+b)/2 is rounde

int lookup_search(double energy)
{
```

```
/* Binary search algorithm */

int a = 0;
int b = nlookup - 1;
int i = (a + b) / 2;

while (i != a && i != b) {
        if (energy > lookup_energy[i]) {
                a = i;
                i = (a + b) / 2;
        } else {
                b = i;
                i = (a + b) / 2;
        }
}

return i;
}

// Interpolates between two points.

double interpolate(double x1, double y1, double x2, double y2, double x)
{

double slope, y;

slope = (y2 - y1) / (x2 - x1);

y = slope * x + (y1 - slope * x1);

return y;

}

int main(int argc, char ** argv) {
/* Open spectrum file */
ifstream file(argv[1]);

if (argc != 2) {
        cout << ”Specify a spectrum file!” << endl;
        return 1;
}

if (!file.is_open()) {
        cout << ”Could not open file!” << endl;
        return 1;
```

```
}

string line;
bool data_start = false;
double A, B, C;

/* Read file line by line */
while (file.good()) {
        /* Read entire lines until the data starts, then read data pairs */
        if (!data_start) {
                getline(file, line);

                /* Get calibration parameters */
                if (line.find(”A\tB\tC”) >= 0)
                        file >> A >> B >> C;

                /* Find line from which data starts */
                if (line.find(”Channel\tData”) >= 0) {
                        cout << ”# Channel\tEnergy\tCounts\tCorrected_Counts\tE
<< endl;
                        data_start = true;
                }
        } else {
                /* Read data (channel, counts) */
                double channel, counts;
                file >> channel >> counts;

                /* Make calibration, channel to energy in KeV */
                double energy = channel * B + A;

                /* Check energy range. Energy < 3 keV: noise */
                if (energy < lookup_energy[0] || energy > lookup_energy[nlookup-
                        continue;

                /* Correction of counts using the efficiency lookup table and in
                int i;
                double x1, y1, x2, y2, efficiency, corrected_counts;

                /* Find the nearest entry in the lookup table */
                i = lookup_search(energy);

                x1 = lookup_energy[i];
                x2 = lookup_energy[i + 1];

                y1 = lookup_attenuation[i];
                y2 = lookup_attenuation[i + 1];
                efficiency = interpolate(x1, y1, x2, y2, energy);

                /* Make sure efficiency is well defined */
                if (efficiency <= 0)
                        corrected_counts = 0;
                else
                        corrected_counts = counts / efficiency;

                /* Print channel, energy, counts, corrected counts, efficiency,
                cout << channel << "\t" << energy << "\t" << counts << "\t" << c
<< "\t" << sqrt(counts) << "\t" << sqrt(counts)/efficiency << endl;
        }
}

file.close()

return 0;
}
```

Bibliography

[1] K. Gabathuler *PSI46 Pixel Chip - External Specification*, PSI, September 2004.

[2] S. Dambach, C. Eggel, U. Langenegger, A. Starodumov, P. Trüb *CMS Barrel Pixel Module Qualification.*

[3] P. Trüb, *CMS Pixel Module Qualification and Monte-Carlo Study of* $H \to \tau^+\tau^- \to \ell^+\ell^- \not{E}_T$, Diss. ETH No. 17985.

[4] Amptek, *www.amptek.com/x123.html.*