Technische Universität München
Lehrstuhl für Integrierte Systeme

Real-time data acquisition system for X-ray imaging spectroscopy

Dipl.-Ing. Univ. Florian Matthias Aschauer

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzende: Univ.-Prof. Dr.rer.nat. Doris Schmitt-Landsiedel

Prüfer der Dissertation:

1. apl. Prof. Dr.-Ing. habil. Walter Stechele
2. Univ.-Prof. Dr.rer.nat. Peter Fischer, Universität Heidelberg

Die Dissertation wurde am 29.09.2014 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 28.01.2015 angenommen.

Contents

List of Figures

List of Tables

Nomenclature

ADC	**A**nalog-to-**d**igital **c**onverter
ADU	**A**nalog-**D**igital **U**nits
AMBA	**A**dvanced **M**icrocontroller **B**us **A**rchitecture
APS	**A**ctive **P**ixel **S**ensor
ASIC	**A**pplication-**s**pecific **i**ntegrated **c**ircuit
ASTEROID	**A**ctive current **S**witching **Te**chnique **R**ead **O**ut **i**n X-ray spectroscopy with **D**EPFET
AXI	**A**dvanced e**X**tensible **I**nterface
BRAM	**B**lock **RAM**
CALIFA	**Cali**bration **Fa**cility
CAMEX	**C**harge **A**mplifier **M**ultipl**EX**er
CCD	**C**harge **C**oupled **D**evice
CDC	**C**lock **d**omain **c**rossing
CDF	**C**umulative **d**istribution **f**unction
CERN	**C**onseil **E**uropéen pour la **R**echerche **N**ucléaire
CMC	**C**ommon **M**ezzanine **C**ard
CPE	**C**ycles **p**er **e**lement
CTRL	**C**on**tr**ol **L**ogic
DAC	**D**ata **a**nalysis **c**omputer
DAQ	**D**ata **Ac**quisition
DB	**D**ata**b**ase
DCC	**D**AQ **C**ontrol **C**omputer
DDR	**D**ouble **D**ata **R**ate
DEPFET	**De**pleted **P**-Channel **F**ield-**E**ffect **T**ransistor
DHC	**D**ata **H**andling **C**omputer
DMA	**D**irect **M**emory **A**ccess
DNE	**D**ynamic **N**oise **E**stimation

DOE	**D**ynamic **O**ffset **E**stimation
DPB	**D**ata **p**re-processing **B**oard
DPPS	**D**ata stream **p**ost-**p**rocessing **s**oftware
DSCC	**D**etector **S**ystem **C**ontrol **C**omputer
DSP	**D**igitaler **S**ignal **P**rozessor
EDK	**E**mbedded **D**evelopment **K**it
ESA	**E**uropean **S**pace **A**gency
FEL	**F**ree **E**lectron **L**aser
FPGA	**F**ield **P**rogrammable **G**ate **A**rray
fps	**F**rames **P**er **S**econd
FT	**F**rame**T**erm
FWHM	**F**ull **w**idth at **h**alf **m**aximum
GDEPFET	**G**ateable **DEPFET**
GPGPU	**G**eneral-**p**urpose **g**raphics **p**rocessing **u**nits
IC	**I**ntegrated **C**ircuit
IIC	**I**nter-**I**ntegrated **C**ircuit
IP	**I**ntellectual **P**roperty
IP	**I**nternet **P**rotocol
ISE	**I**ntegrated **S**oftware **E**nvironment
IXO	**I**nternational **X**-ray **O**bservatory
LLink	**L**ocal**Link**
LT	**L**ine**T**erm
LUT	**L**ook**u**p-**T**able
lwIP	**L**ight**W**eight **IP**
MBS	**M**ultiple **B**uffer **S**tructure
MCR	**M**aximum **c**hange **r**ate
MIP	**M**inimum **i**onizing **p**articles
MIXS	**M**ercury **I**maging **X**-ray **S**pectrometer
MLL	**M**aier-**L**eibnitz **L**aboratory for Nuclear and Particle and Accelerator Physics
MOSFET	**M**etal-**O**xide-**S**emiconductor **F**ield-**E**ffect **T**ransistors
MPMC	**M**ulti-**P**ort **M**emory **C**ontroller
MPO	**M**ercury **P**lanetary **O**rbiter
NPI	**N**ative **P**ort **I**nterface

PCI-X	**P**eripheral **C**omponent **I**nterconnect e**X**tended
PCIe	**P**eripheral **C**omponent **I**nterconnect **E**xpress (PCI Express)
PCI	**P**eripheral **C**omponent **I**nterconnect
PCores	**P**rocessor **cores**
PDF	**P**robability **d**ensity **f**unction
PLB	**P**rocessor **L**ocal **B**us
RAM	**R**andom **A**ccess **M**emory
RDBMS	**R**elational **d**ata**b**ase **m**anagement **s**ystems
RNDR	**R**epeated **N**on-**D**estructive **R**eadout
ROAn	**R**OOT-based **O**ffline **An**alysis
SDD	**S**ilicon **D**rift **D**etector
SDK	**S**oftware **D**evelopment **K**it
SEU	**S**ingle-**e**vent **u**pset
SFP+	Enhanced **s**mall **f**orm-factor **p**luggable
SFP	**S**mall **f**orm-factor **p**luggable
SGL	**S**catter **G**ather **L**ist
SPI	**S**erial **P**eripheral **I**nterface
TCP	**T**ransmission **C**ontrol **P**rotocol
TEMAC	**T**ri-Mode **E**thernet **M**edia **A**ccess **C**ontroller
UDP	**U**ser **D**atagram **P**rotocol
VHDL	**V**ery High Speed Integrated Circuit **H**ardware **D**escription **L**anguage
WFI	**W**ide-**f**ield **i**mager
XEUS	**X**-ray **E**volving **U**niverse **S**pectroscopy
XFEL	**X**-ray **F**ree **E**lectron **L**aser
XPS	**X**ilinx **P**latform **S**tudio

Zusammenfassung

Das in dieser Arbeit vorgestellte **Su**per-**Mo**dular-DAQ (SuMo-DAQ)-Konzept wurde für die Echtzeitdatenerfassung, -verarbeitung und -analyse von bildgebenden Röntgenspektroskopie-Detektor-Systemen der nächsten Generation entwickelt. Im Vergleich zu konventionellen Kamera-Systemen liefert die bildgebende Röntgenspektroskopie neben der X-/Y-Position und der Ankunftszeit zusätzlich die absolute Photonen-Energie und somit einen vier-dimensionalen Datensatz. Dies ermöglicht eine detaillierte Analyse von Objekten, bei der sogar die örtliche Verteilung der einzelnen chemischen Elemente darstellbar ist. Damit ergeben sich viele Anwendungsmöglichkeiten im Bereich der Medizintechnik, Biologie, Chemie, Sicherheitstechnik und der Teilchenphysik bis hin zur Röntgenastronomie. In dieser Arbeit wurden schwerpunktmäßig **De**pleted **P**-Channel **F**ield-**E**ffect-**T**ransistor (DEPFET)-Detektoren zur Erfassung von Röntgenstrahlung verwendet. Dieser Detektor-Typ wird sowohl erdgebunden in verschiedenen Beschleunigeranlagen als auch in Satelliten eingesetzt. DEPFET-Detektoren bieten eine exzellente Energieauflösung und eine hohe Auslesegeschwindigkeit bei gleichzeitig niedriger Leistungsaufnahme.

Die erwarteten Datenraten der kommenden Detektor-Systeme von mehreren GB/s erforderten die Entwicklung eines neuen Prozessierungs-Konzepts, um die Echtzeitdatenverarbeitung und -analyse dieser Systeme zu ermöglichen. Für die Entwicklung des universell einsetzbaren SuMo-DAQ-Konzepts und dessen Realisierung wurden in dieser Arbeit zuerst die bisherigen DAQ- und Verarbeitungs-Methoden für bildgebende Röntgenspektroskopie-Detektoren analysiert sowie neue Konzepte evaluiert. Des Weiteren wurden verschiedene Implementierungsplattformen wie GPGPU-, CPU-, DSP- und FPGA-Systeme evaluiert und gegeneinander verglichen.

Um verschiedenste Detektor-Systeme der nächsten Generation effektiv betreiben zu können, sind die Kernanforderungen an das SuMo-DAQ-System in Laboranwendungen eine hohe Flexibilität und Modularität, eine gute Skalierbarkeit und Portabilität auf andere Sensorplattformen, die Echtzeitdatenverarbeitungsfähigkeit sowie eine signifikante Reduktion des Datenvolumens zu realisieren. Für die Nutzung des SuMo-DAQ-Konzepts auf Satellitenplattformen sind neben der benötigten Fähigkeit, das System völlig autark operieren zu können, auch die Einschränkung auf Space Qualified Hardware-Komponenten zu berücksichtigen. Aufbauend auf den Evaluierungsergebnissen sowie den Anforderungen der zukünftigen Detektor-Systeme wurde das neue SuMo-DAQ-Konzept auf der Virtex-5 FPGA Platform von XILINX implementiert.

Die Echtzeitverarbeitungfähigkeit der Detektor-Daten wird im SuMo-DAQ-Konzept durch ein FPGA-basiertes Hardware-Pre-Processing und ein nachfolgendes Software-Post-Processing erreicht. Während des Pre-Processings wird der gesamte Datenstrom verarbeitet und für das anschließende Post-Processing reduziert. Die Datenreduktion durch das Pre-Processing ermöglicht eine Reduktion der benötigten CPU-Rechenleistung um den Faktor $\sim 4,7$ für die weitere Datenverarbeitung durch das Software-Post-Processing-System. Durch dieses zweistufige Verarbeitungskonzept ist es zudem möglich, die verschiedenen Prozessierungs-Aufgaben je nach benötigter Rechenleistung, Verarbeitungszeit und den Systemressourcen für die jeweilige Anwendung optimal zu verteilen.

Die neuartigen Algorithmen zur dynamischen Berechnung von Pixel- individuellen Rausch- und Offsetwerten ermöglichen es während einer Messung, wichtige Systemparameter in Echtzeit zu adaptieren. Die permanente Selbstadaption erhöht die Messgenauigkeit und ermöglicht zudem einen vollständig autarken Betrieb des SuMo-DAQ-Systems, wie es auf Satelliten er-

forderlich ist. Durch die geringen Anforderungen in Bezug auf die Hardware-Ressourcen dieser neuen Algorithmen war eine vollständige Implementierung in Hardware auf dem FPGA Pre-Processing System möglich.

In den durchgeführten Labormessungen zur Evaluierung des SuMo-DAQ-Systems konnten die signifikanten Vorteile des neuen Konzeptes deutlich gezeigt werden. Dies gilt besonders im Hinblick auf die Zuverlässigkeit und Präzision der neuen selbstadaptierenden Algorithmen, die Messgenauigkeit in sich ändernden Umgebungen, des autarken System-Betriebs sowie die Vorteile der Echtzeitdatenverarbeitung und des Echtzeit-Detektor-Feedbacks. Während der Langzeit-Messkampagne zur ESA-Qualifizierung der **M**ercury **I**maging **X**-ray **S**pectrometer (MIXS) Detektor-Module, bei der das neue SuMo-DAQ-System bereits eingesetzt wurde, konnte des Weiteren die Zuverlässigkeit und Langzeitstabilität unter realistischen Einsatzbedingungen eindrucksvoll demonstriert werden. Durch die Echtzeitdatenverarbeitungfähigkeit des SuMo-DAQ-Systems konnte zum einen die zu speichernde Datenmenge um den Faktor $36,5$ reduziert werden und zum anderen konnte damit erstmals eine Online-Dosimetrie direkt mit dem zu bestrahlenden DEPFET-Detektor durchgeführt werden.

Summary

The **Su**per-**Mo**dular-DAQ (SuMo-DAQ) concept presented in this thesis has been developed for the real-time data acquisition, processing and analysis of imaging X-ray spectroscopy detector systems of the next generation. In comparison with conventional camera systems, the imaging X-ray spectroscopy provides, in addition to the X/Y position and the time of arrival, the absolute photon energy and thus a four-dimensional data set. This enables a detailed analysis of objects, which even allows for portraying the local arrangement of the individual chemical elements. This leads to many potential applications in the field of medical engineering, biology, chemistry, safety engineering and particle physics right up to X-ray astronomy. This thesis has a focus on the application of **De**pleted **P**-Channel **F**ield-**E**ffect-**T**ransistor (DEPFET) detectors for sensing X-rays. This detector type is used both bound to Earth in various accelerators and in satellites. DEPFET detectors offer an excellent energy resolution and a high read-out speed with low power consumption at the same time.

The expected data rates of the upcoming detector systems of several GB/s required the development of a new processing concept in order to enable the real-time data processing and analysis of these systems. For the development of the universally usable SuMo-DAQ concept and its implementation this thesis first analyzes the present DAQ and processing methods for imaging X-ray spectroscopy detectors and, further, evaluates new concepts. In addition, various implementation platforms such as GPGPU, CPU, DSP and FPGA systems were evaluated and compared with one another.

In order to effectively operate various types of detector systems of the next generation, the core requirements on the SuMo-DAQ system in laboratory applications are high flexibility and modularity, good scalability and portability to other sensor platforms, the ability to process real-time data and a significant reduction of the data volume. For applying the SuMo-DAQ concept in satellite platforms it is, apart from the ability to operate the system completely autarkically, crucial to regard the limitation to space-qualified hardware components. Based on the evaluation results and the requirements of the future detector systems, the new SuMo-DAQ concept was implemented on the Virtex-5 FPGA platform from XILINX.

In the SuMo-DAQ concept, the ability to process real-time detector data is achieved by means of FPGA-based hardware pre-processing and subsequent software post-processing. During the pre-processing the entire data stream is processed and reduced for the subsequent post-processing. The data reduction brought about by the pre-processing enables a reduction of the required CPU processing power by a factor of ~ 4.7 for the further data processing by means of the software post-processing system. Further, this two-stage processing concept allows for the optimal distribution of the various processing tasks depending on the required processing power, processing time and the system resources for the respective application.

The novel algorithms for the dynamic calculation of pixel-individual noise and offset values make sure that during a measurement important system parameters can be adapted in real time. The permanent self-adaption increases the measurement accuracy and, additionally, allows for an entirely autarkic operation of the SuMo-DAQ system, which is essential in satellites. As a result of the low requirements regarding the hardware resources of these new algorithms, a complete implementation in hardware on the FPGA pre-processing system was possible.

In the laboratory measurements carried out for evaluating the SuMo-DAQ system the sig-

nificant advantages of the new concept could be illustrated. This is particularly true regarding the reliability and precision of the new self-adapting algorithms, the measurement accuracy in changing environments, the autarkic system operation and the advantages of the real-time data processing and the real-time detector feedback. Within the scope of the long-term measurement campaign for the ESA qualification of the **M**ercury **I**maging **X**-ray **S**pectrometer (MIXS) detector modules, in which the new SuMo-DAQ system has already been in use, further the reliability and long-term stability could be impressively demonstrated under realistic operating conditions. Due to the ability of the SuMo-DAQ system to process real-time data, on the one hand the amount of data to be saved could be reduced by a factor of 36.5, while on the other hand an online dosimetry could be carried out for the first time directly with the DEPFET detector to be irradiated.

Danksagung

Die in dieser Dissertation dokumentierte Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Max-Planck-Institute for Extraterrestrial Physics. Mein ganz besonderer Dank gebührt meinen beiden Doktorvätern Prof. Dr.-Ing. Walter Stechele vom Lehrstuhl für Integrierte Systeme (LIS) der Technischen Universität München (TUM) und Prof. Dr. Peter Fischer vom Lehrstuhl für Schaltungstechnik und Simulation der Universität Heidelberg, die diese Dissertation durch die Übernahme der Betreuung erst ermöglicht haben, sowie während der gesamten Zeit der Dissertation motivierend und unterstützend zur Seite standen. Mein weiterer Dank gilt Dr. rer. nat. Johannes Treis vom Halbleiterlabor (HLL) der Max-Planck-Gesellschaft für die Unterstützung während meiner Tätigkeit am Halbleiterlabor und die fruchtbaren und inspirierenden Diskussionen in dieser Zeit.

Herrn Prof. Dr. Lothar Strüder und Dr. Norbert Meidinger möchte ich für die große Unterstützung von Seiten des Max-Planck-Institute for Extraterrestrial Physics danken, ohne die das Anfertigen dieser Arbeit nicht möglich gewesen wäre.

Ich möchte auch allen meinen jetzigen und ehemaligen Kollegen am MPE, HLL und LIS für die Unterstützung, die anregenden Diskussionen und die grandiose Arbeitsatmosphäre danken. Ohne die Bereitschaft meiner Kollegen, ihr Wissen zu teilen, wäre die Arbeit in dieser Form nicht möglich gewesen.

Dr. Robert Andritschke, Sabine Walter, Dr. Stefan Bonerz und Julia Rempel danke ich für das intensive Korrekturlesen der Arbeit und die zahlreichen Verbessungsvorschläge.

Das wichtigste Dankeschön gebührt meiner Familie, meinen Eltern Christine und Matthias Aschauer, meinem Bruder Dr. Stefan Aschauer, sowie meiner Großmutter Helga Eder. Ihre moralische Unterstützung, ihr Rückhalt und ihre Geduld führten letztendlich zum Gelingen dieser Arbeit. Letztendlich sei auch all denjenigen gedankt, die sich hier aus Platzgründen nicht wiederfinden, jedoch meine Promotionszeit in jedweder Form bereichert haben.

Chapter 1

Introduction

Ever since the launch of the first imaging X-ray telescope in 1978 the Einstein Observatory (HEAO-2) [63] X-ray astronomy scientists have been demanding continuously faster, larger, and more sensitive detectors to reach their scientific goals. Semiconductor detectors have made a significant progress in technology since their invention and have become one of the most important detector types today. The unique features of these detectors make them also interesting for nuclear physics, elementary particle physics and, in recent times, even for material investigations and medical applications. The most important feature of silicon semiconductor detectors for imaging spectroscopy is the ability to measure the origin and the energy of a photon with high precision at the same time. In the meantime the design and manufacturing improvements make it even possible to build detectors with a high frame rate, a high spatial resolution, and a large sensitive area.

High frame rates are driven by scientific requirements and detector issues. Precise temporal resolution is essential for the scientific analysis of intensity fluctuations in X-ray sources such as pulsars. Furthermore, a single photon energy resolution is crucial for many applications. To achieve this, the clear distinction of the different photons on the detector is necessary. In high flux environments, for example during the observation of bright X-ray sources, a high frame rate is mandatory. Additionally, higher frame rates are required to reduce the leakage current at a high level of radiation damage and in high temperature environments. Radiation damage, high temperatures and high X-ray fluxes are the most challenging requirements to be met for the upcoming projects.

Higher spatial resolution is essential for astronomy to enable the separation of two objects in the field of view with a small angular distance. The resolution of a telescope is determined by the minimal distance between two separable objects. The spatial resolution of the detector can be increased by reducing the pixel size. A higher resolution enables furthermore the investigation of the fine structure of an observed object with higher accuracy.

A large sensitive detector area is necessary for a wide field of view and to collect as many X-ray photons as possible to gather sufficient statistics. This reduces the time required for the observation of a particular sky area or object. In combination with the requirement for high spatial resolution this leads to a high number of detector pixels.

For an optimal operation of such detectors a high performance **d**ata **acq**uisition (DAQ) system is necessary. In addition, the acquired data have to be processed and analyzed. Without a high-quality DAQ and processing system the maximum performance cannot be extracted from the latest cutting edge detectors. The requirements on the next generation of DAQ systems for X-ray imaging spectroscopy therefore include not only the ability to manage the data stream; it also requires real-time processing capabilities. To cope with the raising requirements, a new DAQ system with real-time processing capabilities was developed, built, and evaluated during this thesis at the Semiconductor Laboratory of the Max Planck Society.

The Semiconductor Laboratory of the Max Planck Society was established in the early 1990s to develop and produce new sophisticated semiconductor X-ray detectors with unequaled performance for all kind of scientific research applications. The detectors developed since then are successfully used in various accelerator facilities around the globe as well as in several satellite missions. However, in order to fully utilize the outstanding performance of the detectors the data acquisition and processing system must be correspondingly powerful.

The processing and full analysis of data obtained from these X-ray detectors is, up to now, performed offline after recording the entire data set by a data analysis software. Such an approach is only useful for detection systems with a low output data rate and without a real-time data processing requirement. Due to the increasing requirements, the frame rate and the number of pixels will increase dramatically with the next generations of X-ray detector systems. A completely new approach for data processing is required for an efficient handling of the data rates in these X-ray detector systems. Processing the entire raw data stream in real time is desirable for laboratory applications and essential for data processing on board of satellites to reduce both the required down-link bandwidth to a ground station and the needed data rate for on-board data storage.

In the framework of this thesis, a new real-time DAQ and processing system for X-ray imaging spectroscopy applications was developed to meet the requirements of the upcoming detectors. The major key points of the new concept are the real-time data processing capability and the scalability of the data processing system. The real-time data processing capability is needed for live monitoring and autonomous operated systems, in particular for satellite applications, while the scalability is required to enable the operation of detector systems continuously growing in number of pixels and frame rate even in the future. A detector system of the new generation, in which these DAQ system functionalities have been realized, is the **M**ercury **I**maging **X**-ray **S**pectrometer (MIXS). The following section will give an overview of MIXS, an instrument on board of the ESA's BepiColombo mission.

1.1 X-ray imaging spectroscopy

State-of-the-art scientific research experiments with X-rays mainly use imaging spectroscopy technology. Standard detector systems can be divided into two classes: firstly, imaging systems, which provide two-dimensional information on the position of an incoming X-ray photon and, secondly, spectroscopic systems, which measure the energy of an X-ray photon, but do not provide information concerning the position. The X-ray imaging spectroscopy combines these two measuring technologies and adds on a new dimension of information to the recorded data. This new dimension increases the complexity of the required data processing algorithms, but allows for new scientific goals. The possibility to calculate the precise element distribution in an X-ray image is one of the most convincing new options. An example is the observation of Tycho supernova (SN 1572) by the **X**-ray **M**ulti-**M**irror - **Newton** (XMM-Newton) [27, 23] Mission telescope. The measured spectrum is depicted in Figure 1.1. The elements distribution map, calculated from the raw data, is illustrated in Figure 1.2 [24] [1]. The ability to calculate the precise element distribution map opens up new scientific possibilities. It allows for obtaining a better understanding of how the universe operates and enables more accurate comparisons to simulations.

A new instrument with X-ray imaging spectroscopy functionality is the **M**ercury **I**maging **X**-ray **S**pectrometer (MIXS) [32, 60, 80, 81]. MIXS has been designed to perform an **X**-**r**ay **f**luorescence (XRF) analysis of the Mercury surface. The instrument is currently in the final development stage and will be one of the instruments on board of the ESA's "cornerstone" mission BepiColombo [5] to Mercury. The launch of BepiColombo is scheduled for 2016 from the Space Center in French Guiana on board of an Ariane 5 rocket provided by Arianespace; it will orbit Mercury from 2022 onwards.

[1] The data is based on observations, obtained from XMM-Newton, an ESA science mission with instruments and contributions directly funded by the ESA Member States and NASA

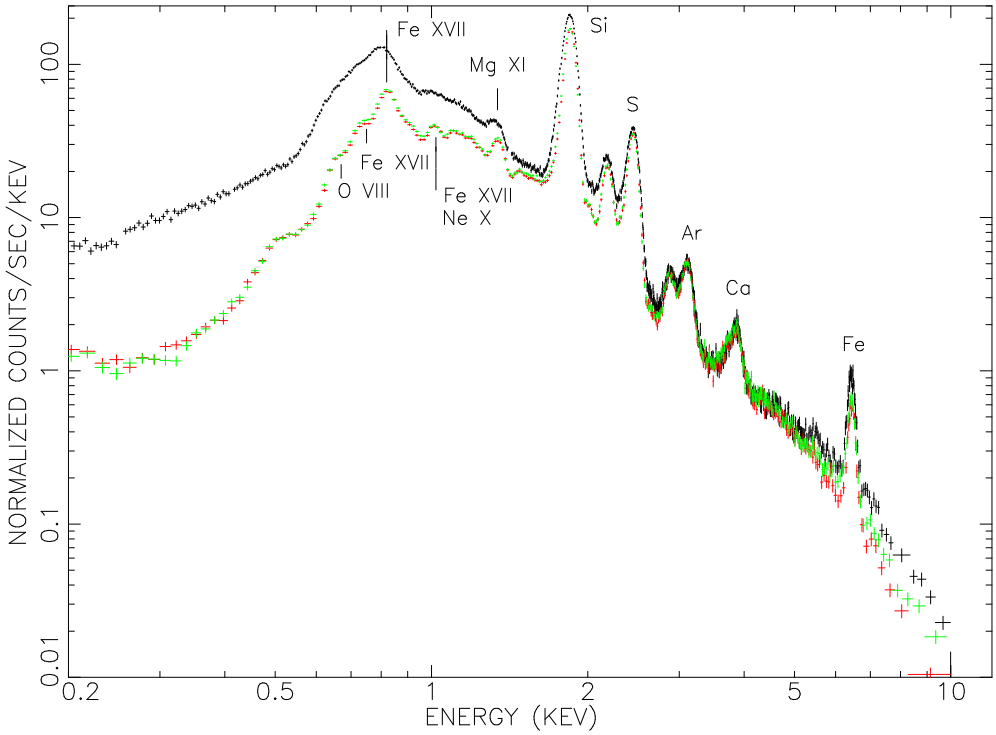
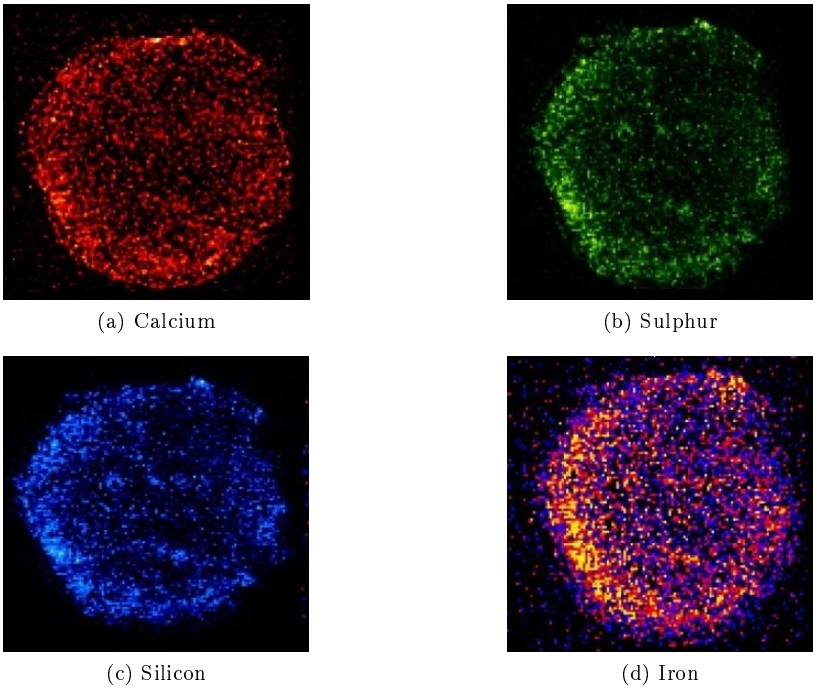


Figure 1.1: The Spectrum of Tycho supernova (SN1572) measured by the **X**-ray **M**ulti-**M**irror - **Newton** (XMM-Newton) Mission. The emission lines in the spectrum are labeled with their correspondent chemical element. Figure taken from [17].



(a) Calcium

(b) Sulphur

(c) Silicon

(d) Iron

Figure 1.2: The element distribution maps of the four selected elements of calcium, sulphur, silicon and iron from Tycho supernova (SN1572). The calculation was performed on data sets extracted from measurements made by the **X**-ray **M**ulti-**M**irror - **Newton** (XMM-Newton) Mission. Figures taken from [24].

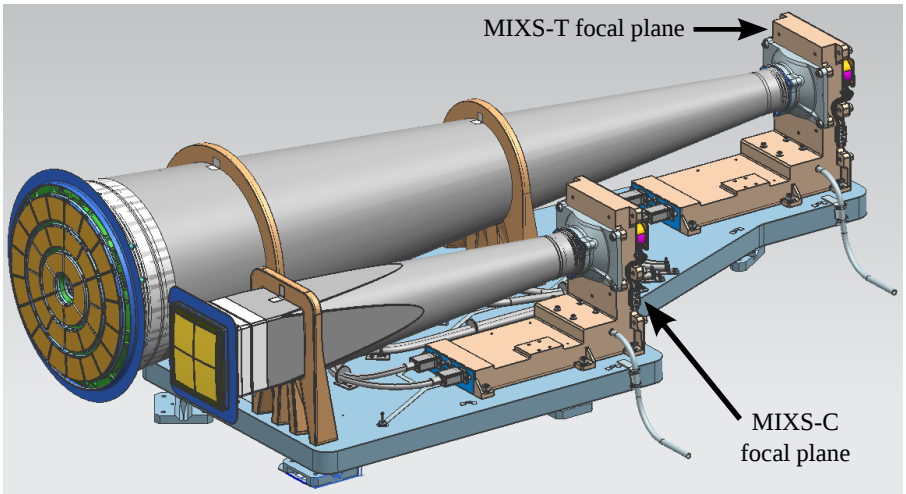


Figure 1.3: The two MIXS channels, MIXS-T and MIXS-C on board of ESA's "cornerstone" mission BepiColombo to Mercury. The instrument MIXS-T is equipped with a high-resolution X-ray telescope, while MIXS-C is an observation instrument which uses a colimator optic to have a large field of view. Figure received from [21].

The instrument consists of two detector channels: one with a large field of view and one with a high spatial resolution for narrow field observations. Figure 1.3 shows a sketch of the two MIXS instruments. MIXS-T and MIXS-C are mounted inside the **M**ercury **P**lanetary **O**rbiter (MPO) and both detector heads face the Mercury surface. MIXS-T has a high spatial resolution, but a small field of view and lower photon collection efficiency. Due to this fact, MIXS-T requires a higher solar X-ray flux from the Sun and is used for imaging the fluorescence radiation of the Mercury surface. MIXS-C is the instrument with a large field of view for planetary mapping and allows for large scale observations in case of a low solar X-ray intensity.

The two MIXS camera heads use an **A**ctive **P**ixel **S**ensor (APS) based on **De**pleted **P**-Channel **F**ield-**E**ffect **T**ransistors (DEPFETs) on the focal plane. The APS sensors are read out with approximately 6,000 frames per second and have a matrix dimension of 64×64 pixels. Each MIXS camera head uses two analog channels for the detector readout to achieve this frame rate. The data from the two analog readout channels have to be converted and processed in real time at this frame rate. The readout speed and the requirement for pixel reordering to reconstruct a complete and consistent image frame from the two input channels makes the MIXS detector system a very interesting platform for the performance evaluation of the new data processing DAQ system.

In addition, MIXS is a technology demonstrator instrument for the **A**dvanced **T**elescope for **H**igh **EN**ergy **A**strophysics (ATHENA+) mission, an L-class mission candidate in the ESA's Cosmic Vision [22] 2015-2025 program. This large X-ray observation telescope will be equipped with a **w**ide-**f**ield **i**mager (WFI) based on silicon active pixel sensor technology. This active pixel sensor with 640×640 pixels allows for a broad-band X-ray survey and imaging in combination with a large field of view. A full frame rate of 780 fps produces an incredible raw data rate of 4,898 Mbit/s on board of the satellite. Compared to this raw data rate, the nominal satellite downlink data rate of only 450 kbit/s is more than 11,000 times smaller. The autonomous handling of such a raw data rate in real time on board of a satellite requires a new data processing concept.

The newly developed real-time data processing DAQ system presented in this thesis was used in the development phase of the MIXS detector system and during a long-term irradiation campaign at the **M**aier-**L**eibnitz **L**aboratory for Nuclear and Particle and Accelerator Physics (MLL) [18].

1.2 Data processing for X-ray imaging spectroscopy

Up to now for the development and testing of DEPFET X-ray imaging spectroscopy detectors a full frame DAQ system in combination with an offline data analysis software has been used. The full frame recording simplifies the DAQ system structure because each digitized pixel value is transferred to a storage system without any further data processing. For this data acquisition approach, standard **c**ommercial **o**ff-**t**he-**s**helf (COTS) hardware components can be used. The use of standard hardware components minimizes the required development effort to build the DAQ system.

In case that no data processing is done at the run time of the DAQ system, it is absolutely necessary to store every pixel data value during a measurement. The drawback of the reduced development effort is the high output data rate and the large amount of data that is required to store a measurement. For previous generations of DEPFET X-ray imaging spectroscopy detectors, where the number of pixels and the frame rate was significantly lower, this was manageable. All the required data processing necessary to analyze the data set was done offline without any real-time processing requirement. This also reduces the demands and complexity of the analysis software.

This old full frame DAQ system and processing concept delivered excellent analysis results and allowed for the operation of DEPFET detector systems on the physical limit for silicon X-ray detectors, the Fano limit. The platform was optimal for algorithm development and optimization. Difficulties with the concept occurred due to the rapid improvement of the DEPFET detector systems and the increasing data rates.

The handling of the full-frame output data for the next generation of DEPFET detectors is inefficient and expensive. Furthermore, the data processing of the large amount of stored data requires a lot of processing power and time. Another limitation of this concept is that no real-time feedback of the current detector condition is available during a measurement. This is important for the optimization of detector parameters.

1.3 Requirements for a new data processing concept

In this thesis, a newly developed real-time data processing concept called **Su**per-**Mo**dular-DAQ (SuMo-DAQ) for the next generation of X-ray imaging spectroscopy detector systems will be presented. This includes novel dynamic processing algorithms specially designed to handle the requirements of the upcoming X-ray imaging spectroscopy systems. This enables the system to automatically adapt the data processing to the specific detector conditions and operation situations. The system hardware used in this thesis has been designed for laboratory operation; nevertheless the platform can be used to demonstrate the ability and advantages of the newly developed algorithms for satellite on-board use. Verification of the system performance is demonstrated by simulations as well as by measurements with various state-of-the-art X-ray imaging spectroscopy detector systems. Intensive tests of the developed SuMo-DAQ system were made during a long-term irradiation test campaign of the **M**ercury **I**maging **X**-ray **S**pectrometer (MIXS) to qualify the flight detector modules.

The upcoming requirements are diversified and depend on the field of application. A summary of the core requirements for such a new concept is described below.

System flexibility and adaptability

The most important point for the development concept is flexibility. The first field of application will be the laboratory environment, where the capability of the system has to be demonstrated before use on board of a satellite can be taken into consideration. In the laboratory environment, a wide range of different X-ray detector systems is available. Each of these detector systems has individual requirements and characteristics in terms of the number of detector pixels, pixel arrangement, amount of readout nodes and readout directions; even the detector type can be different in these setups.

The data processing system needs to be flexible enough to handle all these individual requirements of the detector system efficiently. Additionally, it has to be capable of adding new data processing features to the system for future detector systems.

Scalability and data rate handling

Both the size of detector matrices and the frame rates are growing. It is essential for a new concept to have the ability to handle the fast increase in raw data rate efficiently also for the next generation of detector systems. Fast detector systems use multiple parallelly operated output channels to achieve the targeted frame rates. Therefore, the new DAQ system and the data processing concept must be able to handle multiple concurrently operated data channels. The data processing of each of these input data channels should be independent from information on other channels. Otherwise the system complexity increases significantly due to the communication required between the different processing units.

Data reduction

The handling and storage of the raw data rates produced by the next generation of high-speed X-ray detectors is already a challenging task in laboratory applications. Real-time data reduction can lower the required amount of storage bandwidth and space to reduce the system costs. For satellite applications, the reduction of the data rate to the minimum is mandatory to get along with a very limited satellite downlink bandwidth. Furthermore, data sets which contain only relevant information allow, in addition, a reduction of the processing time for the final analysis.

Real-time feedback and data processing

Real-time data processing is an important feature to allow for live monitoring of detector system performance. The ability to visualize key system parameters in real time during the optimization process of new detectors makes it possible to increase the optimization efficiency. During the normal detector system operation, real-time monitoring enables better detector surveillance and provides feedback of the current detector condition to the system operator. In applications where the detector system is used as a trigger to start the data acquisition of other measurement systems also the processing delay has to be as short as possible.

A new challenging type of application is real-time dosimetry. Up to now the radiation dose placed in the silicon detector is estimated from the expected photon rate and the exposure time. In this configuration the precise radiation dose applied to the detector will not be known until the complete offline data analysis is finished. To measure the applied radiation dose more precisely and in real time the full data stream needs to be processed during the measurement.

Autonomous system operation and self-adapting algorithms

A further requirement is the possibility to react autonomously to parameter drifts and external disturbances of the detector system. An autonomous reaction is essential in a satellite environment, where the possibility of user controllability cannot always be guaranteed and often has a long delay. Disturbances of the system cannot only be generated by temperature or voltage drifts; also infrared and bright light sources in the field of view can require the adaption of data processing. In long-term measurements, parameter drifts have an even greater influence on the measurement precision. This has to be considered in real-time data processing. Dynamically adapting algorithms can also help to reduce the time and number of recalibration cycles required for precise measurements with the detector system. This allows for increasing the available observation time of the detector system.

Chapter 2

Detector and DAQ system fundamentals

The newly developed **Su**per-**Mo**dular-DAQ (SuMo-DAQ) system presented in this thesis is not limited to a specific detector size, speed or type. During the development stage, the system was tested with **C**harge **C**oupled **D**evice (CCD) detectors as well as **De**pleted **P**-Channel **F**ield-**E**ffect **T**ransistor (DEPFET) detectors. However, the majority of the tests and measurements in this thesis where made with a DEPFET detector system. Therefore, the introduction to the X-ray detection in this chapter is focused on DEPFET detector systems. The subchapters 2.1 and 2.2 give an introduction to the DEPFET detector principles and explain how X-rays are measured with semiconductor detectors for a better understanding of this detector type. Additional system components are required to operate a DEPFET matrix; these components are described in subchapter 2.3. Finally, the DAQ system, which has been used for DEPFET detector systems up until now, is presented in subchapter 2.5. This PCI-based DAQ system is used later on as benchmark reference for the new **Su**per-**Mo**dular-DAQ (SuMo-DAQ) system.

2.1 X-ray detection with DEPFET detectors

The most common base material for electronic semiconductor circuits is silicon (Si). The well-established process technology for this material allows for the production of sophisticated silicon X-ray detectors. The principle of X-ray detection with semiconductors for imaging spectroscopy is the electron-hole pair generation in the detector bulk material by incoming photons. These electron-hole pairs are measured and used to calculate the deposited energy.

Figure 2.1 shows a typical analog signal chain for semiconductor X-ray detectors. An absorbed photon creates a cloud of electron-hole pairs in the silicon bulk. The charge is then

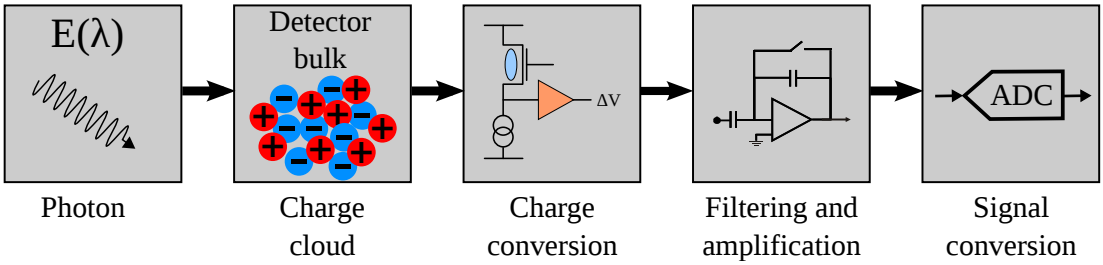


Figure 2.1: An analog signal chain for photon detection with semiconductor detector systems. The charge cloud created in the semiconductor by absorbed photons is converted, filtered, amplified and finally digitized.

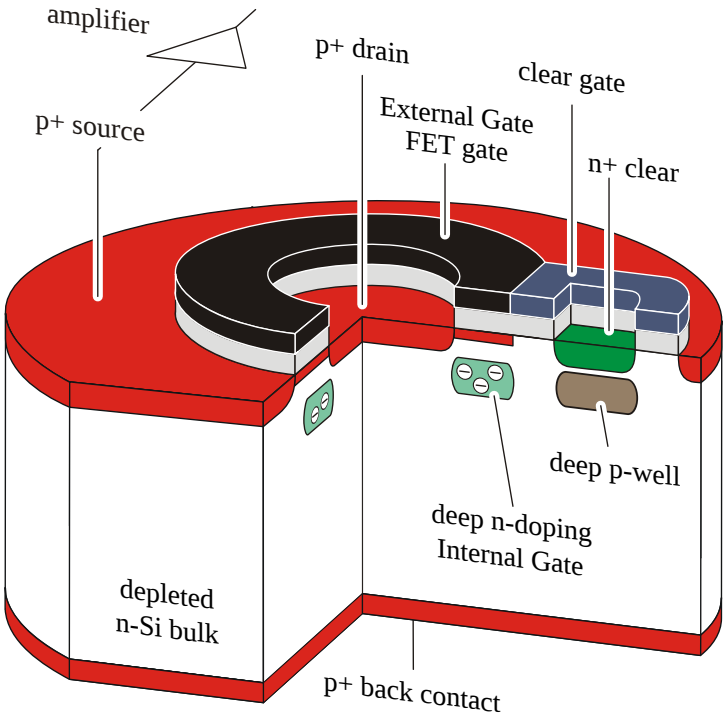


Figure 2.2: The structure of a circular **De**pleted **P**-Channel **F**ield-**E**ffect **T**ransistor (DEPFET) pixel. The "Internal Gate" is situated directly below the "External Gate" or "FET Gate" of the MOSFET, where the MOSFET channel is located.

converted into a current or voltage signal by the detector. This signal is finally filtered, amplified and converted into a digital value for further data processing and analysis. With the knowledge of the physical properties of the used semiconductor material and the number of generated electron-hole pairs the photon energy can be calculated.

2.1.1 Depleted P-Channel Field-Effect Transistor

Depleted **P**-Channel **F**ield-**E**ffect **T**ransistor (DEPFET) detectors are silicon detectors developed and produced at the **H**alb**l**eiter**l**abor (HLL) [39], the semiconductor laboratory of the Max Planck Institute for Extraterrestrial Physics [30] and the Max Planck Institute for Physics [31]. The DEPFET device was introduced by Lutz and Kemmer in 1987 [45]. First measurements with a DEPFET structure were made in 1990 [46] at the HLL. The basic DEPFET structure is depicted in Figure 2.2 and the equivalent circuit in Figure 2.3 [56].

The equivalent circuit in Figure 2.3 shows that a DEPFET is build up from two specially coupled **M**etal–**O**xide–**S**emiconductor **F**ield-**E**ffect **T**ransistors (MOSFET) [77]. The charge collected in the silicon bulk is stored in a region called "Internal Gate". This Internal Gate and its physical location in the device is depicted in Figure 2.2. The Internal Gate is close to the channel of MOSFET (1) and has, therefore, a strong influence on the channel conductivity of MOSFET (1). The "External Gate" contact of the DEPFET is similar to the Gate contact known from a conventional MOSFET structure. The influence of the Internal Gate on the channel conductivity is similar to the influence of the External Gate of MOSFET (1) [56]. If the External Gate is always adjusted in the same way, the current through MOSFET (1) is entirely steered by the charge stored in the Internal Gate. In this configuration, the MOSFET (1) current is directly proportional to the charge collected in the Internal Gate.

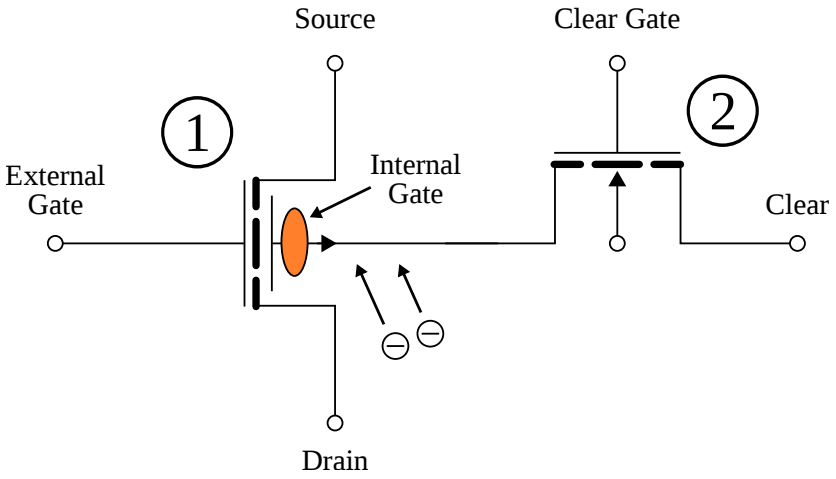


Figure 2.3: A schematic of the equivalent circuit of a DEPFET device. A DEPFET is composed of two tightly coupled MOSFETs and an implanted volume called "Internal Gate". The charge created by incoming photons is collected and stored in the "Internal Gate". This volume is located directly below the conductive channel of MOSFET (1) to enable a strong influence of the collected charge on the conductivity of the MOSFET channel. MOSFET (2) is required to remove the collected charge from the "Internal Gate".

The control function on the channel conductivity of the collected charge stored in the Internal Gate is used to convert the quantity of stored electrons into a proportional current or voltage signal. The amplification characteristic of the MOSFET is directly used in the DEPFET device to create a first signal amplification stage very close to the point where the electrons are stored. This reduces the impact of noise disturbances on the transmission line.

The Internal Gate is created by a special n-doped region in the silicon bulk below the External Gate, of MOSFET (1). A circular DEPFET structure is depicted in Figure 2.2. DEPFETs can be designed in a wide variety of shapes, but the basic concept with the Internal Gate remains always the same. To remove the stored charge from the Internal Gate of MOSFET (1), the second MOSFET, MOSFET (2), is switched into the conductive state. In this configuration, the Internal Gate is connected to the "Clear" contact, which acts as a current sink.

The DEPFET is usually operated under conditions when MOSFET (1) is in the saturation region. Equation 2.1 [75] describes the current of a P-channel MOSFET in the saturation region $I_{DS_{Sat}}$ in dependence of the gate-to-source threshold voltage V_{tp}, the gate-to-source voltage V_{GS}, the electron mobility μ_p, the thin oxide dielectric constant ϵ_{0x}, the transistor thin oxide thickness T_{0x}, the effective transistor gate width W and length L. The current drive strength of a MOSFET is described by the constant K, shown in Equation 2.2.

$$I_{DS_{Sat}} = -\frac{\mu_p \epsilon_{0x}}{2 * T_{0x}} \frac{W}{L} (V_{GS} - V_{tp})^2 \tag{2.1}$$

$$K = \frac{\mu_p \epsilon_{0x}}{T_{0x}} \tag{2.2}$$

The standard P-channel MOSFET Equation 2.1 has to be modified to consider the influence of the charge stored in the Internal Gate on $I_{DS_{DEPFET}}$. The control effect of this charge is similar to the control effect of the V_{GS} voltage. The charge-equivalent control voltage can be calculated with Equation 2.3 [56] for the DEPFET device. Where the stored charge is represented by Q_{Store}, the coupling factor of the Internal Gate to the channel of MOSFET (1) is described by f and the MOSFET gate-channel capacitance as $C_G = WL\frac{\mu_p \epsilon_{0x}}{T_{0x}}$. With this charge equivalent control voltage V_{Sig} the MOSFET current Equation 2.1 can be modified to the DEPFET current Equation 2.4 [56].

$$V_{Sig} = \frac{f * Q_{Store}}{C_G} = \frac{f * Q_{Store}}{WL\frac{\mu_p \epsilon_{0x}}{T_{0x}}} \tag{2.3}$$

$$I_{DS_{DEPFET}} = -K\frac{W}{2 * L}\left(V_{Sig} + V_{GS} - V_{tp}\right)^2 \tag{2.4}$$

For the following considerations, Equation 2.4 is approximated by a first-order Taylor series. For the description of the Taylor series, the derivatives with respect to the variables Q_{Store} and V_{GS} are required. Equation 2.5 calculates the charge amplification factor in dependence of the current through MOSFET (1) I_{DS} and the charge stored in the Internal Gate Q_{Store}. For standard DEPFET structures a common conversion factor is $g_Q \cong 300\frac{pA}{e^-}$.

$$\begin{aligned} g_Q &= \left.\frac{\partial I_{DS_{DEPFET}}}{\partial Q_{Store}}\right|_{V_{GS=const.}, V_{DS=const.}} \\ &= -K\frac{W}{L}\frac{f}{C_G}\left(\frac{f * Q_{Store}}{C_G} + V_{GS} - V_{tp}\right) \end{aligned} \tag{2.5}$$

The second important DEPFET parameter is the transconductance g_m, which is calculated by Equation 2.6 and describes the changes on I_{DS} created by the gate-source voltage V_{GS}.

$$\begin{aligned} g_m &= \left.\frac{\partial I_{DS}}{\partial V_{GS}}\right|_{V_{DS=const.}} \\ &= -K\frac{W}{L}\left(\frac{f * Q_{Store}}{C_G} + V_{GS} - V_{tp}\right) \end{aligned} \tag{2.6}$$

The total current through MOSFET (1) is finally described by Equation 2.7 as first-order Taylor series, where I_0 is the I_{DS} current at the linearization point.

$$I_{DS} = I_0 + \Delta V_{GS} * g_m + \Delta Q_{Store} * g_Q \tag{2.7}$$

The charge stored in the Internal Gate of a DEPFET pixel (Q_{Store}) can be measured in a three-step readout process with Equation 2.7. Figure 2.4 shows a sketch of the I_{DS} current for the readout process. At the time (T1) MOSFET (1) is switched on and the readout process starts. The channel conductance is measured during the time t_1. During the second phase t_2, the so-called clear process, the charge stored in the Internal Gate is removed. Finally a second measurement of the channel conductance is made during the time t_3. The difference between the channel conductance of the first measurement and the second measurement is proportional to the amount of charge, which was stored in the Internal Gate.

There are many further developments and improvements of the DEPFET device. In this thesis, not all of them can be discussed in detail. Therefore, only a short summary of the new features will be given at this point.

Non-linear DEPFET structures are introduced to increase the dynamic range of the device and, on the one hand, allows for distinguishing single photons; on the other hand, they have the ability to measure a very large number of photons without any loss. This is required especially in **F**ree **E**lectron **L**aser (FEL) applications such as the newly built European **X**-ray **F**ree **E**lectron **L**aser (XFEL) in Hamburg. In non-linear DEPFETs, the gain of the MOSFET (1) decreases with the number of collected electrons in the Internal Gate because of a specifically designed charge collection volume.

One device currently used in laboratory measurements has the functionality of a built-in shutter. This modified DEPFET device structure is called **G**ateable **DEPFET** (GDEPFET) [57] pixel. The shutter allows for a device configuration, in which the charge generated in the silicon bulk is not collected in the Internal Gate, while the charge already stored in the Internal Gate remains unchanged. This enables a precise selection of time windows, where the device

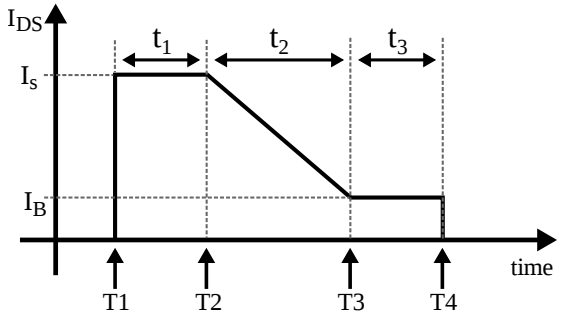


Figure 2.4: A sketch of the basic readout sequence for DEPFETs. The combined base line and signal level is measured during the time window t_1. Charge from the "Internal Gate" is removed during the time window t_2. The base line level is measured during the time window t_3. The difference between I_S and I_B is proportional to the charge stored in the "Internal Gate".

is sensitive or insensitive. An example of application is the investigation of the sodium layer in the upper Earth's atmosphere with a laser beam. Suppression of disturbances from light outside the measurement time window with the laser beam is important to reduce the night sky background and requires a very fast shutter functionality.

"Infinipix" is an enhancement of the DEPFET structure, which integrates two DEPFETs into a single pixel. This allows for observations without an insensitive time window during the pixel readout process. The charge generated in the silicon bulk can be directed into one or the other DEPFET in this pixel. If the first DEPFET is read out, the charge is directed and collected in the second DEPFET and vice versa.

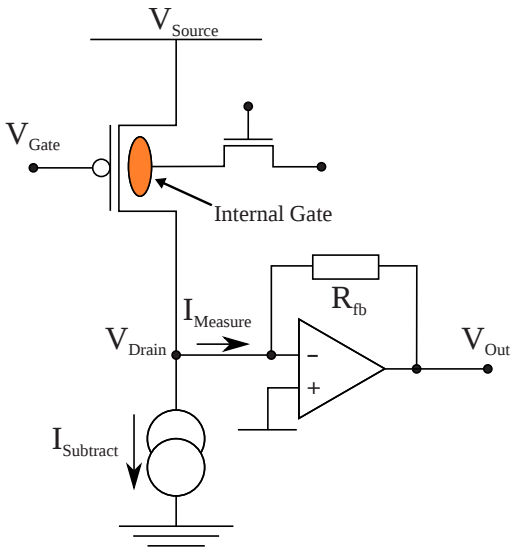
Multiple readouts of the same charge allow increasing the measurement precision with statistical methods. As described above, the charge stored in the Internal Gate is removed from the device during the readout process when a standard DEPFET is used. With the **R**epeated **N**on-**D**estructive **R**eadout (RNDR) [86] [66] structure, the charge collected in the Internal Gate of the DEPFET can be measured multiple times. This is a demonstrated and promising feature to achieve very precise energy resolutions. Measurements [87] show the ability to overcome the $1/f$ noise and allow for sub-electron measurement resolution.

2.1.2 Readout schemes for Depleted P-Channel Field-Effect Transistors

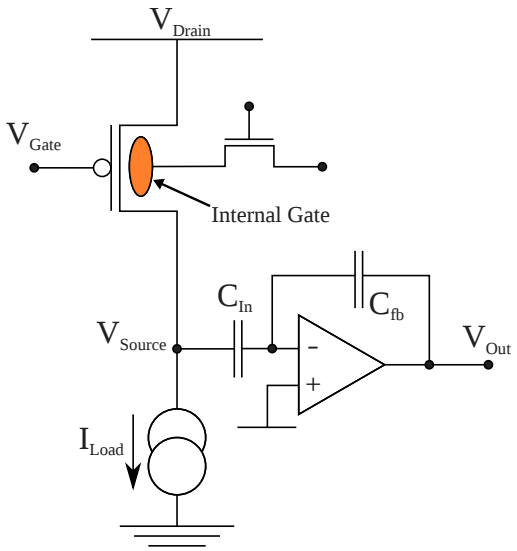
The charge stored in a DEPFET can be determined by two methods, namely the source follower or the drain-current readout configuration. The commonly used method is the source follower readout configuration. The drain current readout structure requires a more complex readout ASIC design. First investigations of this readout configuration and measurements with newly developed ASICs can be found in the literature [8, 9]. Due to the development stage of the drain-current readout ASIC this method was not used in this thesis.

Drain current readout method for Depleted P-Channel Field-Effect Transistors

In the drain current readout configuration the DEPFET channel current is directly measured by a current amplifier. As shown in Figure 2.5a, a current sensitive amplifier and a constant current source are directly connected to the drain contact of the DEPFET. The Op-amp is used to convert the weak current signal from the DEPFET into a voltage signal V_{out}, while the constant current source is used to adjust the operation point of the DEPFET pixel and the current amplifier to an optimal point. The subtracted current does not change the final result, as the output signal only depends on the difference of the DEPFET current before and after clearing the Internal Gate and not on the absolute current value. The change in current depending on the stored charge is described by Equation 2.8. In this equation, $I_{DS,1}$



(a) Drain current readout configuration for a DEPFET pixel. The current source subtracts a constant $I_{Subtract}$ value from the DEPFET channel current. The Op-amp, DC coupled to the source node of the DEPFET, converts the weak current signal from the DEPFET into a voltage signal V_{out}.



(b) Source follower readout configuration for a DEPFET pixel. The current source drives a constant current I_{Load} through the DEPFET pixel. The Op-amp, AC coupled to the source node of the DEPFET, amplifies the weak voltage signal from the DEPFET to the output voltage V_{out}.

Figure 2.5: The schematics of a drain current and source follower readout configuration for DEPFET pixels.

represents the DEPFET current before the clear process and $I_{DS,2}$ the DEPFET current after the clear process. The first-order Taylor approximations for $I_{DS,1}$ and $I_{DS,2}$ are calculated with Equation 2.7 and used in Equation 2.8. $\Delta Q_{Store,1}$ is the charge stored in the Internal Gate before the clear is applied and $\Delta Q_{Store,2}$ is the remaining charge in the Internal Gate of the DEPFET after the clear process.

$$\begin{aligned} \Delta I_{DS} &= I_{DS,1} - I_{DS,2} \\ \Delta I_{DS} &= (I_{0,1} + \Delta V_{GS,1} * g_m + \Delta Q_{Store,1} * g_Q) \\ &\quad - (I_{0,2} + \Delta V_{GS,2} * g_m + \Delta Q_{Store,2} * g_Q) \end{aligned} \tag{2.8}$$

Equation 2.9 can be deduced from Equation 2.8 with $I_{0,1} = I_{0,2}$ and $\Delta V_{GS,1} = \Delta V_{GS,2}$. The assumption $\Delta V_{GS,1} = \Delta V_{GS,2}$ is correct if a genuine relaxation time to stabilize V_{GS} is considered. $I_{0,1} = I_{0,2}$ is a valid assumption, because the transistor itself is not changed during the measurement and the gate-source voltage V_{GS} is also not changed during the drain readout procedure.

If the DEPFET is operated with an optimized set of parameters, the complete charge stored in the Internal Gate is removed during the clear process and, therefore, $\Delta Q_{Store,2}$ can be set to zero. The change in the drain current ΔI_{DS} is then proportional to the charge collected and stored in the DEPFET. The DEPFET-specific amplification factor of the first and built-in amplification stage is expressed by g_Q.

$$\Delta I_{DS} = (\Delta Q_{Store,1} - \Delta Q_{Store,2}) * g_Q \tag{2.9}$$

Source follower readout method for Depleted P-Channel Field-Effect Transistors

The state-of-the-art readout method is the source follower configuration. All DEPFET measurements in this thesis are made with this configuration. The AC coupling of the source follower

readout structure reduces the complexity of the analog readout ASIC and it can handle radiation damages of the DEPFET better. The disadvantage in readout speed can be compensated by increasing the number of parallel readout nodes if required.

The readout structure is depicted in Figure 2.5b. In this readout configuration, the drain current I_{DS} driven through the DEPFET by the constant current source is independent from the amount of charge stored in the Internal Gate. This is represented by Equation 2.10.

$$I_{DS,1} = I_{DS,2} \tag{2.10}$$

With the fundamental Equation 2.7 for the DEPFET drain current and the fact that I_{DS} is the same before and after the clear process, Equation 2.11 can be derived.

$$I_{0,1} + \Delta V_{GS,1} * g_m + \Delta Q_{Store,1} * g_Q = I_{0,2} + \Delta V_{GS,2} * g_m + \Delta Q_{Store,2} * g_Q \tag{2.11}$$

The transistor itself is not changed during the measurement; therefore, $I_{0,1} = I_{0,2}$ is a valid assumption for the source follower readout configuration and Equation 2.12 follows.

$$(\Delta V_{GS,1} - \Delta V_{GS,2}) = (\Delta Q_{Store,2} - \Delta Q_{Store,1}) * \frac{g_Q}{g_m} \tag{2.12}$$

The DEPFET clear process is not influenced by the source follower readout configuration and with the right set of DEPFET operation parameters, the complete charge stored in the Internal Gate is removed during the clear process. For the source follower readout configuration, $\Delta Q_{Store,2} = 0$ is therefore a correct assumption. Equation 2.13 follows if $\Delta Q_{Store,2} = 0$ is applied to Equation 2.12. The voltage step created during the clear process at the readout node can be calculated by means of Equation 2.13. In this equation, $\Delta Q_{Store,1}$ represents the charge collected in the Internal Gate. In combination with some physical properties of the used semiconductor material, $\Delta Q_{Store,1}$ is required to calculate the photon energy.

$$(\Delta V_{GS,2} - \Delta V_{GS,1}) = \Delta Q_{Store,1} * \frac{g_Q}{g_m} \tag{2.13}$$

The factor $\frac{g_Q}{g_m}$ from Equation 2.13 represents the DEPFET gain and is calculated during a calibration measurement with a known reference source. This gain value is defined by the geometry and dimension of the processed MOSFET itself and will, therefore, not change during a measurement.

Time sequence for source follower readout of Depleted P-Channel Field-Effect Transistors

In the time sequence of the DEPFET operation the accumulation and the readout phase have to be distinguished. During the accumulation phase, the DEPFET control voltages remain constant in a setting where no current follows through the DEPFET pixel. The accumulation phase is typically much longer than the readout phase and a switched off DEPFET helps to reduce the power consumption in this phase. For the readout phase the DEPFET control voltages are changed to switch the transistor into the conductive state and allow measuring and removing the collected charge. The complete readout process for a DEPFET pixel in the source follower configuration is depicted in Figure 2.6.

A three-step measurement procedure is typically used to measure the charge stored in the Internal Gate of the DEPFET. The DEPFET is first switched on at t_0 by applying the DEPFET Gate voltage. After a certain settling time the combined signal and base line level is measured during the time interval t_{m1}. The second so-called clear step starts at t_3, where the voltage at the DEPFET Clear contact is switched to the high level. After the settling of the DEPFET Clear voltage the Cleargate voltage on the DEPFET is switched to the high level. This voltage configuration applied to the DEPFET pushes the charge stored in the Internal Gate into the DEPFET Clear contact depicted in Figure 2.3. During the time interval t_C the charge stored

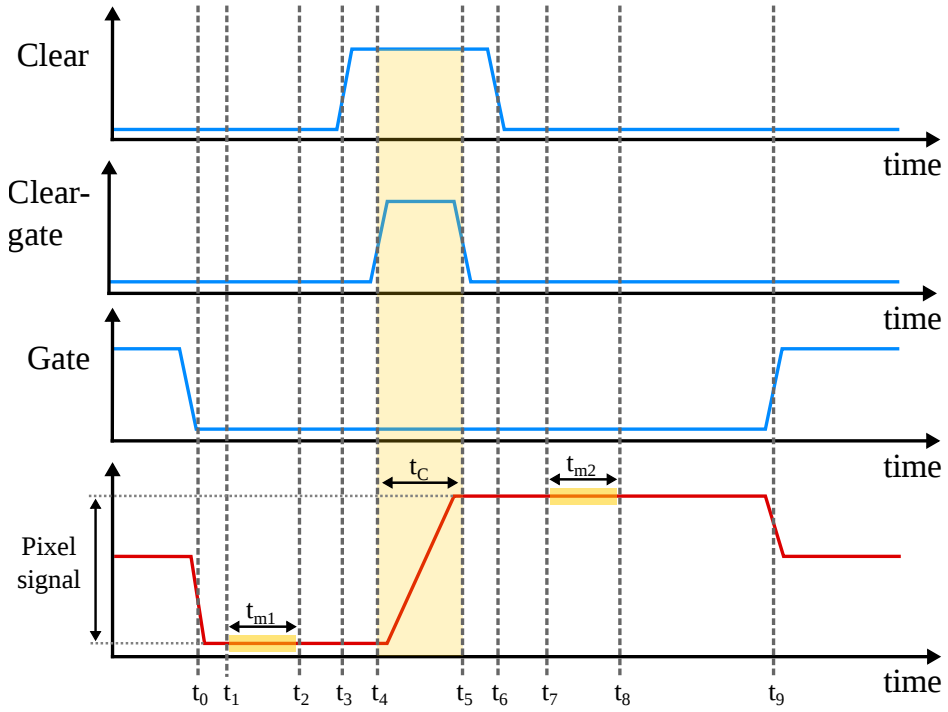


Figure 2.6: Time sequence of a DEPFET readout and the resulting analog output signal for the source follower configuration. In this example, the DEPFET analog output signal shows an "Internal Gate" filled with charge. The stored charge is removed during the time interval t_C. Measurements of the analog output level are made during t_{m1} and t_{m2}.

in the Internal Gate is completely removed from the DEPFET. The Cleargate and the Clear voltages are then switched back to their low levels. The third and final step is the second measurement phase; only the pixel base level is measured now. The amount of charge, which was stored in the Internal Gate $Q_{Store,1}$, can then be calculated by the Equation 2.13 with the values $V_{GS,1}$ measured during t_{m1} and $V_{GS,2}$ measured during t_{m2}.

2.1.3 Depleted P-Channel Field-Effect Transistor matrices

A DEPFET single pixel was explained in the previous chapters 2.1, 2.1.1 and 2.1.4. This chapter will describe how such a single pixel can be used to create a complete DEPFET detector matrix.

Figure 2.7 shows an example of a 3×3 DEPFET matrix. The routing of the interconnection signals in this configuration allows for an expansion of the DEPFET matrix by placing the DEPFET single pixel block, which is depicted in Figure 2.7, on a fixed 2D grid. The three control signals of clear, gate and cleargate, which are required to control the readout process, are horizontally routed. The analog signals of drain and source, which are required to measure the stored charge, are routed vertically. In this configuration, all control signals can be connected from the left or right side of the DEPFET matrix and all analog signals used for the charge readout are accessible from the top or bottom side of the matrix. Row by row connection of the clear, gate and cleargate signals reduces the number of required switches to control the DEPFET matrix. This not only reduces the number of required bond connections between the ASIC and the matrix, but also the power dissipation of the control ASICs. This configuration can also be used to build large matrices, because the signal routing on the silicon die and the number of required signal lines are independent from the matrix size. Selecting an entire detector row allows, in addition, for the parallel readout of all these DEPFET pixels, which increases the possible matrix readout speed.

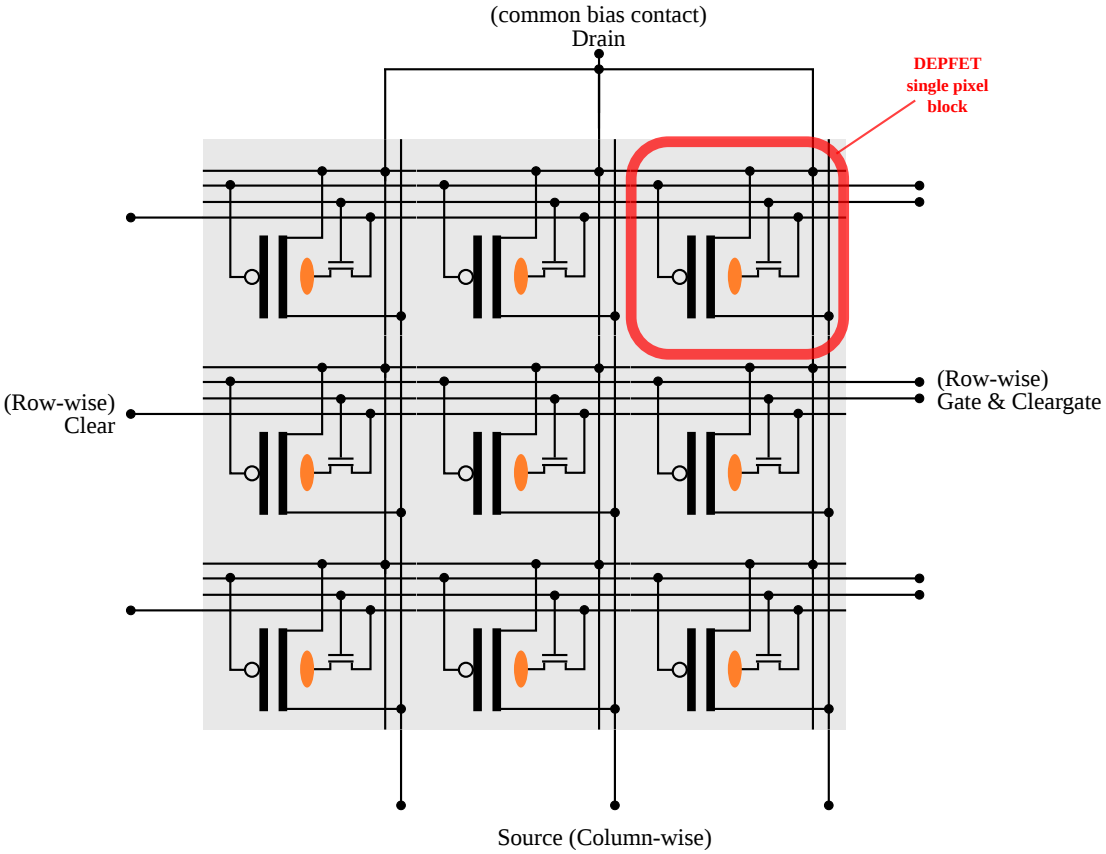


Figure 2.7: Example of a 3×3 DEPFET matrix composed of DEPFET single pixel blocks. Interconnection routing for DEPFET control signals is done horizontally; analog readout signals are routed vertically. A DEPFET matrix built with such a structure can be controlled from left and right and read out from top and bottom.

An overview of different DEPFET detector matrices in use or under development with their readout configurations are shown in Table 2.1. Matrix size, frame rate and pixel size have to be adapted for each mission depending on their requirements and scientific goals. A comparison of the different matrix sizes on a 6-inch wafer is shown in Figure 2.8 [51, p. 60].

2.1.4 Macropixel Depleted P-Channel Field-Effect Transistor

The two main goals while operating a DEPFET X-ray detection system are the precise measurement of the position and energy of each incoming photon. The energy of a photon has to be determined from the measured signal amplitude, while the position is given by the pixel coordinate where the charge was collected. The achievable spatial resolution is, therefore, mainly determined by the DEPFET pixel size.

The size of a Depleted P-Channel Field-Effect Transistor, as described in Chapter 2.1.1, is typically only a few square micrometres. With the spatial resolution of today's X-ray optics, these small detector pixel sizes cannot be used reasonably in most applications. To achieve a wide field of view, which is a central requirement for most X-ray telescopes, a large sensitive area is needed to compensate for the X-ray optic lag. However, with each additional detector pixel the required electric and cooling power for the detector system is generally increased. To get the optimum ratio between spatial resolution and power consumption, which is especially important for satellites, the pixel size needs to be adapted to the application requirements. Therefore, the possibility to adjust the size of the sensitive DEPFET area is essential.

Mission name	Matrix size	Frame rate (fps)	readout configuration	DEPFET type
Simbol-X	64×64	1,200	1 Ch source follower, 1 hemispheres	500 μm macropixel
MIXS	64×64	6,000	2 Ch source follower, 2 hemispheres	300 μm macropixel
ATHENA+	640×640	780	10 Ch source follower, 2 hemispheres	130 μm macropixel
IXO	1,024×1,024	1,000	16 Ch source follower, 2 hemispheres	100 μm macropixel

Table 2.1: Overview of currently available and planned DEPFET matrices for planetary and astrophysical satellite missions.

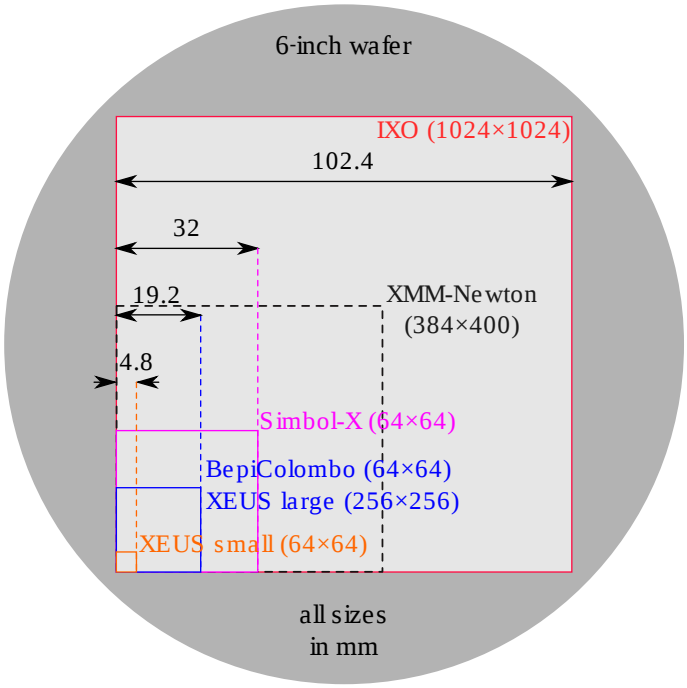


Figure 2.8: Comparison of different DEPFET detector matrices and their physical die size on a 6-inch wafer. The number of pixels is printed in parentheses for each matrix. Dependent on the project requirements, the pixel size can vary. Therefore, the physical detector die size is not directly given by the number of pixels. As a reference the size of the CCD on board the XMM-Netwon telescope is also shown in this figure.

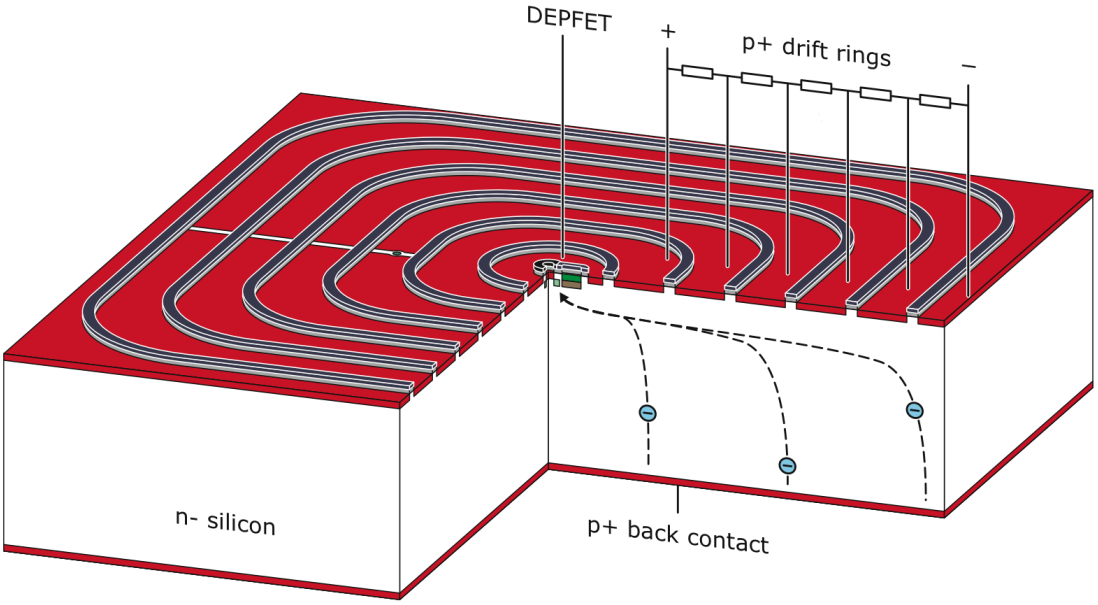


Figure 2.9: Sketch of a DEPFET macropixel. The DEPFET standard structure is in the center of the pixel and surrounded by the drift rings. The electric potential in the silicon bulk generated by these drift rings pushes all electrons below this structure into the "Internal Gate" of the DEPFET. Figure received from [52].

A solution for this pixel size adaptation is depicted in Figure 2.9. The so-called Drift Rings placed on the silicon surface create an electric potential distribution in the silicon bulk suitable to push all electrons generated in the volume below these Drift Rings into the center of the pixel, where the electrons are collected in the Internal Gate of the DEPFET. This approach allows the adaptation of the pixel dimension exactly to the size required by the particular application. In satellite applications the optimal pixel size is typically determined by the achievable resolution of the used X-ray optics. Macropixel DEPFET detectors currently use pixel sizes between $60\,\mu m^2$ and $1\,cm^2$. With this Drift Ring approach not only the depicted rectangle pixel shape is possible, but also circular and hexagonal pixel shapes can be designed. In larger pixels the drift distance for the generated charge is longer and, therefore, a complete charge collection in the DEPFET structure requires more time. For the high-speed XFEL detector system, a hexagonal pixel shape instead of a conventional rectangle pixel shape was chosen to reduce the maximum charge drift distance.

Further reduction in the number of required pixels, without losing spatial resolution for a particular X-ray optic, can be realized by **s**ubpixel **e**vent **r**epositioning (SER) [4, 54] algorithms. This improvement in position resolution despite a reduced number of pixels can be used for further reduction of the required power for the detector system on board of a satellite.

2.2 X-ray spectrum and detector characteristics

A reference source with a known spectrum and with peak energies is needed to determine the energy resolution of a detector system. It is typical to use a ^{55}Fe source for semiconductor detectors because of the well-suited radioactive half-life and the peak energies for this application. This ^{55}Fe source provides a spectrum with two quasi mono-energetic fluorescence lines MN-K$_\alpha$ at $5,899$ eV [79] and MN-K$_\beta$ at $6,490$ eV [79]. These two energies are in an interesting energy band for X-ray astronomy. A typical energy calibrated data spectrum from a DEPFET measurement with such a ^{55}Fe calibration source is shown in Figure 2.10.

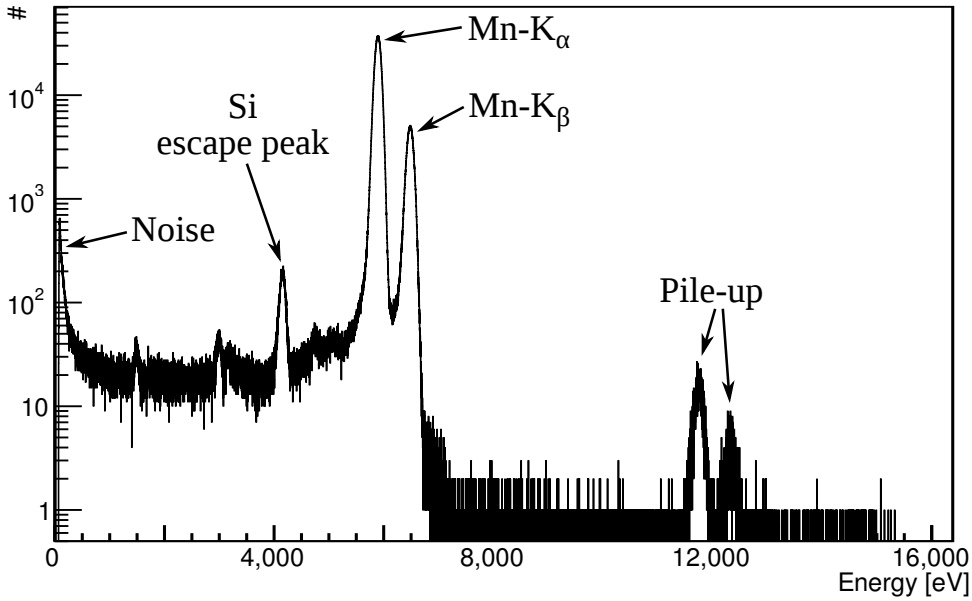
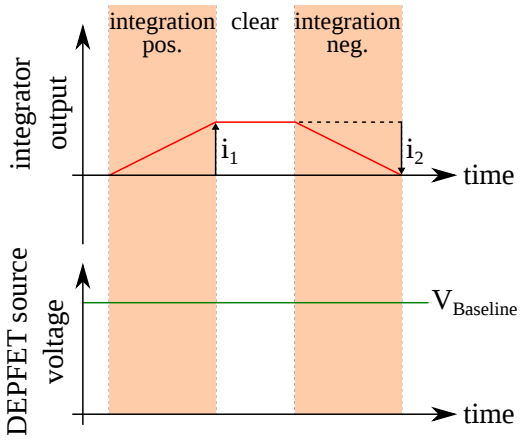


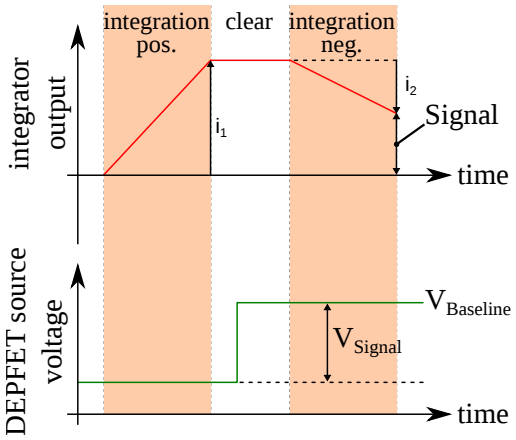
Figure 2.10: Energy-calibrated data spectrum of a DEPFET measurement with a ^{55}Fe calibration source. The noise, Si escape, pile-up and the two signal peaks Mn-K$_\alpha$ and Mn-K$_\beta$ are labeled in the spectrum. The increased level between the noise and the Si-escape peak is called background. The shown pile-up peaks are created by multiple photon hits in a single pixel.

Figure 2.10 exhibits the two characteristic signal peaks Mn-K$_\alpha$ and Mn-K$_\beta$, the noise peak, the pile-up peaks and the Si-escape peak for a ^{55}Fe reference measurement. The Si-escape peak is created by fluorescence photons, which are escaping from the silicon detector. During the absorption of an incoming photon, a fluorescence photon can be generated. If this photon escapes from the detector, the energy E_f carried away by this photon is not measured. The measured energy in this case is $E_m = E_0 - E_f$, where E_0 is the original energy of the incoming photon. For silicon the most prominent fluorescence energy E_f is Si-K$_\alpha$ at $1,740$ eV. The pile-up peaks are created by multiple photon hits in the same pixel in a single frame. The increased level between the noise peak and the Si-escape peak is called background. This background level is created by a combination of noise, charge loss and split events, where multiple entries with a fraction of the total photon energy are added to the spectrum. The background level can be reduced if an event recombination, where partial events are merged together, is done during the data processing phase. For rating of this background level the **p**eak-to-**b**ackground (P/B) value is calculated from the pixel gain corrected data spectrum during the data analysis. This P/B value is the ratio between the highest signal peak and the mean value in a certain interval in the low energy region. The default range of ROAn for this interval is between 900 eV and $1,100$ eV.

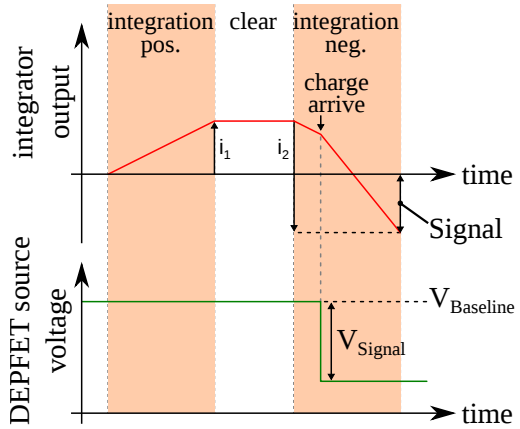
All data values below zero in the offset-corrected raw data spectrum are created by noise or misfit events. A so-called misfit occurs if a photon arrives while the pixel is read out. With the three-step readout sequence described in Section 2.1.2, positive and negative misfits have to be distinguished. In order to reduce high-frequency noise during the measurement, the signal is low-pass filtered by an integrator. For this three-step readout sequence, a positive and a negative integration phase is used to measure the difference in the DEPFET channel conductivity before and after the DEPFET clear. The difference is proportional to the charge which was stored in the Internal Gate of the DEPFET. In Figure 2.11 the signal sequence of the DEPFET source node and the integrator stage are depicted in detail for the four possible cases of an empty DEPFET, a normally filled DEPFET, a positive misfit and a negative misfit. Figure 2.11a shows the signals for the DEPFET readout if no charge was collected in the DEPFET during



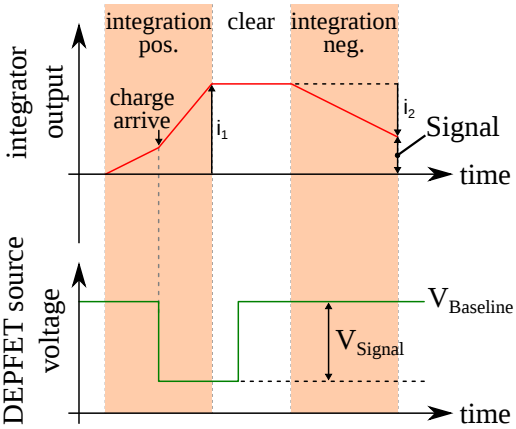
(a) Readout of a DEPFET when no charge was collected.



(b) Readout of a DEPFET when the charge of a photon was normally collected.



(c) Negative misfit.



(d) Positive misfit.

Figure 2.11: Signal sequences of the DEPFET source and the integrator output voltage during the DEPFET readout. Figure similar to [51, p. 109]

the accumulation phase. The DEPFET source voltage is not changed by the clear because there is no charge to remove. Therefore, the integrator output signal is zero after the first positive and the second negative integration phase. Figure 2.11b shows the case where the DEPFET has collected the charge of a photon during the accumulation phase. For the source follower readout configuration the DEPFET source voltage is increased after the clear phase in this case. This leads to a positive integrator output after the positive and negative integration phase. Figure 2.11c shows the case of a negative misfit. At the beginning of the readout phase the DEPFET is empty in this example and during the negative integration phase charge arrives in the DEPFET and changes the DEPFET source voltage during the integration phase. This change results in a negative output signal of the integrator. The charge is finally measured correctly during the next readout phase of the DEPFET. The sensitive time window for negative misfits in the readout sequence depicted in Figure 2.6 is between t_5 and t_8. Figure 2.11d depicts the case of a positive misfit. In this example the DEPFET is empty at the beginning of the readout sequence and during the positive integration phase charge arrives in the DEPFET and changes the DEPFET source voltage. This results in a positive output signal of the integrator stage, but the value is smaller than if the same amount of charge would have been collected during the accumulation phase. The precise influence on the output signal of this additional

collected amount of charge depends on the exact arrival time during the integration phase. The later the charge is collected in the Internal Gate, the lower is the influence of this charge on the output signal. During the clear phase, the charge is removed from the DEPFET and can, compared to the negative misfit, not completely be measured during the next readout phase. In the readout sequence depicted in Figure 2.6 the time window, where the detector system is sensitive to positive misfits, is between t_1 and t_2. To reduce the probability of virtual charge losses by positive misfits, this time window should be as small as possible. The output signal from a positive misfit can, compared to negative misfits, not be distinguished from a normal photon signal. Therefore, the detection of positive misfit events in a real measurement is not possible.

For further scientific use of the measured raw data values a calibration of the data values is needed. After preparing the data set by several data processing steps, the factor to convert the detector signal into the absolute photon energy can be calculated from a known signal peak in a reference measurement. A ^{55}Fe source is typically used as a benchmark for the comparison of different detector types and their possible energy resolution. The precision of the energy measurement is specified by the full width half maximum (FWHM) of a Gaussian curve fitted to the Mn-K$_\alpha$ peak in the measured spectrum. Equation 2.14 [83] describes the relationship of the standard deviation σ and the FWHM of a Gaussian curve. The derivation of this relation is shown in Appendix A.4.1.

$$\mathrm{FWHM} = 2\sqrt{2\ln(2)}\sigma \approx 2.3548 \cdot \sigma \tag{2.14}$$

For semiconductor detectors the best achievable energy resolution is limited by the statistical fluctuation in the numbers of generated electron-hole pairs. The average number of electron-hole pairs $\langle N \rangle$ created in semiconductors by an absorbed photon with the energy E_{Photon} can be calculated with Equation 2.15 in dependency of the mean electron-hole pair creation energy w of the semiconductor material.

$$\langle N \rangle = \frac{E_{Photon}}{w} \tag{2.15}$$

The Fano factor F [28] allows for calculating the variance $\langle \triangle N^2 \rangle$ of the created electron-hole pairs from the mean number of generated electron-hole pairs$\langle N \rangle$. Equation 2.16 [48] describes this ratio. Strictly speaking, the Fano factor F depends on many parameters such as detector material, temperature and photon energy, but usually the ratio is treated as a constant for a specific detector material. For silicon semiconductors is $F = 0.115\,(300\,K)$ [1] and the mean energy for electron-hole pair generation is $w = 3.62\,\mathrm{eV}\,(300\,K)$ [48].

$$\sigma = \sqrt{\langle \triangle N^2 \rangle} = \sqrt{F * \langle N \rangle} = \sqrt{F * \frac{E_{Photon}}{w}} \tag{2.16}$$

For a certain photon energy E_{Photon} the best achievable FWHM is calculated by means of Equation 2.17. This Fano-limited FWHM is the lower physical boundary for the achievable energy resolution. To get the result in [eV] instead of electron-hole pairs, σ is multiplied by w in this equation.

$$\begin{aligned} \mathrm{FWHM}_{FanoLimit}[\mathrm{eV}] &= 2\sqrt{2\ln(2)}\sigma \cdot w \\ &= 2\sqrt{2\ln(2)}\sqrt{\frac{F \cdot E_{Photon}}{w}} \cdot w \end{aligned} \tag{2.17}$$

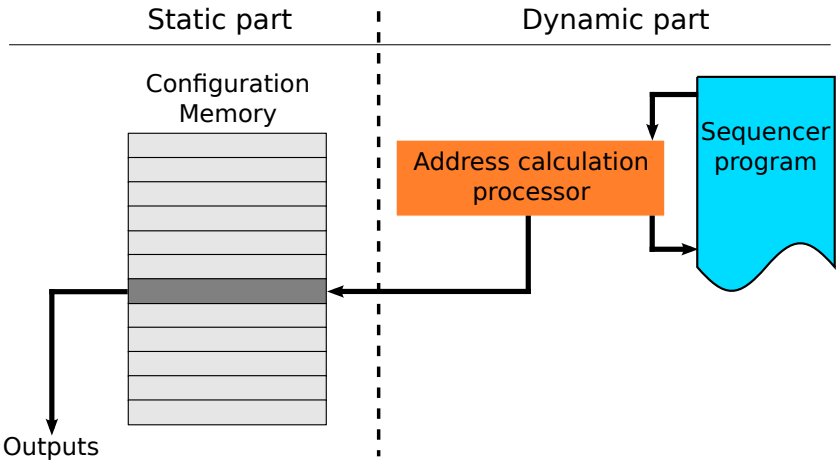


Figure 2.12: Block diagram of the "i-Seq" (**i**ntelligent **Seq**uencer) structure implemented on the X-Board to generate control sequences for the DEPFET detector system. The sequencer is divided into two parts: a static part, where all occurring output combinations are stored and the dynamic part, where the memory address with the next output combination is calculated. The sequence is described by the sequencer program, written in a customized sequencing language.

The best theoretically achievable energy resolution with DEPFET detectors or, more general, with silicon semiconductors for the Mn-K_α peak at 5,899 eV [79] is calculated in Equation 2.18. The Energy resolutions reached in real measurements with the DEPFETs are only a few eV above this theoretical limit; such detector systems are usually called Fano-limited.

$$\begin{aligned} FWHM_{\text{MN-K}_\alpha}[\text{eV}] &= 2\sqrt{2 * \ln(2)} * \sqrt{0.115 * \frac{5,899}{3.62}} * 3.62 \\ &\approx 116.7\,\text{eV} \end{aligned} \tag{2.18}$$

2.3 Detector system components

To create a complete detector system, a sequencer unit, an analog readout ASIC and a steering ASIC is needed besides the DEPFET matrix to create a complete detector system. The functionality and the integration of these components in the detector system is described in this subchapter.

2.3.1 Sequencer unit

For a synchronous operation of all detector system components and the digitization stage a common control unit is required. For this task the "X-Board" [37] was developed at the Semiconductor Laboratory as a programmable and universal sequencing unit. Core component is a FPGA, where all sequencer output channels are synchronously generated. In addition, the FPGA contains all the required programming and configuration interfaces to control the different detector system components. As the X-Board is the central control unit for the complete hardware, the user interface software directly communicates with the X-Board in order to control the entire detector system. This communication is currently done via a standard USB interface. With the next version of the sequencer, this will be changed to an Ethernet interface for a higher flexibility.

The latest FPGA firmware called "**i**ntelligent **Seq**uencer" (i-Seq) [36] can be separated into two parts as shown in Figure 2.12: a static configuration memory, where the used output combinations are stored and a programmable address calculation unit to select the active output

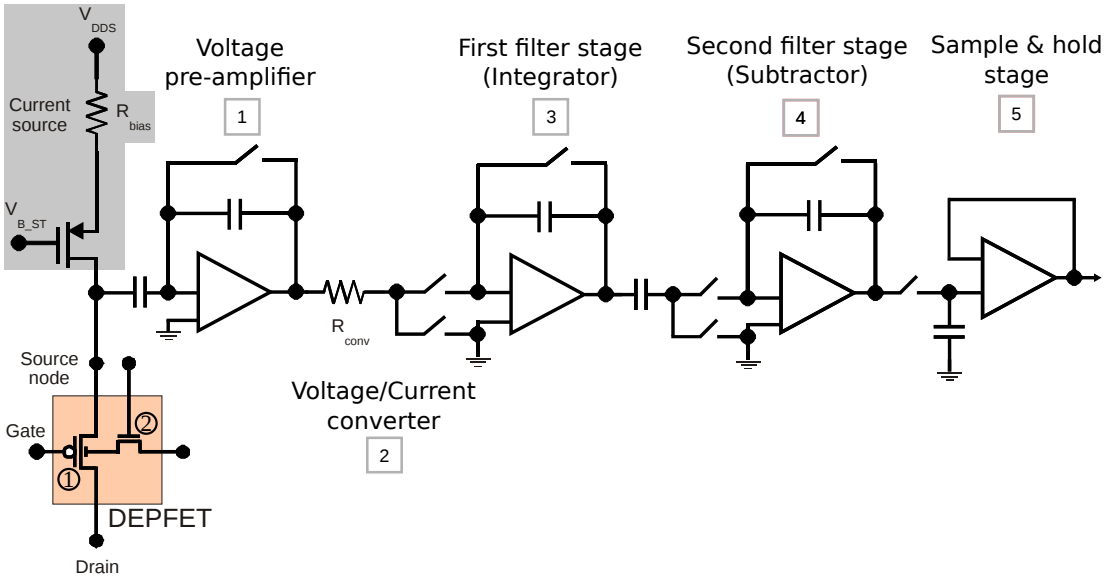


Figure 2.13: Block diagram of a single analog channel of the ASTEROID readout ASIC for the DEPFET source follower readout configuration. Stage (1) is an AC-coupled input amplifier followed by a voltage-to-current converter as a second stage (2). The third stage (3) integrates this current for a certain time during the two measurement windows. In stage (4) the difference of these two measured signal levels is calculated. Finally, a sample and hold stage (5) drives the analog channel output.

combination. This structure is optimal to control a DEPFET detector system, where approximately 60 different output combinations are required, while the sequence itself is much longer and contains many repetitions of these output combinations. The base clock frequency of the FPGA is currently 80 MHz. For most DEPFET systems this is too fast. For the pattern generation the i-Seq has, therefore, a clock divider option. The sequencer working frequency $f_{Seq} = \frac{80\,\mathrm{MHz}}{n+1}$ is derived from the base clock frequency and adjusted by the divider factor $n \in \mathbb{N}_0$. A more detailed description of the X-Board and the i-Seq with a list of all commands is available in [37],[36].

2.3.2 ASTEROID analog readout ASIC for Depleted P-Channel Field-Effect Transistors

The second stage in the analog signal chain for a DEPFET detector system (Figure 2.1) is the **A**ctive current **S**witching **T**echnique **R**ead **O**ut **i**n X-ray spectroscopy with **D**EPFET (ASTEROID) [35] analog readout ASIC. This ASIC is the interface between the DEPFET pixel and the following digitization stage of the detector system and includes the complete analog signal processing.

The analog readout ASIC ASTEROID is the replacement of the old **C**harge **A**mplifier **M**ultipl**EX**er (CAMEX) ASIC and implements the source follower readout structure with a trapezoidal weighting function. This weighting function represents the theoretically optimal filter for time-limited measurements under the presence of white series noise [67]. The ASTEROID is normally mounted close to the DEPFET detector matrix to minimize disturbances on the analog interconnection lines and to achieve the best possible signal-to-noise ratio. Each analog input channel of the ASIC is directly connected to the DEPFET matrix via a bond wire.

In Figure 2.13, a block diagram of one ASTEROID channel is depicted. Each channel includes a current source, as shown in Figure 2.13 (top left), to force a constant current through the MOSFET (1) of the DEPFET below. The voltage at the source node then depends on the

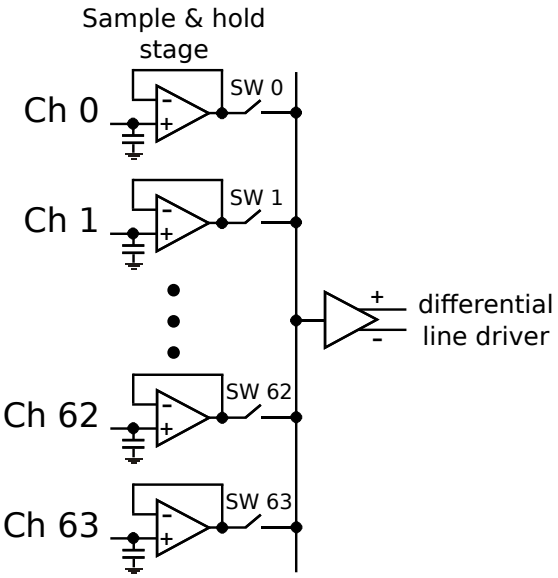


Figure 2.14: Block diagram of the analog serialization stage of the ASTEROID readout ASIC. This figure shows only the last part of all 64 analog channels of the ASIC, namely the sample and hold stage. The output of the sample and hold stage can be connected via a channel-individual switch to the ASIC internal analog bus. A differential output driver provides the analog bus level on the transmission line to the digitization unit of the data processing system.

conductivity of the DEPFET. The further analog signal processing chain inside the ASTEROID can be divided into five consecutive stages and implements the readout sequence exhibited in Figure 2.6. The first ASTEROID stage (1) is an AC-coupled pre-amplifier for further amplification of the weak signal from the DEPFET pixel. Stage (2) is a voltage-to-current converter and drives the following integrator stage (3). The difference between the first integration, where the sum of base line and signal is measured and the second integration time, where only the base line is measured, is calculated in stage (4). The last stage (5) is a sample and hold circuit. This stage allows operating the system in the so-called "duplex mode", where an output signal value is sampled by the digitization stage, while the analog signal chain already processes the next pixel readout.

Currently available versions of the ASTEROID ASIC are equipped with 64 of these analog channels. All these channels are operated in parallel, allowing to read out 64 pixels simultaneously. To reduce the number of the digitizing units required, a serialization stage is integrated on the ASIC in the following processing stage. This allows the use of a single analog output driver and output line for all of the 64 analog channels. The signal level for each channel is driven for a certain time window on the analog output line. The serialization stage of the ASIC is depicted in Figure 2.14. The output driver of each sample and hold stage is connected to an internal bus via an analog switch. This internal bus is connected to a more powerful and differential output driver. The differential output is used to reduce the signal sensitivity to external disturbances on the transmission line with the output voltage limited by the ASIC technology to ± 1.7 V. The output of this driver is connected to the transmission line for the digitization unit of the data processing system. Nevertheless, the performance of this output driver is currently the most limiting factor for a readout speed increase. Faster readouts lead to **i**ntersymbol **i**nterference (ISI) of consecutive pixel values and, therefore, decrease the achievable energy resolution.

The ASTEROID ASIC contains a small internal configuration memory in order to control and steer all internal switches during the readout sequence. The sequencer described in Chapter 2.3.1 steers the entire readout process and also controls the currently active row in the

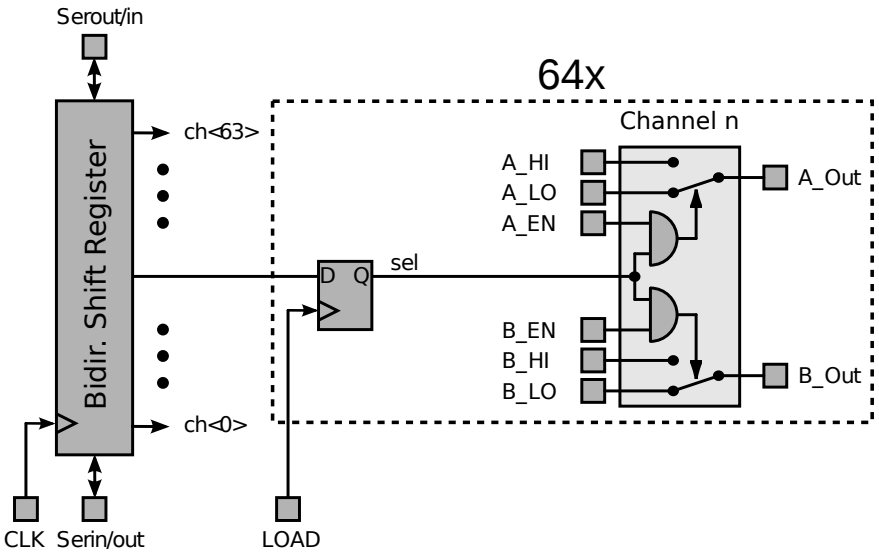


Figure 2.15: Simplified block diagram of the switcher control ASIC. This figure shows one of the 64 dual-port analog multiplexer channels, the bidirectional shift register to control the channels and the channel-individual D flip-flop to store the particular output state.

configuration memory. Changes in the readout sequence and speed are, therefore, completely controlled by the X-Board sequencing unit.

2.3.3 Switcher ASIC for detector matrix steering

The readout sequence described in Chapter 2.1.2 requires the ability to independently switch the clear, cleargate and gate signals for each detector row. The so-called Switcher ASIC is a 64×2 channel analog multiplexer ASIC, specially designed for the row steering of a DEPFET sensor matrix.

A simplified block diagram of the latest Switcher developmental stage, the Switcher-S control ASIC, is depicted in Figure 2.15 [29]. This figure shows one of the 64 dual-port channels implemented on the ASIC. Each of these channels is equipped with two analog output multiplexer ports, A and B. A_HI and A_LO are the two output levels for port A, which are externally provided and selectable by the multiplexer. For port B it is the same with B_HI and B_LO. To enable the selection of an arbitrary number of active channels out of the 64 ones, a bidirectional shift register is used. For the channels selected by the bidirectional shift register the two global signals A_EN and B_EN allow for switching the multiplexer channels A and B between the two possible output levels A_HI/B_HI and A_LO/B_LO. The bidirectional shift register is also required to adjust the detector readout direction and to allow for the mounting on both sides of the detector matrix. An example configuration is shown in Figure 2.16 [29]. In addition, the output signal of the bidirectional shift register can be passed to neighboring ASICs for the creation of a daisy chain. This is needed for the readout of large detector matrices, where 64 output channels are not sufficient to steer the complete matrix. Therefore, the shift register output signals SO_TOP at the top side of the chip and SO_BOT at the bottom side as well as the two shift register input signals SI_TOP and SI_BOT are available.

For a higher flexibility in selecting and deselecting channels, the bidirectional shift register does not directly control the output driver of the N channels. An additional D flip-flop controlled by the LOAD signal stores the current output configuration for each channel and decouples the pattern generation from the output driver. This allows implementing special readout schemes, such as the window mode. The window mode was introduced to read out a region of interest on the detector matrix with a higher rate, while the rest of the detector

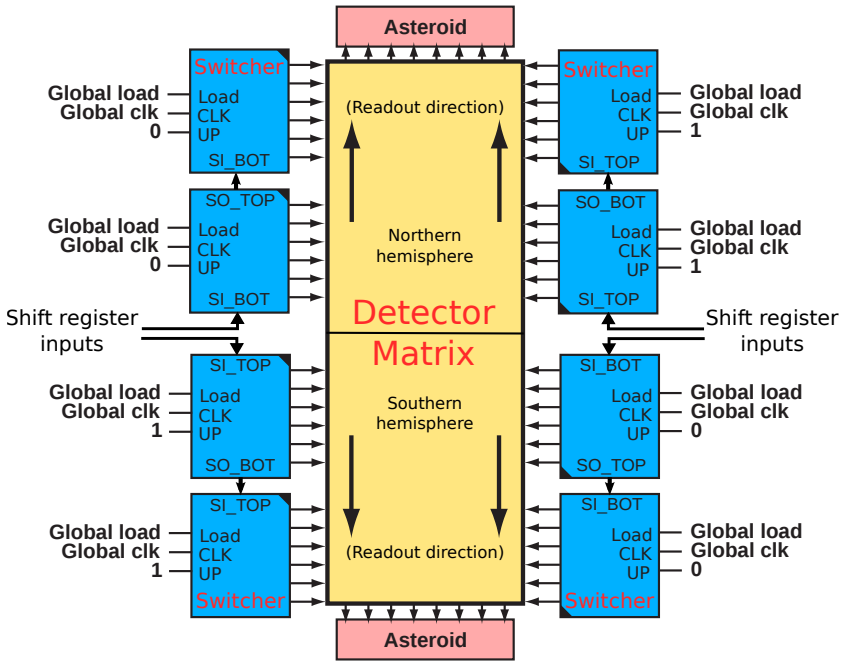


Figure 2.16: Configuration example for a detector matrix readout in two directions. The two upper right and two lower left Switcher ASICs are running in upwards direction, while the others are running in downwards direction. The external shift register inputs in combination with the global load and clk signal are used to activate the currently selected detector row for readout.

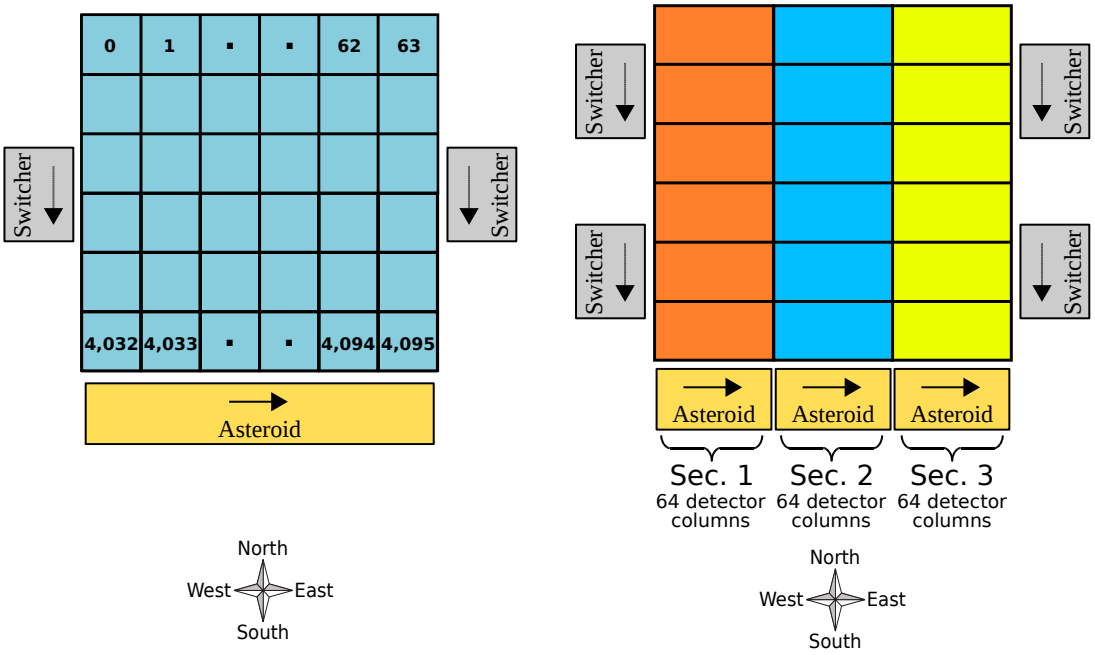
matrix is skipped. To read out only a fraction of the entire detector matrix, the row selection is not necessarily sequential and must be able to contain jumps.

The configuration depicted in Figure 2.16 shows a detector matrix readout in two directions. For this configuration the two upper right and the two lower left Switcher ASICs are running in upwards direction while the other four switchers are running in the opposite downward direction. The run direction indicates the shift direction of the bidirectional shift register. If the ASIC is running in the upwards direction the shift register input on the top side, where channel 0 is located, is used as input and the shift direction is towards higher channel numbers. All external shift register inputs are connected and controlled by the X-Board sequencer unit. Special read-out sequences, such as the window mode, where only a part of the detector matrix is read out, can directly be controlled by the sequencer unit, which is the central steering system of the detector system.

2.3.4 Matrix readout layouts

DEPFET matrices can be designed for a variety of different readout configurations, dependent on the application requirements. Common to all readout schemata the row-wise matrix steering by Switcher ASICs is placed on the left and right side of the detector matrix. The ASTEROID readout ASICs are connected to the top and bottom side of the DEPFET matrix. This orthogonal steering and readout structure can also be used for large DEPFET detector matrices up to the mega pixel range.

The readout speed of a single analog channel is limited by the ASTEROID output line driver. A higher serialization speed increases the **i**ntersymbol **i**nterference (ISI) on the transmission line to the following digitization stage. This crosstalk generates a virtual signal in the subsequent channel and can, therefore, reduce the achievable energy resolution of the detector system. To increase the readout speed without a performance loss, multiple analog channels have to be



(a) The figure shows a single readout node structure for a 64×64 matrix. The plotted pixel number denotes the pixel order after the serialization done by the ASTEROID readout ASIC.

(b) The figure shows a triple readout node structure. Each of the three shown sectors have their own ASTEROID analog readout node. The matrix in this example has 3×64 = 192 columns.

Figure 2.17: Two versions of a single-side readout scheme for DEPFET matrices. The use of multiple ASTEROID readout ASICs is required for detector matrices, where more than 64 readout channels are operated concurrently.

used simultaneously in the detector system.

The simplest readout schema with a single analog channel for a 64×64 DEPFET matrix is depicted in Figure 2.17a. The steering Switcher ASICs are placed at the east and west side of the matrix and an ASTEROID analog readout ASIC is placed on the south side. The numbers plotted in Figure 2.17a denote the pixel order in which they appear at the digitization stage after the analog pre-processing and serialization by means of the ASTEROID. For the data processing system the information about the pixel order is essential for the data stream encoding.

The ASTEROID ASICs currently used have 64 analog input channels. If the DEPFET matrix is broader than 64 pixels, multiple ASTEROIDs are required to process an entire detector row simultaneously. Such a configuration is shown in Figure 2.17b. In this configuration, a single ASTEROID processes only a section of the entire detector matrix; therefore the pixel order inside a sector is the same as in Figure 2.17a. However, the sequential pixel arrangement inside of such a matrix sector does not directly correspond to the global detector matrix coordinates. This has to be considered in the event clustering and data stream encoding, where a coordinate transformation is required for each detector segment to obtain global detector coordinates.

Figure 2.18 shows a configuration, where the detector matrix is splitted into a Northern and a Southern detector hemisphere. Both hemispheres are operated simultaneously to double the detector readout speed. For the concurrent operation of both hemispheres each one needs its own steering and readout ASICs. This results in a more complex detector system and has a higher power dissipation but cannot be avoided for larger and fast readout DEPFET detectors. Clustering of this double-side readout configuration is done in a similar way as in the previously described extension of the single-readout node version.

A fully parallel readout configuration for ultra-high frame rates up to 5 Mfps is currently

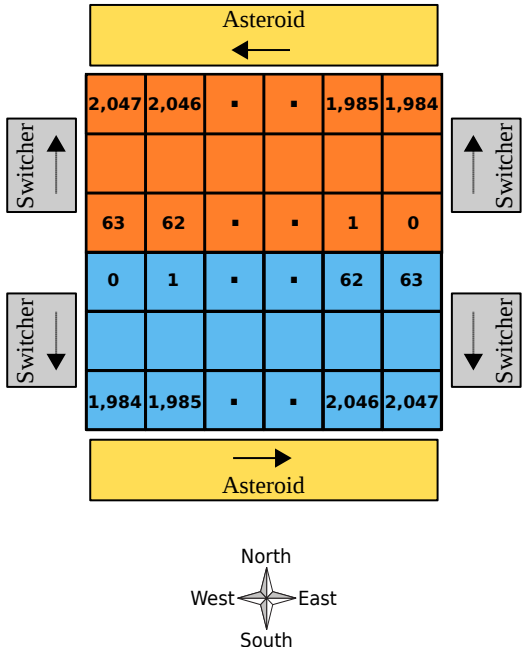


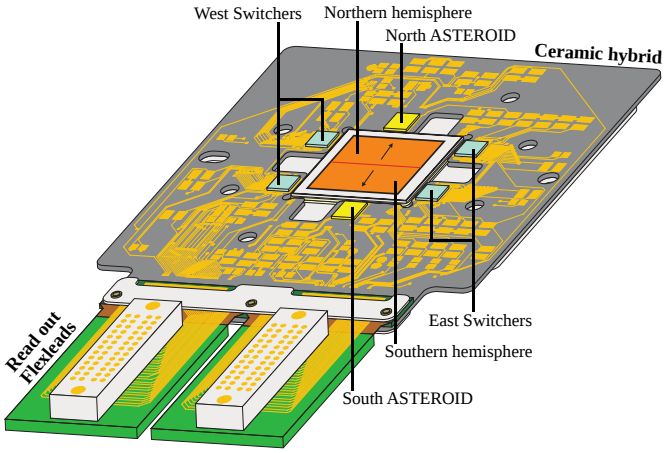
Figure 2.18: The figure shows a double-side readout structure to double the readout speed for a 64×64 matrix. The plotted pixel number denotes the pixel order in the particular readout segment after the serialization done by the ASTEROID readout ASIC.

under investigation. In order to realize such a concept, the readout ASIC has to be connected to each pixel individually. This configuration requires a unique ADC for each detector pixel. Therefore, a totally new detector and ASIC design with bump bond technique is required. The major drawback, especially for satellite applications, is currently the power dissipation of all the ADCs. This concept will not be discussed further at this point, because it has not been available for measurements yet and was not used during this thesis.

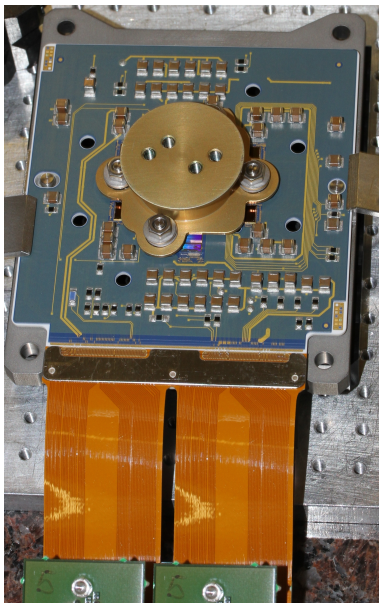
2.4 DEPFET system assembly structure

In DEPFET setups typically ceramic hybrids are used to carry the ASICs and the detector matrix. This is done because of their improved temperature stability and excellent thermal conductivity compared to conventional PCBs. These ceramic hybrids are operated in ultra-high vacuum to avoid icing of the system even at temperatures below $-100°C$. Cooling of the semiconductor detector decreases the leakage current and improves, therefore, the achievable energy resolution. The ceramic hybrid is directly connected to the cooling mask and is used to transport the dissipated thermal energy from the detector matrix and the ASICs to the externally placed cooler. It is important to have a good thermal conductivity to get low thermal gradients in the system. The ceramic is manufactured in a thick film technology, where on top of the ceramic carry made of Al_2O_3, multiple layers of gold and isolation glass are printed to distribute the different voltages and programming signals. Steering and analog readout ASICs as well as the detector chip are glued on top of the ceramic hybrid. Electrical contacts between the chips and the hybrid are established via bond wires.

With the previously presented components, DEPFET matrix, ASTEROID, Switcher and the intelligent Sequencer, different structures for the detector assembly can be realized dependent on the project requirements. In the following, the two detector setups used most often during this thesis are described in detail.



(a) Drawing of the MIXS detector ceramic and the flexleads. On the flexleads the connectors for the front-end electronic interface are mounted. The detector matrix is divided into two hemispheres and read out by the north and south ASTEROID. The matrix is steered by the two west switchers and the two east switchers on the ceramic hybrid.



(b) Picture of the MIXS flight detector ceramic and the two readout flexlead. In this picture the ASICs and the DEPFET matrix is hidden by the mounted cooling adapter in the center of the ceramic hybrid.

Figure 2.19: Flight-qualified detector ceramic hybrid for the **M**ercury **I**maging **X**-ray **S**pectrometer (MIXS).

2.4.1 Mercury Imaging X-ray Spectrometer setup

The **M**ercury **I**maging **X**-ray **S**pectrometer (MIXS) detector system was used for evaluation and benchmark tests of the new dynamic processing algorithms and the entire SuMo-DAQ data processing concept. The high frame rate of the MIXS detector system with 6,000 fps, the two simultaneously operated analog output channels and the non-consecutive pixel order allow demonstrating the performance and flexibility of the new data processing concept.

The structure of the MIXS ceramic hybrid is depicted in Figure 2.19a. MIXS is read out in two directions to double the detector frame rate and is, therefore, equipped with two ASTEROID readout ASICs. One ASTEROID is for the readout of the northern and the other one for the southern hemisphere. Four Switcher ASICs are required to allow for a simultaneous operation of two independent detector rows. The two Switchers on the west side steer the Clear and Cleargate signals in their hemisphere and the two Switchers on the east side are controlling the Gate signal for their hemisphere. The ceramic outline is quite complicated for MIXS because of the cooling power limitation on board of BepiColombo, where only the detector matrix and not the entire ceramic hybrid can be cooled. The four cut-outs in the ceramic close to the DEPFET matrix are required for the mounting structure of the matrix cooling system. This cooling adapter is shown in Figure 2.19b. The detector hybrid is connected via flexleads to the electronics box, where all required voltage and control signals for the detector and the ASICs are generated and the data processing is done.

The parameter optimization for all 7 flight-graded MIXS modules as well as for the 5 additional laboratory modules where made at the Semiconductor Laboratory (HLL) with an X-ray test facility called **Cali**bration **Fa**cility (CALIFA). Figure 2.20 shows the entire CALIFA setup in the HLL laboratory. The multi-target X-ray tube is a conventional X-ray tube with the option to select different targets and filters for the operation. Compared to a conventional

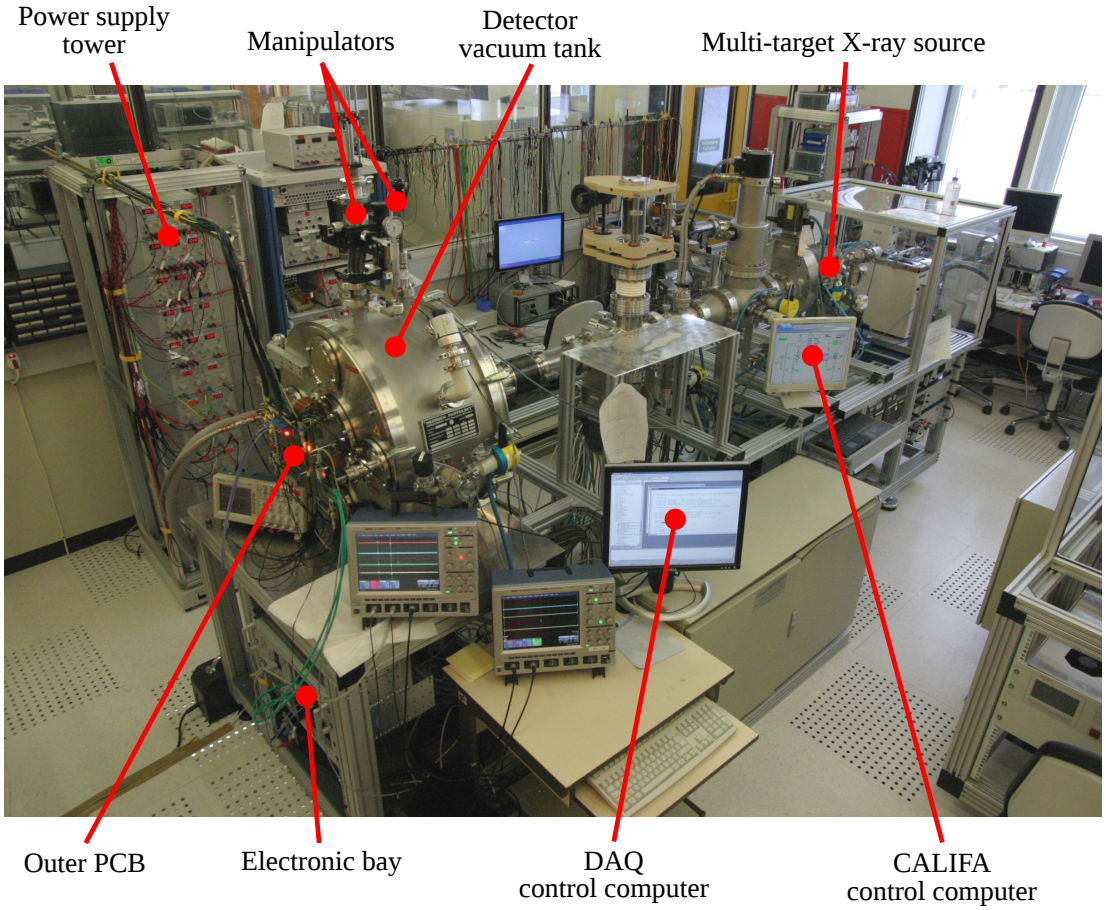


Figure 2.20: Picture of the **Cali**bration **Fa**cility (CALIFA), one of the X-ray test facility at the Semiconductor Laboratory. The multi-target X-ray source is located in the rear of the CALIFA setup. The X-ray beam is guided in a vacuum tube from the source into the detector tank, where the detector hybrid is mounted. A radioactive calibration source is mounted on a manipulator located on the upper side of the detector tank. A second manipulator located right beside it holds a baffle and can be moved into the X-ray beam to create a certain photon pattern on the detector.

radioactive source such as the ^{55}Fe calibration source the luminosity of an X-ray tube can be adjusted. Nevertheless, a manipulator carrying a ^{55}Fe source is connected to the tank. An additional manipulator on the CALIFA vacuum tank allows adjusting and placing a baffle in the X-ray beam, directly in front of the detector. This baffle allows creating a certain photon pattern on the detector and is helpful to verify that the coordinate transformation and the frame reconstruction works as expected. In Figure 2.21 an accumulation of all events recorded during a measurement with such a baffle in front of the detector is depicted. Such an event accumulation map is called hitmap and is normally used to check the homogeneity of the illumination created by the X-ray source.

In Figure 2.22 the internal structure of the CALIFA X-ray test facility with the mounted MIXS detector system is depicted. On the outer PCB all connectors for the support equipment and digitization system are located. From this outer PCB the signals are routed via flexleads and vacuum feedthroughs to the inner PCB. The requirements on board the **M**ercury **P**lanetary **O**rbiter (MPO) spacecraft demands dedicated plug connectors for the detector module interface. These plug connectors cannot directly be mounted and soldered on the ceramic hybrid. The hybrid is, therefore, equipped with flexleads, where the plug connectors are mounted. This

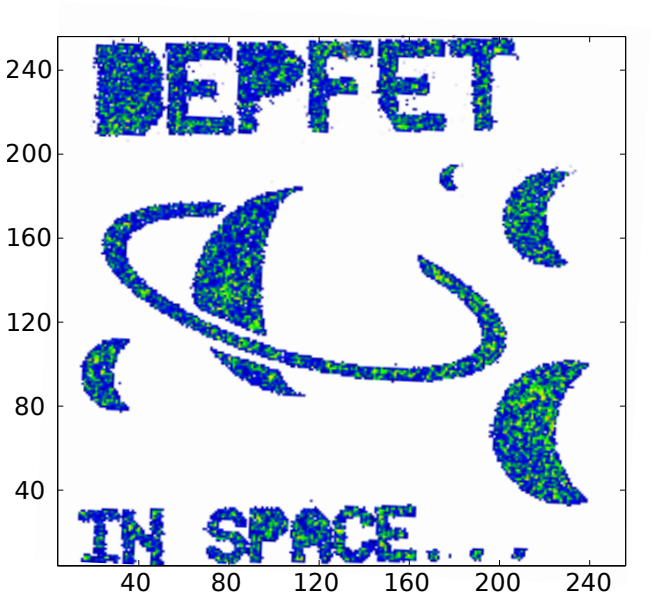


Figure 2.21: This figure shows a hitmap recorded during a measurement made with a baffle in front of the detector. In a hitmap, all events which occurred during an entire measurement are accumulated. During a measurement without such a baffle the hitmap allows for checking the homogeneity of the illumination. The measurement with the baffle is used to verify the coordinate transformation and the frame reconstruction works as expected.

MIXS detector module assembly is depicted in Figure 2.19. The cooling power in the CALIFA setup is provided by a stirling cooler. The advantage of this cooling system is the good adjustability of the detector temperature. Later, this can be used to simulate temperature drifts of the detector system on board a satellite. The radioactive ^{55}Fe calibration source connected to one of the manipulators is used for quick measurements to check the detector system performance. The radioactive source, in the CALIFA setup a ^{55}Fe source, can be moved with the manipulator. In the depicted standby position (S), the source is moved upwards and the detector matrix is shielded from photons by the vacuum tank. This standby position is the default configuration to minimize the detector degradation by radiation damages and is also used for the recording of dark frame data sets. These dark frames are required by the old DAQ system for the calculation of the static offset and noise map. Dark frames should not include any photon event because these signals would disturb the calculation of the static offset and noise map. These static maps are constant for the analysis of the entire data set. A precise and undisturbed calculation of these parameters is, therefore, crucial. In the measurement position (M) the radioactive source is positioned directly opposite the DEPFET detector on the hybrid. Photons from the source can reach the detector and a calibration data set can be recorded. The multi-target X-ray source connected to the CALIFA vacuum tank via a vacuum tube is not shown in Figure 2.22, which gives the MIXS overview. Such a multi-target X-ray source allows for a more precise verification of the energy resolution of the detector system at different photon energies and illuminations. The second independent manipulator is carrying the baffle to create a certain test pattern on the detector matrix.

The MIXS detector system on board of the BepiColombo mission will analyze the surface of Mercury, the least explored terrestrial planet in our solar system. Mercury is the innermost planet in our solar system and close to the Sun. For the BepiColombo mission, this means it is a very long journey to Mercury and because of the close distance to the Sun the environmental conditions are very harsh. Close to the Sun the high environment temperatures and radiation damages created by solar flares are the two most challenging circumstances for the BepiColombo

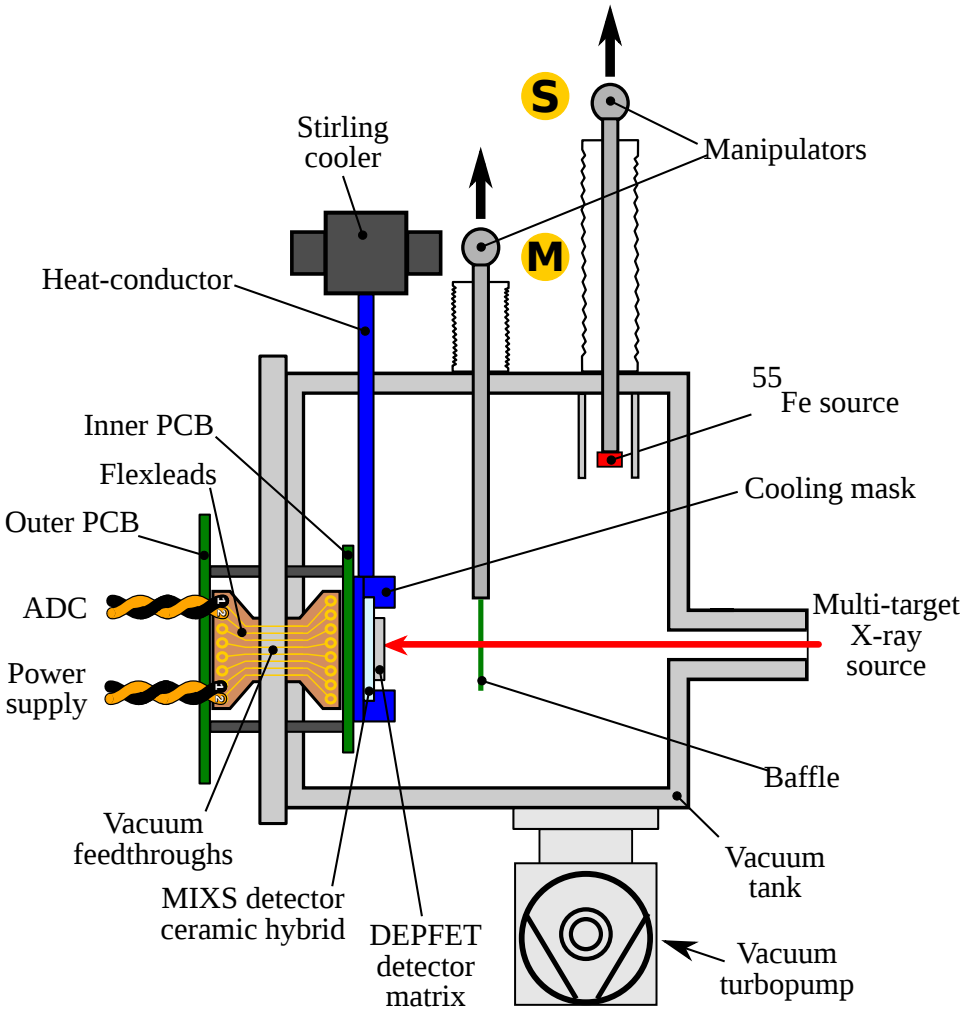


Figure 2.22: Sketch of the MIXS detector setup at the CALIFA X-ray test facility. All external electronic signals are routed via outer PCB, flex leads and vacuum feedthroughs to the inner PCB. The MIXS detector ceramic is connected by two plug connectors, mounted on flexleads, to the inner PCB. Thermally the detector hybrid is coupled via the cooling mask to the Stirling cooler. With the stirling cooler attached to the heat-conductor a precise regulation of the detector temperature is possible. Two independent manipulators are available at the CALIFA tank. The manipulator carrying a radioactive ^{55}Fe calibration source is placed in the standby (S) position. The second manipulator carrying a baffle to create a certain photon pattern on the detector matrix is placed in the measurement (M) position. The multi-target X-ray source allows for testing the detector performance at different photon energies and illumination levels.

mission. For the BepiColombo mission ESA expects approximately $1.1 * 10^{10}$ of 10 MeV proton equivalent during the entire mission life time of 6 years transfer and up to 2 years of operation.

To verify the proposed MIXS detector system performance at the expected mission end of BepiColombo mission the inhouse X-ray test facility CALIFA is not powerful enough. For these radiation hardness tests of the detector system two of the MIXS detector modules where irradiated at the Maier-Leibnitz Laboratory for Nuclear and Particle and Accelerator Physics (MLL) [18]. The available Tandem-van-de-Graaff accelerator at the Maier-Leibnitz Laboratory with up to 14 MV acceleration voltage is used to produce high energy protons for a wide range of scientific research projects. This accelerator can provide the 10 MeV protons required for the ESA qualification procedure of the MIXS detector modules. During the measurement campaign at the MLL the new real-time data processing DAQ-sytem was continually operated for a week and used to record every single proton hit on the detector. Additionally, the new DAQ system provided important detector system parameters such as the noise and offset map for live detector system monitoring.

2.5 Data acquisition system for X-ray detectors

Figure 2.23 illustrates the structure of the DAQ system used up to now for DEPFET detectors with all required system components. The detector hybrid is mounted in the vacuum tank and connected to the inner **p**rinted **c**ircuit **b**oard (PCB). The detector hybrid is operated in vacuum for two reasons: firstly, the short absorption length of low energy photons in standard pressurized air; secondly, to avoid icing at the low operation temperature required to minimize the current leakage in the detector matrix. The inner PCB is used as an interconnection between the detector hybrid and the vacuum feedthroughs. An additional buffer is placed on this PCB for the ASTEROID analog output channels to minimize the capacitive load for the ASTEROID output driver. The outer PCB is connected on the other side of the vacuum feedthroughs. This PCB is used as an interface board to all external system components such as the power supply unit, the sequencer and the digitization unit outside the vacuum. The power supply unit generates all the required voltages to operate the detector matrix and other electronic components, such as analog drivers, optocouplers, ASICs and the digital control units of the entire detector system.

The X-Board depicted in Figure 2.23 is used as central hardware control unit that generates the steering signals for the ASICs on the detector hybrid and concurrently controls the ADCs for a synchronous system operation. The sequencer implemented in the FPGA on the X-Board additionally provides the information about the current readout position to the digitization card. Then a FPGA on this digitization card converts the ADC values and the sequencer control signals into a formatted output data stream, which is transmitted to the data handling system. This formatting contains information about line and frame ends and at the beginning of each frame an identification number to permit the detection of data losses in the stored raw data file. Up to now, a PCI or PCI Express bus has been used for the "data carrying" interconnection between the ADC card and the DAQ system computer. The received raw data stream is finally stored on a mass storage system for the offline data conversion and analysis process.

In this PCI-based DAQ system the received raw data stream is directly stored without any further data processing or checking. All data consistency checks and data processing steps are done later offline. Offline data processing without real-time processing requirements drastically reduces the complexity of the data processing system and the required amount of processing power. The drawbacks of such an approach are the huge amount of storage space required for data recording, the long data analysis times and the operation of the detector system without a real-time feedback of the current detector condition.

The data flow in the PCI-based DAQ system is shown in Figure 2.24. In this figure the DAQ system computer combines the control and data handling tasks in a single system. The operator can control the detector system via a system control interface. This system control

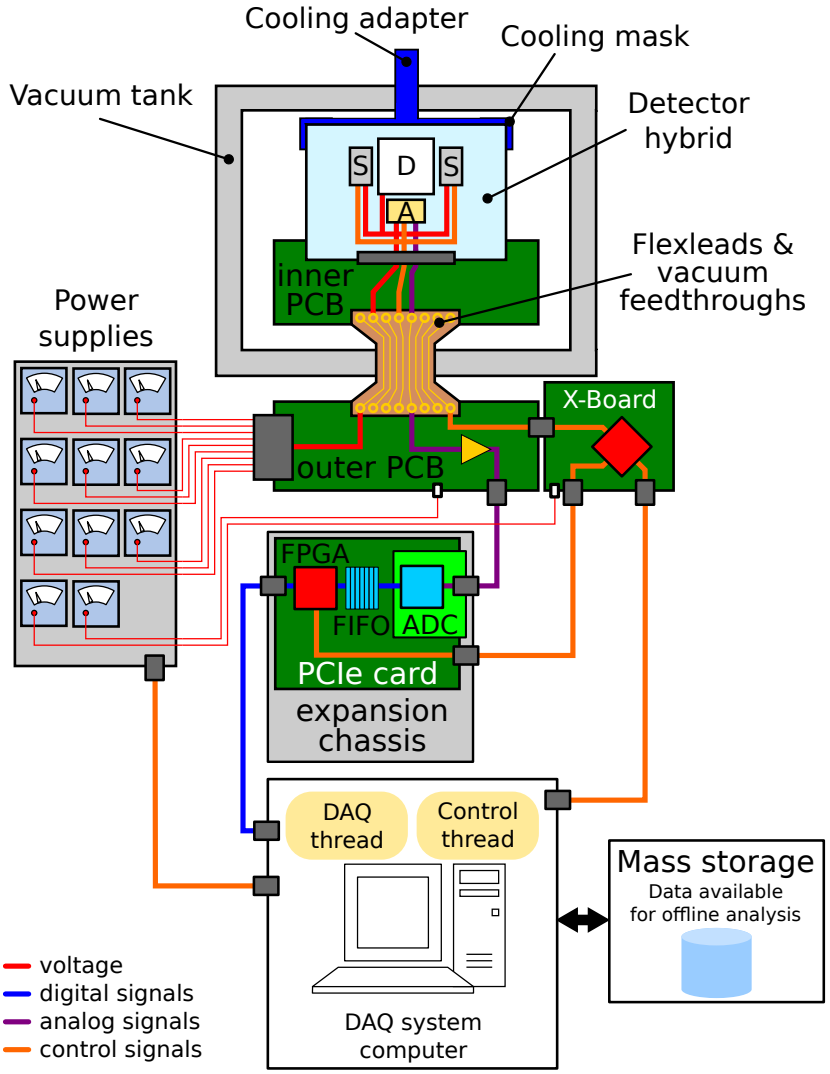


Figure 2.23: [51, p. 62] Schematic of a DEPFET detector setup with all required subcomponents. The colling adapter, which is connected to the cooling mask, is placed on top of the vacuum tank. The detector hybrid is mounted inside the vacuum tank. On this hybrid the detector (D) itself, Switcher (S) steering ASICs and ASTEROID (A) read-out ASICs are placed. The detector hybrid is connected to the inner PCB, where a more powerful line driver for the ASTEROID analog output is located. Outside the vacuum tank the outer PCB is used as a central interface for power supplies, X-Board and ADCs. In this configuration the depicted computer acts as a user-system interface and as the interface to the mass storage system. The user-system interface enables the control of the X-Board, where the complete hardware steering is done by the sequencer unit.

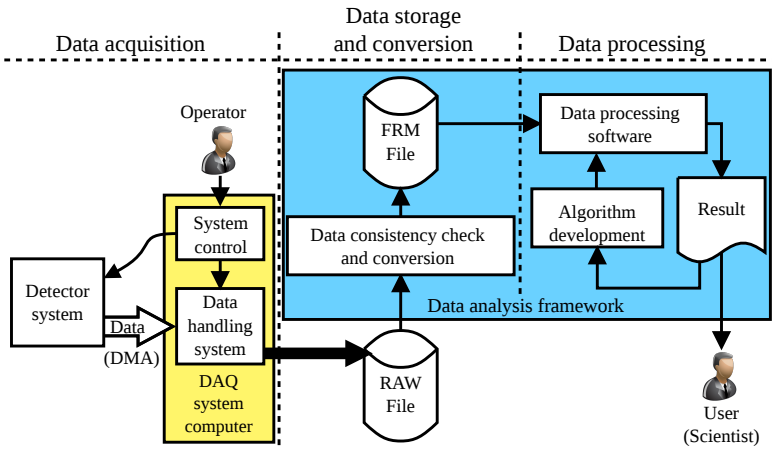


Figure 2.24: System overview of the PCI-based DAQ system for DEPFETs. The system can be divided into three section: the data acquisition section, where the raw detector input data steam is received and handled during the detector operation; the data storage and conversion section to prepare the stored raw data for the offline data processing in the third section.

interface allows steering the detector system as well as the data handling system. The data handling system manages the raw data transfer from the detector system into the raw data file. Decoupled from this data acquisition are the offline data conversion and the analysis steps, which are described in detail in Subchapter 3.1.

The formatted raw data stream from the ADC card is first transferred by **D**irect **M**emory **A**ccess (DMA) into the memory of the data handling system. In the second step the data handling system copies the formatted input data into the raw data file. For the currently used DAQ system a DMA operational mode called **M**ultiple **B**uffer **S**tructure (MBS) is used for the data transfer between the ADC card and the data handling system. This DMA technique is depicted in Figure 2.25. The MBS mode combines multiple conventional DMA memory blocks and organizes the DMA transfers into the different buffers by software. The producer thread shown in Figure 2.25 initiates and controls the DMA transfers into the different memory buffers B1, B2 and B3. For the thread synchronization each of these buffers is assigned to a mutex. In Figure 2.25, the mutexes M1, M2 and M3 are filled with different colors, depending on the current state of the mutex. A red mutex is locked by the producer thread, a green mutex is locked by the writer thread and a white mutex is currently unlocked. The producer thread first acquires an unused DMA buffer by locking the corresponding buffer mutex M1, M2 or M3 and then initiates a new DMA transfer for this buffer. After the DMA transfer has been completed the DMA memory buffer is marked to be processed by the writer thread. This writer thread afterwards copies the memory buffer content of marked buffers into a raw data file and sets the DMA buffer status to empty. This raw data file is later used as input file for the offline data processing and analysis framework. A DMA operation mode, where the organization of the different DMA memory blocks is done in hardware, is called **S**catter **G**ather **L**ist (SGL). This SGL mode was implemented and tested as well for the DAQ system but not used due to frequent data transfer cancellations. The reason for these cancellations are short data congestion as result of the high input data rate and continuous DMA transfers. The DMA hardware engine cancels the entire SGL transfer if such a congestion occurs during the DMA operation regardless of whether the writer thread is momentarily too slow or whether other operations on the data handling computer are disturbing the DMA transfer. With the MBS transfer technique, instead only data loss occurs because of data cache overflows on the ADC card; however, the DAQ system continues recording data. This data loss is detected during the offline data processing and the corrupted data section is then skipped.

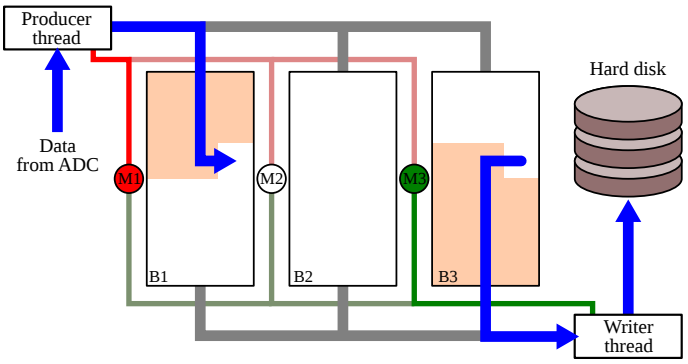
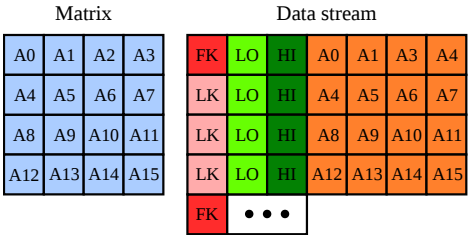


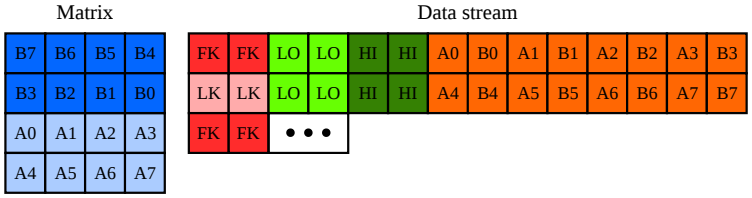
Figure 2.25: **D**irect **M**emory **A**ccess (DMA) transfer with the **M**ultiple **B**uffer **S**tructure (MBS) technique for the raw data transfer from the ADC to the storage system. In this configuration three DMA buffers B1,B2 and B3 with their corresponding mutex M1, M2 and M3 are shown. In this illustration a mutex locked by the producer thread is colored in red, a mutex locked by the writer thread is colored in green and a currently unlocked mutex is white. The producer thread acquires unused buffers and initializes the DMA transfers. The writer thread waits for completely filled buffers and copies their contents into a raw data file. The figure is similar to [51, p. 71].

The raw output data rate for the PCI-based DAQ system can be calculated with the Equation 2.19. The raw data stream encoding format for the standard and the interleaved encoding method is depicted in Figure 2.26. For single readout node systems the standard encoding scheme is used. In this configuration, at the beginning of each encoded detector row, there is a frame keyword (FK) or a line keyword (LK) placed with a size of 2 bytes. The following 4 bytes are a hardware time-stamp, which is subdivided into a 16-bit high (HI) and low (LO) word. This hardware time-stamp at the beginning of each detector row enables data consistency checks and allows for the frame reconstruction even in case of partial data loss. The entire 6 bytes are representing the complete line identifier. For each pixel in this detector row a 2 bytes large ADC value follows this line identifier in the sequence. This is depicted in Figure 2.26a. Each of these ADC values is encoded into the data stream with 2 bytes. The interleaved encoding method is used in detector systems with two read-out nodes. Both ADC values are sampled at the same time on the ADC card. To avoid the requirement of intermediate ADC value storage and reordering, both values are encoded side by side even if the pixels are not adjacent. This non-consecutive pixel order is corrected later in the data processing software. To enable the distinction of the standard data stream to the interleaved data stream, the data encoding method is slightly different. In the interleaved mode, two line keywords or two frame keywords are placed at the beginning of each double line followed by the 8 bytes for the two hardware time-stamps. After these 12 bytes for the double line identifier the 2 bytes large ADC values for the input channels of two detector rows are encoded in the sequence as depicted in Figure 2.26b.

For the used MIXS detector system with a 64×64 DEPFET matrix readout at a frame rate of 6,000 fps, the resulting continuous raw data rate is approximately 49.07 MB/s (see Equation 2.20). For short reference and calibration measurements this data rate can be handled with today's technology, but it becomes quite challenging for long-term measurements. For the next generation of detector systems with a much higher data rate this approach becomes inefficient for lab applications and unusable on board of satellites. For example, the expected raw data rate produced by the detector system for the proposed ESA mission ATHENA+ is calculated in Equation 2.21 to approximately 612.23 MB/s. This is an increase in raw data rate by a factor of 12 for the next generation of planned satellite detector systems.



(a) Standard data stream encoding scheme of the PCI-based DAQ system.



(b) Interleaved data stream encoding scheme of the PCI-based DAQ system.

Figure 2.26: Raw data stream encoding formats.

$$\text{Raw data rate} = (6\,\text{Bytes} + \text{NCols} * 2\,\text{Bytes}) * \text{NRows} * \text{FrameRate} \tag{2.19}$$

$$\text{MIXS}_{\text{Raw data rate}} = (6\,\text{Bytes} + 64 * 2\,\text{Bytes}) * 64 * 6000\,fps \approx 49.07\,\text{MB/s} \tag{2.20}$$

$$\text{ATHENA+}_{\text{Raw data rate}} = (6\,\text{Bytes} + 640 * 2\,\text{Bytes}) * 640 * 780\,fps \approx 612.23\,\text{MB/s} \tag{2.21}$$

PCI or PCI Express based DAQ systems are widely used in industry as well as in research applications, because the bus system is simple to use and cheap. PCI Express is the newest type in the PCI bus family and a serial high-speed replacement of the older parallel PCI and PCI-X versions. The replacement was required due to the limitations in the achievable bus bandwidth for the parallel PCI bus architecture. Table 2.2 shows an overview of the different PCI, PCI-X and PCIe versions and their theoretical bandwidth. Currently, a PCIe x1 v 1.1 interface is used for the interconnection between the ADC card and the DAQ control computer. The PCIe bus standard is based on a point-to-point topology, where every device is connected via a dedicated serial link to the host. This topology makes the flexible distribution of data streams across multiple computers and thus, the aggregation of multiple data streams on a single device becomes quite challenging. In addition, the PCIe standard has been designed as a board-level interconnect system and is not directly usable for longer system interconnections. These two facts make it complicated to scale such a DAQ system to a large number of analog input channels.

Real-time data processing at this data rate with an entirely software-based approach and the complex X-ray data processing algorithms requires a huge amount of processing power. Processing only a fraction of this data stream is an intermediate solution, which is currently used for CCD detector systems in a software tool called **R**eal-time **A**nalysis for **C**cds **O**ffline **O**nli**n**e (RACCOON). The data processing concept used in RACCOON takes the latest frame from the raw input data stream and processes this frame. When the frame processing is finished, the process is repeated. For a single input data stream this is easily possible. For larger detector systems, where the detector readout has to be divided into multiple data streams, extracting

Type	Bus width (bits)	Clock (MHz)	Theoretical bandwidth (MB/s)
PCI (parallel)	32	33	133
PCI (parallel)	32	66	266
PCI-X (parallel)	64	66	528
PCI-X (parallel)	64	133	1,064
PCIe x1 v1.1(serial)	1	2,500	250
PCIe x1 v2.0 (serial)	1	5,000	500
PCIe x1 v3.0 (serial)	1	8,000	1,000

Table 2.2: Overview of different specified PCI, PCI-X and PCIe versions with their theoretical performance.

continuous frame parts out of multiple raw data streams, this is much more complex. The RACCOON software always tries to process as many frames as possible. But with the output data rates of today's detector systems real-time data processing of the complete data stream is by far not possible nor can a fixed ratio of total processed frames be guaranteed by this concept. The calculation made by RACCOON represents, therefore, only a rough estimation of the final results provided by the offline data analysis of the entire raw data stream. Nevertheless this approach provides additional information about the detector system and its current status to the operator during the system operation even if only with a low update rate. However, there are a few other drawbacks to this concept:

- Storage of the entire raw data stream for a complete offline data analysis is still required.
- Shown information are only estimations of the final analysis results.
- Long analysis time to process the complete data set offline is still required. This leads to significant delays in the availability of the final analysis result.
- System scaling is difficult for large detector streams with multiple input data streams.
- Data and frame loss during the data recording cannot be excluded due to temporary data congestion on the data handling system.
- The data processing approach is not usable in a satellite environment because no built-in data reduction for the entire raw data stream is available.

Chapter 3

Exploration of X-ray data processing

The **R**OOT-based **O**ffline **An**alysis (ROAn) [51, 50] software with the algorithms described in Subsection 3.1 has, up to now, been the standard data analysis tool for DEPFET matrix setups. This software uses well-tested algorithms and provides reliable analysis results with high accuracy. Therefore, the ROAn algorithms are used as a baseline for the further optimization and development process. This well-established code base is also used as a reference for the new processing algorithms and their analysis precision and speed. A first step was to analyze the ROAn code to find bottlenecks and possible approaches for performance improvements. This was followed by an evaluation of currently available processing platforms for the different calculation tasks. Each of these processing platforms has its specific advantages and disadvantages for a particular type of processing task. The distribution of the individual processing steps to the most appropriate system components is a non-trivial problem, but essential in order to achieve the optimal system performance with the lowest resource utilisation.

The use of **G**eneral-**p**urpose **g**raphics **p**rocessing **u**nits (GPGPUs) as co-processors and multi-threaded data processing approaches to accelerate the ROAn analysis were evaluated as entirely new concepts for the SuMo-DAQ system after the analysis of the ROAn source code.

3.1 X-ray data processing with ROOT-based Offline Analysis

After the data recording is completed, the data set needs to be analyzed to extract the matrix performance data from the measurement. For the offline data analysis of X-ray measurements made with DEPFETs and the old PCI-based DAQ system, the **R**OOT-based **O**ffline **An**alysis (ROAn) [51, 50] software package is used. The ROAn software package uses the ROOT [11, 26, 78] framework published by the **C**onseil **E**uropéen pour la **R**echerche **N**ucléaire (CERN) [14]. ROOT is an object-oriented framework for scientific data analysis tasks written in C++. Due to the large user group the framework is regularly updated, well maintained and documented. The source code of ROOT is open source and the framework supports a wide range of operating systems. This makes the analysis software independent from any software vendors and avoids license fees. The ROOT framework additionally provides an operating system abstraction layer for the supported platforms. This allows for an operating system-independent software implementation of the ROAn data analysis software.

ROAn itself is also written in C++ and uses an object-oriented programming style. Figure 3.1 schematically depicts the structure of the ROAn analysis software and the relation to the ROOT framework. The ROOT framework provides basic functions needed for any data analysis software such as histogram generation, data containers, statistical and mathematical

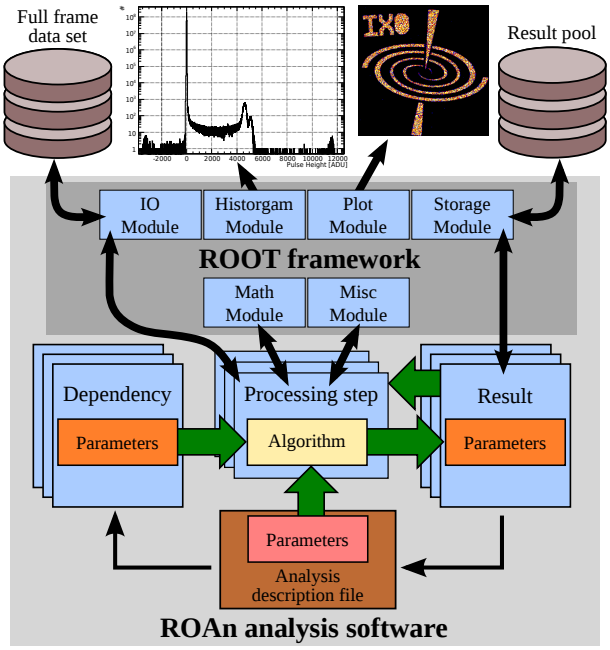


Figure 3.1: Schematic representation of the ROAn data analysis software and the relation to the ROOT framework. ROOT provides basic functions needed for any data analysis software and includes an operating system abstraction layer. Processing algorithms needed for the X-ray data analysis are part of the ROAn analysis software. Analysis results are calculated by specific processing steps from the input data and other previously calculated results. The step-based computing concept allows for easy exchanging and adding of new algorithms.

functions, data plotting, storage and export functions. All data processing algorithms and special analysis functions required for the analysis of the X-ray data sets are part of the ROAn framework.

The ROAn framework uses a step-based computing concept for the data analysis, where all results are calculated in individual data processing steps. Each result can be used as input for other steps to calculate their own results. The results needed to calculate another result are called dependencies. The sequence in which the different processing steps are applied to the data set depends on their specific input dependencies and output results. A processing step can calculate its results only if all required input dependencies have already been calculated. To control the data analysis process, an analysis description file is used. In this description file, every desired analysis result and the processing step for its calculation is specified. In addition, this file can contain parameters to control the algorithms in the different processing steps. For standard X-ray data analysis tasks, templates of this analysis description file are available. These templates make it easier for users to analyze their X-ray data sets with the ROAn software and allow creating comparable analysis results.

This step-based computing concept makes the ROAn analysis framework modular and expandable. Optimized or newly developed algorithms can be exchanged or added easily. In combination with the recorded full frame data sets and the availability of every detector pixel value the ROAn framework can efficiently be used for comparing and evaluating new processing algorithms. Multiple analysis runs with different processing algorithms can be made on exactly the same input values. This enables precise performance comparisons between the different algorithms. The drawbacks of this analysis approach are the required large storage space for the full frame data sets and the non real-time analysis speed with the high latency until the final results are available. This makes the data processing concept inefficient especially for long-term measurements and continuously operated detector systems. Furthermore, the ROAn analysis

software is not usable for autonomous system operation due to the necessary configuration input from the operator.

A more detailed description of the internal principles and methods used in ROAn are available in [51] and on the ROAn documentation website [50].

For a consistent terminology, the terms **Frame, Event, Hit, Pattern** and **Cluster** will be defined first.

Frame: the unprocessed raw data value of all pixels in a detector matrix at the sampling time. Software-internally, the data values of a frame are stored and handled in a 1-dimensional array. For the user a frame is presented as a 2-dimensional array as depicted in Figure 2.21.

Event: a photon absorbed in the detector matrix and producing a cloud of electron-hole pairs is described as an event. The created charge cloud can be collected in a single pixel or distributed during the charge drift over multiple pixels.

Hit: if a pixel shows a signal above the associated pixel threshold a hit is created. The charge distribution over multiple pixels can lead to several hits created by a single event. However, hits can also be generated by noise in a pixel.

Pattern: neighboring hits, which share a common pixel edge, can be combined to patterns. Patterns can have any number of member pixels and an arbitrary shape.

Cluster: hits in the same pixel in consecutive frames can be grouped in clusters. Clusters are used to tag incompletely removed events, which are so-called "clear-correlations". The first cluster member is called precursor while consecutive hits are named successors. The cluster length is defined by the number of hits grouped in this cluster. A precursor without any successor hits is a cluster with length 1.

After calculating all the user-desired results by the ROAn analysis software, these results have to be displayed or prepared for printing. The most common representation forms are **Histograms and Spectra, Maps, Map value histograms and Tables.** These standard forms of data representation are provided by ROAn analysis software. This allows creating the output of the majority of the expenditure without additional formatting work for the ROAn user.

- **Histograms and Spectra** are used for graphic representations of the data value distribution. Typically global measurement results like the raw data spectrum are depicted in this form. The processed data values can additionally be filtered during the spectrum generation to create special histogram types such as the single pattern (SNG) histogram. These filters are especially useful during the investigation of not yet understood detectors or DAQ system characteristics.
- **Maps:** DEPFET detectors are two-dimensional matrix devices. To display detector images or the distribution of specific pixel properties, such as the offset and noise values of the detector matrix, maps are used.
- **Map value histograms** are used to visualize the frequencies of map values. These histograms allow for rapid capturing of parameters, which would otherwise not be directly visible.
- **Tables** are used to summarize key calculation results and allow for a direct performance benchmark of different analysis results. Examples for these values are the SNG energy resolution or the peak-to-background value.

The typical data processing cycle for the analysis of a DEPFET data set with ROAn is depicted in Figure 3.2. This figure shows all required data processing steps in the time sequence as they are applied to the data set. The following subsection will give an introduction to all these data processing algorithms required for the analysis of DEPFET X-ray imaging spectroscopy

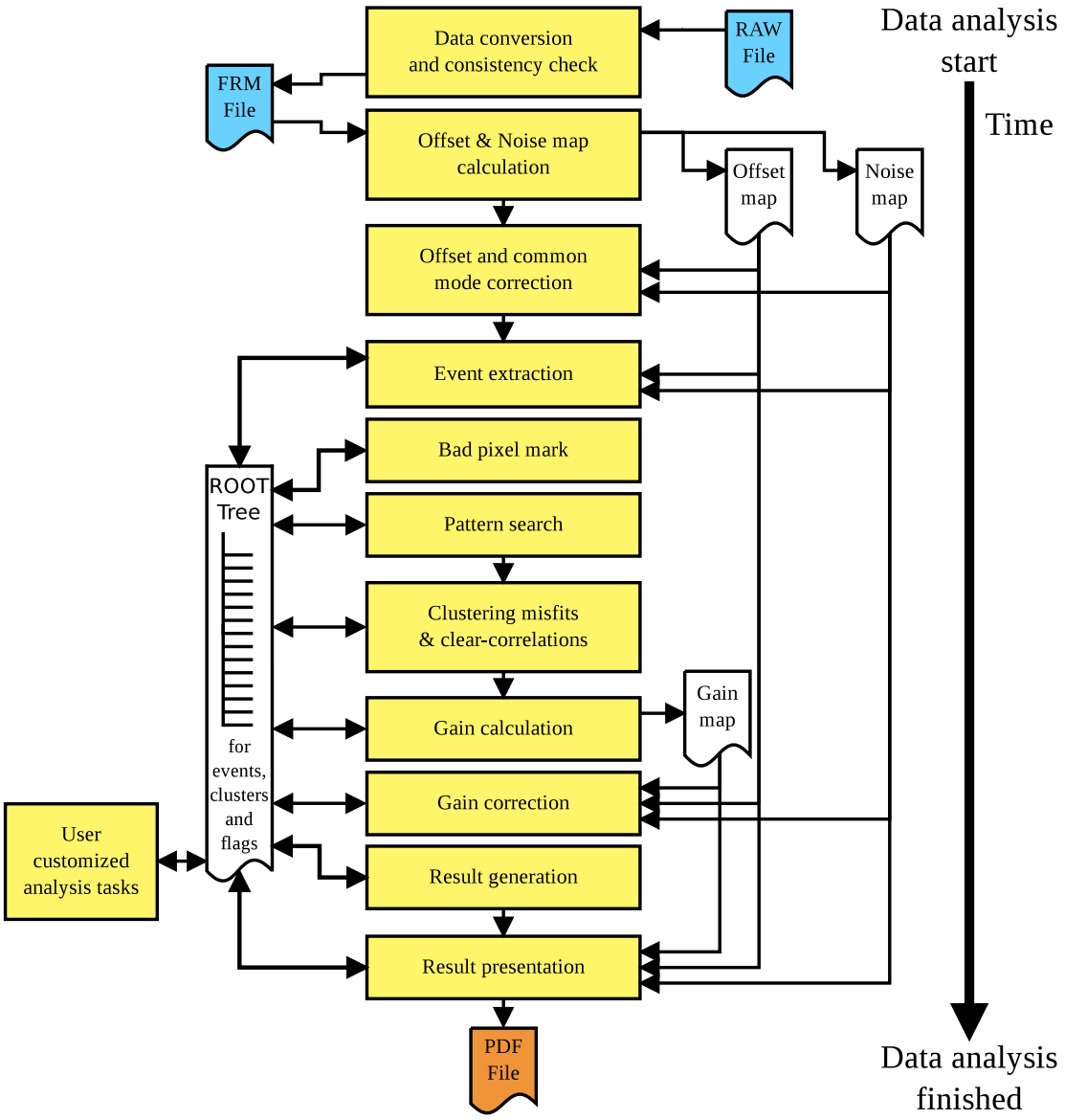


Figure 3.2: Processing order in ROAn to calculate the analysis results for a standard DEPFET measurement data set.

Format version	Min. Col	Max. Col	Min. Row	Max. Row
Extended	0	255	0	255
5	0	255	64	319
6	0	65,535	0	65,535

Table 3.1: Overview of the frame sizes limits for the three currently most often used frame file formats.

data sets. Afterwards, the use of multi-threaded data processing and the application of **G**eneral-**p**urpose computing on **g**raphics **p**rocessing **u**nits (GPGPU) as a coprocessor to improve the data analysis performance of ROAn are evaluated and compared in detail.

3.1.1 Data conversion and consistency check

The PCI-based DAQ system stores the raw data stream without ensuring that the frames are complete. For the data processing with the ROAn analysis software only complete frames can be used. The raw data format contains key words inserted by the DAQ hardware, which enables to recognize data loss (Figure 2.26). The conversion and data consistency process starts at the first frame key word in the raw data file. With the difference between the hardware time stamp stored at the first frame key word and the hardware time stamp at the next frame key word in the raw data file, the frame size can be calculated and checked. If the hardware time stamp contains a jump larger than the correct frame size for the particular detector, partial or total frames are lost in between. The difference in the hardware time stamp can then be used to calculate the number of missing frames and the frame number of the next complete frame can be determined. The verified frames are stored in a special frame file format to avoid multiple data conversion runs in case that multiple analysis runs are applied on the same data set. This file format was originally designed at the **H**alb**l**eiter**l**abor (HLL) as a data format for the **X**-ray **M**ulti-**M**irror-**Newton** (XMM-Newton) CCD camera. Meanwhile, the data format has been further developed and is used for all CCD and DEPFET detector systems in the laboratory. The main difference between the three currently available format versions at the HLL are the maximum number of rows and columns they are supporting. Table 3.1 shows an overview of the three format capabilities.

The base structure is identical for all three frame file formats. The file structure commonly used is version 6, which is shown in Figure 3.3. This frame file format is used for the storage of the full frames. In this file format, the file header has a size of 1,024 bytes and the frame header a size of 64 bytes. Global information for a measurement, such as measurement comments and other constant detector parameters are stored in the file header. Frame-dependent parameters such as frame ID number and current detector system parameters such as date, time and temperature, are stored in the frame header. The size of the following data section, where the pixel values are stored, depends on the detector size of the used detector system. Each pixel is stored as an unsigned 2-byte data value without any additional position information. The constant frame size in a data set enables fast access to specific frames by directly jumping to the calculable frame start address. The entire frame file size can be calculated with Equation 3.1. For a typical calibration measurement of the MIXS detector system with about $2 * 10^6$ frames and a FrameData size of 8,192 bytes, the resulting file size is approximately 15.4 GB.

$$\text{Frames file size} = FileHeader + (FrameHeader + FrameData) * Frames \tag{3.1}$$

The pixel order in the raw data file depends on the used readout schemata, the number of matrix columns and rows, readout hemispheres and the matrix readout direction. In Subsection 2.3.4 various readout schemes for used detector systems and their specific pixel order in the formatted output data stream are presented. To enable a detector- and readout schema-independent data analysis, a standardized pixel order in the frame files is necessary. Therefore,

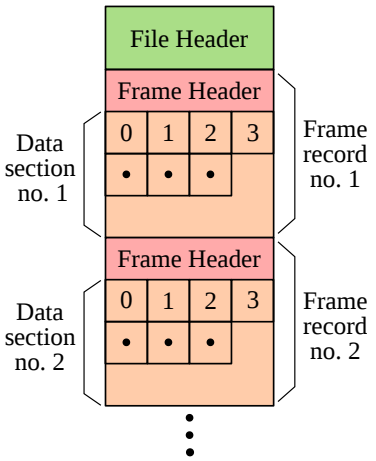


Figure 3.3: Frame file data format used to store converted and complete frames from CCD and DEPFET detector systems. Measurement dependent information such as file format, measurement comments and other constant detector parameters are stored in the file header. The frame header contains a frame ID number and current detector system parameters, such as date, time and temperature. After the frame header, the value for each detector pixel in this frame follows in a predefined and constant pixel order for all detector systems.

a pixel reorder algorithm is applied to the complete and checked frames in the processing step. The standard pixel arrangement in the data section of the frame file start is described by Equation 3.2. In this equation, $DPosX$ represents the absolute X coordinate of the detector pixel, $DPosY$ the corresponding absolute Y pixel position and $DPosX_{max}$ the number of pixels per row for the used detector system. The position coordinates $DPosX$ and $DPosY$ as well as the $ArrayPos$ start counting at 1.

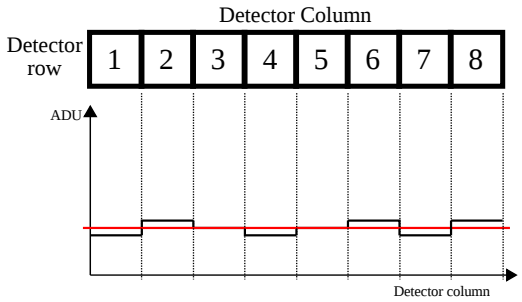
$$ArrayPos = DPosX + (DPosY - 1) * DPosX_{max} \tag{3.2}$$

A more detailed description of the meaning of each bit in the frame file formats is available in [51] and on the ROAn documentation website [50]. For DEPFET detector systems, however, all optional data fields such as temperature, date and time in the frame header are currently not used. The further data processing with ROAn is made exclusively on these frame files. Storing each pixel requires a lot of storage space, but in case of unusual or strange effects in the data set all information is available and can be used for a detailed offline analysis.

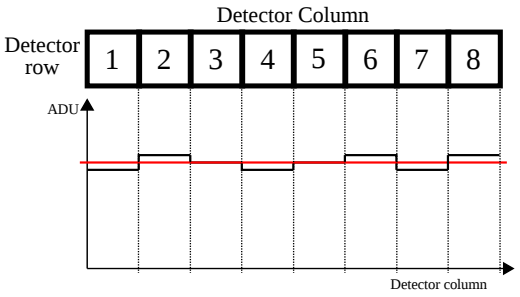
3.1.2 Common mode

As described in Subsection 2.3.2, the ASTEROID analog front-end ASIC samples all its analog input channels in parallel. External disturbances will therefore affect all analog input channels of the currently processed row in the same way. To eliminate these disturbances, the common mode value for each processed detector row is calculated and then subtracted from each pixel value in this row. An example for the pixel values of the same detector row in two different frames without a photon signal is shown in Figure 3.4. The calculated common mode value is marked by the horizontal red line. After the common mode subtraction the pixel values have the same base level in any frame.

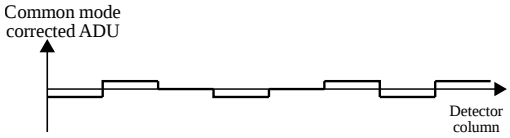
The common mode value is calculated as median value. This calculation requires more computational resources, but compared to a calculation as mean value this approach is more robust against outliers and disturbances by pixels showing a signal. The median value of a finite list of numbers $(x_1, x_2, \ldots, x_n)$ is calculated with Equation 3.3 [10] after the list is sorted from the lowest value to the highest value.



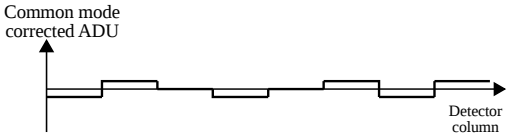
(a) Detector row n in frame j with a low common mode level.



(b) Detector row n in frame $j+1$ with a high common mode level.



(c) Common mode corrected detector row n in frame j.



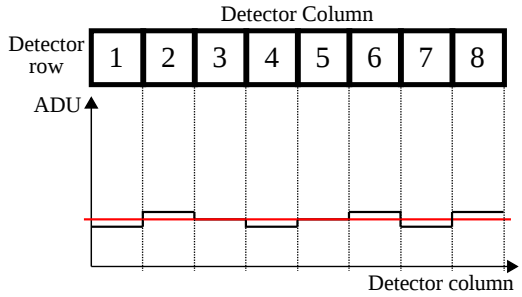
(d) Common mode corrected detector row n in frame $j+1$.

Figure 3.4: Example pixel values of the same detector row in different frames without a photon signal and two different common mode levels before and after the common mode correction.

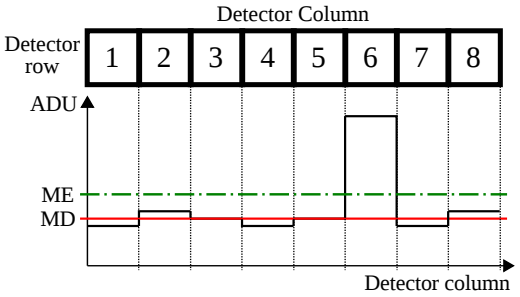
$$\tilde{x} = \begin{cases} x_{m+1} & for\ n = 2m+1 \\ \frac{x_{m+1}+x_m}{2} & for\ n = 2m \end{cases} \tag{3.3}$$

A comparison of the differences in calculation results is shown in Figure 3.5. In the case of Figure 3.5a, where no pixel shows a hit, the result is the same. For a row, which contains a signal, such as pixel number 6 in Figure 3.5b, the median value for this row, shown as solid red line, differs from the mean value, shown as dot-and-dashed green line. This example shows that the median value compared to the mean value is not disturbed by the pixel number 6. But not only photon signals could disturb the common mode calculation. On DEPFET matrices used for developing and test setups, pixels sometimes are called "bright". These are damaged pixels, which often show an extraordinary high signal value.

For a rough estimation of the difference between both approaches a typical signal value of 4,500 ADU and a common mode value of 50 ADU is used. With an occupancy of approximately



(a) Example row without a pixel containing a signal. The median and mean value cannot be distinguished in this figure (ME ≈ MD).



(b) Example row with pixel number 6 containing a signal. MD is the median value and ME is the calculated mean value for this row.

Figure 3.5: Example of a row common mode calculation for a detector row with and without a photon signal.

5 %, three pixels contain a signal in a detector row with 64 pixels. The this case, the median is still 50 ADU compared to the mean value of 258.6 ADU; this would result in a calculation error of 208.6 ADU.

The common mode can be seen as a detector row individual offset correction. This individual correction value is highly frame-dependent and corrects fast disturbances. The following subsection will describe the pixel-individual offset correction, which in comparison is calculated statically and updated only for a new measurement data set.

3.1.3 Offset

If a DEPFET detector matrix is used for the X-ray detection, the offset level is a pixel-individual value. A combination of unavoidable variations in silicon processing, matrix control signals and influences from the matrix readout sequence causes a pixel-individual base level. This base level is, furthermore, amplified by several independent amplification stages up to the digitization stage. The amplification factor for each pixel also varies slightly over the DEPFET matrix. Each detector hybrid has, therefore, its individual and characteristic offset map. An example is shown in Figure 3.6.

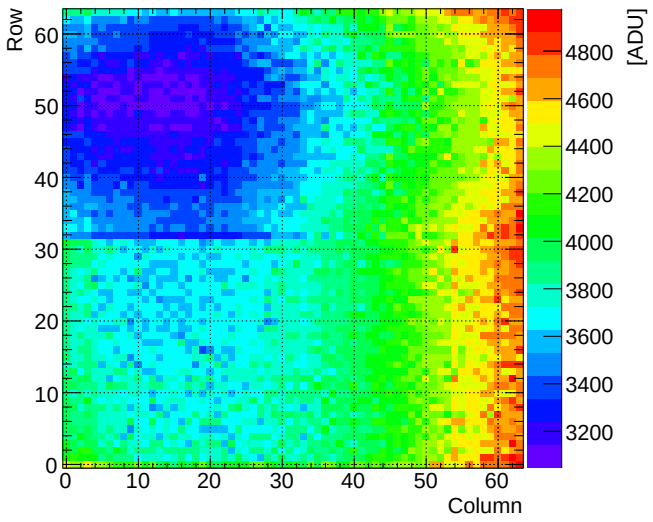


Figure 3.6: The figure shows a typical offset map of a 64×64 MIXS detector matrix. MIXS detectors are read out in two directions, the north and south hemisphere. The hemisphere border in row 32 is visible in the offset map. This offset map was measured with the MIXS detector P07_W33_H19 mounted on LM 5 hybrid.

For the data analysis with ROAn, at the beginning of each measurement set a block of so-called dark frames has to be recorded. Dark frames are recorded in absence of any source and do not contain photon signals. This allows for the calculation of the pixel-individual offset value O_{ij} out of the first N dark frames of a measurement with Equation 3.4 without disturbance from photon signals. The pixel value for the position i, j in frame k is represented in Equation 3.4 by $x_{k_{ij}}$.

$$O_{ij} = \frac{1}{N} * \sum_{k=1}^{N} x_{k_{ij}} \tag{3.4}$$

It is assumed that the offset map calculated at the beginning of the measurement is constant over the complete measurement. Once determined, the offset map is subtracted from each

frame in this measurement data set. Especially in long-term measurements this approach is not accurate because of parameter drifts during the measurement time. In addition, reacting on offset changes created by disturbance sources in the field of view is not possible with this static calculation approach.

3.1.4 Noise

Noise estimation is an essential, but usually a memory- and computational-intensive task in X-ray data processing. The estimated noise level σ is used as a base for the event extraction. An inaccurate or wrong estimation of the noise σ can lead to an inferior energy resolution. For DAQ systems, where an event filter is applied to reduce the output data rate, inaccurate noise values σ can additionally increase the required output bandwidth.

Up to now, the pixel individual noise level is calculated from dark frames at the beginning of each measurement data set and is assumed to be constant for the entire measurement time. This calculation approach reduces the required amount of processing power drastically, but the error made with this assumption increases with the measurement time.

For each pixel $[i, j]$ of the detector, the noise σ_{ij} is calculated with the two Equations 3.5 and 3.6 over N consecutive Frames.

$$\mu_{ij} = \frac{1}{N} \sum_{k=1}^{N} x_{ij_k} \tag{3.5}$$

$$\sigma_{ij} = \sqrt{\frac{1}{N} \sum_{k=1}^{N} (x_{ij_k} - \mu)^2} \tag{3.6}$$

Compared to standard image or video noise calculation algorithms [70][55][106] the noise calculation is done individually for each pixel and not blockwise. This requires a time series of data values for each pixel x_{ij_k}. The reason for the pixel-individual calculation is that the noise level is used for the calculation of the event extraction threshold and, therefore, the pixel noise and not the image noise is relevant. Figure 3.7 shows an example map, where two crescents can be seen at the left detector border in the noise map. This is the place, where the two switcher ASICs are connected to the matrix via bond-wires. The increased noise level in this region is caused by infrared light emitted from the two switcher ASICs.

3.1.5 Event extraction

A central processing step to reduce the data volume is the event extraction. The step uses a pixel-individual threshold level to select all pixels which have to be further processed and analyzed. This approach is similar to the zero suppression technique for data compression. The information thrown away by this step cannot be reconstructed and, therefore, it is also a very critical processing step to achieve the best possible system performance.

Two pixel-individual levels are typically used for the event extraction, the primary and the secondary threshold, with the relation ($T_{Primary} \geq T_{Secondary}$). Each pixel above the primary threshold is a so-called hit and is directly used for further processing. A pixel above the secondary threshold is marked with a special flag and is only used for further processing if a neighboring pixel is above the primary threshold. Common values for the primary threshold are 5 sigma of the pixel noise and 3 sigma for the secondary threshold. Increasing these values will reduce the output data rate, but can deteriorate the energy resolution. Too small threshold levels will increase the number of noise hits and produce a higher output data rate. The higher number of noise events also raises the spectrum background level in the low-energy region. A precise and reliable noise estimation method is, therefore, an important key component of the data processing system.

Figure 3.8 shows the influence of different thresholds on the hit extraction. A high threshold value and the resulting loss of the charge fraction q for a photon event in the neighboring pixel is

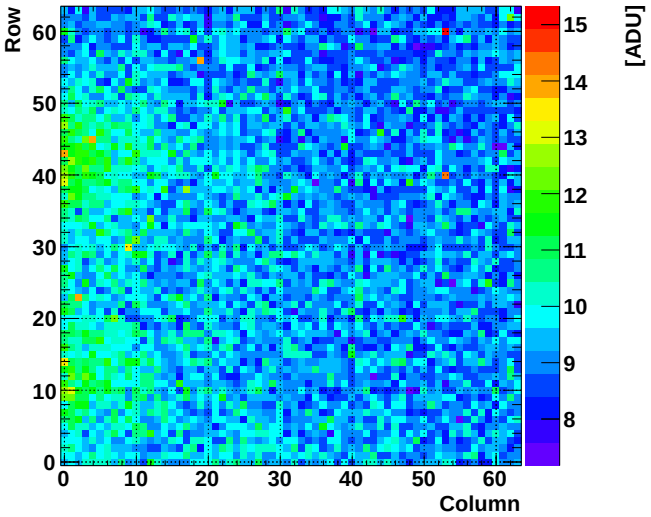
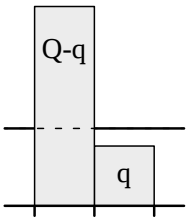
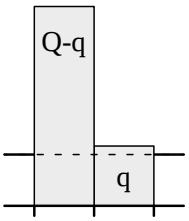


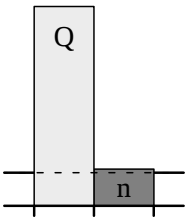
Figure 3.7: Noise map example of a hybrid equipped with a DEPFET matrix. At the left detector boundary two crescents of increased noise are visible. This results from infrared light being emitted by the two switcher ASICs at this position.



(a) High threshold values can lead to missing charge in a neighboring pixel.



(b) Example for an event where an optimal threshold value is applied.



(c) Low threshold values can lead to pseudo events and collect charge created by noise from neighboring pixels.

Figure 3.8: These figures show the influence of different threshold values.

depicted in Figure 3.8a. The optimal threshold level applied to a particular event is depicted in Figure 3.8b; in this case the complete charge created by the photon event can be reconstructed. Finding the adequate threshold level is a difficult task, because it strongly depends on the X-ray detector used, its probability for charge split events and the split ratio. Low threshold values as depicted in Figure 3.8c increase the number of noise events and transform real single events into multiple events. This directly affects several parameters:

- The amount of charge collected for these events is erroneously increased
- The output data rate of the event extraction step is increased
- More processing power required to process the increased number of events

3.1.6 Pattern searching

Events extracted by the event extraction step are further investigated in the pattern searching step, where multiple neighboring hits are combined to a single object called pattern. Detecting and recombining of these patterns is important to decrease the low energy background for the so-called split events. A split event occurs when a photon hits the detector matrix close to the border of the neighboring pixel and the created charge is not entirely collected in a single pixel. If such a split event is not recombined to a single event, charge is lost and a wrong total event energy is calculated. This will deteriorate the energy resolution of the data processing system.

The patterns are categorized by their multiplicity:

- Singles (SNGs)
- Doubles (DBLs)
- Triples (TRPs)
- Quadruples (QUDs)
- Others

For the multiplicity classification, the number of hits in the pattern is used. With the pixel sizes used in DEPFET macropixel detectors today the maximum pattern size for a single photon event is 2×2 pixels. All larger appearing patterns are grouped together in the "others" category. These patterns can only be created by **m**inimum **i**onizing **p**articles (MIP), noise coincidences or multiple photon events in close proximity on the detector matrix.

An illustration for the charge distribution over multiple pixels is shown in Figure 3.9. In this figure the charge cloud created by photon (a) is close to the center of the DEPFET pixel above and the entire charge created by this photon is collected in the Internal Gate of a single pixel. Photon (b) creates the charge cloud directly below the border of two DEPFET pixels; in this case the charge is split and collected in the Internal Gate of these two pixels. The charge split ratio depends on the position of the photon hit in relation to the pixel border.

Not all possible hit combinations can be explained by the charge deposition and splitting of a real photon event. Therefore, in each type class valid and invalid pattern combinations have to be distinguished. All pattern combinations depicted in Figure 3.10 are valid pattern. In these figures, pixels above their individual primary event threshold are marked with "P" and pixels above their secondary threshold are marked with "S". The double pattern shown in Figure 3.10b is created by a photon hit somewhere between the center of the primary pixel and the pixel border to the secondary pixel; otherwise, the pixel above the secondary threshold would collect more charge. In vertical direction the charge cloud is created close to the center line between the two pixels, because no pixel above or below shows a significant signal. An algorithm calculating the center of charge can be used to precisely estimate the point where the charge cloud was created. Further investigations [47] showed that a sub-pixel size precision is achievable with such an approach.

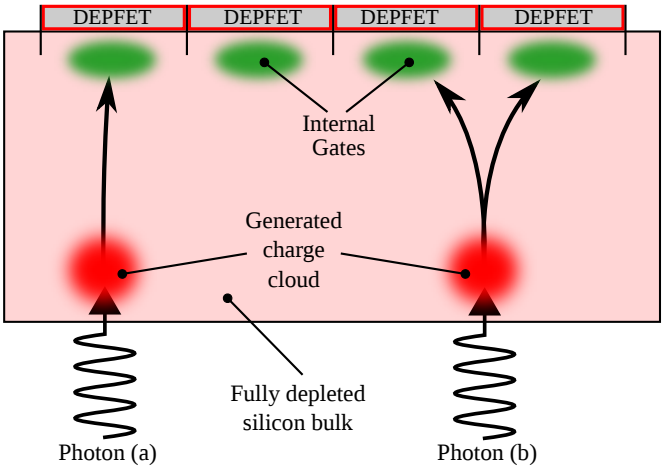
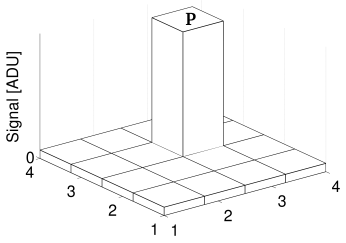
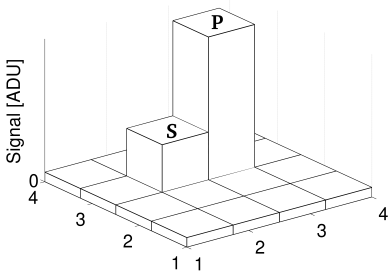


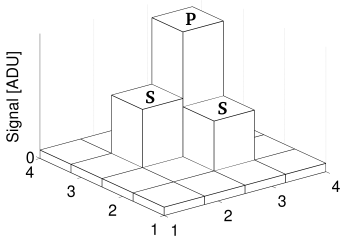
Figure 3.9: Charge collection and spread over multiple pixels dependent on the arriving point of the photon. The charge created by photon (a) is entirely collected in one pixel because the charge cloud in the silicon bulk is created directly below the Internal Gate of this pixel. Photon (b) creates the charge cloud below the border between two DEPFET pixels. This charge cloud is divided into two fractions during the drift into the internal gates and collected in two pixels. The split ratio depends on the relative position where the charge cloud is created in the silicon bulk in respect to the pixel border. Figure similar to [51, p. 86].



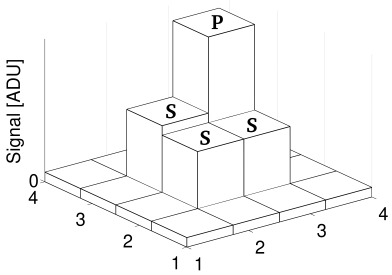
(a) Valid single pattern.



(b) Valid double pattern.

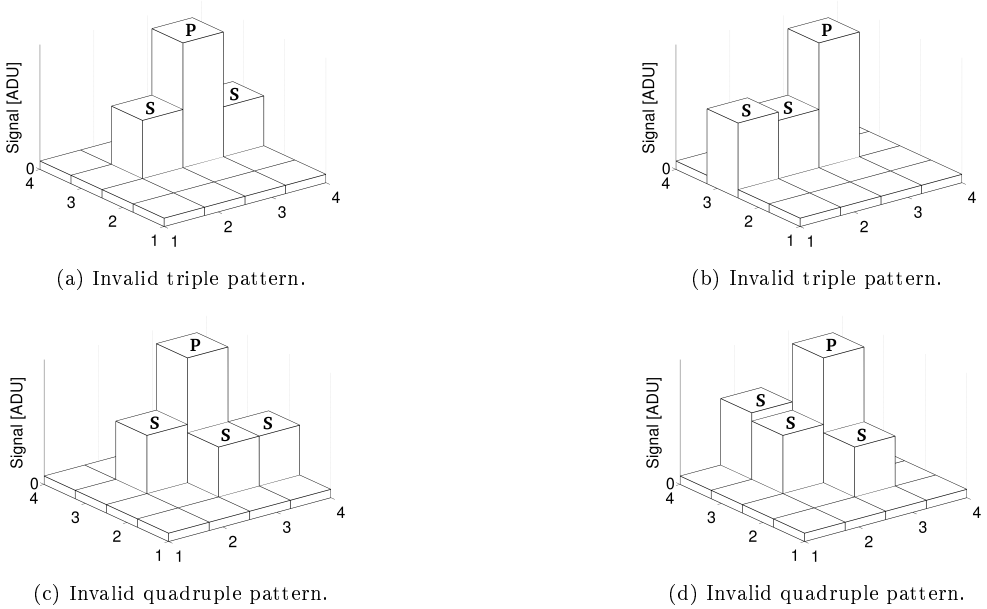


(c) Valid triple pattern.



(d) Valid quadruple pattern.

Figure 3.10: Overview of possible valid base pattern types.



(a) Invalid triple pattern.

(b) Invalid triple pattern.

(c) Invalid quadruple pattern.

(d) Invalid quadruple pattern.

Figure 3.11: Overview of invalid base pattern types.

Invalid pattern combinations are depicted in Figure 3.11. These patterns cannot be created by a single photon event. The triple pattern in Figure 3.11a, for example, cannot be created by a single photon event, because the left pixel above the secondary threshold indicates that the photon created the charge cloud close to the pixel border on the left side; on the other hand, the right pixel above the secondary threshold would require exactly the opposite. Such a pattern can be created in two ways; first, the pattern can be created if the left or right pixel above the secondary threshold is created by a noise event (noise coincidence) and, secondly, the pattern can be created if one photon arrives during the detector integration phase near the left pixel border and a second photon arrives near the right pixel border (pile-up).

3.1.7 Gain calculation and calibration

Each matrix pixel includes an individual first amplification stage. Due to production process variations during the manufacturing of the DEPFET matrix this amplification factor is not constant. Additional amplification stages in the ASTEROID readout ASIC, signal buffers in the analog transmission line and on the ADC-card also contribute to the gain variation, since not every pixel sees the same gain value up to the ADC. The ASTEROID gain changes over the detector matrix columns, because each matrix column is processed in an individual analog channel with independent amplifiers (see Chapter 2.3.2). The transmission line and ADC input buffers amplify the serialized signal from the ASTEROID for the entire matrix section of this readout branch. Therefore, each matrix pixel has to be seen as an individual readout channel with a specific amplification factor. In Figure 3.12 the total signal amplification chain for a DEPFET readout system is shown schematically for a single pixel.

To enable Fano-limited measurements with the detector system this variation in the pixel-individual gain needs to be corrected in the data analysis. During the calibration process, the ROAn analysis software calculates the gain value for the total signal chain of each detector pixel. This allows for a direct gain correction for every pixel with a simple multiplication operation.

A typical gain map and histogram from a MIXS detector module is depicted in Figure 3.13. The two main Gaussian peaks in this figure are a result of the bidirectional readout schemata, where two independent analog signal chains with slightly different gain values are used up to the ADC-cards.

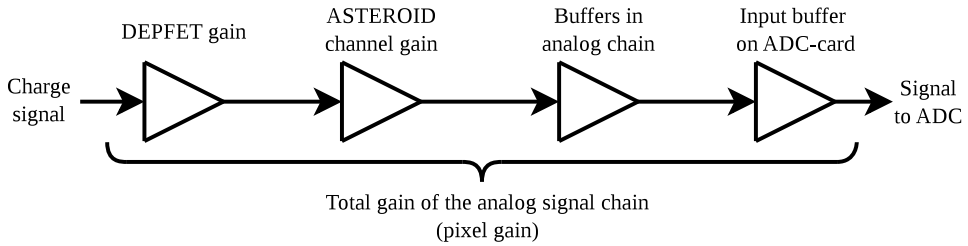
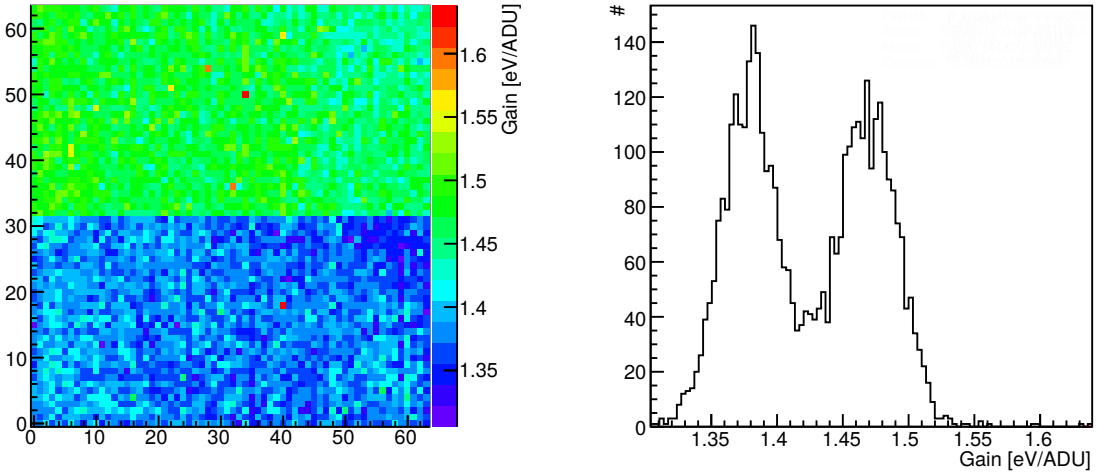


Figure 3.12: Signal amplification chain for the complete DEPFET readout chain.



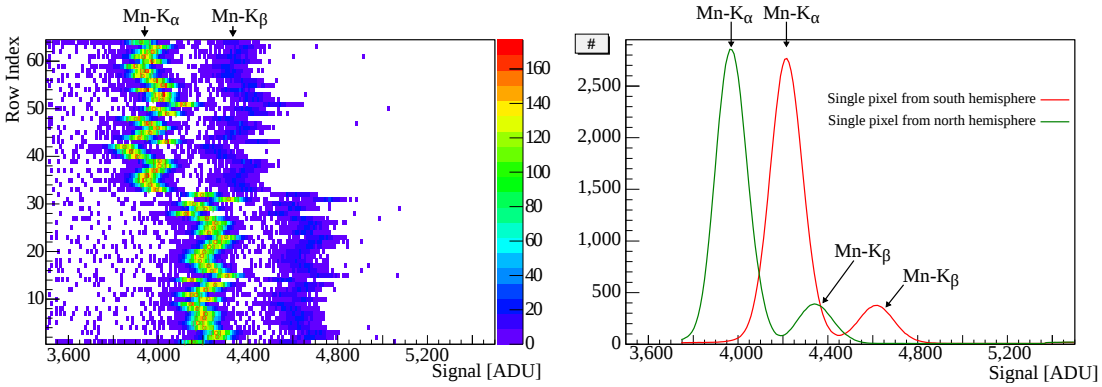
(a) Pixel gain map from a MIXS detector module. The detector matrix is separated into the north and south matrix hemisphere. Both hemispheres are read out by an individual readout chain with slightly different gain factors.

(b) Map histogram of the pixel gain from a bidirectional readout MIXS detector module. The signal gain values in both readout branches are slightly different. This creates the two main Gaussian peaks in the histogram.

Figure 3.13

For the pixel gain calibration process a reference measurement with the detector system and an X-ray reference source with known mono-energetic X-ray emission lines is required. The ^{55}Fe sources used to determine the energy resolution of a detector system as described in Chapter 2.2 also permit a precise pixel gain calibration due to their two photon energy peaks Mn-K$_\alpha$ at 5,899 eV and Mn-K$_\beta$ at 6,490 eV [79]. ROAn typically uses only single (SNG) patterns from the Mn-K$_\alpha$ peak to calculate the pixel-individual gain value. Using only SNG pattern allows for achieving the highest possible calculation precision, but with this method only a fraction of the recorded photon events can be used for the calibration. Therefore, the required number of photon events to calibrate the detector system is higher and leads to a longer recording and calibration time. In addition, the illumination homogeneity of the X-ray reference source on the detector matrix is important for a successful gain calibration of each pixel. For detector systems with large matrix dimensions this is a challenging task and requires a strong X-ray source. Weak X-ray sources or inaccurately adjusted sources result in pixels where no gain value can be calculated. The data analysis software for CCD detector systems uses an alternative processing algorithm, where all recorded photon events can be used for the calculation. This iterative gain calibration method has, up to now, not been adapted to the DEPFET data analysis.

For the current gain calibration method in the DEPFET data analysis, the mean value of the single (SNG) pattern is needed for each pixel. This calibration signal s_i is calculated for the detector pixel i from the pixel-individual SNG histogram. In Figure 3.14 an example SNG pixel



(a) SNG column map (b) SNG pixel histogram

Figure 3.14: SNG column map and SNG pixel histogram from the P07_W33_H19 detector module.

histogram from a MIXS detector module is shown. These SNG pixel histograms contain the values from all single patterns recoded during the calibration measurement in the specific pixel. To extract the calibration signal value s_i from these SNG histograms, a two-stage Gaussian fit process is used. First, a Gaussian fit is calculated where the mean value is fixed to the maximum position in the SNG histogram. For the second Gaussian fit, the fit range is set to $\pm 3\sigma$ of the first Gaussian fit, while all other parameters are free. This two-stage fitting approach provides a highly reliable and precise process for the automatic value extraction.

The gain g_i for a pixel i is defined in Equation 3.7 as the ratio between the calibration signal s_i from the measured reference data set and the known calibration energy E_c of the used reference source. This pixel gain value depends on a variety of different system parameters such as ASIC configuration, system temperature, readout speed and matrix voltages. Therefore, this gain value can be only used to calibrate measurements made with the same detector system and with identical system parameters. For measurements made under other conditions the pixel gain value g_i needs to be recalculated.

$$g_i = \frac{s_i}{E_c} \tag{3.7}$$

With the pixel gain value g_i and the Equation 3.8, all occurring patterns can be calibrated. The total energy of the pattern E_{Pat} is the sum of all contributing signal parts corrected by the corresponding gain value.

$$E_{Pat} = \sum_{1}^{N_{Pat}} \frac{s_i}{g_i} \tag{3.8}$$

3.1.8 Bad pixels

Due to manufacturing failures, detector matrices can contain pixels with irregular behavior. To avoid disturbances of the analysis results from these bad pixels and their neighbors, all patterns containing these pixels have to be excluded for the data analysis. For DEPFETs three types of bad pixels can be distinguished:

- **Bright** pixels show compared to other pixels, a clearly higher offset value.
- **Dark** pixels have a significantly lower gain value than other pixels. Dark pixels are directly identified in the gain map by applying a certain threshold.
- **Noisy** pixels have a drastically higher noise level compared to all other detector pixels. These pixels are identified in the noise map by applying a certain threshold.

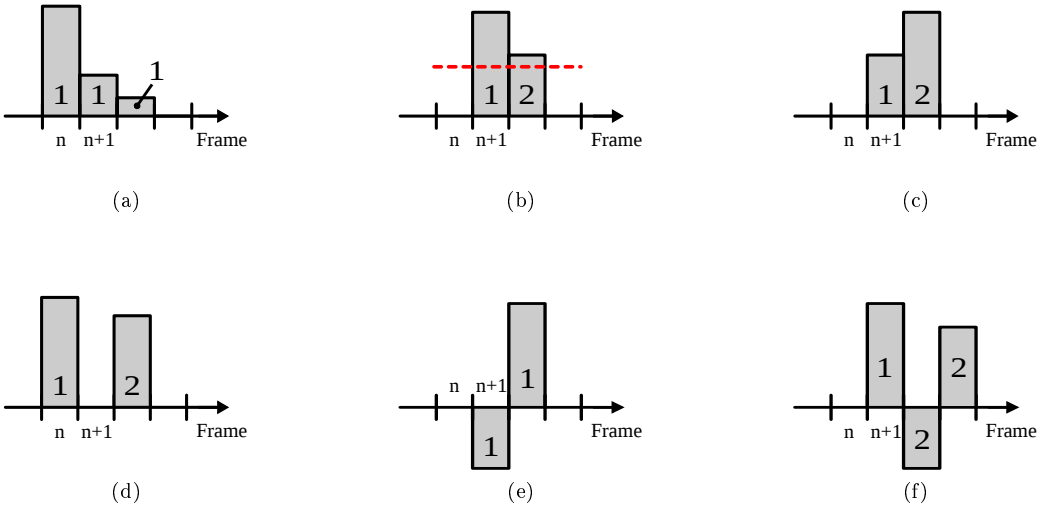


Figure 3.15: Different pixel signal sequences of one pixel in consecutive frames. In 3.15a a clear-correlation cluster is depicted. 3.15b shows two consecutive pixel signals, where the second signal is treated as a separate hit because it is above a certain detection threshold. Whenever the pixel signal increases (3.15c) or an intermediate frame is without a signal (3.15d), the cluster is terminated. Negative pixel signals as depicted in 3.15e and 3.15f are representing misfits.

All patterns containing these bad pixels are excluded from the data analysis process. For a high reliability of the automated data analysis bad pixels are automatically detected and extracted from the residual offset, noise and gain map and collected in the bad pixel map. For a higher flexibility and adjustability the bad pixel map can also be created or combined with a manually built bad pixel map.

3.1.9 Clear-correlation and misfit clustering

A clear-correlation is created by incomplete charge clearing from the Internal Gate of the DEPFET. This results in an additional signal in the consecutive readout frames created by the remaining charge. This clustering of clear correlations improves the energy resolution due to the reduction of wrong signals appearing in the low-energy background.

Figure 3.15 shows possible variants for pixel signal sequences of one pixel in consecutive frames. The numbers in these figures indicate the event number to which the signal is assigned. For the physical explanation of a clear correlation the pixel signal needs to degrade continuously from one frame to the subsequent frame as depicted in Figure 3.15a. This figure shows a single event with a cluster length of three frames. If the signal of this pixel in the consecutive frame is below an adjustable threshold value, the clear-correlation cluster will be continued. Otherwise, the consecutive pixel signal is evaluated as a separate photon signal. In Figure 3.15b the consecutive pixel signal is above the assessment threshold value and is, therefore, treated as a second event. The threshold value can be adjusted to the matrix characteristics to avoid wrong cluster assignments. Intermediate frames without a signal in the pixel or an increased pixel signal as depicted in the Figures 3.15d and 3.15c terminate the clear-correlation cluster. Negative pixel signal values, as depicted in Figure 3.15e and 3.15f, are never part of a clear-correlation cluster. As described in Subchapter 2.2, negative pixel signals can only be created by misfits.

All occurring misfit events are grouped in misfit clusters. This misfit clustering process is similar to the clear-correlation clustering with the difference that the pixel signal values are analyzed for negative signal values. If a negative pixel signal is found, the consecutive frame contains the proper charge signal for this pixel. Negative signals can be created if charge is

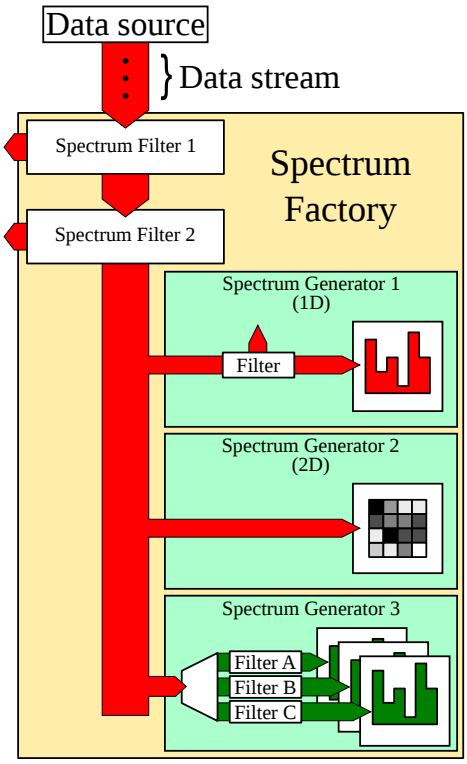


Figure 3.16: Sketch of the internal structure of Spectrum Factories. Spectrum Factories are containers for an arbitrary number of result calculation algorithms called generators. Data stream reading and filtering is shared to save computation resources. A similar figure can be found in [51].

collected in the DEPFET during the second measurement phase of the ASTEROID. This charge is correctly measured and removed from the DEPFET during the next readout of the pixel. The length of such a misfit cluster is, therefore, fixed to two consecutive frames compared to the clear-correlation cluster, where any cluster length is possible.

This additionally stored information will be used in the analysis result generation step (Subchapter 3.1.10) for fast filtering of relevant data entries.

3.1.10 Analysis result generation

All hits in the data set are now extracted, tagged, clustered and gain-corrected. The next step is to calculate all requested histograms and statistics parameters. For this task, the entire event tree needs to be rescanned and filtered by several algorithms event by event. To increase the processing performance and avoid multiple file reread operations, suitable algorithms are grouped together in a data processing container called “Spectrum Factory”. The internal structure of such a spectrum factory is shown in Figure 3.16. The individual processing algorithms in the Spectrum Factory are called generators. The input data stream can be filtered by an arbitrary number of filter units before the input data stream is provided to each of the associated generators. Thus, for a certain combination of patterns, the data stream filtering is done only once during the entire data processing. This reduces the processing overhead compared to a data stream filtering in each Generator. Separating the filter algorithm from the Generator algorithm also enables a compact and clean software architecture with the flexibility to place any filter combination in front of a certain generator. The applied filters can use the value of the data entry itself and the previously calculated results of the clustering and tagging steps to decide if the current input value is dropped or forwarded to the next processing unit. The

base algorithm of the generators can thereby be used to calculate the desired result for any cluster type and **r**egion **o**f **i**nterest (ROI). This concept fits optimally into the primary task of the ROAn software to create a universal and highly flexible offline analysis tool.

The Spectrum Factory configuration depicted in Figure 3.16 contains two shared filter modules. Clear-correlation clusters with a length of more than one frame are, for example, removed in the first filter, while the second filter removes misfit clusters. The first Spectrum Generator contains an additional local filter to select, for instance, only single (SNG) clusters and then calculates the calibrated SNG energy spectrum. The second Spectrum Generator has no supplementary filter and extracts a two-dimensional cluster distribution map called Hitmap. The third Generator in Figure 3.16 inherently provides multiple result histograms. This is used if results of a Generator have to be calculated in multiple variations such as the pattern type statistics. This result has to be calculated for all cluster sizes such as double, triple and quadruple.

3.1.11 Passport creation

The final step in the ROAn data processing is the creation of the analysis passport. This passport is a composition of all initially requested histograms, maps and statistical parameters from the various generators and the analysis results previously calculated by other processing steps. The passport creation itself does not process data at all. This step only collects results from other processing steps and takes care of the result presentation form. The page layout is defined via XML layout templates. ROAn provides standard XML layout files for the different results to enable a fast presentation of analysis results without any further efforts for the user. For further use, the analysis passport is usually exported into a postscript file.

3.1.12 Custom analysis and processing

Customized analysis and processing steps allow adding special processing algorithms for application-specific analysis functions directly into the ROAn software. The step-based processing approach of ROAn enables the direct integration of customized processing algorithms into the data analysis software. The user needs to implement only the specific processing algorithm, while all the data access and storage functionality is provided by the ROAn core framework. The standard ROAn processing steps can be used as code templates for the implementation of the customized steps.

Chapter 4

Investigation of the ROOT-based Offline Analysis

The analysis of the ROAn processing performance and the investigation of different processing acceleration techniques in this chapter is an important input for the SuMo-DAQ development. Furthermore, the knowledge about the performance and capability of the ROAn algorithms is crucial to avoid performance bottlenecks in the design of the SuMo-DAQ system. In addition, the investigation of different processing platforms for the X-ray data processing allows for identifying possibilities to improve the processing performance of the current ROAn implementation.

Performance evaluation of the ROOT-based Offline Analysis

The analysis speed of the ROAn data processing software was analyzed on the base of multiple real measurement data sets recorded with the old PCI-based full frame DAQ system. For the run-time analysis the ROAn source code was extended with timers to measure the execution time for the different processing steps. In order to obtain exact execution times and to change the run-time conditions as little as possible, the use of statistical analysis tools such as the profiling tools PERF and OProfile [49, 15] was avoided. The execution times listed in Table 4.1 are the mean durations of several identical analysis runs on the same data set. ROAn utilizes only a single CPU core for the data processing. The complete ROAn processing time is, therefore, the sum of the measured CPU time and the I/O time.

Table 4.1 shows the processing time of the different ROAn processing tasks for a typical detector calibration measurement data set. The data file size of such a calibration measurement made with the MIXS detector system (Subsection 2.4.1) is approximately 18 GB and contains roughly $2.3 \cdot 10^6$ frames. This amount of data is necessary to have enough statistic for an accurate detector characterization and calibration. To avoid disturbances of the measured analysis run time by file I/O operations, all analysis runs where made on a powerful machine of the latest generation with 512 GB of RAM to enable the caching of the entire input data file. Nevertheless, the ROAn analysis software required on average $1,786.8\,\mathrm{s}$ ($\approx 29.8\,\mathrm{min.}$) to process the full data set, while the total data recording time is only $360\,\mathrm{s}$ ($6\,\mathrm{min.}$). This shows the gap between the produced output data rate and the capability of the current data processing speed solution. Even for the detector systems currently used the difference is approximately a factor of 5. With the next generation of detectors with more than 1 MPixels and a frame rate of $1,000$ frames per second this factor will increase to roughly 200.

The evaluation of the different task execution times (Table 4.1) shows that the analysis tasks, which are by far the most time-consuming ones are data conversion, combined offset and common mode correction. The data conversion, where the complete raw data stream has to be scanned for keywords to verify the frame consistency, consumes more than one third of the

Processing task	Processing time in percent	Absolute processing time [s]
Data conversion	33.02	589.92
Static offset & noise map calculation	∼0	0.10
Offset and common-mode correction	30.87	551.52
Event extraction	14.86	265.44
Bad pixel map	∼0	0.01
Pattern search	5.35	95.54
Gain map calculation	2.34	41.81
Event calibration	3.11	55.70
Analysis result calculation	10.42	186.17
Passport creation	∼0.03	0.59
Sum		1,786.8

Table 4.1: ROAn processing time for the analysis of a data set used for the detector calibration. The data set contains roughly $2.3 \cdot 10^6$ frames and has a size of approximately 18 GB. This size is typically used to have enough statistic for a precise detector calibration.

entire processing time. The full frame data recording strategy creates large data sets. The data conversion and consistency check not only needs to read the entire data file; after checking and converting the raw input data nearly the same amount of data needs to be written. For a single analysis run, where the file caching is not effective, the generated I/O load puts high demands on the storage system.

Almost the same time is required for the combined offset and common mode correction (Subsection 3.1.2), where all of the approximately $2.3 \cdot 10^6 * 64 = 147.2 \cdot 10^6$ rows in the data set have to be sorted to calculate the row median value and to subtract it from the $2.3 \cdot 10^6 * 4,096 = 9.42 \cdot 10^9$ pixels. The ROAn software combines the offset subtraction and the common mode correction in a single processing step due to performance reasons. Therefore, these analysis processing parts where not separated for the performance investigation. Due to the data sorting required, the processing is a computational intensive task. ROAn uses a function provided by ROOT to calculate the median. For the sorting itself, this ROOT median function calls the "std::sort()" function provided by the C++ Standard Library [76]. The specific sorting algorithm implemented in the library can vary across different libraries, but often the introspective sorting [61] algorithm is used. Independent of the particular algorithm implementation, the complexity should be on average $\mathcal{O}(n \cdot \log(n))$. More detailed runtime measurements showed that the sorting requires already more than 55 % of the entire processing time needed for the combined offset and common mode correction.

The third most time-consuming processing step is the event extraction. Comparing every single pixel with multiple pixel-specific threshold values and the insertion to the data structure used for the further event data processing is time-consuming. Nevertheless, this step is crucial to reduce the amount of data which has to be processed by the following steps and helps to decrease the residual processing time. The event extraction is, furthermore, an extremely sensitive processing step to achieve the best analysis results. In this step, each misfiltered event can degrade the energy resolution and increase the required storage space.

The pattern search benefits from the data reduction and is applied only to the events already extracted. Therefore, the number of pixels to be checked for adequate neighbors (to create a cluster) is significantly reduced, which saves a lot of processing time. The contribution of the pattern search to the total processing time is merely 5 %.

Up to now, only single events are used for the gain map processing. For this reason, the gain map processing needs to be placed after the event filtering and the pattern search. This requires an additional walk through the entire cluster list to generate pixel histograms needed for calculating the gain values.

The analysis report generation applies further filters on the pattern list to calculate all the spectra, histograms and statistical parameters requested by the user. This processing task strongly depends on the analysis goal of the user. Here the flexibility of the ROAn design shows all its advantages. The analysis task does not necessarily require real-time processing capability because this part is rather used to investigate special detector characteristics and behavior. If some unusual effects are visible during the first data analysis, a more detailed analysis is launched in a second cycle to understand these effects.

All the other processing tasks consume only a small fraction of the complete processing time and are not analyzed in more detail. A major advance in order to enable real-time data processing, therefore, cannot be achieved by the optimization of these tasks.

Discovered limitations of the ROOT-based Offline Analysis

The PCI-based full-frame DAQ system and the ROAn X-ray data processing software commonly in use until now are two completely decoupled systems. This means that the DAQ hardware is used only as data acquisition interface to the detector system and does no data processing at all. The ROAn software cannot process data during the data acquisition phase because of its software design. Each processing step in the ROAn software design [51] needs the entire data set to calculate its results, which are then used by the subsequent processing steps. The concept limitations generate a lot of delay time between the end of a measurement and the availability of the analysis result.

Furthermore, large data sets have to be recorded and stored with this system. The required data conversion and consistency check in the ROAn software doubles the amount of required storage space and creates a high I/O load on the storage system.

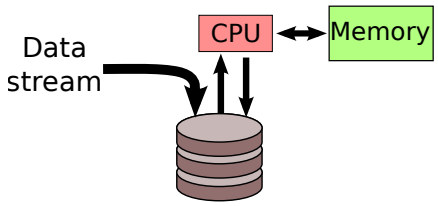
Single-threaded applications are no longer state-of-the-art for the processing of large data sets and cannot benefit from the further development of current multi-core processor architectures.

For the creation of the complete analysis report by the ROAn software, multiple sequential processing runs through the entire data set and the cluster list are necessary. This significantly decreases the data processing efficiency even on powerful machines with a lot of RAM, which enables full data caching.

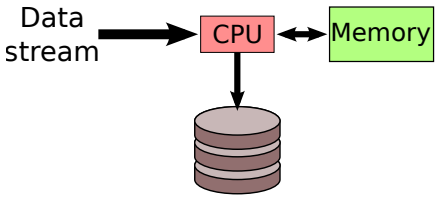
4.1 Evaluation of data processing platforms

This chapter presents and evaluates various architectures and processing platforms for the X-ray data analysis. To achieve the best system performance, the requirements and abilities of the used processing device and architecture have to be considered for the implementation of the different processing algorithms. The evaluation results are used later to design a suitable architecture and real-time data processing concept for the SuMo-DAQ system.

The single CPU processing architecture currently used is depicted in Figure 4.1a. The data stream is transferred on to the storage system by the DAQ system and then read again from the system for the processing. Direct data stream processing as shown in Figure 4.1b could reduce the I/O load on the storage system. As the previous analysis of the processing performance has shown, a single CPU cannot handle the data stream in real time even from small detector systems. The system concept does, furthermore, not allow scaling the processing system to operate the next generation of X-ray detectors. The single-thread limitation of the ROAn design and the inability of data stream processing prevents the use of the ROAn architecture even for small detector systems. Design modifications and code rewriting in the ROAn analysis software is necessary in any case to increase the analysis throughput. Multi-threading extension of the data processing is investigated in the following subchapter.



(a) Currently used single CPU offline data processing architecture. The Data stream is directly transferred to a storage device and read again from the storage device for the data analysis.



(b) Direct data stream processing with a single CPU. The processing power of a single CPU core is not sufficient to process the complete data stream.

Figure 4.1: Single-CPU processing architectures for one input data stream.

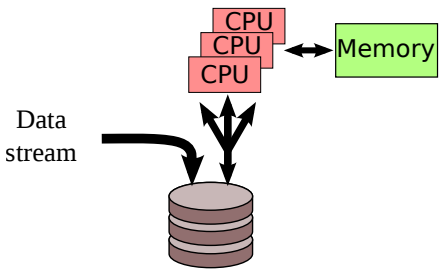
4.1.1 Multi-threaded data processing on CPUs

This subchapter investigates and evaluates the standard x86 CPU computation platform for the X-ray data processing task. Multi-threading extensions of the X-ray data analysis can benefit from the latest multi-core architecture development of the platform while single-threaded applications have not been able to see any major improvement during the last years. The reason for this limitation is the achieved clock frequency boundary of CPU cores on silicon.

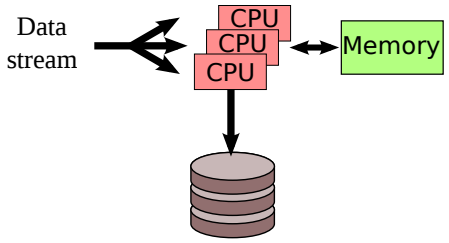
The ROAn data analysis software makes heavy use of the TTree [12] data storage structure from ROOT to store and exchange the extracted events. Up to now, the TTree implementation in ROOT has unfortunately not been completely thread-safe. ROOT manages all files created during the analysis process and also handles the file I/O operations of the TTree storage structure. Without a thread safe implementation of these ROOT code parts, the implementation of multi-threading algorithms in ROAn is quite complex. For the implementation of parallel processing steps in ROAn, the file I/O operations need to be protected and synchronized manually by mutexes in a particular processing step. This creates an extensive and difficult-to-read source code for the new processing steps.

Figure 4.2 shows two possible architectures for multi-threaded X-ray data processing. First, the data processing from a storage system as depicted in Figure 4.2a is discussed.

To enable the reuse of other data processing steps from the ROAn analysis software the code modifications have to be made without any change in the produced result data structure. The single-threaded code processes all frames sequentially and strictly ordered by the frame number. For this reason, the calculated result data structure is also ordered. To produce



(a) Multi-core data processing from a storage system. This architecture generates a high I/O load on the storage system.



(b) Multi-core data stream processing.

Figure 4.2: Evaluated real-time processing architectures for a single input data stream.

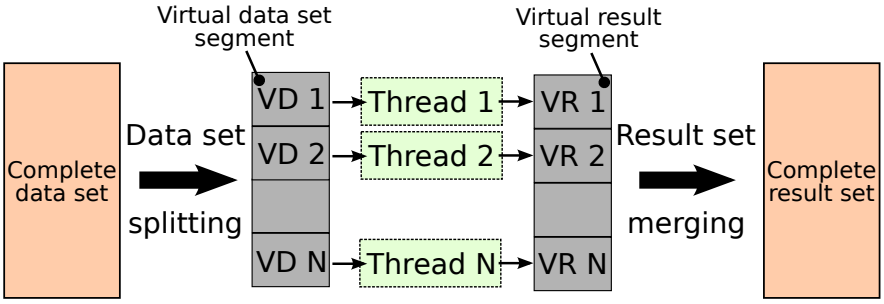


Figure 4.3: Data set splitting for efficient multi-threaded offline data processing in the ROAn software.

exactly the same data structure with a multi-threaded algorithm implementation the complete data set is divided into multiple virtual data sets. The splitting, processing, and merging is shown in Figure 4.3. Each data set segment is handled by an individual thread. After the processing of the virtual data sets is finished, the results are concatenated to the final result. The frame order in this final data structure is then exactly the same as for the sequential data processing. This segment processing additionally reduces the necessary thread synchronization effort and increases the processing throughput. The data file splitting is done in bytes for the data conversion and the consistency check and in frames for all other processing steps. The runtime difference between the threads for the processing of the various data set segments can be neglected for large data file sets.

For the multi-threading extension of the ROAn analysis, the offset and common-mode correction is combined with the event extraction in a single processing step. The measured analysis time and the calculated speed-up factor for this combined processing step is shown in Figure 4.4. For realistic speed-up results even at high numbers of threads, the measurements where made on a high-end computer system with 32 real processing cores. The best achieved speed-up factor for the combination of these processing tasks was slightly above 6 for concurrently utilizing 16 threads. A higher number of concurrent threads did not allow for further significant improvements in processing speed due to the necessary processing overhead for data set splitting and I/O synchronization for the threads. The minimal processing time for the combined step was roughly 134 seconds. This is a significant reduction compared to the single-thread data processing, where approximately $551 + 265 = 816$ seconds (Table 4.1) were needed. However, this would just be sufficient to adapt the data processing speed for the offset, common-mode, and event extraction tasks to the current detector data rate. The speed-up curve in Figure 4.4 shows that a further increase of the processing speed cannot be achieved simply by increasing the number of used processor cores.

The required analysis time and speed up for the multi-threaded data conversion step with and without enabled data caching is depicted in Figure 4.5. The shortest measured analysis time for the data conversion with enabled data caching was ~ 126 seconds and was achieved with 16 processing threads. This corresponds to a peak speed-up improvement of ~ 4.7. For a one-time analysis of a data set, where the data caching does not work properly, the storage system had to provide an average data rate of $\frac{18 * 1{,}024\,\mathrm{MB}}{126\,\mathrm{s}} * 2 \approx 292.6\,\mathrm{MB/s}$ in random read/write access. Providing this bandwidth continuously in random read/write accesses is a tough requirement for a storage device and limits the scalability of the data conversion.

The influence of file caching and the used file system is difficult to determine and strongly depends on the used storage system. To identify the influence of the storage device in a test measurement, a **r**edundant **a**rray of **i**ndependent **d**isks (RAID) system with 8 disks in a hardware RAID level 5 configuration was used. The file system cache of Linux was continuously flushed during the analysis performance evaluation without data caching. The measurement results depicted in Figure 4.5 show that even on a high-end storage system the conversion with disabled data caching is significantly slower than with enabled data caching and limits the

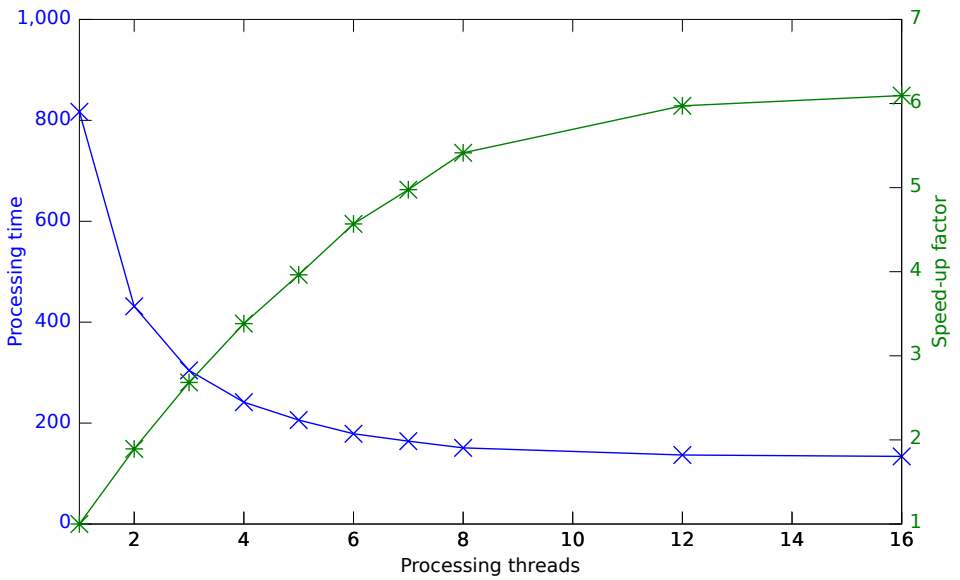


Figure 4.4: Measured processing time and speed up for the combined offset, common-mode and event extraction step of the multi-threaded implementation.

scalability. For more than 8 concurrently working threads on the storage device the performance starts to decrease. The reason for this is most probably the I/O limitation for random accesses on the storage system.

The achieved speed-up factor matches the theoretical speed-up calculation with Amdahl's law [3]. Since Amdahl's law is a pessimistic assumption [38] for the estimation of the speed-up factor, it has been repeatedly used as a basis for advanced and more precise formulas. The law is sufficiently accurate at this point, since only a rough estimate of the possible increase in performance has to be made. Equation 4.1 describes Amdahl's law, where P is the algorithm fraction, which allows for parallel processing and N is the number of used processor cores. Figure 4.6 shows the calculated speed-up factor for various P values. By comparing the measured speed up and the theoretically estimated speed up, the P value for the combined offset, common-mode correction and event extraction in ROAn is approximately $P = \frac{8}{7} * (1 - \frac{1}{5.95}) \approx 95\,\%$. With this processing parallelism, the speed-up factor for ROAn would not be enough for the next generation of X-ray detectors even with 128 CPU cores.

$$Speedup(N, P) = \frac{1}{(1 - P) + \frac{P}{N}} \tag{4.1}$$

The I/O load on the storage system could be reduced by a direct data stream processing architecture as shown in Figure 4.2b. Nevertheless, the speed-up factors realized with the multi-threaded data processing for the computational intensive offset, common-mode and event extraction step show that an operation of the next generation of X-ray detectors in real time cannot be achieved with this technique. Furthermore, the current ROAn software architecture does not support data stream processing. A modification of the ROAn data processing structure allows for reusing only very little software parts. Without the ability to reuse the source code, no significant advantage for the data acquisition and processing concept base on the ROAn software exists.

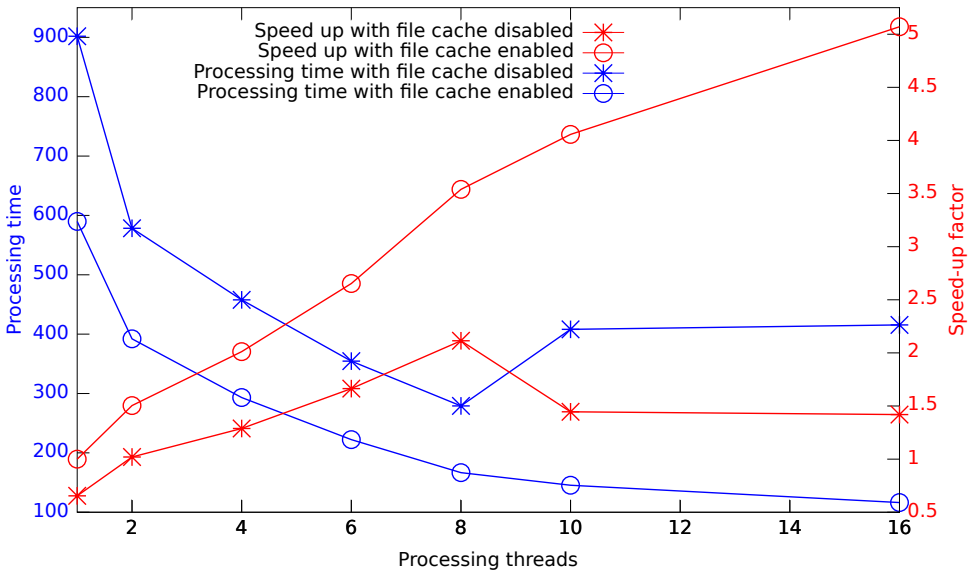


Figure 4.5: Measured processing time and speed up for the multi-threaded data conversion and consistency check with enabled and disabled data caching.

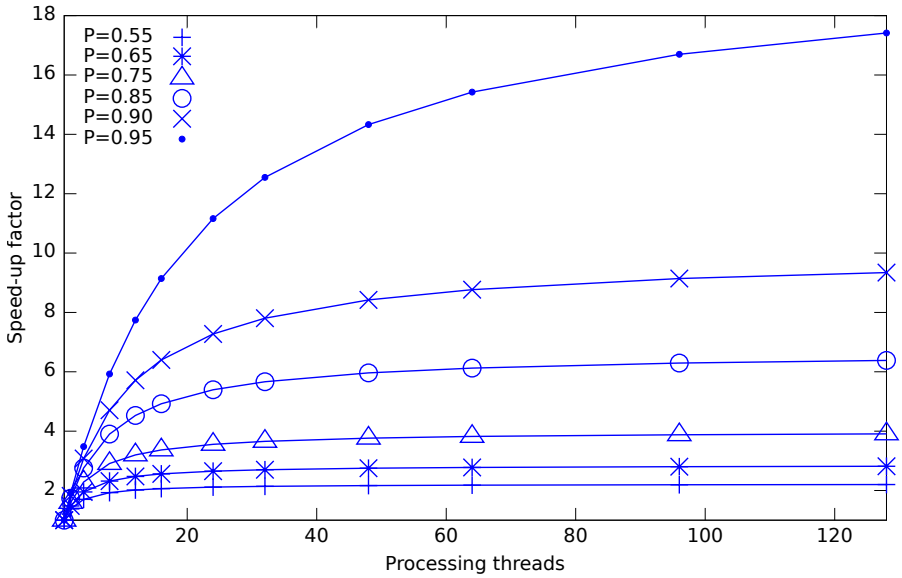


Figure 4.6: Estimated speed-up factors for various parallel processing portions of the algorithm with Amdahl's law.

4.1.2 Data processing on DSPs

Digital **s**ignal **p**rocessors (DSP) can provide a significant processing performance advantage for suitable tasks in combination with reduced power consumption compared to conventional high-performance CPUs. DSPs are specialized microprocessors, in which the architecture and the instructions are adapted to the requirements of digital signal processing.

The two most time-consuming analysis steps are the data conversion as well as the offset and common-mode correction. These steps have to process the complete data stream and spend most of the processing time for searching in the data conversion and consistency check step and for sorting in the common-mode calculation. For the estimation of the performance achievable by DSP processor cores the speed up of these tasks in comparison to general-purpose x86 CPUs is most important and has to be analyzed. For the acceleration of the processing by DSPs, a direct data stream processing architecture has to be used. Data processing from a storage system would require access to a file system on the data storage system. This is complex to realize, produces a lot of overhead on the DSP system and will run into the same I/O bandwidth challenge as the pure multi-threaded GPP solution.

DSP processor cores run with a clock frequency of up to 1.25 GHz [82], which is significantly lower than the 4 GHz [82] clock frequency of conventional high-end general purpose processors. Therefore, a single operation on a DSP takes longer than on a high-end GPP. The performance advantage of DSPs is generated by specialized signal processing instructions and a high degree of parallelization. In the near past, DSPs lost a lot of the parallelism advantage over x86 general-purpose processor cores due to the fast movement of the x86 CPUs to multi-core architectures and **s**ingle **i**nstruction **m**ultiple **d**ata (SIMD) operations. DSPs are prominent for their **m**ultiplier-**ac**cumulator (MAC) units, which are essential for **F**inite **I**mpulse **R**esponse (FIR) filters. Therefore, DSPs usually have an advantage in classic signal processing tasks. To utilize the full potential of DSPs, it requires assembler level programming. This makes the development and maintenance of the source code difficult. For non-signal processing algorithms, which usually consume the majority of the code space, DSPs show a lower performance than high-end GPPs because of their lower clock frequency.

A precise performance comparison between GPPs and DSPs is difficult. Pure performance metrics such as **m**illions of **i**nstructions **p**er **s**econd (MIPS) are widely used to describe the processing performance of a device. But in the real world this is often misleading because on different processing cores algorithms require a different amount of instructions for their calculations. Therefore, an algorithm kernel benchmark and application profiling [25] approach will be used to provide a good estimation precision of the expectable processing performance.

The speed of sorting algorithms is crucial for the median calculation performance and - as the analysis of the ROAn software has shown - this is a significant portion of the entire analysis time. Texas Instruments has implemented and evaluated various sorting algorithms on their c64x+ DSPs [43]. For the benchmark a fixed-point TCI6488 DSP with 1.0 GHz clock frequency and the ability to execute up to 8 instructions in parallel was used. The merge sort algorithm showed the best performance on this DSP platform. The highly optimized implementation of Texas Instruments needs approximately 50 **c**ycles **p**er **e**lement (CPE). For the performance of sorting algorithms on the x86 architecture the results from an Intel technical report [71, 72] are used as a base line. This report indicates 52 CPE on an Intel Core i7 CPU.

For the sorting task no performance improvement can be expected from a DSP processing solution over a high performance GPP solution due to the lower clock frequencies of the DSPs. Real-time processing of the full data stream from a next generation X-ray detector system is therefore not possible with DSPs. The achievable I/O bandwidth reduction by the direct data stream processing is necessary, but not sufficient for the acceleration required. An advantage of the DSP approach is a lower power consumption. Especially in satellite on-board processing this is a critical factor. For laboratory applications the higher system costs and the complexity is disproportionate to the power benefit.

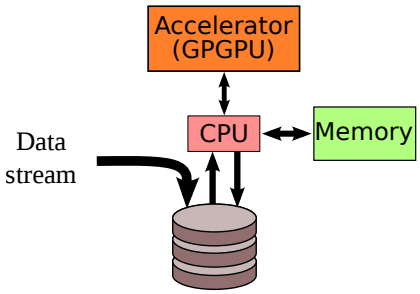


Figure 4.7: Implemented architecture for the evaluation of GPGPU-accelerated data processing.

4.1.3 General-Purpose Graphics Processing Units

General-**P**urpose **G**raphics **P**rocessing **U**nits (GPGPUs) are a new option to provide computational power at a better *Operation/Cost* ratio than **G**eneral-**P**urpose **P**rocessors (GPPs). Low costs and the possibility to use a small add-on card to boost the processing performance is very attractive for customers. Therefore, GPGPUs have become increasingly popular for technical and scientific data processing over the last years. However, not every algorithm can efficiently be processed on GPGPUs with their massive parallel data processing capability but slow sequential data processing structure.

The integration of a GPGPU into a data processing system as a coprocessor to offload computational intensive algorithm parts from the GPP cores is, at the moment, still a complex task. In comparison to conventional GPPs, not only a different programming language is necessary for the programming of GPGPUs, but also the required programming style is different. The different programming concept on GPGPUs is required because of their ability to execute tens of thousands of concurrently working threads, but the single-thread computation performance is low. Furthermore, the amount of local memory per thread is very limited. In addition, a vendor-specific programming language has to be used to obtain the best performance on a GPGPU. This makes it difficult to switch between GPGPUs from different hardware vendors.

For NVIDIA GPGPUs the programming language CUDA is recommended, while for GPGPUs from ATI the STREAM programming language provides the highest performance. The Khronos Group [33] tries to establish a common programming language for GPGPUs, which is called OpenCL. OpenCL should create a de facto standard, as it is the C/C++ language for GPP programming.

To speed up a program or an algorithm part by using a GPGPU, usually the code has to be completely rewritten because of the new programming language and the different programming style. Estimating the achievable speed up for an algorithm part, which is ported from the CPU to the GPGPU, is in general a difficult task and usually not very precise. At this time, one of the major bottlenecks for offloading calculations to the GPGPU is the available PCI-Express bandwidth between the CPU and the GPGPU as well as the high communication latency. Today, the peak data transfer rate between CPU and GPGPU memory is 8 GB/s, while the memory bandwidth on CPUs is more than 32 GB/s and on GPGPUs about 144 GB/s. For short algorithms it is inefficient to copy the required program code and data to the GPGPU and start the calculation there and copy back the results after finishing the task. For algorithms, which require large amounts of input data and produce a lot of output data, the PCI-Express bandwidth limitation is a critical factor. However, not only the bandwidth limitation can significantly reduce the performance. Code section with branches [34], as often used in control parts of the algorithm, cannot be efficiently processed on GPGPUs and should be avoided. The reason for this is that a GPGPU serializes the execution in cases where tasks are choosing different branches. The built-in hardware parallelism can then no longer be used efficiently. The event extraction and the pattern search algorithm in ROAn are classic examples for such algorithm

System configuration	Speed-up factor
Original single-threaded CPU code	1
Tesla C1060 OpenCL	0.87
Quadro 4000 OpenCL	1.97
Quadro 4000 CUDA without simultaneous computation on CPU and GPGPU	2.15
Quadro 4000 CUDA with simultaneous computation on CPU and GPGPU	2.67

Table 4.2: Measured ROAn analysis speed-up factors for various system configurations on CPU and GPGPU. The offset subtraction, common-mode calculation and correction, and parts of the histogram calculation were accelerated by the GPGPU for these measurements. For more details see [58].

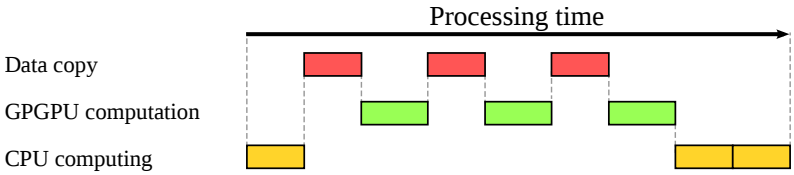
parts. The measurements have shown that these steps were processed faster on a CPU than on a GPGPU. The speed up created by GPGPUs, is in the real world, often significantly smaller than the vendors are claiming. The claimed huge improvements are typically achievable only for small fractions of the program and perfect for the GPGPU-fitted algorithms. Independent investigations as in Reference [53] show that the performance enhancement is closer to 2.5 than the often reported orders of magnitude.

For the evaluation of the achievable processing acceleration, the ROAn software was extended by analysis steps which offload proper processing parts to a GPGPU. For the implementation, a storage-based architecture as shown in Figure 4.7 was used. This allows comparing the single-threaded, multi-threaded and GPGPU data processing performance on the identical input data. If the results promise a sufficient speed up, this code can be reused in a direct data stream processing architecture, where the storage I/O limitation can be reduced.

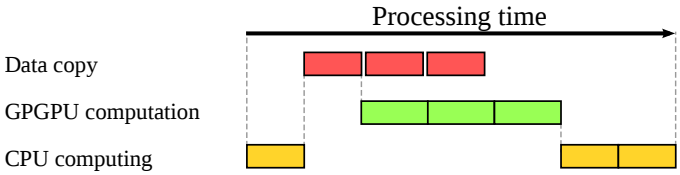
The offset and common-mode correction, the event extraction and the pattern clustering step were implemented in CUDA as well as in OpenCL for performance measurements. These processing steps were, furthermore, extended and calculating additional information such as the pixel histogram needed for the gain calibration. This enables to save processing time also in other processing steps. The test system was equipped with an Intel Core i7 870 processor clocked at 2.93 GHz, 4 physical CPUs, and 8 GB of RAM. For the acceleration measurements a Quadro 4000 and a Tesla C1060 GPGPU from NVIDIA were used. The offset subtraction, common-mode calculation and correction, as well as parts of the histogram calculation could be accelerated on the GPGPU. Runtime measurements of the implemented event extraction and pattern clustering showed that these parts were processed slower on GPGPU than on CPU and are, therefore, further processed on CPU. The GPGPU performance was mainly limited by the branching in these code parts. First, the performance study will be concentrated on the analysis task and does not include the data conversion step.

Table 4.2 shows the best measured speed-up factors for various GPGPUs, programming languages, and implementations. All GPGPU implementations use the copy compute overlap technique. This allows for a better utilization of the available PCI-Express bandwidth by overlapping data transfer between CPU and GPGPU and data processing on the GPGPU device. Figure 4.8 depicts such a copy compute overlap sequence. For the CUDA implementation, furthermore, the possibility of the asynchronous API to compute simultaneously on CPU and GPGPU was used. This allows for a better utilization of both processing systems and reduces the idle run time during the ROAn analysis.

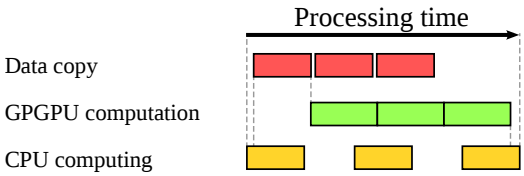
The discovered performance difference between CUDA and OpenCL for the accelerated steps was approximately 40 %. This is consistent with the comparison results from Reference [44]. The best overall ROAn analysis speed-up factor with the GPGPU acceleration was 2.67 and was achieved with the CUDA implementation on the Quadro 4000. This acceleration is much lower than expected for the X-ray data analysis and does not enable real-time analysis performance



(a) Sequence without copy compute overlap.



(b) Sequence with copy compute overlap.



(c) Sequence with copy compute overlap and simultaneous computation on CPU and GPGPU.

Figure 4.8: Overlap techniques on GPGPUs.

of the complete raw data stream with an optimized version of the ROAn software. The main reason for this small performance improvement by GPGPUs was the unfavorable ratio between the amount of data needed to be transferred and the computational complexity. The ratio between the computational complexity and the necessary amount of data transfer cannot further be increased for the X-ray data analysis, because other processing steps are not suitable for GPGPU acceleration and would reduce the overall performance. Furthermore, the GPGPU data processing acceleration allow only dedicated computer systems to benefit from the speed increase because of their special hardware and driver software requirements. In comparison to that multi-threaded data processing can increase the performance on almost every modern computer system. A further drawback of GPGPUs as processing accelerators for the X-ray data processing is the limitation to laboratory applications. The power dissipation of state-of-the-art GPGPUs under full processing load is a few hundred watts and there are no chips available in a radiation hard technology. Both reasons make it impossible to use such devices on satellites.

4.1.4 Summary of the investigation results

The investigation of the different processing platforms has shown that an acceleration of the ROAn analysis software is possible. The achievable factor, however, is small and not sufficient for real-time processing of the complete detector data stream.

The main limitation identified in the studies was the large amount of raw data rates produced by the detector systems. This becomes even more challenging with the next generation of detector systems, where the output data rates will increase significantly by a factor of more than 40. This increase is beyond the expectable improvements of the processing platforms in the near future. Closing the gap between the software processing performance and the detector output data rates requires a completely new concept for the DAQ and data processing system.

Implementation	Pro.	Con.
ROAn software single-thread (standard)	High processing flexibility Easy step adding	Low processing speed Worse scalability No real-time processing capability
ROAn software multi-thread	High processing flexibility Easy step adding	Higher code complexity High I/O load Speed up < 5 Speed up not sufficient for real-time processing
DSP	Usable in satellites	Low clock frequency Low expected speed up Assembler programming
GPGPU	Cost-efficient processing platform	Not usable in satellites Speed up < 3 Limited by CPU to GPGPU bandwidth Special programming language Special HW for data analysis

Table 4.3: Summary of the investigation results.

Chapter 5

Super-Modular-DAQ system design

This chapter describes the newly developed architecture, dynamic algorithms, and the hardware for the **Su**per-**Mo**dular-DAQ (SuMo-DAQ) system to overcome the discovered limitations of the PCI-based DAQ system.

5.1 The system architecture of the Super-Modular-DAQ system

Requirements on the SuMo-DAQ

Flexibility is a key feature for the new SuMo-DAQ architecture and data processing concept. It should be able to handle the data stream from a wide variety of detector systems with a variable number of concurrently operated analog channels and different readout configurations. Simple and fast adaptability of the SuMo-DAQ to a new detector system is crucial for laboratory applications. For a long system life time, the scalability of the architecture is essential. With this, the operation of large and high-speed next-generation detector systems can be ensured. Modularity of the data processing concept is necessary to allow for adding and adapting the system to specific processing requirements for an application. To improve the turn around time during the detector optimization process the DAQ system has to provide feedback of key detector parameters in real time to the operator. Furthermore, the concept should be able to extend the real-time processing with application-specific algorithms. The use of the next-generation detector systems requires a significant degree of data reduction during the processing to enable an efficient use of storage space. Data reduction also helps to reduce the required processing power to handle and analyze the information. The need of the autonomous detector system operation requires self-adaptation functionality from the data processing system. This enables the system operation under changing environment conditions and inaccessible systems such as satellites, where a manual adjustment of the system is not directly possible. The self-adaptation is also required to improve the measurement precision if disturbing objects are in the field of view as well as in long-term measurements to reduce the necessity of detector recalibration cycles.

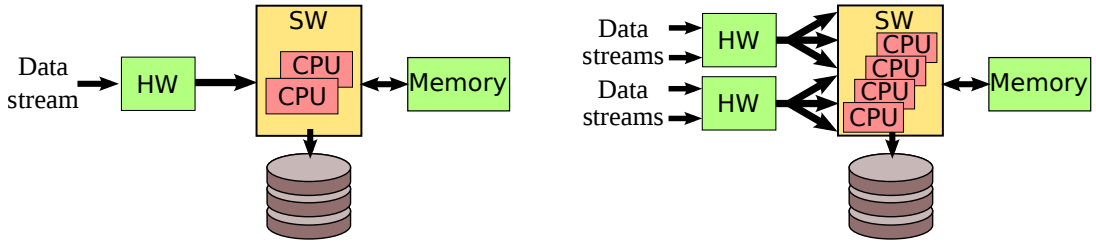
The requirements on the SuMo-DAQ system are summarized in Table 5.1.

5.1.1 SuMo-DAQ architecture and processing task distribution

The SuMo-DAQ system concept is based on a hardware pre-processing and software post-processing architecture as depicted in Figure 5.1. This processing architecture allows for an optimal distribution of the various X-ray data processing tasks between hardware and software. Hardware and software co-design increases the system complexity, but allows using the

Requirement	Reason
Flexibility and scalability	Necessary for the handling of a wide variety of detector systems with a variable number of concurrently operated analog channels and different readout configurations
Modular and adaptable	For a fast adaption to application and detector system-specific processing tasks
Autonomous operation	To enable the operation of stand-alone and remote detector systems as in satellites
Data reduction	For large and high-speed detector systems, a significant degree of data reduction is crucial to allow for an efficient data set storage
Self-adaptation	Feature required for measurements in changing environmental conditions, especially for autonomous system operation and long-term measurements with a higher precision.
Real-time processing	For a significant degree of data reduction to enable the operation of large detector systems and to provide live feedback of the detector condition to the operator.

Table 5.1: Summary of the requirements on the SuMo-DAQ architecture and data processing concept.



(a) Configuration for small detector systems with a single input data stream.

(b) Configuration for larger detector systems with multiple input data streams.

Figure 5.1: SuMo-DAQ real-time processing architecture with hardware data stream pre-processing and software post-processing.

advantages from both fields. Highly optimized hardware structures enable high-speed data processing and high data throughputs, while very complex algorithm parts can be implemented more efficiently in software.

The majority of the processing time during the ROAn-based software data analysis is used in parts where the full frame data stream has to be handled. This was shown in Table 4.1 during the analysis of the ROAn software. The main limitation discovered during the evaluation of various processing platforms was the low computational complexity to data transfer ratio. To avoid this identified bottleneck in the new SuMo-DAQ system, one solution could be the realization of the full frame data stream processing in hardware up to the point where a significant degree of data reduction is achieved. The hardware pre-processing minimizes the amount of required processing power in the software part and enables real-time processing performance. Extending the hardware processing part to a complete data stream analysis system, however, is not a reasonable solution, because not all processing steps can be implemented efficiently in hardware. Some pattern filters, for instance, require a high degree of algorithmic complexity to generate special histograms.

The task distribution for real-time data processing is, therefore, mainly restricted by the necessity to significantly reduce the data volume by the pre-processing before the data stream is handed over to the software part. The necessary X-ray data analysis steps where investigated during the ROAn software exploration (Chapter 3) and their processing sequence is shown in

Figure 3.2. The highest degree of data reduction is provided by the event extraction step. The achievable data reduction ratio is analyzed and discussed in Appendix A.2. This analysis shows only a negligible advantage in the data rate of the clustered event data stream compared to the event data stream for the typical system operation conditions. The key processing step is, therefore, the event extraction, and all tasks up to this processing step in the chain have to be implemented in hardware. Realizing less processing steps in hardware leads to a drastically increased data volume and would limit the real-time capability of the system. Adding further processing steps only makes sense for dedicated DAQ systems and special applications such as on-board satellite processing, where power dissipation is a critical factor.

5.1.2 Hardware processing devices

For the implementation of the hardware processing part reconfigurable **f**ield-**p**rogrammable **g**ate **a**rray (FPGA) and hard-wired **a**pplication-**s**pecific **i**ntegrated **c**ircuit (ASIC) devices can be used. These devices allow for the precise implementation of the required logic functionality to solve a specific processing task. This enables higher data processing throughput at lower clock frequencies as in conventional CPUs. Furthermore, this makes a reduction of the power consumption for the processing system possible.

The development with FPGAs is cheaper and provides a higher flexibility due to their faster turnaround time compared to ASICs because of the device programmability. The drawback of FPGAs is a higher power dissipation and a lower processing performance compared to the realization of the same functionality in an ASIC. To enable the operation of a wide range of detector types and readout configurations, the flexibility and fast adaptability of the DAQ and data processing system are key features. Therefore, the programmable FPGA processing solution is the preferred choice for the SuMo-DAQ development. The rapid modification of the FPGA systems not only allows for faster adaptation of specific detector systems, but it also enables to realize and test system optimizations directly on hardware. Furthermore, an ASIC can be developed fast based on the design of the FPGA processing system. This would allow for the reduction of the power required for a satellite mission.

5.1.3 SuMo-DAQ data processing structure

This subsection describes the SuMo-DAQ processing structure for a detector system with a single analog channel. A stream-based processing structure is used for the data handling. The data stream from the detector system is a continuous, rapid, and an infinite sequence of ADC data elements. As shown in Figure 5.2 this data stream is directly fed into the FPGA on the **d**ata pre-**p**rocessing **b**oard (DPB). On this board, the entire hardware pre-processing is realized for the SuMo-DAQ. The serialized input data stream from the detector system is processed pixel by pixel in different hardware units. Afterwards, the result is handed over to the following unit. This structure enables continuous processing of the serial detector data stream with a minimum of required data storage capacity in hardware.

This approach even enables processing independent from the detector format. The last pixel of a detector row and the last pixel of a detector frame is indicated by a flag encoded to the data stream. Compared to a solution, where the frame is first completely stored and then fully processed, the maximum frame size is not limited by the available amount of hardware storage capacity. This makes it possible to use the same processing chain without any change for different detector systems.

The processing chain structure allows for adapting the system to specific requirements by simply exchanging processing modules. This creates a modular and highly flexible hardware processing system for the SuMo-DAQ. A single hardware unit solves only a small and specific processing task. This enables faster development, testability, improves the readability of the source code and allows for the efficient reuse of **i**ntellectual **p**roperty (IP) hardware blocks.

The software processing part is realized on the **d**ata **h**andling **c**omputer (DHC). This is a standard computer system in laboratory applications, while for satellite onboard processing

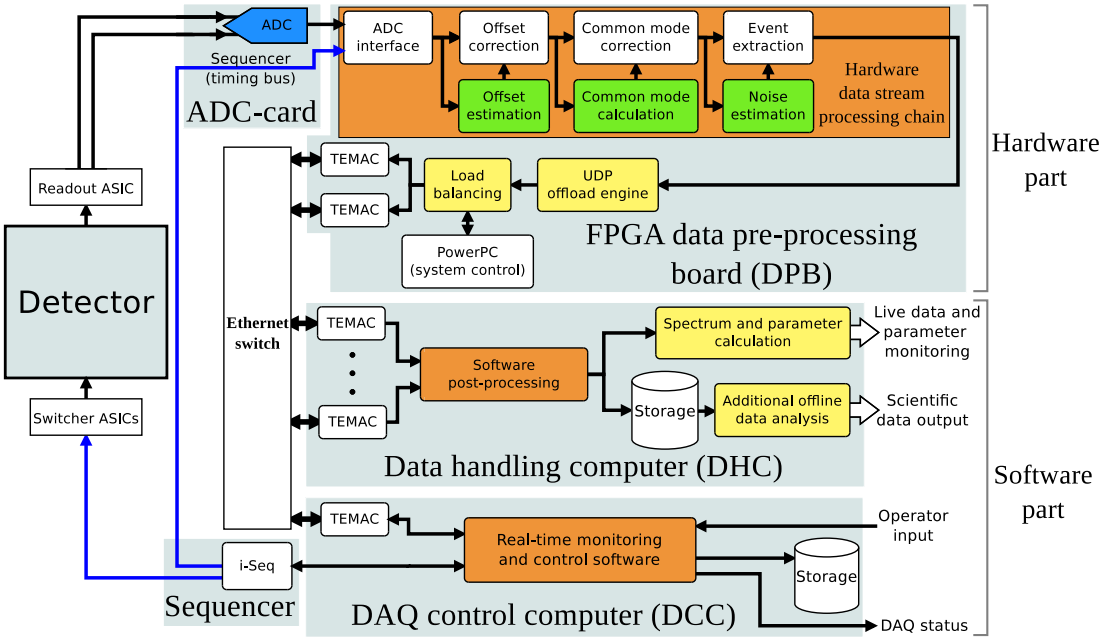


Figure 5.2: SuMo-DAQ data processing structure for a detector system with a single ADC channel. The differential analog signal from the readout ASIC is digitized on the ADC card. The data and timing bus signals are provided to the DPB board, where the hardware pre-processing is made. The results from this processing are forwarded via an Ethernet connection to the DHC for the software post-processing and data storage. The DCC system controls all sub-systems and provides the user interface for the SuMo-DAQ system.

the DHC is replaced by an embedded CPU or DSP solution. Depending on the performance requirements, the PowerPC CPU integrated in the FPGA can also be utilized for this task. The software on the DHC handles the data transmitted from the DPB to the software post-processing system. The processing tasks vary with the type of the input data stream. The first task for all streams is a check of the encoded frame ID number to detect frame losses. The event packets are then distributed over several worker threads on the DHC for the software post-processing. Along the way of the processing, the event statistic, hitmap, and the raw data spectrum are extracted from the data stream for live monitoring. Furthermore, the events are clustered and stored for an additional and more detailed offline analysis.

The SuMo-DAQ system control is realized by the **D**AQ **c**ontrol **c**omputer (DCC). The two main tasks of the DCC are providing the system control GUI and real-time monitoring GUI. The system control GUI is the operator interface to communicate with the DPB control software and steer the data acquisition. The real-time monitoring GUI enables to observe the detector system status and the SuMo-DAQ data processing system. The real-time monitoring data is recorded by the DCC to enable the documentation and understanding of unusual behavior. In laboratory environments, the DHC and the DCC functionality is often combined on a single computer system for mid-size detector systems. This was also the case during the long-term irradiation campaign for MIXS. The structure was chosen to enable the scalability of the SuMo-DAQ system beyond the capability of a single DHC. For large-scale detector systems additional DHC systems are added and the data streams from the DPBs are distributed over these systems. In such a configuration, each sub-system handles the data streams only for a specific detector part. This enables to control and limit the input data rate for the processing systems also for large and high-speed detector systems to avoid overload situations.

5.1.4 Hardware processing module interconnection

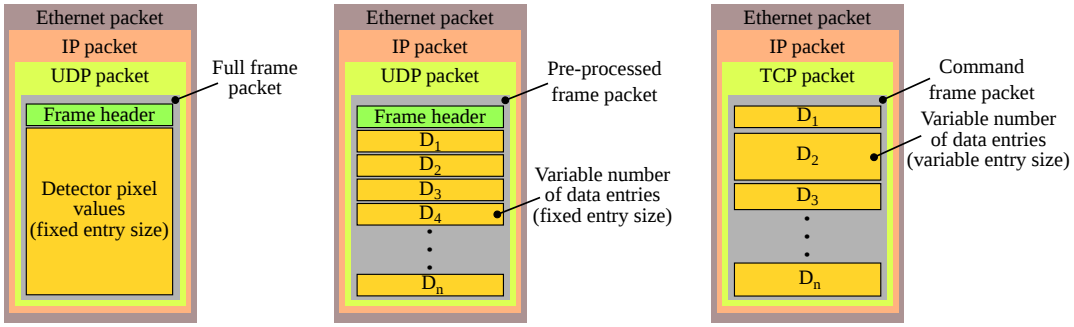
For the changeability of different hardware modules in the processing chain, all modules use the same **i**nput/**o**utput (I/O) interface standard. This enables adding, removing, replacing and rearranging the processing modules independently from the individual functionality. The base components in the processing chain can, therefore, be used for every system without any change. As I/O standard a slightly modified version of the **A**dvanced e**X**tensible **I**nterface (AXI) [102, 6, 7] stream protocol is used. The AXI-Stream protocol has been specially designed for high-speed streaming applications and has a minimal hardware footprint. The customized protocol version named SuMo-Stream is presented in detail in Appendix A.1. For the flexibility of the SuMo-DAQ system this is a key feature to enable a fast adaptation of the processing chain to various applications or detector-specific requirements.

5.1.5 SuMo-DAQ communication

For the communication between the SuMo-DAQ FPGA hardware pre-processing board and the software post-processing part an Ethernet [42] network is used. For the high-speed data link from the FPGA to the post-processing system the **U**ser **D**atagram **P**rotocol (UDP) [68] is used, while for the DAQ control communication the **T**ransmission **C**ontrol **P**rotocol (TCP) [2, 13, 69] is used.

The UDP is a simple connectionless protocol and the lack of retransmission delays makes it suitable for real-time applications. Furthermore, UDP does not require to store already transmitted packets until the complete reception of the packet is confirmed by the receiver. The simplicity of UDP and the low storage requirements allow for the full implementation of the protocol in hardware. This enables the handling of the data stream without any software interaction on the FPGA. The drawback of the UDP is the possible loss of data packets. In dedicated network areas, however, the real packet loss is almost zero. The pre-processed data for a single frame are merged into a single UDP packet. Even if a packet is lost, the information loss is extremely small compared to the overall information. Merging all the information of a frame to a single packet enables fast frame checks and by choosing network components, which support the hardware verification of packets, the software can offload this task. Each of these packets contains a section at the beginning, where the current frame number and the optional information are encoded.

The TCP, however, is a more complex and connection-oriented protocol. TCP requires handshakes to establish an end-to-end communication channel between two devices. The user is able to send data in both directions over the TCP communication channel and the protocol ensures that the complete message arrives at the receiver. This generates a lot of overhead and limits the TCP use for the necessary DAQ control communication, which is realized in software.



(a) Full frame packet structure with a single data entry of a fixed size.

(b) Pre-processed packet structure with a variable number of fixed size data entries.

(c) Command frame structure with a variable number of data entries and a variable entry size.

Figure 5.3: Structures of the different data packets used in the SuMo-DAQ.

Replacing the point-to-point communication of the PCI system by an Ethernet-based communication approach enables to control and operate multiple SuMo-DAQ systems with the same software post-processing system. With Ethernet communication, the scaling of the SuMo-DAQ concept to more FPGA boards and more analog detector channels is possible. Depending on the application requirements, the different data streams from FPGA boards can be aggregated to a single post-processing system or distributed over multiple systems by simply changing the communication settings of the FPGA boards. Distributed laboratory systems, where a single software system is utilized to process and store the data from different detector systems can additionally reduce the necessary hardware effort and cost. Each FPGA board transmits multiple types of data streams to the software processing part. TCP and UDP port numbers are used to distinguish the different data streams and assign the correct detector area, where the data are collected by the FPGA board. The precise UDP and TCP packet structures of the data streams used for the SuMo-DAQ are described in [68] (UDP) and [2, 13, 69] (TCP). The SuMo-DAQ concept has three base types of data streams: full frames, pre-processed events and command and status frames.

Full frames

Full frame data streams are used to transmit complete frames between the hardware pre-processing system to the software post-processing system. The format shown in Figure 5.3a is similar to the data format of the PCI-based DAQ system. This format is utilized if the full and unprocessed detector data stream has to be transmitted for debugging or test purposes. Furthermore, it is used to record the different maps calculated in hardware at a certain time and to update the operator live monitoring status. The values of all pixels are sequentially encoded into such a packet. This data stream type has, therefore, a fixed and constant packet size, which depends on the size of the transmitted detector area. The size limitation of fragmented IPv4 [41] packets of $65,535$ bytes on Ethernet determines the maximum detector area, which can be transmitted by a single packet. The maximum payload size of an UDP packet transmitted by means of the IPv4 protocol is $65,507$ bytes. Detector areas larger than $32,753$ pixels have to be split up into several packets.

Pre-processed events

The pre-processed event data stream (Appendix A.6) is custom-formatted by the hardware pre-processing system. All events extracted from a single detector frame are sequentially encoded into a single packet in a fixed format. The packet size is variable because of the fluctuating number of events per frame. The base structure of such a packet is depicted in Figure 5.3b. Each data entry D_i represents an extracted event and has a fixed data size in this structure. This reduces the necessary processing power to extract the information on the software post-processing system. Due to the fixed data entry size the number of entries and the position of various information can be calculated directly. This enables fast information extractions from the data stream.

The frame header contains the frame number of the frame the events where extracted from. This number is essential for further processing and enables synchronization of the data stream from multiple DPB systems. The frame number is calculated on each DPB from the sequencer timing-bus signals. For each of the extracted events the X/Y position, the signal value, and the primary/secondary flag is encoded in a data entry. The primary/secondary flag indicates if the event was above the secondary or the primary event extraction threshold. In Appendix A.6 the encoding of an event in a data entry is shown in detail.

Command and status frames

Command and status streams are low data rate streams, which are only used for control and feedback communication between the different SuMo-DAQ processing and control components.

High flexibility is, therefore, more important than performance. The packet structure is depicted in Figure 5.3c. The command type encoded in this data entry is identified by a key word at the beginning of each data entry and followed by an optional command/status data section with a variable size. This enables fast and flexible adding of new commands and allows for directly including necessary parameters such as filter values into the command frame. For the feedback from the DPB systems to the DHC the same structure is used and allows encoding a variable number of system parameters such as temperatures, voltages, and processing statistic values. The available command and status key words are described in Appendix A.6.

5.1.6 Data storage

In addition to the filtered events for each detector frame in the pre-processed data stream the SuMo-DAQ delivers also the up-to-date versions of the calculated offset and noise maps in certain time intervals. For documentation purposes, these full frame data streams should also be recorded during a measurement. During debugging and special detector development tasks the hardware processing chain can be skipped and the full detector data stream is transmitted to the DHC. This operation mode is similar to the PCI-based DAQ system. Furthermore, the processed and the raw data stream can be recorded at the same time. This is necessary for a direct performance comparison between the SuMo-DAQ system with the new processing algorithms and the data analysis flow of the old PCI-based system with the ROAn analysis software in evaluation measurements. Two different storage strategies are used for the full frame and the pre-processed data stream: full frame data stream storage and pre-processed event data stream storage.

Full frame data stream storage

The full frame data stream is stored directly in the **fram**es (FRM) file format by the SuMo-DAQ software. The FRM file structure is the standard data file format in the ROAn software for storing frames and is described in [51]. This enables the use of the ROAn viewer and file interface to read and display the files.

The dynamically adapted offset and noise maps from the SuMo-DAQ system can be stored with a selectable time interval between one and ten seconds. The total number of such maps is very small compared to the total number of detector frames. The incoming amount of data can be handled also for long-term measurements without problems. The recording of these maps is optional and can be used to verify the stability of the algorithms.

The storage of the full detector data stream similar to the old PCI-based DAQ system, however, requires a powerful and large storage system. This optional SuMo-DAQ operation mode is used only for short reference or comparison measurements and not for long-term measurements. With the full frame data stream format generated by the DPB the data consistency check can be realized on the fly in the DHC software. In contrast to the PCI-based DAQ system, where an unchecked RAW file is recorded, the data conversion step is not necessary during the ROAn data analysis. This saves approximately 33 % of the analysis time as shown in Table 4.1 for the comparison measurements.

Pre-processed event data stream storage

For storing the pre-processed event data stream, a **d**ata**b**ase (DB) is used instead of a conventional data file. The use of a database has significant advantages especially for multi-threaded and distributed high-performance systems. The utilized MySQL [88, 62] database is a **r**elational **d**ata**b**ase **m**anagement **s**ystems (RDBMS) and is highly optimized to handle an arbitrary number of concurrent data accesses with a good scalability. For large systems, where concurrently multiple data streams have to be stored and synchronized, the database reduces the system complexity and enables the SuMo-DAQ scaling without hardware changes. The database also enables to execute analysis tasks on the data set during an active measurement. The protection

Data field name	MySQL Type	Size	Description
FrameID	BigInt	8 Bytes	Sequential frame identification number.
FrameTime	BigInt	8 Bytes	Frame recoding time (optional).
EventData	LongBlob	Variable (maximum 4 Gbytes)	Contains all encoded events for this frame.

Table 5.2: SuMo-DAQ database table structure for the event data stream storage.

of the data with mutexes is handled by the database itself and simplifies the DAQ software. It is possible to store data in an RDBMS system because of the significantly reduced data rate of the pre-processed event data stream. Storing full frames would be inefficient on a database system.

The information in a RDBMS database is stored in tables. A table represents a collection of related data entries called records, which are the table rows. In the SuMo-DAQ database, for each measurement a new table is created, where all frames from this measurement are stored. For each recorded frame a particular record is created in this table. A record consists of multiple data fields, which represents a column in this table, where a specific data content is stored. The data type used for a field depends on the content to be stored. Table 5.2 shows the database table structure used by the SuMo-DAQ system to store the event data stream. The FrameID field is a unique number in the table and identifies the detector frame, where the events in this record are extracted from. The FrameTime field contains the frame receiving time stamp of the DHC. All event information for this frame are encoded in the binary data field of the record. The encoding of the events in this binary data field is the same as in the event data stream (Appendix A.6). Using the same data encoding in the database and data stream avoids additional reformatting tasks. This enables direct data storing with the same data access advantages as in the data stream. Frames where no event was extracted are also stored in the table with an empty EventData field. Storing empty frames requires only a small amount of space, but enables to check the consistency of the table if all frames are stored. This is crucial for the data stream synchronization, if multiple DPB systems are operated concurrently.

The alternative storage approach with one record per event and data fields for X/Y position, signal Value, primary/secondary flag, frame time and frame number was too inefficient in terms of processing power and storage space.

5.2 Hardware used for the Super-Modular-DAQ system

The FPGA board shown in Figure 5.4 was used as development base for the SuMo-DAQ hardware pre-processing system. The data processing board is equipped with a Virtex-5-LX100T (XC5VFX100T) [91] FPGA from XILINX. This FPGA contains 16,000 CLB slices, 64,000 LUTs, 8,208 Kb of Block RAM, two hardware TEMACs and two PowerPC processor blocks. The two available PowerPC CPU cores on the FPGA are used to realize the software control part on the DPB board without using additional logic resources for an alternative soft-core processor. The board is additionally equipped with 256 MB of DDR, 256 MB of DDR2 RAM, a Compact-PCI interface and two **s**mall **f**orm-factor **p**luggable (SFP) slots. These SFP slots are utilized for the network connection of the DPB board. The slots can be equipped with Gigabit Ethernet modules for copper or fiber-optic connections. The fiber-optic modules are the standard option and enable galvanic decoupling of the SuMo-DAQ DPB board from other system components to avoid disturbances. The Compact-PCI interface is only used to supply the power to the board and enable a compact arrangement of multiple FPGA boards in a 19-inch rack-mount system.

A standardized **C**ommon **M**ezzanine **C**ard (CMC) interface on the FPGA board is used

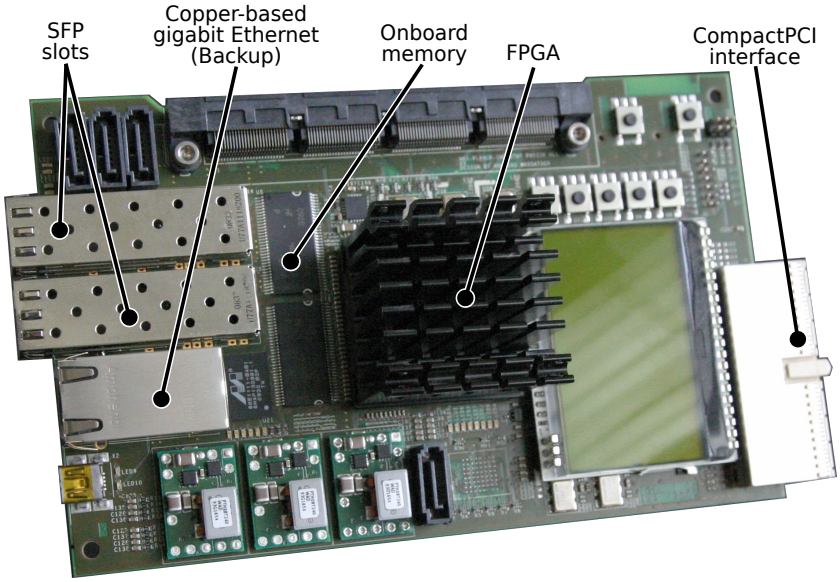


Figure 5.4: FPGA **d**ata pre-**p**rocessing **b**oard (DPB) used in the SuMo-DAQ system. The main system components on the front side of the DPB are labeled in the figure. The CMC interface for the ADC card is on the backside of the board and not visible in this figure.

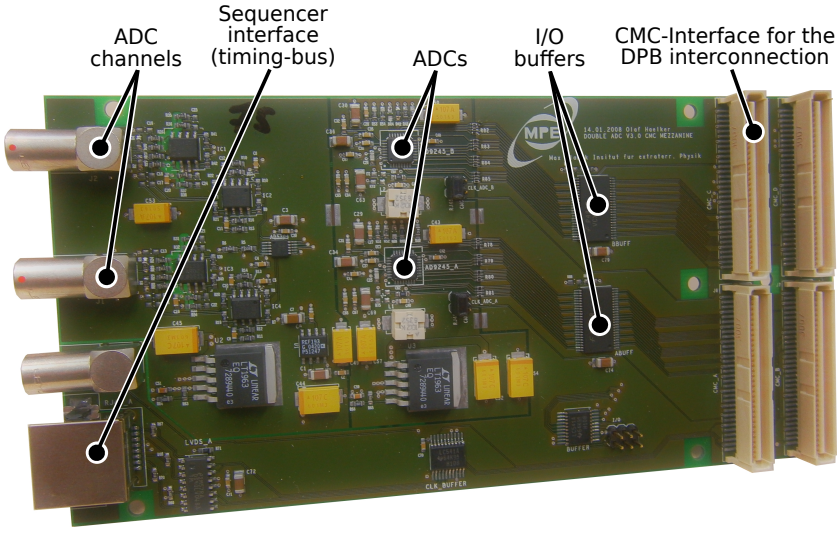


Figure 5.5: Two channel ADC extension card with a CMC interface for the connection to the FPGA **d**ata pre-**p**rocessing **b**oard (DPB). The card is powered and controlled via the CMC interface by the DPB board.

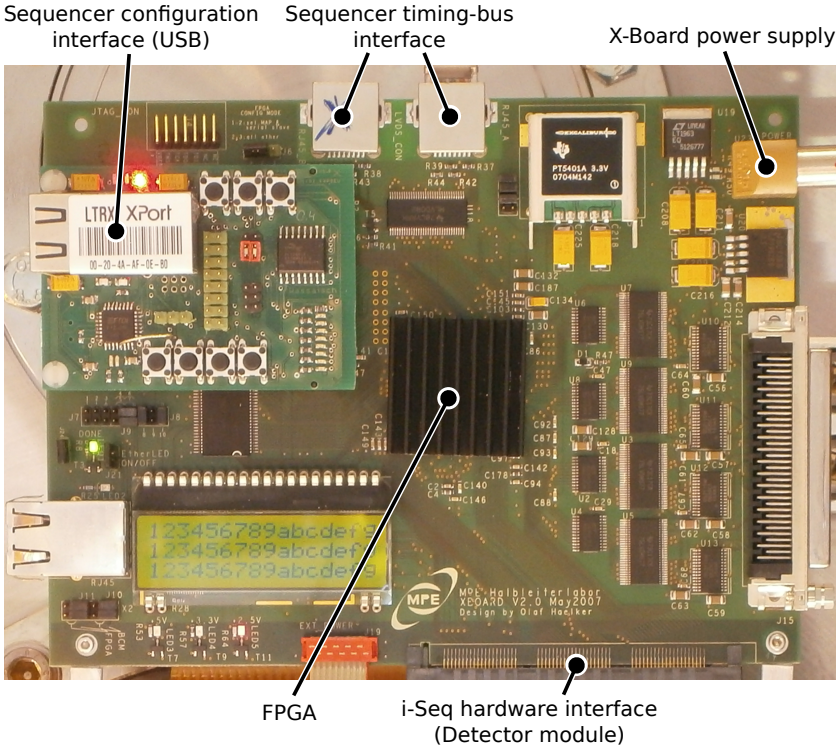


Figure 5.6: Sequencer FPGA board (X-Board), where the control signals for the detector system and the timing-bus signals for the SuMo-DAQ system are generated synchronously.

to connect a daughter card with high-speed ADCs. The CMC interface is on the backside of the FPGA processing board and is not visible in Figure 5.4. Figure 5.5 depicts the two channel ADC extension card used. This card is equipped with two 14-bit ADCs, which are operated at 80 MHz, the necessary differential signal amplifiers, and the reference voltage source. The ADC extension card is fully powered and controlled by the DPB via the CMC interface. These versions of the ADC extension card as well as the DPB board are used for the complete development, test, and evaluation process of the SuMo-DAQ system in this thesis.

The i-Seq sequencer in the first SuMo-DAQ hardware version is implemented on the X-Board, which is depicted in Figure 5.6. The X-Board is a FPGA board particularly designed for the sequence generation task and is equipped with a special interface connector to the detector system. For the distribution of the sequencer timing bus signals to the DPB board, a high-quality Cat 6a Ethernet cable is used. The sequencer timing bus carries the ADC-clock, detector line termination, detector frame termination and the ADC gate as differential signals. These signals are used by the ADC interface hardware unit on the FPGA to generate the SuMo-Stream for the data processing chain.

The DDR and DDR2 memory on board the DPB board is used in the SuMo-DAQ system for storing the program and data of the control software running on the PowerPC CPUs. In addition, the offset and noise maps continuously updated by the hardware processing chain are stored in the on-board memory. Therefore, a sufficiently large memory bandwidth is necessary to enable the maximum SuMo-DAQ processing chain throughput. The DDR memory on the DPB board provides a theoretical bandwidth of $4\,\text{Bytes} * 125\,\text{MHz} * 2 = 1,000$ MB/s. The DDR2 memory delivers an additional 1,000 MB/s bandwidth on the DPB. For both memory systems, the **M**ulti-**P**ort **M**emory **C**ontroller (MPMC) [98] provided from Xilinx is used as universal and high-performance memory interface. The achievable data throughput in a real system environment with the MPMC memory interface is shown in Table 5.3. The **N**ative **P**ort **I**nterface (NPI) port performance represents the true memory bandwidth, which a MPMC core

Port type	Memory interface	Maximum data throughput (MBytes/sec)	
		Read	Write
NPI	DDR with 125 MHz / 32 bit	800	666
PLB	DDR with 125 MHz / 32 bit	635 (single port)	400 (single port)
NPI	DDR2 with 125 MHz / 32 bit	696	552
PLB	DDR2 with 125 MHz / 32 bit	348 (single port)	222 (single port)

Table 5.3: **M**ulti-**P**ort **M**emory **C**ontroller (MPMC) performance [98] for burst length of 16 double words on Virtex-5 devices scaled for the configuration on the DPB system. A single **P**rocessor **L**ocal **B**us (PLB) port cannot utilize the full MPMC bandwidth, which is represented by the **N**ative **P**ort **I**nterface (NPI) bandwidth. Multiple PLB ports on the MPMC have to be used to maximize the MPMC throughput.

can provide. In the SuMo-DAQ system, the **P**rocessor **L**ocal **B**us (PLB) is used, because in comparison to the NPI interface, where each port is dedicated to a single device, the PLB bus enables to interconnect multiple bus devices. In addition, the PLB bus can directly be used to control the different system components on the FPGA. To allow for the utilization of the complete available memory bandwidth on the DPB board, multiple PLB buses have to be connected to different ports on the MPMC and used concurrently. The various system components are distributed over the buses in such a way that the expected load on each bus is about the same. Although the use of multiple buses requires more logic resources, it helps to avoid problems with the achievable clock frequency on the FPGA if many system components have to be interconnected.

The second generation of SuMo-DAQ hardware is currently under development and uses a Spartan 6 FPGA from XILINX instead of the Virtex 5. Lower costs and power dissipation are significant advantages for building large systems. Due to the similar structure of the internal FPGA architecture the porting of the SuMo-DAQ system to the new hardware version will easily be possible. All comparisons and evaluation results presented in this thesis are based on the first SuMo-DAQ hardware version.

5.3 Real-time data processing and dynamic algorithms for X-ray data processing

Calculating the pixel-individual offset value from dark frames at the beginning of each data set has been the standard approach up to now and was described in Chapter 3.1.3. This approach is not usable for direct data stream processing and has, furthermore, functional drawbacks. The main drawback of this static offset calculation approach is that offset changes during a running measurement cannot be detected and corrected. In long-term measurements, recalibration phases must, therefore, be included after a certain time of measurement. The automatic scheduling of the recalibration phases is difficult, but necessary for autonomous system operation. As described, a number of dark frames needs to be recorded for the recalibration. To avoid interferences during the offset calculation, any photon source has to be removed from the field of view. This is cumbersome especially on board of a satellite, where a suitable detector shielding has to be provided in form of a movable baffle.

To enable the data stream-based processing for an autonomously operated system, new algorithms, which allow for the calculation of necessary parameters directly from the raw data stream, are required. For the SuMo-DAQ data processing algorithms it is essential to avoid data dependencies between pixels and to minimize the necessary pixel history information.

Data dependencies increase the complexity of the processing steps as well as the required storage space. The static offset and noise map calculation in ROAn, for example, requires in minimum 200 history frames to determine these values. For real-time processing on storage-limited FPGAs this approach is not usable. Filter-based methods as described in Appendix A.3 have a high hardware complexity and high memory bandwidth requirements.

The new SuMo-DAQ algorithms described in this subsection have been designed and optimized for minimal data dependencies, pixel-individual calculation, and low computational complexity. This is necessary to enable the integration of the entire data processing path for multiple raw detector data streams on the FPGA platform and allows for achieving real-time processing performance.

5.3.1 Dynamic offset estimation algorithm

The new **d**ynamic **o**ffset **e**stimation (DOE) algorithm described by Equation 5.1 was developed to realize a hardware implementation on FPGAs to overcome the drawbacks of the static algorithms. The DOE algorithm uses an iterative calculation approach and needs to store only a single history value, has low resource requirements and can be implemented efficiently in hardware.

$$O_i(n+1) = \begin{cases} O_i(n)+1 & for\ \ I_i(n) - O_i(n) > 0 \\ O_i(n) & for\ \ I_i(n) - O_i(n) = 0 \\ O_i(n)-1 & for\ \ I_i(n) - O_i(n) < 0 \end{cases} \tag{5.1}$$

The new offset value $O_i(n+1)$ for the detector pixel i in frame $n+1$ is calculated by an increment or decrement operation from the current offset value $O_i(n)$ for this pixel and the current pixel value $I_i(n)$. The difference between $O_i(n)$ and the current pixel value $I_i(n)$ in this frame n defines the direction of adjustment. This algorithm converges against the median value. For this value the **c**umulative **d**istribution **f**unction (CDF) of a random distribution is equal to $\frac{1}{2}$. The precise mathematical definition of the median value is given in Appendix A.4.2.

To increase the stability of the offset value calculated from a noisy input signal, a filter is added. Equation 5.2 is an extension of the base DOE Equation 5.1 with a built-in filter option. The filter parameter M in this equation allows for adapting the calculation of the intermediate offset value $\overline{O_i(n+1)}$ to the application-specific requirements. The absolute pixel offset can finally be calculated by Equation 5.3, where the filter has been removed from the intermediate offset value $\overline{O_i(n+1)}$.

$$\overline{O_i(n+1)} = \begin{cases} \overline{O_i(n)}+1 & for\ \ I_i(n) - \frac{\overline{O_i(n)}}{M} > 0 \\ \overline{O_i(n)} & for\ \ I_i(n) - \frac{\overline{O_i(n)}}{M} = 0 \\ \overline{O_i(n)}-1 & for\ \ I_i(n) - \frac{\overline{O_i(n)}}{M} < 0 \end{cases} \tag{5.2}$$

$$O_i(n+1) = \frac{\overline{O_i(n+1)}}{M} \tag{5.3}$$

Depending on the application-specific requirements, the filter option can be adapted for a faster settling or a more stable estimation value. Larger M values result in a higher stability of the calculated offset value, while smaller M values allow for faster reactions on changes. For a hardware implementation M should be selected as a power of two to minimize the hardware resource requirements. The **m**aximum **c**hange **r**ate (MCR) for the calculated offset value can be calculated by Equation 5.4 for a certain filter parameter M. Equation 5.5 allows for an

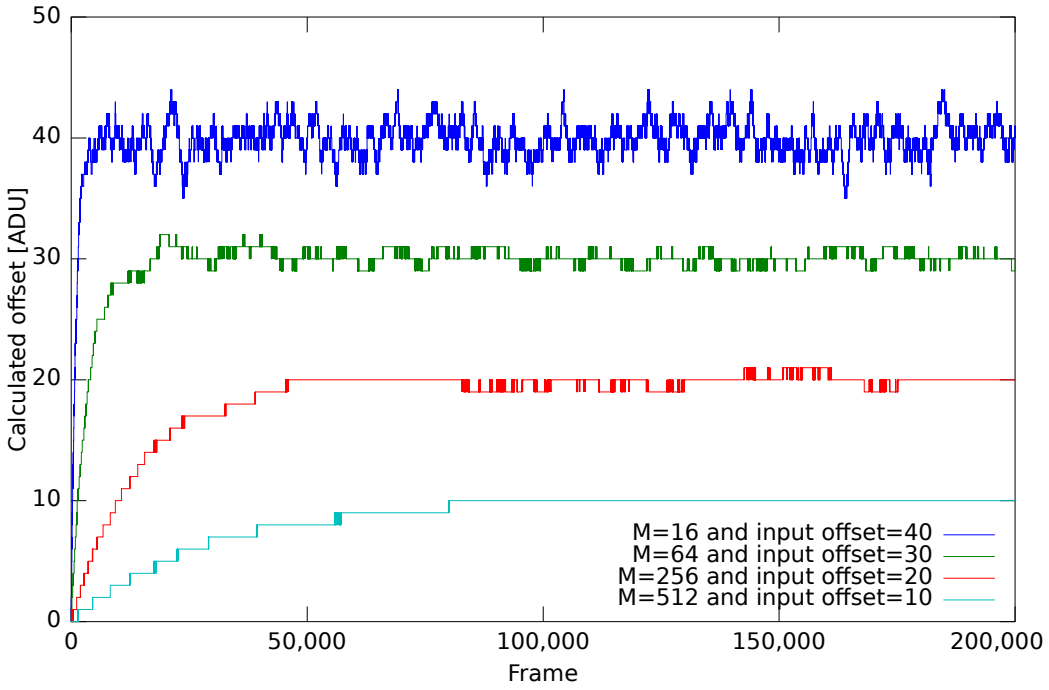


Figure 5.7: Offset time series calculated by the DOE algorithm with different filter parameters M and different input offset levels for a signal with 45 ADU noise sigma. Smaller M values enable faster change rates but lead to a lower stability of the calculated offset value.

estimation of the clock cycles necessary to compensate an offset step of O_{Step} ADU to an accuracy of better than 90 % with an input signal noise level of $Noise_{Sigma}$ ADU for a specific filter parameter M. The coefficients in Equation 5.5 and 5.6 are extracted from simulations of the algorithm. As the simulation results depicted in Figure 5.7 show, the mean value of the calculated offset O_{DOE_i} for pixel i converges for any filter parameter against O_{Real_i}, the real offset input value. The filter parameter M influences only the stability and standard deviation of the calculated offset time series. The filter option enables to reduce the impact on disturbances created by noise on the input signal. For $M = 256$, the stability of the calculated offset value can be approximated by Equation 5.6. The algorithm calculation precision is shown in a detailed comparison of the input signal and the calculated offset output in Figure 5.8 for an input offset $O_{Real} = 100$, 15 ADU noise sigma and a filter parameter of $M = 256$.

$$MCR_{Offset} = \frac{1}{M} \tag{5.4}$$

$$Cycles_{Stabiliz} \approx O_{Step} * M * 0.9 + M * 0.65 * Noise_{Sigma}^{1.25} \tag{5.5}$$

$$std\left(O_{DOE_i}\right) \approx 0.0079 * Noise_{Sigma} + 0.3684 \tag{5.6}$$

The suggested minimum filter parameter M for a certain input noise level of $Noise_{Sigma}$ ADU can be selected as a power of two value for the DOE algorithm with a resulting standard deviation of the calculated offset below 1 ADU by Equation 5.7.

$$M \gtrapprox 2^{\left\lceil \frac{log\left(Noise_{Sigma}\right)}{log(2)} \right\rceil} \tag{5.7}$$

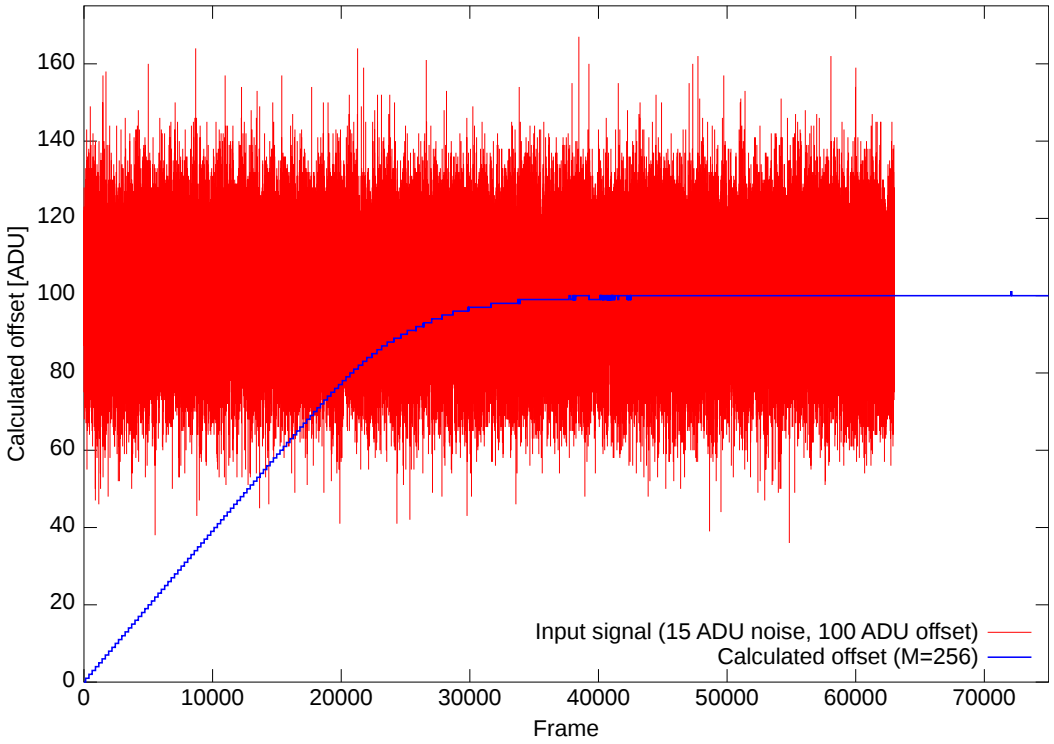


Figure 5.8: Detailed example for an offset time series calculated by the DOE algorithm with a filter parameter $M = 256$. The offset is calculated from the input signal depicted in red with 15 ADU noise sigma and an input offset $O_{Real} = 100$ ADU.

The maximum offset value, which can be calculated by the DOE algorithm, depends on the data type and the applied filter option M. These two parameters define the representable data values for a certain configuration. The maximum manageable offset can be calculated by Equation 5.8. For the hardware implementation of the DOE algorithm value of $O_i(n+1)$, respectively, $O_i(n+1)$ are, furthermore, bound to the maximum data range of the used data type to avoid over- or underflows if the system is operated in improper ranges.

$$\begin{aligned} O_{max} &= \frac{DataType_{max}}{M} \\ O_{min} &= \frac{DataType_{min}}{M} \end{aligned} \tag{5.8}$$

Photon signals always lead to a correction in the same direction (incrementing) of the offset and can, therefore, not be corrected with such a filter approach. In such a case, Equation 5.9 is no longer suitable. Equation 5.10 allows for a rough estimation of the calculated offset in this condition for a filter parameter $M = 256$, 15 ADU noise sigma on the input signal, and a photon probability of $R \in [0, 0.5[$ on the input signal. The coefficients in Equation 5.10 were extracted from simulations.

$$mean\left(O_{DOE_i}\right) \approx O_{Real_i} \tag{5.9}$$

$$\begin{aligned} mean\left(O_{DOE_i}\right) &\approx O_{Real_i} + O_{DOE\,Signal\,Error_i} \\ &\approx O_{Real_i} + R * 13.7 + R^2 * 28.3 \end{aligned} \tag{5.10}$$

To increase the offset calculation precision under a high photon hit frequency, the iterative adaptation for pixels with a photon hit has to be skipped in the particular frame. This modification of the algorithm is described by Equation 5.11, where $Sig_i(n)$ is the photon signal

detection threshold for pixel i. Such a correction eliminates also the requirement for a **m**inimum **i**onizing **p**article (MIP) filtering for the offset calculation as applied in the ROAn algorithm.

$$\overline{O_i(n+1)} = \begin{cases} \overline{O_i(n)} & for\ \ I_i(n) \geq Sig_i(n) \\ \overline{O_i(n)} + 1 & for\ \ Sig_i(n) > I_i(n) - \frac{\overline{O_i(n)}}{M} > 0 \\ \overline{O_i(n)} & for\ \ I_i(n) - \frac{\overline{O_i(n)}}{M} = 0 \\ \overline{O_i(n)} - 1 & for\ \ I_i(n) - \frac{\overline{O_i(n)}}{M} < 0 \end{cases} \tag{5.11}$$

The approach in Equation 5.11, however, is not feasible as at this early stage the signal detection threshold $Sig_i(n)$ is not available within the processing chain. Moreover, an evaluation according to Equation 5.11 is required to anticipate the result of the data processing. Nevertheless, the calculation approach allows for reversing the effect of a false incrementation due to photon events. If a photon is detected in a pixel, the correction factor for the offset value $O_i(n+1)$ is known (-1) without storing any additional information and so the false adjustment can be reversed. The algorithm from Equation 5.11 is, therefore, often split into two parts; Equation 5.12, where the offset is adapted and Equation 5.13, where the event correction is applied if necessary.

$$\overline{O'_i(n+1)} = \begin{cases} \overline{O_i(n)} + 1 & for\ \ I_i(n) - \frac{\overline{O_i(n)}}{M} > 0 \\ \overline{O_i(n)} & for\ \ I_i(n) - \frac{\overline{O_i(n)}}{M} = 0 \\ \overline{O_i(n)} - 1 & for\ \ I_i(n) - \frac{\overline{O_i(n)}}{M} < 0 \end{cases} \tag{5.12}$$

$$\overline{O_i(n+1)} = \begin{cases} \overline{O'_i(n)} - 1 & for\ \ I_i(n) \geq Sig_i(n) \\ \overline{O'_i(n)} & for\ \ I_i(n) < Sig_i(n) \end{cases} \tag{5.13}$$

If the application requires a higher convergence speed of the calculated offset time series, the increment/decrement adjustment can be changed to an adding/subtracting operation with a variable adjustment value S as described by Equation 5.14. This enables a higher flexibility and the adaptation of the algorithm to the current situation. The adaptation of the S parameter has the opposite effect as the M parameter in the DOE algorithm. In comparison to the M parameter the S parameter can be changed dynamically during the runtime without producing jumps in the calculated offset value. As an example for a higher change rate during the system initialization, S is set to values higher than one and after a certain number of frames to $S = 1$. This is especially an advantage for systems, in which high M values are used. Figure 5.9 shows an example of the system initialization for three offset time-series calculated with different S parameters from the same input signal with $O_{Real} = 50$ ADU and 45 ADU noise sigma. This demonstrates the capability of the dynamic change rate adjustment with the S parameter. However, it must be considered that a higher S parameter increases the deviation of the calculated offset time series. Therefore, S should be reduced after a certain initialization period. Sometimes, the necessity arises that the operator has to fix the offset values and stop the automatic adaptation of the DOE algorithm. With the S parameter, this can instantaneously be realized by setting the S parameter to zero.

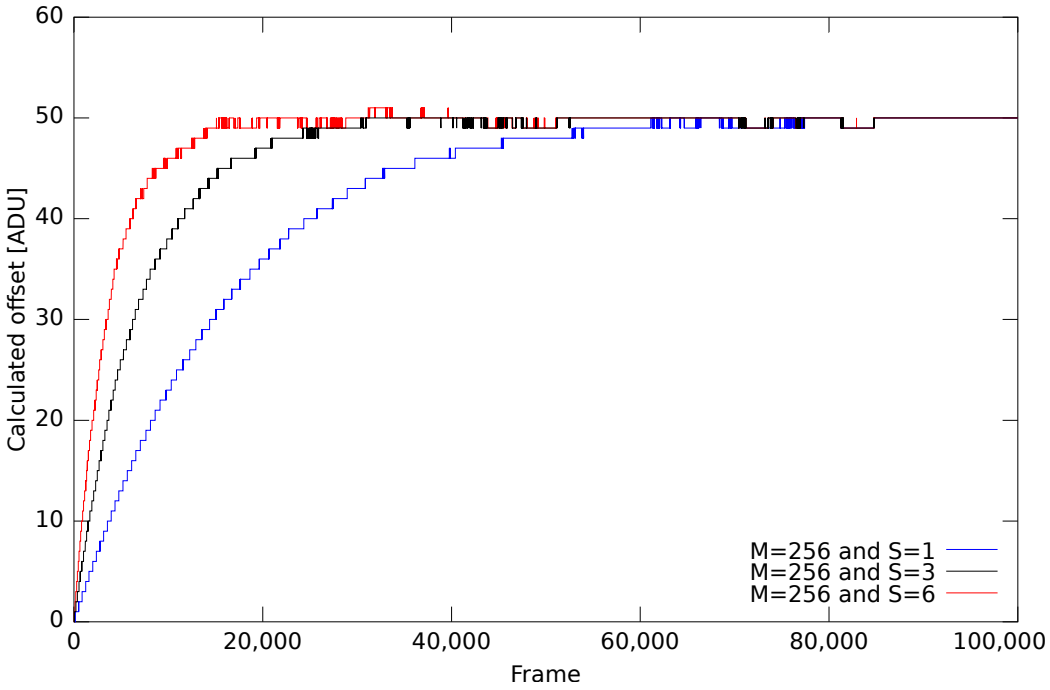


Figure 5.9: Three offset time series calculated by the DOE algorithm with different S parameters for an input signal with $O_{Real} = 50$ ADU and 45 ADU noise sigma. This figure shows the ability of the change rate adjustment with the S parameter.

$$\overline{O'_i(n+1)} = \begin{cases} \overline{O_i(n)} + S & for\ \ I_i(n) - \frac{\overline{O_i(n)}}{M} > 0 \\ \overline{O_i(n)} & for\ \ I_i(n) - \frac{\overline{O_i(n)}}{M} = 0 \\ \overline{O_i(n)} - S & for\ \ I_i(n) - \frac{\overline{O_i(n)}}{M} < 0 \end{cases} \tag{5.14}$$

$$\overline{O_i(n+1)} = \begin{cases} \overline{O'_i(n)} - S & for\ \ I_i(n) \geq Sig_i(n) \\ \overline{O'_i(n)} & for\ \ I_i(n) < Sig_i(n) \end{cases} \tag{5.15}$$

The DOE Equations 5.2, 5.12, and 5.14 require only the last estimated offset value $\overline{O_i(n)}$ or $O_i(n)$, respectively, and the current pixel value $I_i(n)$ to calculate the new offset value $\overline{O_i(n+1)}$ or $\overline{O'_i(n+1)}$, respectively. This allows for processing each pixel independently from others and enables the concurrent processing of multiple input data streams in parallel without the requirement of data exchange or synchronization. Even fully parallel processing of the entire frame would be possible.

The memory bandwidth required for the DOE calculation approach is significantly lower than for the moving average filter approach described by Equation A.1. The presented algorithm requires only one memory read transaction and one memory write transaction to process an input pixel. The moving average filter requires n memory read operations and one write operation; in total, $n + 1$ memory transactions are necessary. The memory reduction factor in dependency of n, the length of the moving average window, can be calculated with Equation 5.16.

$$r(n) = \frac{n+1}{2} \tag{5.16}$$

For a typical history length of $n = 200$ frames, the memory bandwidth can be reduced by a factor of approximately 100 in comparison to the moving average method. The memory bandwidth required for the DOE to process a complete detector matrix can be calculated by Equation 5.17. This results in a necessary bandwidth of $B_{MIXS} = 93.75$ MB/s for the MIXS detector system.

$$B(NumPixels, fps) \quad = \quad NumPixels * fps * 2 * 2\,Bytes \tag{5.17}$$

In many cases the reduction in necessary memory space is even more important than the lower memory bandwidth requirement. In a straightforward implementation the moving average approach requires to store the entire history of n frames compared to a single frame for the DOE algorithm. For resource-limited FPGA systems in a space environment this is a significant advantage.

5.3.2 Dynamic noise estimation algorithm

Noise estimation directly on the raw data stream during a running measurement has not been possible up to now. The static algorithms always require dark frames, where no photon signals are included, at the beginning of each measurement for the calculation. For autonomous system operation, this approach is unusable. With the novel **d**ynamic **n**oise **e**stimation (DNE) algorithm it is now possible to estimate the noise even when the data stream contains photon signals. This is one of the major improvements of the new X-ray data processing system, which for the first time allows for a continuous real-time feedback of the current noise σ level from the detector system during an active measurement. The live monitoring of an essential system parameter is very useful to check if the system is in a healthy state and if the detector system performs as expected. The implementation of a DNE algorithm on FPGA is crucial for the real-time data processing of large and fast readout detector systems.

The most common noise distribution function for physical signals is the normal distribution and has also been observed in the DEPFET detector systems. The **p**robability **d**ensity **f**unction (PDF) of this normal distribution [10] with the parameters μ for the mean or expected value and the σ for the standard deviation is described by Equation 5.18 [10] and depicted in Figure 5.10a.

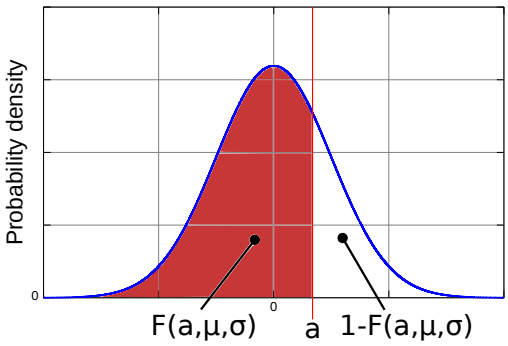
$$f(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} * e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{5.18}$$

The **c**umulative **d**istribution **f**unction (CDF) is derived by integrating the PDF and expressed by Equation 5.20. With the error function erf (x) [10], which is defined by Equation 5.19, the CDF can be simplified as also shown in Equation 5.20. The CDF $(F(x;\mu,\sigma))$ describes the probability of a random input value being within the interval $(-\infty\ ,\ x]$.

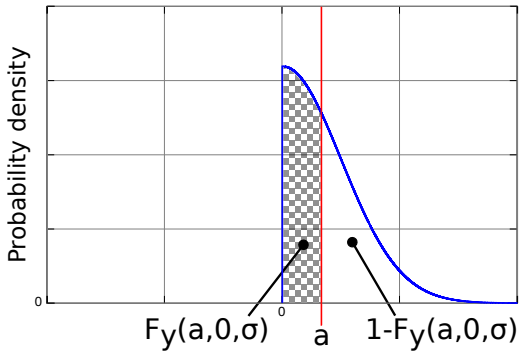
$$\mathrm{erf}\,(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \tag{5.19}$$

$$F(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{(t-\mu)^2}{2\sigma^2}} dt = \frac{1}{2}\left[1 + \mathrm{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right] \quad , x \in \mathbb{R} \tag{5.20}$$

The calculation of the expected value μ and the standard deviation σ from a data series with a normally distributed random variable can be made with the Equations 5.21 and 5.22 [10]. For the normal distribution, about 68 % of all input values are within the standard deviation σ from the expected value μ. The calculation of this 68 % border or σ value in the CDF is a



(a) A plot of the normal **p**robability **d**ensity **f**unction (PDF). The red area represents the result of $F(a)$ the integrated area over the probability density function.



(b) A plot of the half-normal **p**robability **d**ensity **f**unction (PDF). The chequered area represents the result of $F_y(a)$ the integrated area over the probability density function.

Figure 5.10

complex task and an implementation on the FPGA would require a lot of hardware resources. The basic idea behind the DNE algorithm is to reduce the calculation complexity for the 68 % (σ) border value in the CDF of the input signal. The new approach calculates the 50 % border value in the CDF. The noise σ value can be calculated from this 50 % border value by a simple multiplication.

$$\mu = E(X) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{+\infty} x e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx \tag{5.21}$$

$$\sigma^2 = E\left[(X-\mu)^2\right] = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{+\infty} (x-\mu)^2 e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx \tag{5.22}$$

For more efficient processing of the 50 % border, the calculation is made on the absolute input value $|X|$ instead of the direct input values X. The normal distribution for $Y = |X|$ results in the so-called folded normal distribution described in [74]. In the X-ray data processing chain of the SuMo-DAQ system the input offset is corrected before the noise calculation takes place. Therefore, the expected value μ can be assumed to be zero. Under these conditions ($\mu = 0$; $Y = |X|$) the half-normal distribution [84] can be used as special case of the general folded normal distribution. The PDF of the half-normal distribution is given by Equation 5.23 and depicted in Figure 5.10b. The CDF for this distribution is given by Equation 5.24 and the key equation for the DNE algorithm. The median value ($\tilde{y}$) for the half-normal distribution is determined by resolving Equation 5.24 to y for $F_y(y;\sigma) = \frac{1}{2}$. The result of this calculation is given by Equation 5.25. The precise mathematical definition of the median value is given in Appendix A.4.2. The calculation of the median value $\tilde{y}$ (50 % border) can be efficiently implemented with the same algorithm used for the DOE calculation. With this $\tilde{y}$ value the noise σ can be calculated by resolving Equation 5.25 to σ according to Equation 5.26. In this equation, $\mathrm{erf}^{-1}(x)$ [85] represents the inverse error function. With Equation 5.26 the noise σ can now be directly calculated from the median value of the half-normal distribution ($\tilde{y}$), which can be determined efficently.

$$f_y(y;\sigma) = \frac{\sqrt{2}}{\sigma\sqrt{\pi}} * e^{-\frac{(y)^2}{2\sigma^2}} \quad , y > 0 \tag{5.23}$$

$$F_y(y;\sigma) = \mathrm{erf}\left(\frac{y}{\sqrt{2}\sigma}\right) \quad , y > 0 \tag{5.24}$$

$$\tilde{y} = \sigma\sqrt{2} * \mathrm{erf}^{-1}\left(\frac{1}{2}\right) \tag{5.25}$$

$$\sigma = \frac{\tilde{y}}{\sqrt{2} * \mathrm{erf}^{-1}\left(\frac{1}{2}\right)} \tag{5.26}$$

The newly developed DNE algorithm uses this approach for the iterative calculation of the pixel noise σ_i for each pixel i. The algorithm is separated into two parts. First, Equation 5.27 is used to estimate the absolute median value $\tilde{y} \approx N_i(n+1)$ with the input pixel value $I_i(n)$ for frame $n+1$. The second step then transforms the absolute median $N_i(n+1)$ into the noise $\sigma_i(n+1)$ by a constant conversion factor. For a Gaussian distribution of the noise, $\sigma_i(n+1)$ can be calculated by using the Equation 5.28, where the constant conversion factor is $\sqrt{2}*\mathrm{erf}^{-1}(\frac{1}{2}) \approx 0.67449$. For other distribution functions of the input noise signal, the transformation factor has to be recalculated for the specific distribution function. Appendix A.4.3.1 shows how the transformation factor is calculated for the normal distribution. The new DNE algorithm described by Equation 5.27 and 5.28 has a low processing complexity and does not require to store the complete history of frames to estimate individual pixel noise values. Compared to a time series calculation method, where n consecutive frames have to be stored, the DNE algorithm only requires to store a single frame. In this frame all pixel history information for the iterative calculation and filtering are combined. This enables a hardware resource-efficient implementation of the continuous noise calculation algorithm DNE with low memory bandwidth and storage capacity requirements.

$$N_i(n+1) = \begin{cases} N_i(n) + 1 & for \quad |I_i(n)| > N_i(n) \\ N_i(n) & for \quad |I_i(n)| = N_i(n) \\ N_i(n) - 1 & for \quad |I_i(n)| < N_i(n) \end{cases} \tag{5.27}$$

$$\sigma_i(n+1) = \frac{N_i(n+1)}{\sqrt{2} * \mathrm{erf}^{-1}(\frac{1}{2})} \tag{5.28}$$

All considerations concerning the algorithm properties are similar to the DOE algorithm (Subsection 5.3.1) and are, therefore, presented in a compact form. The standard data type used for X-ray images is a 16-bit integer. This data type can store data values much higher than the usually expected noise value of ~ 12 ADU for a DEPFET detector system. This can be used to incorporate a filter option into the proposed algorithm in order to stabilize the estimation value without increasing the required memory bandwidth for loading and storing of the $N_i(n)$ values by taking the bits, which are not required for the data representation. To minimize the required hardware resources for an FPGA implementation, the value D is typically chosen as power of two. This allows for substituting the division operation by a resource-efficient shift operation. The Equations 5.29 and 5.30 show the integration of the low pass filter option into the DNE algorithm, which is realized in the same way as in the DOE algorithm. The filter parameter D in these equations allows for adjusting the integrated filter to the specific application requirements. Similar to the DOE algorithm, the D parameter determines the maximum change rate of the calculated noise σ value. A small D value is chosen if a fast reaction on changes of the data input is required. To obtain a higher stability of the estimation value, a larger value of D is used. In other words, the cut-off frequency of the low pass filter is decreased by increasing the D value.

$$\overline{N_i(n+1)} = \begin{cases} \overline{N_i(n)} + 1 & for \;\; |I_i(n)| > \frac{\overline{N_i(n)}}{D} \\ \overline{N_i(n)} & for \;\; |I_i(n)| = \frac{\overline{N_i(n)}}{D} \\ \overline{N_i(n)} - 1 & for \;\; |I_i(n)| < \frac{\overline{N_i(n)}}{D} \end{cases} \tag{5.29}$$

$$\overline{\sigma_i(n+1)} = \frac{\overline{N_i(n+1)}}{\sqrt{2} * \mathrm{erf}^{-1}(\frac{1}{2}) * D} \tag{5.30}$$

The maximum determinable noise σ for a specific data word size L can be calculated by the Equation 5.31 in dependency of the filter factor D. For the 16-bit data word size used in the SuMo-DAQ, the maximum noise σ for the typical filter parameter value of $D = 512$ $\left(\sigma_{16\mathrm{bit\ max}}\left(2^{16}, 512\right) = \frac{2^{16}}{\mathrm{erf}^{-1}(\frac{1}{2})\sqrt{2}*512} \approx 189\,\mathrm{ADU}\right)$ is more than a factor 20 above the usual DEPFET detector system noise and provides sufficient contingency for the noise σ increase created by radiation damage of the DEPFET system.

$$\sigma_{\max}\left(L, D\right) = \frac{2^L}{\mathrm{erf}^{-1}\left(\frac{1}{2}\right)\sqrt{2}D} \tag{5.31}$$

Similar to the DOE algorithm an event correction is necessary for the DNE algorithm to avoid disturbances and miss calculations if a high photon rate is expected. The integration of this option into the DNE algorithm is realized by the Equations 5.32 and 5.33 in the same way as in the DOE algorithm (Equations 5.12 and 5.13). The event correction is again split into two parts. First, the DNE calculation itself is done by Equation 5.32 and secondly the correction of false adjustments caused by photon events is made by Equation 5.33. This two-step calculation approach enables to correct the DNE value in a different location than the DNE calculation itself, and allows for keeping the DNE algorithm independent from the event extraction threshold $Sig_i(n)$.

$$\overline{N_i'(n+1)} = \begin{cases} \overline{N_i(n)} + 1 & for \;\; |I_i(n)| > \frac{\overline{N_i(n)}}{D} \\ \overline{N_i(n)} & for \;\; |I_i(n)| = \frac{\overline{N_i(n)}}{D} \\ \overline{N_i(n)} - 1 & for \;\; |I_i(n)| < \frac{\overline{N_i(n)}}{D} \end{cases} \tag{5.32}$$

$$\overline{N_i(n+1)} = \begin{cases} \overline{N_i'(n)} - 1 & for \;\; I_i(n) \geq Sig_i(n) \\ \overline{N_i'(n)} & for \;\; I_i(n) < Sig_i(n) \end{cases} \tag{5.33}$$

For dynamic adjustment of the calculated noise σ convergence speed during the system operation, the increment/decrement operation can be replaced by an adding/subtracting operation. This is shown in Equation 5.34, where S is the variable adjustment value. In the same way as for the DOE algorithm this provides for a higher flexibility and enables the adaptation of the DNE algorithm to application-specific requirements during the calculation process. The S parameter has the opposite effect on the DNE algorithm as the D parameter. Therefore, higher S values not only increase the convergence rate, they also cause a higher fluctuation of the calculated noise time series and should be used only if necessary. The noise σ adaptation by the DNE algorithm can instantaneously be suspended by setting $S = 0$ if desired.

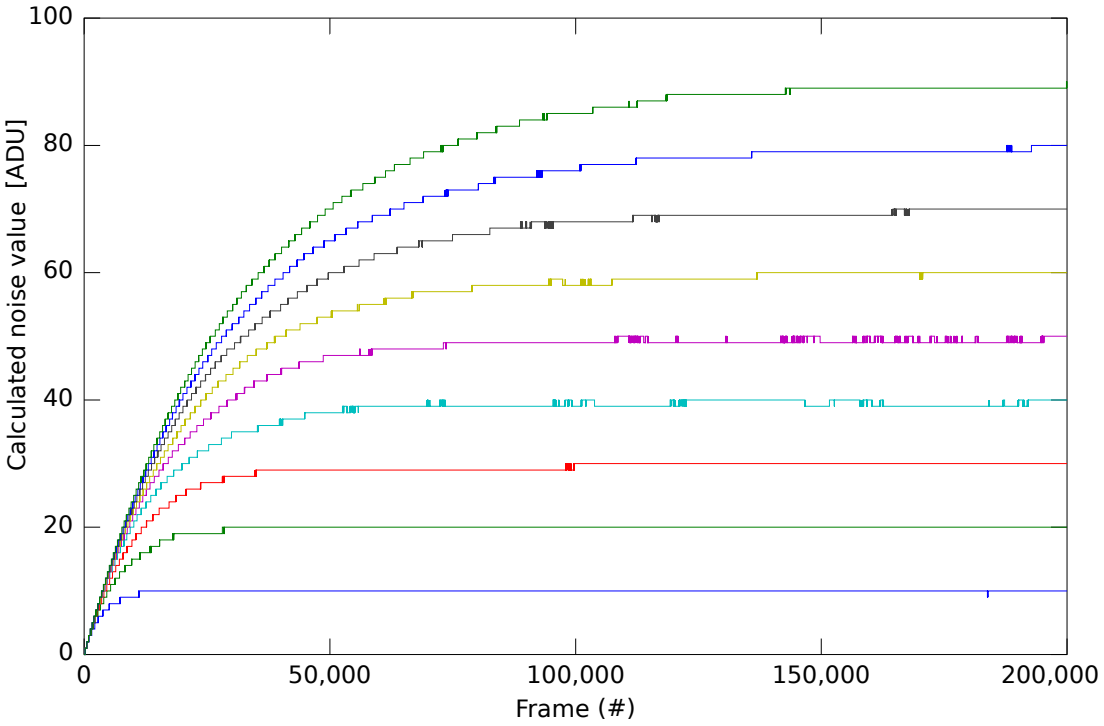


Figure 5.11: Noise time series calculated with the DNE algorithm for a filter parameter of $D = 512$. The input noise σ values were varied between 10 ADU sigma and 90 ADU sigma.

$$\overline{N_i(n+1)} = \begin{cases} \overline{N_i(n)} + S & for \quad |I_i(n)| > \frac{\overline{N_i(n)}}{D} \\ \overline{N_i(n)} & for \quad |I_i(n)| = \frac{\overline{N_i(n)}}{D} \\ \overline{N_i(n)} - S & for \quad |I_i(n)| < \frac{\overline{N_i(n)}}{D} \end{cases} \tag{5.34}$$

The memory bandwidth necessary for the continuous reading and writing of the $\overline{N_i(n)}$ values in the SuMo-DAQ stream processing chain is identical to the DOE algorithm. Equation 5.17 allows for calculating the bandwidth as a function of the detector size and frame rate. For the MIXS detector system a value of $B_{MIXS} = 93.75$ MB/s is calculated.

To demonstrate the high calculation precision and excellence stability of the DNE algorithm for a wide range of operation conditions, simulations in octave [65] were made. An initial noise value of 0 ADU and a filter parameter D of 512 was used in the simulations. The calculated noise time series for input noise values between 10 ADU and 90 ADU are depicted in Figure 5.11. The typical noise level of a DEPFET detector system is approximately 12 ADU. These simulations illustrate the iterative approach of the calculated noise value to the applied level and the settling after an initialization phase. Figure 5.12 shows the noise calculation error of the DNE algorithm extracted from these simulations. For the extraction of the calculation error, the mean value for each simulation run is calculated and referenced to the simulation input value after an initialization phase of approximately 150,000 frames. The maximum calculation error over the complete noise range is ±0.5 ADU.

The stability of the noise value calculated by the DNE algorithm is adjustable with the filter parameter D. The standard deviation of the calculated noise sigma time series is plotted in Figure 5.13 for several input noise levels over the filter parameter D creating the maximum

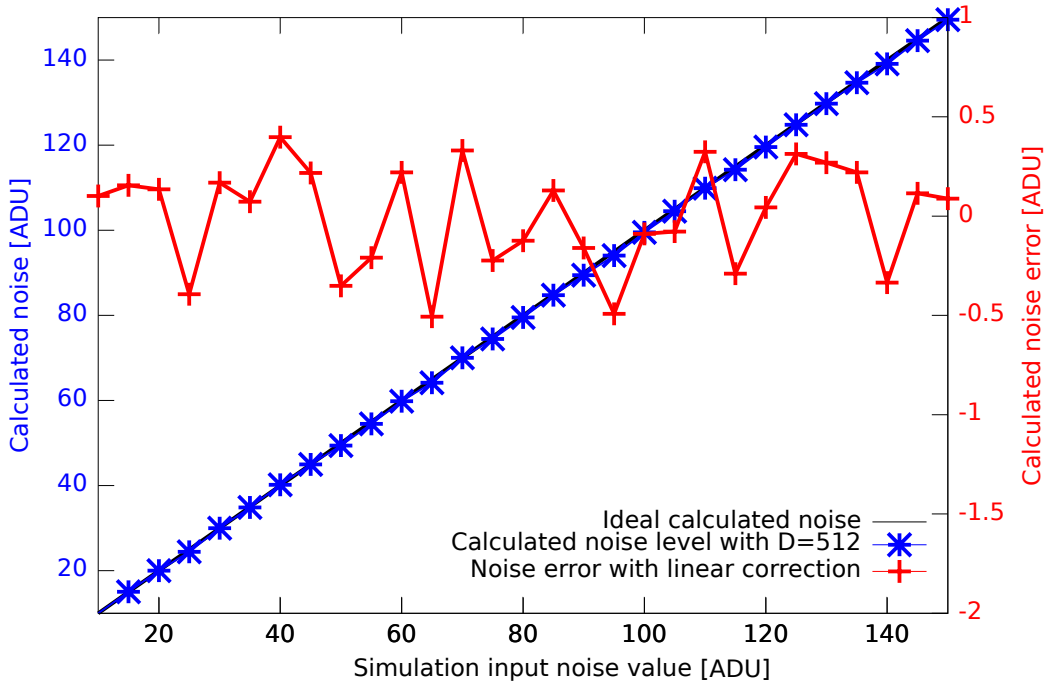


Figure 5.12: Extracted noise calculation error of the DNE algorithm from high-level simulations with an integer data type-based implementation.

signal loss of $5 * \sigma$.

The worst-case standard deviation for the DNE algorithm in fully settled conditions is created if the input noise signal is on the border between two noise sigma levels. Under these conditions, the calculated noise value can continuously jump between two integer values m and $m + 1$. This leads to a worst-case standard deviation of 0.5 ADU for the DNE algorithm in fully settled conditions and with a properly selected filter parameter D. With the SuMo-DAQ standard filter parameter $D = 512$ the DNE algorithm has a standard deviation below 0.5 ADU for all input noise levels up to 100 ADU sigma. This is far beyond the typically expected 12 ADU of a DEPFET detector system and gives plenty of headroom for noise increases during the system operation with a sufficient calculation stability.

5.3.3 Combined operation of the dynamic estimation algorithms

In the previous subsections, the DOE and DNE algorithms were investigated independently from each other. In the SuMo-DAQ processing chain, both algorithms are interconnected and work concurrently on the same data stream. The data flow and the algorithm interconnection in the simulation is shown in Figure 5.14.

The high-level simulation results of such a system constellation with the DOE and DNE algorithms are shown in Figure 5.15. For the simulation input data, a noise sigma of 15 ADU and an offset value of 50 ADU where chosen, while the filter parameters where set to the SuMo-DAQ default values $M = 256$ and $D = 512$. The assumption made in the development of the DNE algorithm ($\mu = 0$) allows for the use of Equation 5.27 only if the input data $I_i(n)$ have a mean value of zero. In the data processing for X-ray images, the offset is removed from the input data before the DNE algorithm receives the data values. Therefore, the assumption is justified if the algorithm is in equilibrium state. If the input is not or not fully offset-corrected, the calculation of the $N_i(n + 1)$ value from the absolute input value $|I_i(n)|$ leads to an offset-

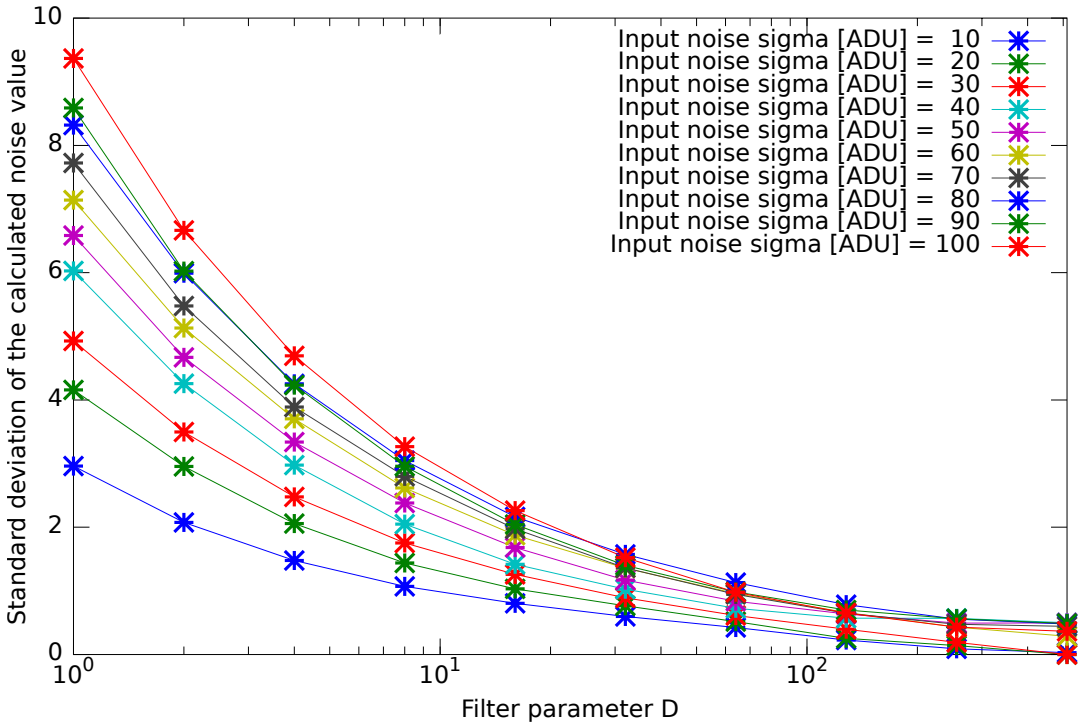


Figure 5.13: Standard deviation or stability of the noise time series after settling calculated by the DNE algorithm for several input signal noise levels plotted over the used DNE filter, parameter D.

dependent calculation error. The simulations of the connected DOE and DNE algorithms, therefore, show an overshoot in the DNE calculation during the system initialization phase. In this phase the offset has not yet been calculated correctly as the assumption ($\mu = 0$) is invalid, which leads to a noise calculation error in the initialization phase. The small overshoot is shown in Figure 5.15d around frame 10,000. Even without the adaptation of the M parameter of the DOE and the DNE algorithm to enable a faster initialization, the system is in equilibrium state after approximately 30,000 frames for these conditions. For the MIXS detector system, this means that the system is fully operational after ~ 5 seconds. For a reliable system initialization, the event correction for the DOE and DNE algorithm has to be deactivated during this time and should be activated afterwards, if necessary. Otherwise, convergence is prevented by the correction of pseudo-events, which are continuously detected because of false offset and noise values.

Simulations of continuously changing environmental conditions are shown in Figure 5.16 for 15 ADU noise sigma and the filter parameters $M = 256$ and $D = 512$. In this simulation, the offset is ramped up from 0 ADU to 75 ADU between frame 100,000 and 200,000. This rate of change is much higher than the expected rates in real detector systems, where the change is mainly created by temperature drifts and radiation damage. The simulation demonstrates the performance of the algorithm even in such extreme situations. The offset calculation error for this simulation run is depicted in Figure 5.16c and shows only a small deviation of a few ADUs during the steep offset change phase. The influence of the dynamic parameter change on the calculated noise value is negligible, as shown in Figure 5.16e.

These high-level simulations show that the interconnected DOE and DNE system performs as expected and delivers an excellent calculation precision after a short system initialization phase. This demonstrates the capability to create an autonomously operating, self-initializing, and self-adapting processing system for X-ray detectors on the base of the DOE and DNE

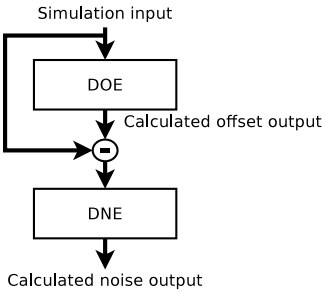


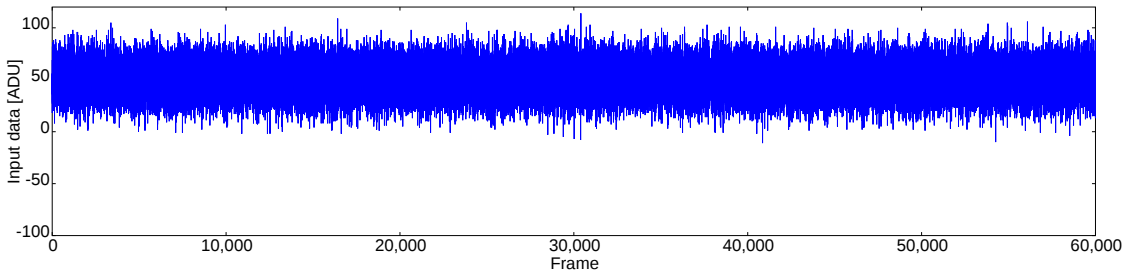
Figure 5.14: Data flow in the combined high-level simulations of the DOE and DNE algorithms.

algorithms. With the dynamic algorithms it is, furthermore, possible to make long-term measurements without system recalibration phases. For conventional static data processing systems, recalibration phases in certain intervals are crucial to hold the measurement precision in the specified range. During these recalibration phases, the system is usually blind and cannot be used for observation. The use of dynamic algorithms allows for extending the available observation time of the instrument. In addition, these algorithms can make the difference between usable and unusable data sets in difficult measurement environments. System disturbances generated by infrared sources in the field of view, for instance, change the offset and noise parameters between the calibration and measurement configuration. Dark frames used in static processing systems are captured with closed shutters and can, therefore, not cover such influences or be utilized to suppress these changes in the offset and noise values.

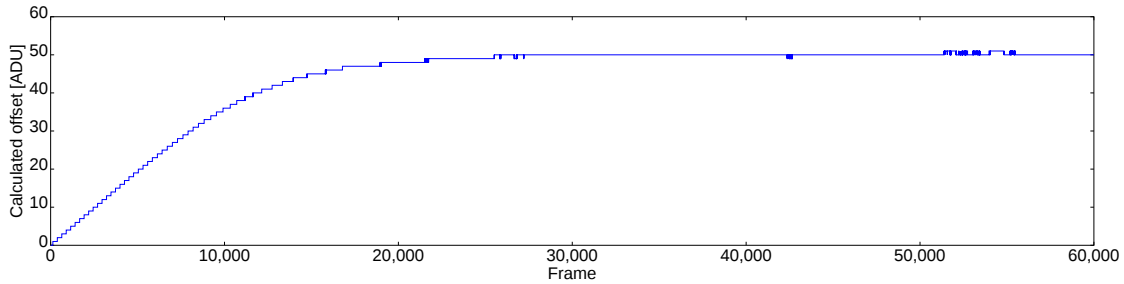
5.4 Hardware implementation of the SuMo-DAQ

The SuMo-DAQ hardware and firmware on the FPGA is assembled with the **E**mbedded **D**evelopment **K**it (EDK) [96, 93] system tool from Xilinx. EDK is a component of the **I**ntegrated **S**oftware **E**nvironment (ISE) Design Suite. This is the Xilinx development product required to implement designs into programmable logic devices from Xilinx. In EDK, the complete system is composed from hardware blocks called **p**rocessor **cores** (PCores) [93] for the implementation and configured on a Xilinx FPGA device. Basic system blocks and functionalities such as memory interfaces, network interfaces, processor cores, system buses and so on are provided by Xilinx as PCores. To enable a high flexibility and interchangeability of the SuMo-DAQ processing system in EDK, all developed and necessary hardware units for the SuMo-DAQ processing chain have been implemented as PCores. These PCores are then interconnected with the help of the software tool to create the complete system on the FPGA. Figure 5.17 depicts an abstract overview of the SuMo-DAQ hardware processing system. The depicted structure is grouped into the different basic functions of the SuMo-DAQ system. How the hardware processing chain is integrated into the complete SuMo-DAQ system was shown in Figure 5.2. The data source interface group (red) connects the ADC channels to the processing chain. The data buffer group (orange) is used for temporary data value storage to smooth the data flow and to avoid data loss if processing parts are temporarily busy. The storage access group (green) enables the reading and writing of data from and to the DPB on-board memory. The system management and debugging group (white) provides statistical and status information on the processing chain and enables to check and control the SuMo-Stream data flow. The network interface group (blue) enables the hardware-controlled data transmitting. The data processing is made by the data stream processing group (gray) and by the dynamic parameter calculation group (yellow).

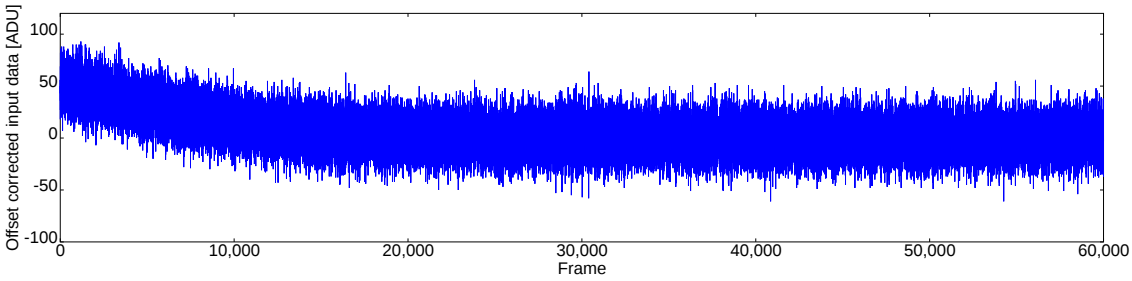
In the following subsection, the functions and implementations of the PCores in the different functional groups of the system are described in detail. The integration of all these PCores into the complete SuMo-DAQ processing chain is finally described in detail in Subsection 5.4.8.



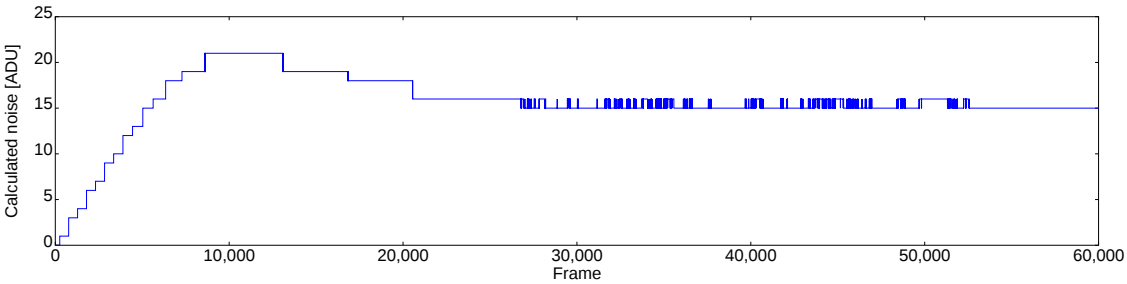
(a) Simulation input data signal.



(b) By the DOE calculated offset value.

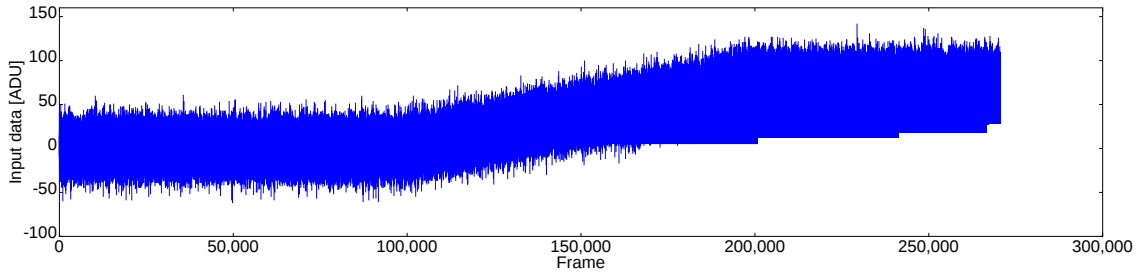


(c) Simulation input data with subtracted offset. This signal is used as input for the DNE algorithm.

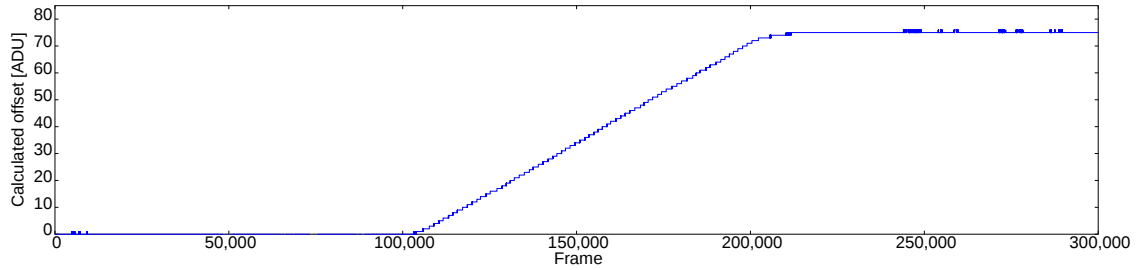


(d) By the DNE calculated noise σ value.

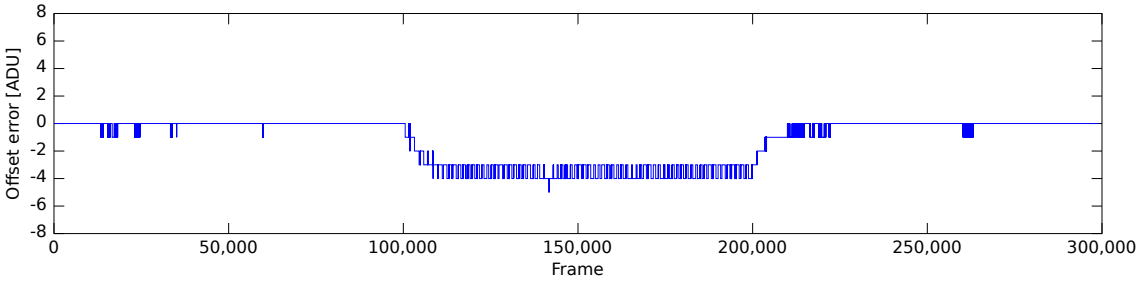
Figure 5.15: Simulation results of the interconnected DOE and DNE algorithms as they are operated in the SuMo-DAQ processing chain. The simulation input offset was 50 ADU, the noise was 15 ADU and the used filter parameters where $M = 256$ and $D = 512$.



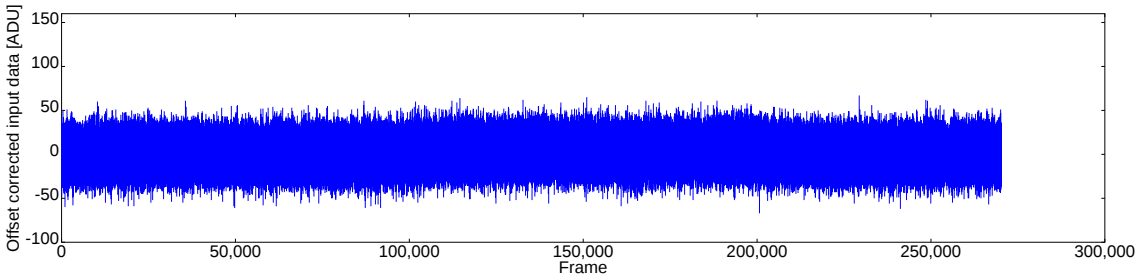
(a) Simulation input data signal.



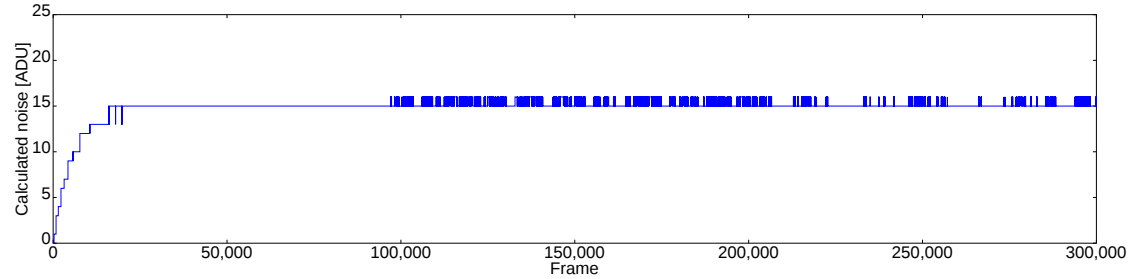
(b) By the DOE calculated offset value.



(c) Error of the calculated offset value.



(d) Simulation input data with subtracted offset. This signal is used as input for the DNE algorithm.



(e) By the DNE calculated noise σ value.

Figure 5.16: Simulation results of the interconnected DOE and DNE algorithms with an offset change from 0 ADU to 75 ADU during the simulation. The simulation input noise was 15 ADU and the filter parameters used where $M = 256$ and $D = 512$.

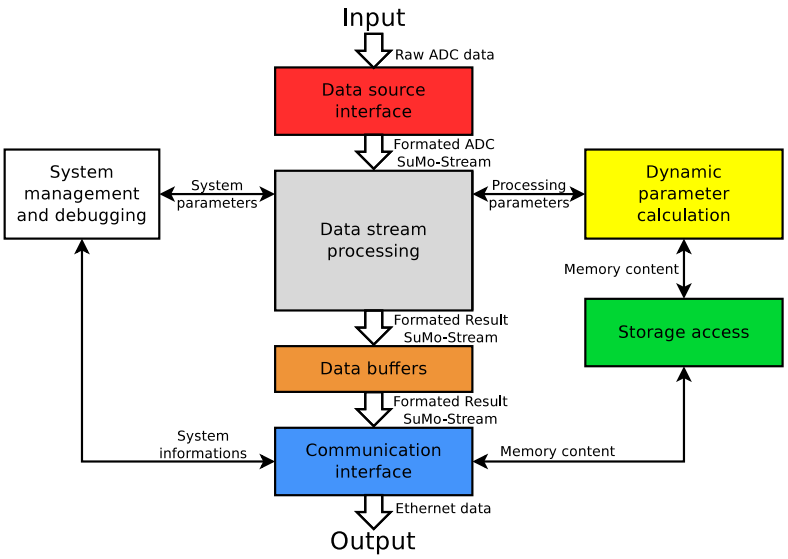


Figure 5.17: An abstract overview of the hardware processing system with functional grouping into the different SuMo-DAQ system tasks. The groups are colored to enhance the clarity of the more detailed SuMo-DAQ processing chain representations.

The main objective for the design, structure, and implementation of the SuMo-DAQ PCores is the creation of a simple and hardware resource-efficient structure to enable a versatile use of the modules.

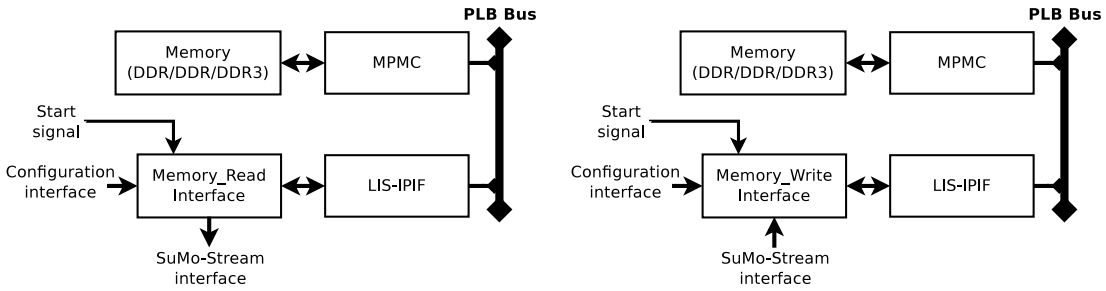
5.4.1 Storage access

Memory interface for read and write access

Two universal memory interface modules for the SuMo-Stream (Subchapter 5.1.4) were developed. The Memory_Read_Interface unit enables to read data from a specific memory address and to convert the memory content into a SuMo-Stream. The Memory_Write_Interface unit allow to write a SuMo-Stream to a certain memory address. This functionality is, for example, used to read and write the offset and noise map for the DOE and DNE algorithms. Furthermore, the units are utilized for support and debugging purposes. During the development and debugging phase, well-known frames are processed to check and verify the processing chain function under real load. For these tests the ADC interface is replaced by a Memory_Read_Interface unit. Together with the UDP packet generation hardware unit the memory interface can also be used as high-speed memory content copy system.

Both interfaces use the **P**rocessor **L**ocal **B**us (PLB) [94] for the memory access via the **M**ulti-**P**ort **M**emory **C**ontroller (MPMC) [98]. For the PLB protocol handling the LIS-IPIF [105] glue logic is utilized. By exchanging the LIS-IPIF PLB bus interface with a **N**ative **P**ort **I**nterface (NPI) or an AXI4 interface, the Memory_Read_Interface and Memory_Write_Interface unit can also be connected by other interface standards to the MPMC. For efficient memory transactions the frame data is buffered in the Memory_Read_Interface and the Memory_Write_Interface unit and then read and written in burst via the PLB and MPMC. The base structure of a SuMo-Stream memory access system is depicted in Figure 5.18. The parallel configuration interface on the Memory_Read_Interface and Memory_Write_Interface units allows for setting the memory base address, frame width and height. The parallel interface avoids a complex configuration interface and enables, furthermore, simple hard-coded static configurations to reduce the hardware requirements if possible.

With the start signal, the transfer of a configured amount of data is triggered. For continuous



(a) The Memory_Read_Interface. (b) The Memory_Write_Interface.

Figure 5.18: Structure of the SuMo-Stream memory access interface. The **M**ulti-**P**ort **M**emory **C**ontroller (MPMC) controls the memory itself and provides an universal interface via the **P**rocessor **L**ocal **B**us (PLB). The Memory_Read_Interface and Memory_Write_Interface units manage the complete memory access and use the LIS-IPIF glue logic for the PLB protocol handling.

operation, as required for the stream processing, a clock signal can be applied as a start signal. The SuMo-Stream data flow control then regulates the data flow on the stream interface.

5.4.2 Data buffers

SuMo-Stream buffer

The stream buffer PCore is used to smooth the data flow through the processing chain and to avoid frequent stops of the processing due to short delays in various PCores. The ADC input part, for example, is a data source driven by an external clock and produces a certain amount of samples per second, which have to be captured. This represents a hard real-time requirement. Furthermore, the buffer can also be used to decouple the processing or data handling of different chain sectors. The network packets, for example, have to be created and transferred blockwise. If multiple SuMo-DAQ streams are transmitted via the same network interface PCore, time slots, where no data transfer is possible for an input SuMo-DAQ stream, can appear. This should not influence the operation of the data processing chain and, therefore, requires SuMo-Stream buffers with a sufficient storage depth.

Two versions of the PCore are available to enable an efficient covering of different buffer sizes. The internal structure for both versions is shown in Figure 5.19. For small buffers with a depth of only a few samples, the FIFO storage structure is realized with **F**lip-**F**lops (FFs). The particular FIFO depth in the small FF-based SuMo-Stream buffer can be selected, but usually the standard depth of four samples is used. This version is typically used inside the PCores as a local stream buffer to compensate for short processing delays.

Large buffers with a depth of a few hundred samples are typically used in between different PCores. Furthermore, this version can be used to store complete detector rows, if necessary. In this version, the internal storage FIFO is realized with BRAMs instead of FFs on the FPGA. This PCore version has a fixed storage depth of $2,049$ samples because of the hardware-defined BRAM size on the FPGA. A reduction of the depth would only waste hardware resources. If a higher storage depth is necessary, multiple of the SuMo-Stream buffers can be concatenated. The hardware resources necessary for different configurations of the PCore are shown in Table 5.4.

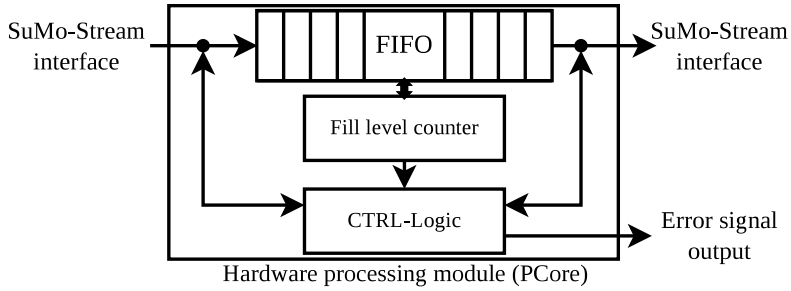


Figure 5.19: Internal hardware structure of the SuMo-Stream buffer PCore.

Implementation type	Slice registers	LUTs	BRAMs	Storage depth	Max. implementation clock frequency [MHz]
Flip-Flops	80	100	0	4	627.7
BRAM	187	99	1	2,049	350.0

Table 5.4: Hardware resource utilization of the two SuMo-Stream buffer configurations.

5.4.3 System management and debugging

SuMo-Stream debug interface

This hardware unit provides no logic functionality and is only used to simplify the debugging process of the SuMo-DAQ processing chain. The complete processing chain is composed in EDK from different PCores. To simplify the interconnection between the different blocks and to enable a fast connection of PCores, all SuMo-Stream signals are composed into a bus. The EDK tool supports bus interconnections between PCores. This enables to connect all signals, which are part of the bus system, at once instead of a complex and error-prone connection of each individual signal. For debugging purposes of the system on the FPGA, the ChipScope Pro [92] logic analyzer from Xilinx can be included by the EDK. Unfortunately, the ChipScope Pro is not able to connect and observe signals from an interconnection bus directly. To enable the use of the ChipScope Pro logic analyzer together with the comfortable bus interconnection the SuMo-Stream debug interface was created. The debugging interface splits the bus into individual signals to allow for the ChipScope Pro logic analyzer connection. Figure 5.20 depicts the internal structure of the hardware unit. Since the PCore does not contain any logic and is used only to get access to the bus signals, no hardware resources are necessary for the implementation on the FPGA.

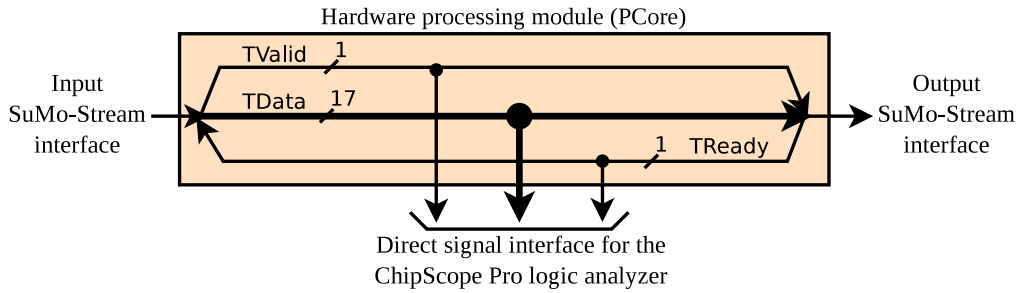


Figure 5.20: Internal structure of the SuMo-Stream debug interface. The bus signals are internally split up into single logic signals to enable the connection of the ChipScope Pro logic analyzer by the **E**mbedded **D**evelopment **K**it (EDK) system tool from Xilinx.

SuMo-Stream switch

The SuMo-Stream switch allows turning on or off the frame transmission over the bus system. Figure 5.21 depicts the internal structure of the hardware unit. Interrupting the data stream has to be synchronized with the transmission of entire frames; otherwise, the state of the subsequent data processing units can be undefined and restarting the processing may cause errors. If the data forwarding is stopped by the switch, all input data values from the sender are accepted and directly discarded to avoid a data concession. By inserting the PCore into the data processing chain, certain processing parts can be connected or disconnected without influencing other system parts. The PCores additionally allow for the automatic extraction of a single frame out of the SuMo-Stream, initialized by an external trigger signal. This option is used for the extraction of frames for status updates from time to time as well as for debugging purposes. As shown in Figure 5.51 the PCore is placed in front of a Memory_Write_Interface PCore, which stores selected frames at a particular address in the DPB on-board memory for further use. The single frame request trigger enables the decoupling of the hardware processing part and the managing software part independently from the stream and CPU speed.

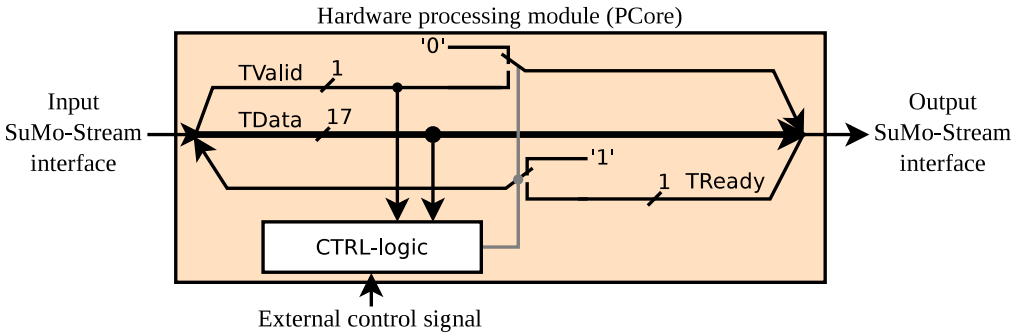


Figure 5.21: Internal structure of the SuMo-Stream switch. The CTRL-logic always ensures the transmission of entire frames over the bus system if they are requested by the external control signal.

SuMo-Stream statistic

This PCore provides some statistical information about the data flow and the SuMo-Stream. The total number of transmitted frames is counted by a 32-bit width counter. In continuously repeated time windows of 650,000 clock cycles, the number of clock cycles is counted where the SuMo-Stream is idle, the receiver is ready to accept data and the transmitter has got valid data. This allows for estimating the current load on the particular SuMo-Stream and the number of frames that was processed by the system.

SuMo-Stream analyzer

The SuMo-Stream analyzer is a PCore for stream checking and system debugging. The core provides the pixel coordinate of the currently transmitted pixel on stream. In addition, the core checks and verifies the transmitted frame size on the stream against an externally provided reference size. During the system development and debugging phase it was very useful to check the data integrity. Furthermore, the coordinate information provided by the PCore simplifies the observation of the stream with the ChipScope Pro logic analyzer. Due to storage limitations only a short time window can be visualized with ChipScope Pro. Therefore, the x/y position information can be useful to trigger the logic analyzer.

SuMo-Stream splitter

The splitter PCore replicates an incoming SuMo-Stream to two output streams. In the SuMo-DAQ data processing chain, it is necessary in several positions to enable the handling of the same data values in different processing units. The PCore controls the complete data flow and makes sure that each output stream receives all data values. This also enables that content stored in the memory such as offset and noise maps have to be read only once during a frame process. This reduces the memory bandwidth and makes the processing system more efficient.

5.4.4 Data source interface

ADC interface

This PCore generates the input SuMo-Stream for the hardware processing chain using the timing-bus signals and the ADC signals. The SuMo-Stream format is described in detail in Appendix A.1. The PCore also realizes the **c**lock **d**omain **c**rossing (CDC) [16] from the external i-Seq clock domain to the FPGA internal domain. The i-Seq domain typically uses an 80 MHz base clock, while the FPGA internal clock is 125 MHz. Driving the FPGA logic with the i-Seq clock signal can cause problems when the i-Seq is stopped for some reasons during the system operation. Furthermore, the 125 MHz clock frequency is necessary to operate and interface the Gigabit network interfaces. To avoid multiple CDC points in the FPGA, the complete hardware processing chain is operated with the 125 MHz clock, and the only crossing point is directly at the ADC input interface. The internal structure of the PCore is depicted in Figure 5.22. The shift register directly after the ADC value input is used to compensate various system delays, which affect the synchronization of the timing-bus signals and the ADC data input. This so-called tab-delay setting is setup-dependent and mainly influenced by the ADC conversion delay, but also by the cabling, configuration, and operation speed of the detector setup. The best value for a certain setup configuration is selected from the shift register by a multiplexer.

The hardware processing chain requires complete detector frames to remain in a defined and synchronous state. In this way, the stopping and restarting of the processing is possible without resynchronization of the chain. The PCore control (CTRL) logic in the i-Seq clock domain ensures that only complete detector frames are captured when they are requested via the control and status signals. If the frame capturing is active, the selected ADC values are combined with the necessary SuMo-Stream format signals and are provided to the output buffer in the FPGA clock domain via a CDC unit. The required hardware resources for the PCore are shown in Table 5.5.

Slice registers	LUTs	BRAMs	Max. implementation clock frequency [MHz]
658	284	1	220.7

Table 5.5: Hardware resources required for the ADC interface PCore. The synthesis and implementation was made with the XST version 13.1.

Old ADC sequence adapter

The timing-bus sequence generated by the i-Seq for the SuMo-DAQ system marks the last pixel of a detector row with a **L**ine**T**erm (LT) and the last pixel of a detector frame with a **F**rame**T**erm (FT) signal. The created data stream structure on the SuMo-Stream by such a new timing-bus sequence is shown in Figure 5.23b.

In sequences used for the old PCI-based DAQ system, the LT or FT signals are indicated in the clock cycle after the pixel value itself. The data stream structure created by such an old timing-bus sequence is shown in Figure 5.23a. As the SuMo-DAQ system encodes the pixel value and the two control signals simultaneously into the data stream, the old timing-bus

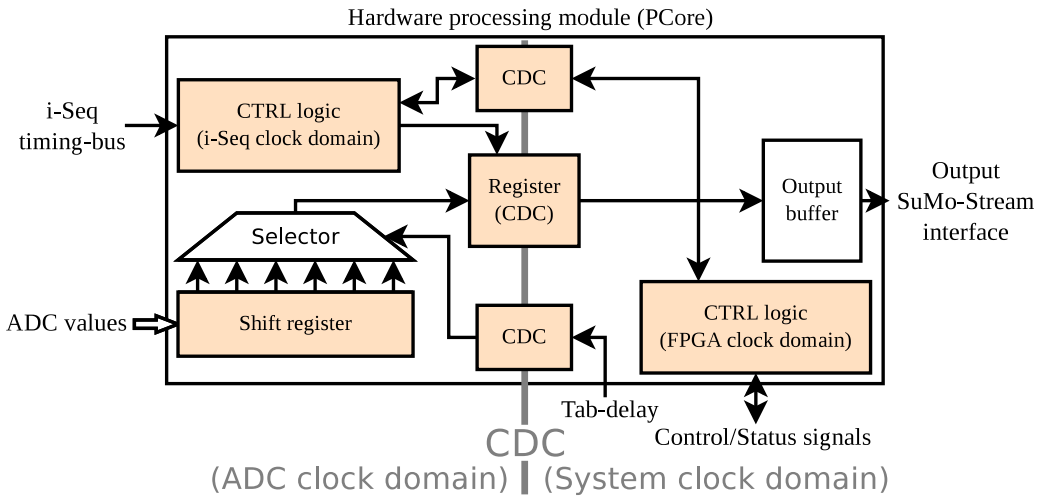
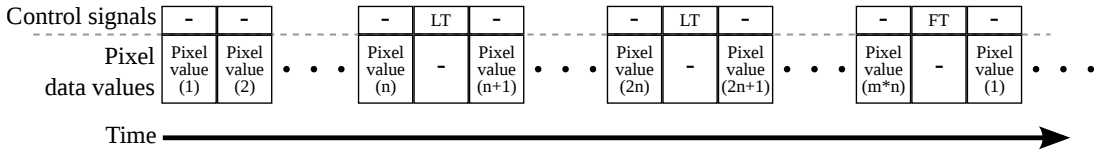


Figure 5.22: Hardware implementation of the ADC interface PCore. This PCore contains the **c**lock **d**omain **c**rossing (CDC) from the i-Seq clock base to the FPGA internal 125 MHz clock base. Via the Control/Status signal the communication with the DPB control software is possible and allows, for example, for starting and stopping the data capturing. The Tab-delay setting is also steered by the DPB control software and enables the synchronization of the timing-bus signals and the sampled ADC values by shifting them against each other.

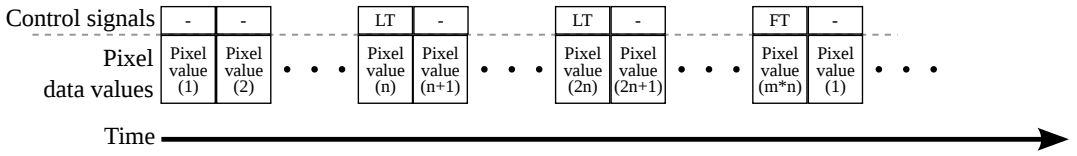
sequence has to be converted to enable the use with the SuMo-DAQ system. The PCore merges the pixel data value and the corresponding LT and FT signals together. This was crucial for compatibility reasons and to enable the operation of the MIXS detector system with the original i-Seq sequences together with the SuMo-DAQ system. Furthermore, this allows for exchanging the DAQ system in a plug-and-play manner during the evaluation and comparison phase.

ADC data stream merger

The performance, throughput, and characteristic of the SuMo-DAQ hardware processing chain allows for handling the data streams from multiple analog input channels in a single chain. The data stream merger PCore combines two input SuMo-Streams into one. This enables a better utilization of the hardware processing chain and allows for increasing the hardware resources efficiency during the SuMo-DAQ system scaling. To enable the processing of the pixel data without any changes on the other PCores the merging has to be realized row-wise. For the common-mode PCore all data values from a complete detector row are expected to be received consecutively. The hardware structure of the PCore is shown in Figure 5.24. The CTRL logic transfers the values of a complete detector row alternately from the "Input Buffer A" and "Input Buffer B" to the common "Output Buffer". Figure 5.24 shows an example constellation of the merging process by the data entries depicted in green. The complete data values from the first row on input channel A are represented by $\mathrm{Row}\,(\mathrm{A}_1)$ and the first row values of channel B by $\mathrm{Row}\,(\mathrm{B}_1)$. On the output SuMo-Stream the complete rows from the two input channels appear in an alternating sequence with the correct order of the row numbers. More details of the various SuMo-DAQ scaling options and abilities are discussed in Subsection 5.6.



(a) SuMo-Stream data structure generated with the old timing-bus sequences.



(b) The new SuMo-Stream data structure used for the hardware processing chain.

Figure 5.23: Comparison of the different pixel encoding structures on the SuMo-Stream. The Control signal LineTerm (LT) marks the last pixel in a detector row and the FrameTerm (FT) signal marks the last pixel of a detector frame. The matrix in this example sequence has n detector columns and m detector rows.

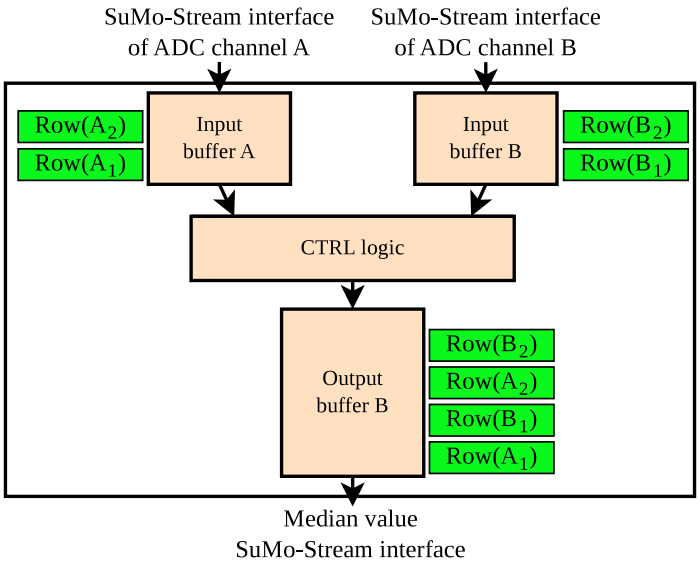
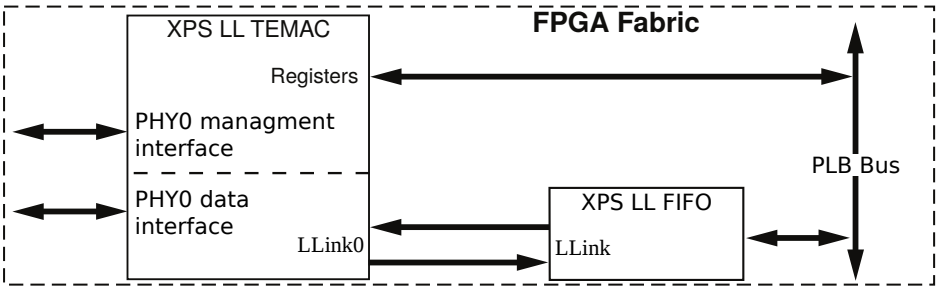


Figure 5.24: This figure depicts the internal hardware structure of the ADC data stream merger PCore. This PCore merges two input SuMo-Streams row-wise into a single SuMo-Stream. A constellation for the merging is shown by the green rectangles, which combine the data values from a complete detector row. Row (A_1), for example, contains all data values from the first detector row received on the input channel A. The complete rows from the two input channels appear in an alternating sequence on the output with the correct order of the row numbers.

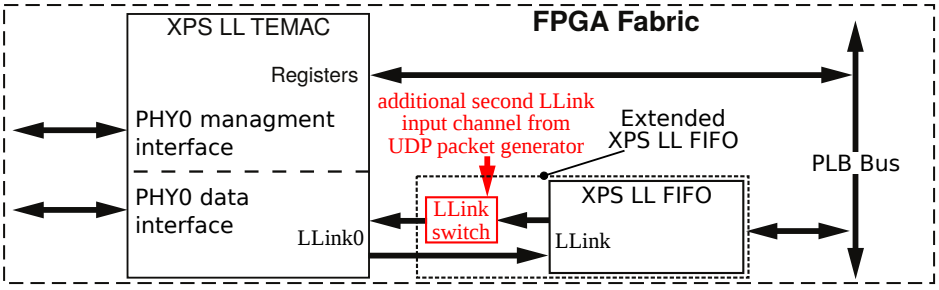
5.4.5 Communication interface

UDP packet generator

The UDP packet generation in hardware by this PCore is an essential feature for the SuMo-DAQ system to enable high data transfer rates and allow for full system performance. Decoupling the packet assembly from the software part reduces the load on the CPU and enables the data stream handling completely in hardware to allow for a more predictable real-time behavior under



(a) Simplified version of the structure suggested by Xilinx for a network interface utilizing a TEMAC and XPS LL-FIFO. The structure is taken from the XPS LL-FIFO product specification[99].



(b) The adapted structure used in the SuMO-DAQ system with the additional second **L**ocal**Link** (LLink) input channel to the TEMAC for a seamless integration of the UDP packet generator PCore into the network connection.

Figure 5.25: Single TEMAC network interface structures.

high system load. Furthermore, this enables the implementation of the SuMo-DAQ system on smaller and less powerful FPGAs without high performance processor cores.

The PCore transforms the SuMo-Stream directly into UDP-compliant packets and hands them over to the **T**ri-Mode **E**thernet **M**edia **A**ccess **C**ontroller (TEMAC) [95] for the transmission on the physical connection. A simplified version of the structure suggested by Xilinx for a XPS LL-FIFO [99] buffered TEMAC interface is shown in Figure 5.25a. This reference structure is described in the XPS LL-FIFO product specification [99]. The realization allows for a buffered read/write access to the TEMAC. The SuMo-DAQ control software, which runs on a CPU on the FPGA, uses this structure for the control communication over Ethernet. Xilinx provides software drivers to configure and use these components with low programming complexity. The commonly used **L**ight**W**eight **IP** (lwIP) [90] stack library is supported by the Xilinx development tools and enables an efficient realization of Ethernet communication systems [103]. As the UDP packets are usually assembled by the PPC core, the structure needs to be extended by a second **L**ocal**Link** (LLink)-Interface [89] input channel for the packet generator. This enables the concurrent utilization of the TEMAC by the software part and the UDP packet generator PCore without restrictions and software driver reprogramming. The LLink is an unidirectional point-to-point connection for packet-oriented data transmitting. The UDP packet generator PCore only transmits data and needs, therefore, only a single LLink channel. A LLink switch was integrated into the TEMAC LLink input channel as shown in Figure 5.25b to merge the data from the XPS LL-FIFO and the UDP packet generator PCore. The packet format for transmitting data to the TEMAC via the LLink-Interface is described in the XPS LL TEMAC product specification document [95].

On the input SuMo-Stream the UDP packet boundaries are marked by the two control signals (LT and FT) of the SuMo-Stream. For the last data word to be encoded into a UDP packet both control signals have to be set. The next valid data word on the SuMo-Stream is encoded into the next UDP data packet. In the UDP packet the datagram length has to be

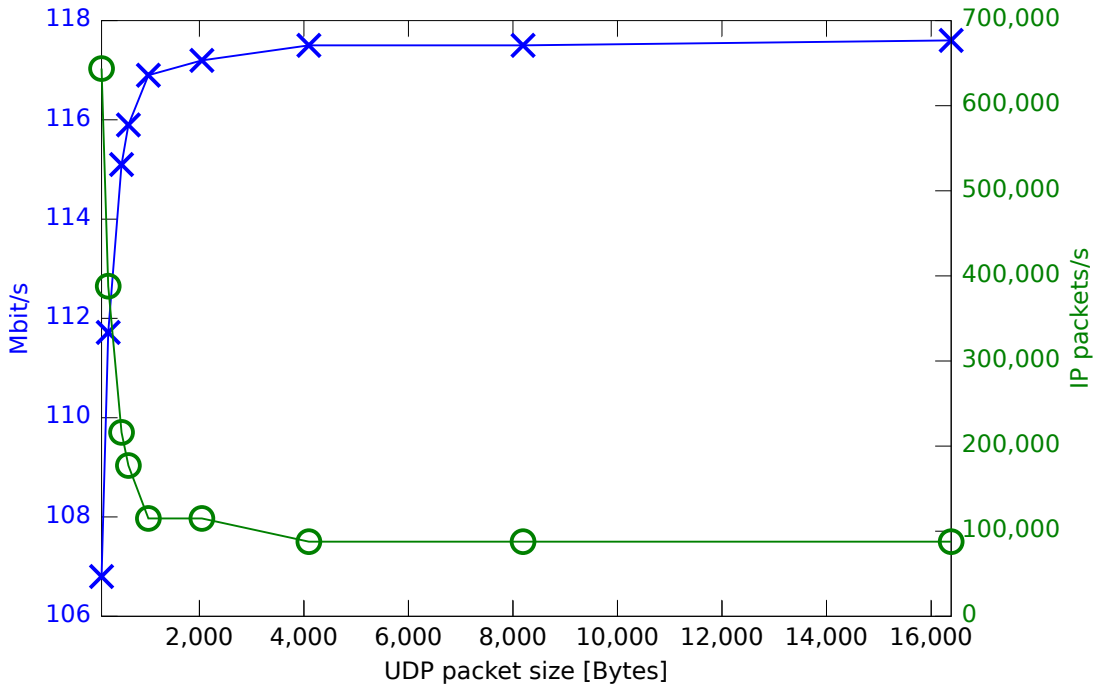


Figure 5.26: Measured performance of a single Gigabit Ethernet link on the DPB. The UDP packets are created with the UDP packet generator PCore on the FPGA. The system delivers line rate performance over a wide range of different packet sizes.

encoded and, therefore, the complete packet payload has to be stored and counted before the packet can be created. In the current hardware implementation a FIFO size of 16,384 bytes was chosen to match the MIXS detector frame size as closely as possible to save BRAMs. Larger UDP packets of up to 65,507 bytes are possible if the buffer FIFO in the PCore is enlarged. The fragmentation [64] required by the **I**nternet **P**rotocol (IP) for large packets is done automatically by the PCore itself. The splitting into smaller pieces is necessary to enable the packet transmission through links with a smaller **m**aximum **t**ransmission **u**nit (MTU) than the original datagram size. To establish a communication channel, only the MAC addresses, IP numbers and port numbers for the sender and receiver have to be provided externally. These parameters are set via dedicated configuration ports of the PCore.

Performance measurements on the FPGA with the implemented UDP packet generator hardware show full Ethernet line-speed performance. The results for different UDP packet payload sizes are depicted in Figure 5.26. The utilization of the two available fiber-optic Gigabit Ethernet connections on the DPB is realized with an additional load balancing unit in front of the TEMACs. The structure is shown in Figure 5.27 and is the default configuration used in the SuMo-DAQ processing system. Furthermore, this allows for a direct scaling of the system to multiple concurrently operated UDP packet generator PCores. As shown in Figure 5.51, multiple parallel data transmissions are used to provide all necessary information from the SuMo-Stream processing system to the software post-processing system. The existing structure for the network connection can be scaled to multiple UDP packet generator PCores by adding an additional LLink switch in front of the load balancer. This is permitted by the packet-based transmission protocol on the LLink and due to the LLink data bandwidth of ~ 500 MB/s this is possible without any limitation. The performance of such a system has been demonstrated by measurements, which showed that the dual TEMAC configuration allows for doubling the usable network bandwidth to up to ~ 225 MB/s. The LLink switch uses a round robin scheduling

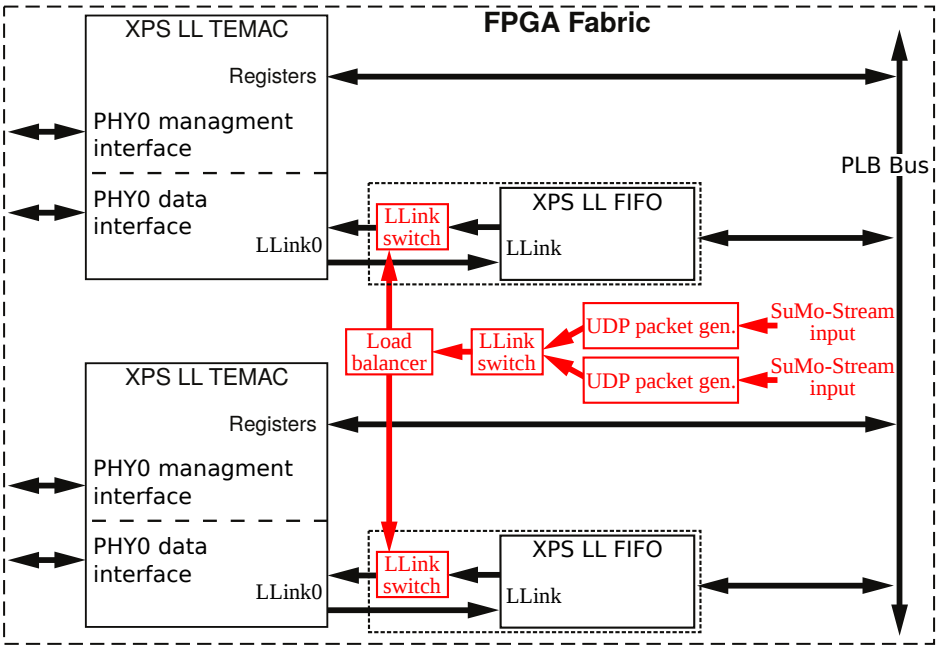


Figure 5.27: Dual TEMAC network interface structure with load-balancing functionality and the ability to connect multiple SuMo-Stream interfaces. This is the standard network interface configuration for the SuMo-DAQ system and allows for fully utilizing both Gigabit Ethernet links available on the DPB.

mode to enable a fair bandwidth distribution even if both input channels produce high load.

Two versions of the PCore are available and only differ in the input SuMo-Stream interface used. The first one has a 16-Bits-wide pixel data SuMo-Stream interface and can directly be used for full frame transmissions as well as for offset and noise map broadcasts. The second version has a 32-Bits-wide event data SuMo-Stream interface and is used to transmit the data stream, which has been pre-processed and, as well, reduced by the SuMo-Stream processing chain.

The internal hardware realization of the two versions of the UDP packet generation PCore is the same and is depicted in Figure 5.28. The two buffers A and B in the PCore are necessary to determine the UDP packet payload size before the first packet header is generated. If the PCore is ready to process a new UDP packet, the data from buffer A are transferred into buffer B until the next packet boundary mark is found. The complete packet payload is then stored in buffer B. The control (CTRL) logic then triggers the creation of the UDP packet and controls the fragmentation into multiple sub-packets if required by the payload size. Furthermore, the CTRL logic provides the FrameID counter, which allows detecting a UDP packet loss by the SuMo-DAQ software post-processing part to the packet generation unit. All additional information, such as MAC and IP addresses, port numbers and so on are stored in the configuration registers. These registers are configured by the DPB control software to enable the network communication. After the UDP packet has completely been forwarded to the TEMAC FIFO via the LocalLink interface and when buffer B is empty, the next UDP packet generation is initiated. The used hardware resources for the PCore implementation are shown in Table 5.6.

5.4.6 Data stream processing

The common internal structure of all data stream processing PCores is depicted in Figure 5.29. The hardware processing core of each PCore is decoupled by buffers from other processing

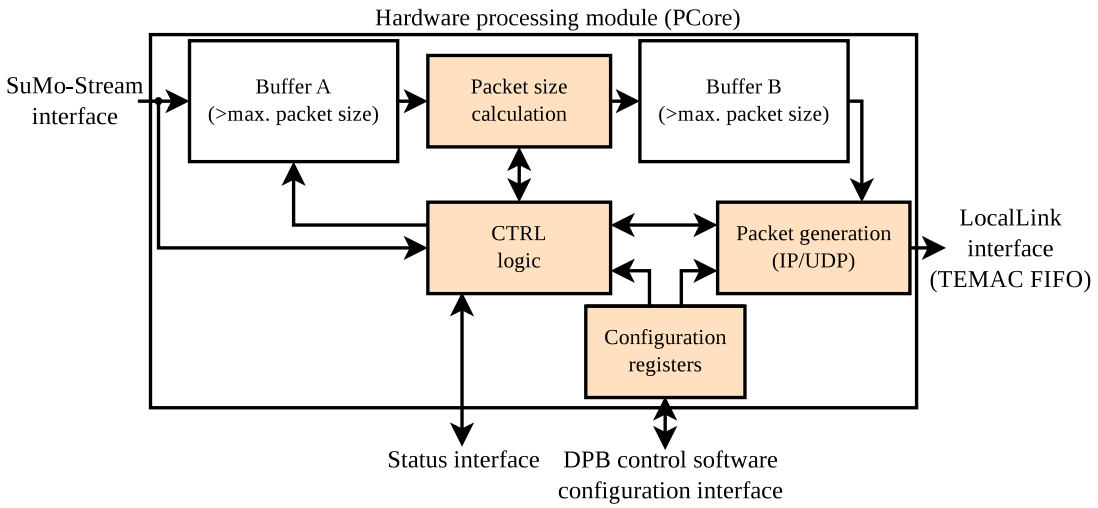


Figure 5.28: Hardware structure of the UDP packet generation PCore.

Input type	Slice registers	LUTs	BRAMs	Max. implementation clock frequency [MHz]
Pixel data stream	1,213	1,375	7	160.4
Event data stream	865	1,206	6	155.0

Table 5.6: Hardware resources required for the UDP packet generator PCore. The synthesis and implementation was made with the XST version 13.1.

parts, which allow for the compensation of small processing delays and enable a smooth data flow. This increases the data processing efficiency due to a reduced number of wait states in the processing chain and enables a more efficient utilization of the complete processing system. Especially in highly branched parts of the SuMo-DAQ processing chain, where wait states can also influence other processing chain branches, this allows for the prevention of performance drops due to frequent processing stops. The buffers typically have a depth of four data samples and, therefore, are realized by FFs on the FPGA. In the following, the SuMo-DAQ hardware processing cores are described.

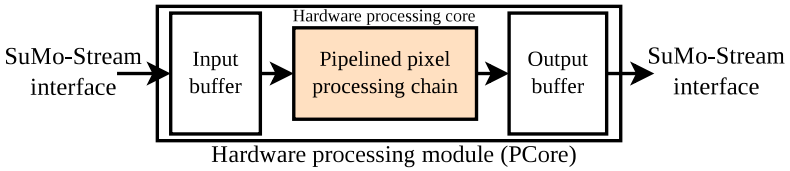


Figure 5.29: Base structure of all data processing PCore modules in the SuMo-DAQ system. Small input and output buffers decouple the internal hardware processing core from other PCores for a smooth data flow through the complete SuMo-DAQ processing chain.

Data shift

This hardware unit subtracts or adds a constant value from all data values transmitted over the SuMo-Stream. The unit is placed at the beginning of the SuMo-DAQ hardware processing chain and enables the adjustment of the baseline as shown in Figure 5.30. The adjustment value is provided externally using a parallel configuration port. For maximum change margins of the

DOE calculation algorithm in both direction the detector baseline should be adjusted to be close to zero. Otherwise, the manageable and correctable offset change can be significantly smaller than the theoretical value given by Equation 5.8. If a detector drift in a certain direction is expected, it is also possible to maximize the margin in this direction. During the MIXS irradiation test campaign, for example, an increase of the leakage current over the time was expected and, therefore, the margin in positive direction was optimized. The adjustment value is typically set during the initialization phase via the SuMo-DAQ control software to allow for an optimal detector baseline shifting. The realized PCore hardware structure is depicted in Figure 5.31 and the necessary hardware resources are shown in Table 5.7.

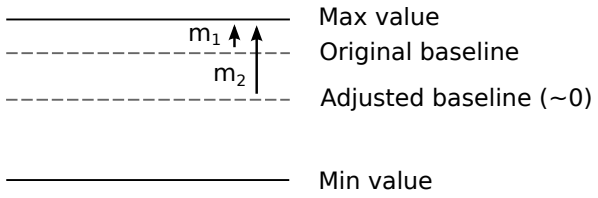


Figure 5.30: Optimizing the offset calculation margin from (m_1) to (m_2) for the DOE algorithm by shifting of the baseline closer to the range center.

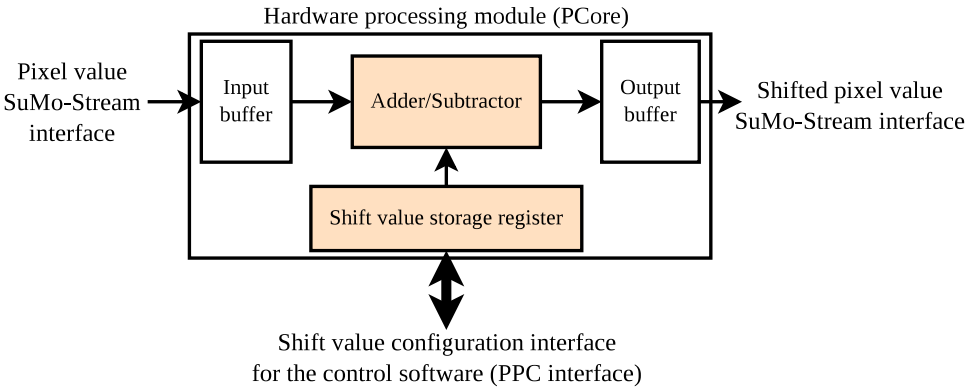


Figure 5.31: Structure of the PCore hardware realization for the data shift module.

Slice registers	LUTs	BRAMs	Max. implementation clock frequency [MHz]
236	92	0	243.1

Table 5.7: Hardware resources required for the data shift PCore. The synthesis and implementation was made with the XST version 13.1.

Bad pixel reset

Bad pixel maps are used in the ROAn software to eliminate data distortions from detector pixels with an unusual behavior. In software, an event which is extracted from such a bad pixel is simply discarded. In hardware, a pixel value cannot be discarded in the full data stream, because there is no pixel position information on the pixel data SuMo-Stream (Appendix A.1) to minimize the necessary hardware resources. The detector pixel coordinate in hardware is defined by the relative position of the pixel value in the SuMo-Stream to the Line- and FrameTerm marks. Therefore, the pixel values are set to a defined value (zero) in hardware instead of discarding these pixels. Furthermore, the data distortions and resulting strange data values

from these pixels should also be suppressed in the offset and noise maps dynamically calculated by the DOE and DNE algorithm. For this reason, the value of the selected bad pixels are set to zero directly in front of the DOE and the DNE PCores in the SuMo-DAQ processing chain as depicted in Figure 5.51. This leads to zero offset and noise for these pixels and, additionally, avoids the event extraction, because the pixel value I_{cmp_i} is always equal to the event extraction threshold Sig_i ($Sig_i = I_{cmp_i} = 0$) and never fulfills the extraction criteria ($Sig_i < I_{cmp_i}$).

Table 5.8 shows the hardware resources requirements of the PCore. In the hardware realization of this PCore the bad pixel map is stored in an external BRAM as shown in Figure 5.32. In order to provide more flexibility and a simpler adaptation of the PCore to different detector sizes, the BRAM is not included into the PCore. Each detector pixel is represented by a single bit in this BRAM, where a bad pixel is indicated by a '1'. The Xilinx BRAMs are dual-port RAMs. One port is used to configure the bad pixel map from the control software side and the other one is used by the PCore hardware to read the data during the SuMo-Stream processing.

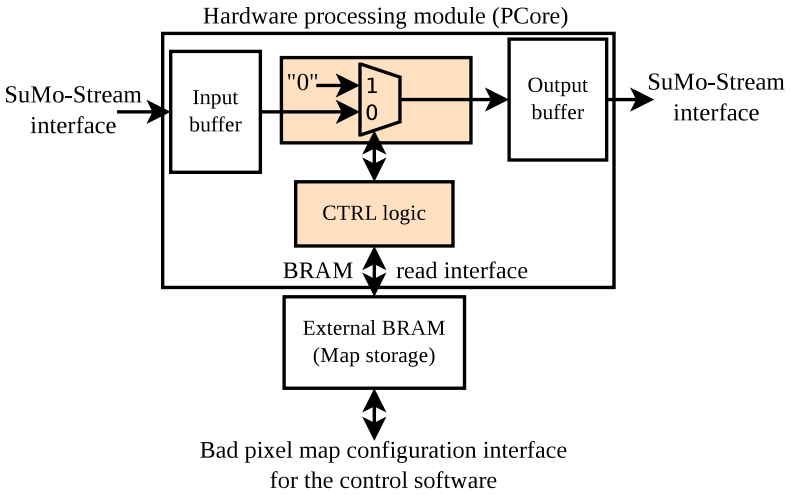


Figure 5.32: Implemented hardware structure for the bad pixel reset PCore. The Bad pixel map is stored in an external dual-port BRAM, where one port is used to configure the bad pixel map from the control software part and the other port is utilized to read the map for the data processing.

Slice registers	LUTs	BRAMs	Max. implementation clock frequency [MHz]
178	116	0	160.5

Table 5.8: Hardware resources required for the bad pixel reset PCore. The synthesis and implementation was made with the XST version 13.1.

Offset correction

This PCore corrects each pixel data value on the SuMo-Stream by the offset value calculated with the DOE algorithm. Figure 5.33 shows the implemented hardware structure of the PCore. The Sumo-Stream synchronization is checked directly after the two input buffers. This check verifies that the pixel input and the offset value input are synchronous, and the correction for each pixel uses the right offset value. For this check, the synchronization of the Line- and FrameTerm signals from both input SuMo-Streams are verified. If differences between the input SuMo-Streams appear, the error output signal is set. In case of an asynchronous SuMo-Stream operation is detected, the DPB control software needs to reset the SuMo-DAQ

processing chain to resynchronize all SuMo-Streams. Theoretically, a desynchronization can happen in system overload situations or due to **s**ingle **e**vent **u**pset (SEU) occurrence. During the entire SuMo-DAQ system evaluation and testing phase, which also includes the long-term irradiation campaign of MIXS, a desynchronization has never been observed. This shows that the buffer sizes are sufficient to enable a smooth data flow under high work load and that the processing chain is stable.

The offset value is divided by the DOE filter parameter M (Subsection 5.3.1) to recover the real pixel offset value before it is subtracted from the actual data value. The offset-corrected pixel values are then provided via the output buffer to the next processing stage.

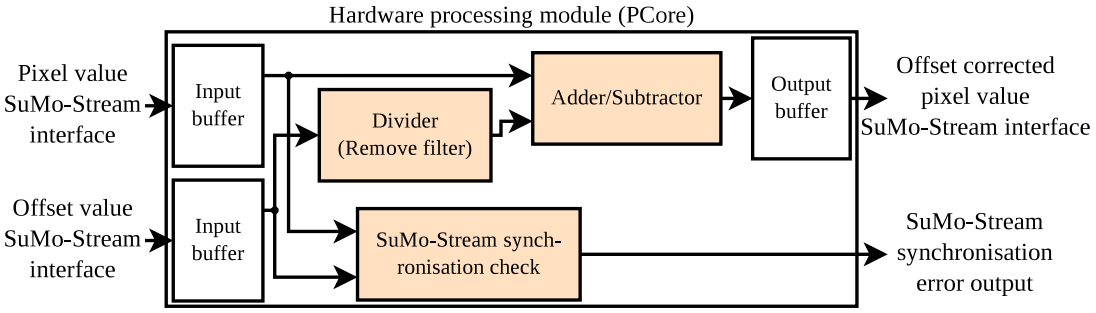


Figure 5.33: Implemented hardware structure for the offset correction module (PCore) with SuMo-Stream input synchronization check functionality.

Slice registers	LUTs	BRAMs	Max. implementation clock frequency [MHz]
551	353	3	170.9

Table 5.9: Hardware resources required for the offset correction PCore. The synthesis and implementation was made with the XST version 13.1.

Common-mode calculation and correction

The common-mode correction PCore contains a complex processing task, the sorting of the detector row pixel values for the median calculation. This sorting task required more than 55 % of the complete analysis time in the software as shown in Chapter 4. This calculation can be realized much more efficiently in hardware. Figure 5.34 shows the implemented hardware structure of the PCore. Key elements of this PCore are the Sort/Store elements in the sorting chain, where the pixel values of a detector row are sorted and stored. Three different hardware implementations of the sorting chain are available. The "Size" version is optimized for low hardware resources utilization with the downside of a low processing performance. The "Balanced" version is a compromise between pixel throughput and hardware resource utilization. The "Performance" version is designed to enable maximum pixel throughput and full utilization of the complete SuMo-DAQ processing chain. Depending on the throughput and scaling requirements the best suitable version can be selected.

Size-optimized version: The hardware implementation of the Sort/Store element used in the "Size"-optimized version of the sorting chain is shown in Figure 5.35. To create a sorting

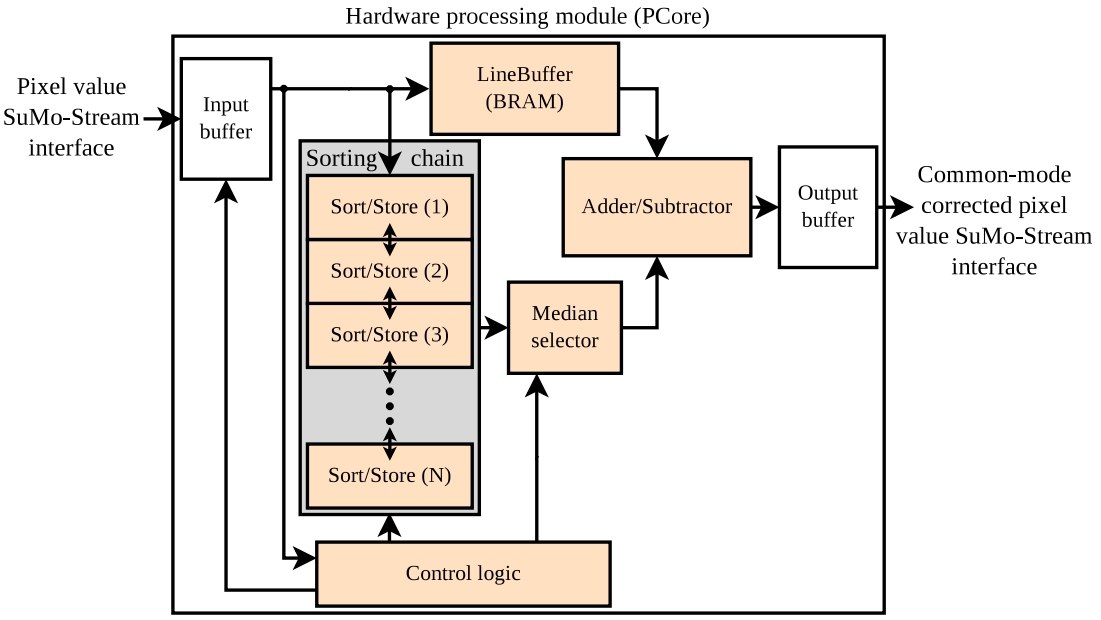


Figure 5.34: Hardware structure of the common-mode calculation and correction PCore. Key components of the PCore are the Sort/Store elements where the data values are sorted before the median value is selected. All pixel values of this detector row are corrected by this median value.

chain, several of these elements are concatenated. Each Sort/Store element contains only a single storage cell and compares the own value with the value stored in the neighboring Sort/Store element. With this Sort/Store element type the chain has to use a two-step sorting approach after the chain initialization. In the first sorting step, a pixel value is loaded into the serial chain of Sort/Store elements. During the second step, the comparison results are used to determine which values have to be exchanged between neighboring Sort/Store elements. Then the first step, where the next pixel value is loaded into the Sort/Store element chain, is repeated. If all row values are finally loaded into the chain, only the sorting step is used to minimize the total sorting time. This sorting procedure is based on the bubble sort algorithm [20]. An example sequence of the sorting process with the "Size"-optimized chain implementation is shown in Figure A.4. The used data input sequence in this example is 5, 1, 6 and 7. A more detailed VHDL simulation of a sorting chain with eight elements for the worst-case sorting scenario is shown in Appendix A.5.1 in Figure A.4. This implementation of the sorting chain does not provide the highest data throughput, but requires to store only one data value per Sort/Store element. In the worst-case situation, the Size version requires $N * 3$ clock cycles for sorting a row with N pixels. If sorting of all row values is completed, the value from the center Sort/Store unit $\left(Y_{N/2}\right)$ is selected as a median value. For an even number of elements the median is mathematically defined as the mean value of the two center elements $\frac{1}{2}\left(Y_{N/2} + Y_{1+N/2}\right)$ as described in Appendix A.4.2. Therefore, the selection of one value is mathematically not absolutely precise, but allows for saving hardware resources while the resulting error for the X-ray data processing is negligible. The selected median value is finally added or subtracted from each pixel in the particular detector row. The incoming pixel values have to be buffered in the LineBuffer until the median value is determined. The LineBuffer depicted in Figure 5.34, is compared to the standard I/O buffers, much larger and realized with BRAMs instead of **F**lip-**F**lops (FFs) from the FPGA **C**onfigurable **L**ogic **B**locks (CLBs) [104]. This component buffers the pixel values until the row median value is determined and the pixel values of this row can be corrected.

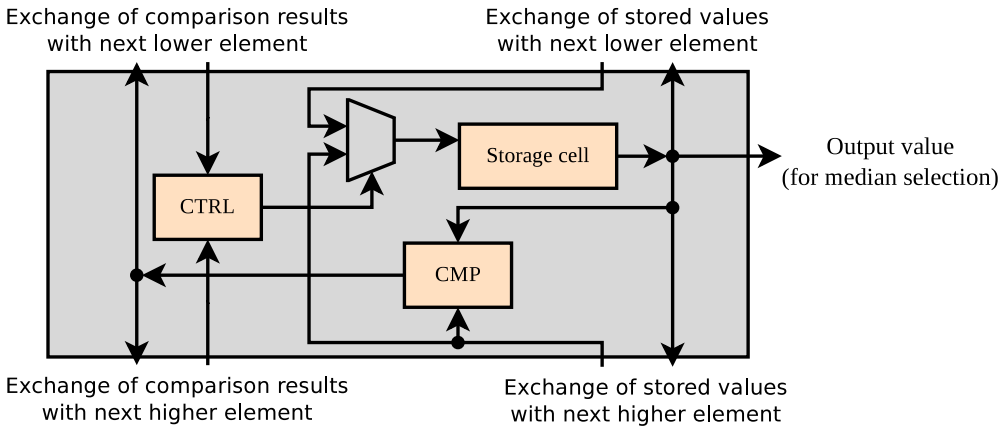


Figure 5.35: Sort/Store element for the sorting chain in the "Size"-optimized version. The element in this version contains only a single data value storage element, a comparison logic, and a control (CTRL) logic. The CTRL logic decides, based on the comparison results from the neighboring Sort/Store elements, whether the data value has to be swapped with a neighboring element.

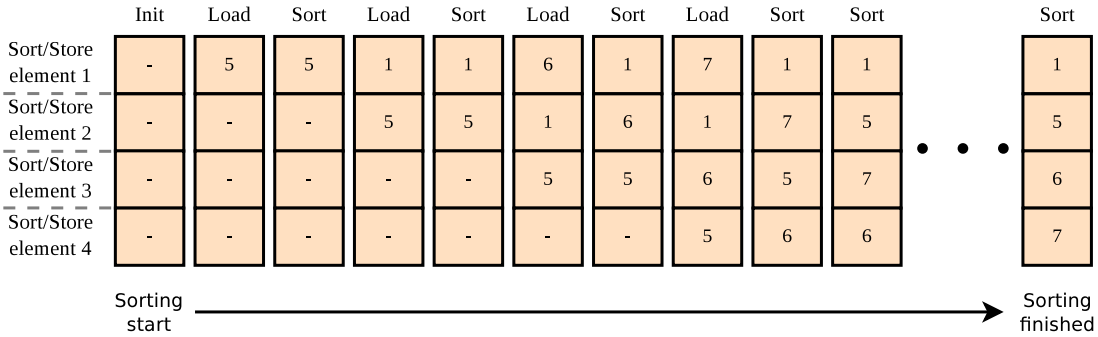


Figure 5.36: Example sequence of the data sorting for the "Size"-optimized implementation of the sorting chain. The data input sequence in this example is 5, 1, 6 and 7. After the initialization of the chain the loading and sorting of the data values is alternated until all values are loaded into the chain. After loading all data values, only sorting steps are necessary to finish the process.

After the chain initialization the loading and sorting of the data values is alternated until all values have been loaded into the chain. After loading all data values only sorting steps are necessary to finish the process.

Balanced and performance-optimized version: The two other hardware implementation versions of the PCore differ only in the realization of the Sort/Store element in the sorting chain. Both versions use the Sort/Store element depicted in Figure 5.37. This element has two storage cells: a local comparison logic and an element control (CTRL) logic. Compared to the Sort/Store element in the "Size" version this realization requires slightly more hardware resources, but enables the concurrent loading and sorting of data values to increase the throughput of the sorting chain. After the chain initialization, the input data values are fed into the first element and pass through the chain up to the Sort/Store element, where the data value has to be inserted to establish the sort order. If both storage places in the element are empty,

a received data value is stored in the storage place 1. In case that storage cell 1 is filled and a new data value is received, the new value is stored in storage cell 2. Then both values are compared and the larger one is provided to the next Sort/Store element in the chain, while the smaller one is moved to storage cell 1. Due to the concurrent sorting and loading of the chain, the number of sorting elements in the chain for the median calculation can be reduced to $\lceil \frac{n}{2} \rceil$. The correct order of the values above $\lceil \frac{n}{2} \rceil$ is not relevant for the median calculation and the values can be rejected. This enables to reduce the number of necessary clock cycles for sorting an N pixel counting row from $N*2$ to $N*\frac{3}{2}+1$ cycles in the worst-case situation. Furthermore, this allows reducing the required hardware resources significantly compared to a sorting chain with a length of n elements.

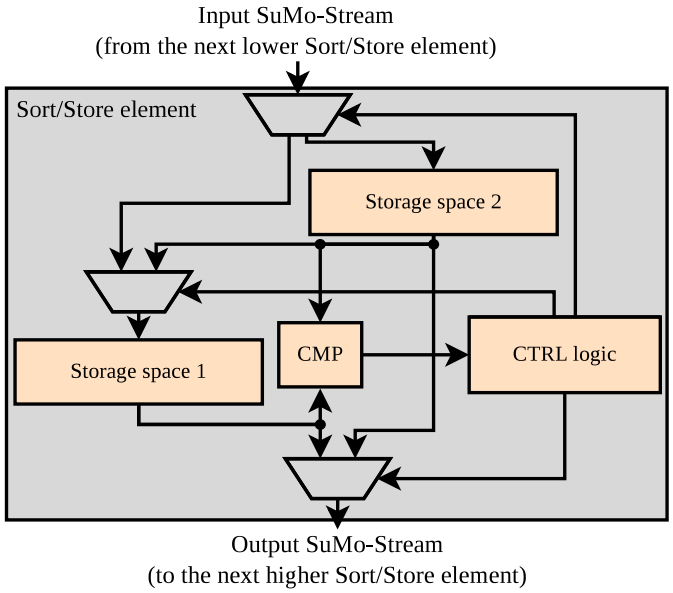


Figure 5.37: Internal hardware structure of a full Sort/Store element for the "Balanced" and "Performance" version. With the two storage cells and the CTRL logic the element is able to perform the sorting of values stand-alone. The smallest value received during the sorting process is stored, while all other values are forwarded to the next higher Sort/Store element.

Compared to the "Balanced" version, in the "Performance" version two sorting chains are used in parallel to maximize the pixel throughput and to enable the full use of the SuMo-DAQ processing chain performance. Figure 5.38 shows the realized hardware structure for the "Performance" version. The two sorting chains are alternately loaded with all data values from a particular row by the "Loader logic". The median values calculated by the two chains are provided to the output SuMo-Stream by the "Reader logic" in the correct order. This dual-chain configuration enables to reduce the hold time on the input SuMo-Stream during the ongoing sorting process and allows for increasing the chain throughput even further than in the "Balanced" configuration. The average number of clock cycles for sorting N values is reduced to $N+1$, while the latency of the processing chain is not reduced. A VHDL simulation of a sorting chain in the Performance configuration is depicted in Figure A.5 for the worst-case sorting scenario.

Slice registers	LUTs	BRAMs	Cycles for the sorting of N values in the worst case	Implementation type	Storage type
1,619	1,893	1	$N * 3$	Size	BRAM
2,863	2,394	1	$(N * 3/2) + 1$	Balance	BRAM
3,954	3,455	1	$N + 1$	Performance	BRAM

Table 5.10: Hardware resources requirements and performance of the different sorting chain implementation versions to calculate the median value from a detector row with 64 pixels. The synthesis was made with the Xilinx ISE tool in the version 13.1.

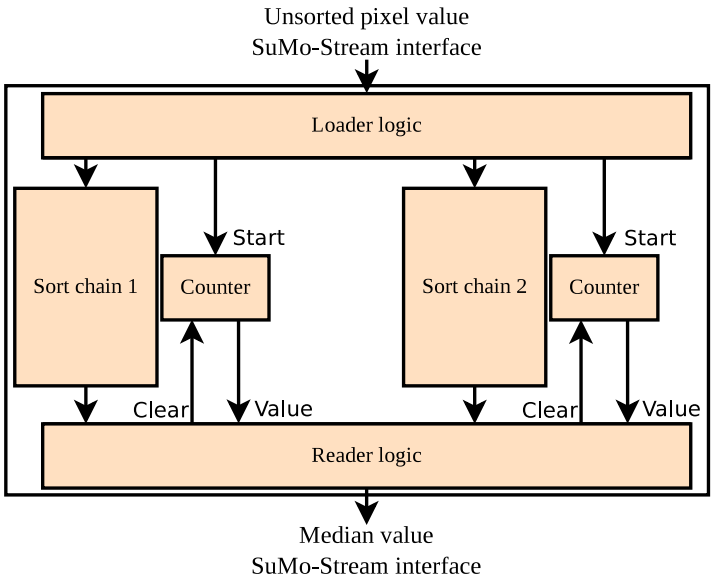


Figure 5.38: Median calculation unit in the "Performance" version. Both sort chains are assembled from several concatenated and stand-alone Sort/Store elements. The "Balanced" version has the same structure, but uses only one sorting chain.

The required hardware resources and sorting performance for the different versions of the common-mode PCore for detector rows with 64 pixels are shown in Table 5.10. For the synthesis of the modules the Xilinx **I**ntegrated **S**oftware **E**nvironment (ISE) [97] tool in the version 13.1 was used.

Event extraction

The event extraction PCore provides a key functionality for the SuMo-DAQ real-time data processing system. Each pixel, which exceeds the threshold value, is encoded into the event SuMo-Stream (Appendix A.1) and transmitted to the software post-processing system on the DHC, while all other pixel values are discarded. This leads to the necessary degree of data reduction for the further software-based real-time data processing on the DHC and provides the scalability of the SuMo-DAQ system necessary to operate the next generation of large X-ray detector systems.

The hardware realization of this PCore is shown in Figure 5.39. For the event encoding, the pixel position on the detector matrix is necessary. This information is extracted from the pixel input SuMo-Stream by two counters for the row and column index in the PCore. These counters use the Line- and FrameTerm control signals on the SuMo-Stream to determine the coordinate of each pixel. This allows using the event extraction module independently of the detector size for all systems without any module reconfiguration or adaptation. The pixel

coordinates derived with these counters depend, of course, on the detector readout configuration and may not correspond directly to the absolute pixel coordinates on the detector matrix. The coordinate recalculation potentially required can be made in the SuMo-DAQ post-processing software on the DHC or in hardware with the event transformation PCore described later on in this subsection.

The noise sigma value for each pixel has to be multiplied by a certain factor to obtain the pixel-specific event extraction threshold. The typical multiplication factor for a primary event is $X_P = 5$, and for a secondary event it is $X_S = 3$. These threshold values are provided together with the pixel value and coordinate information to the selector and event encoder block. This block encodes each pixel above the certain pixel threshold value into the output event SuMo-Stream up to a maximal number of encoded events per frame. The limitation of the maximal number of encoded events per frame is a safety feature to avoid overflow problems during the system initialization, where some system parameters are possibly not in a steady state condition.

The event data structure for the encoding is depicted and described in Appendix A.1. This event SuMo-Stream has 32 data bits and 2 control bits. The increase from 16 data bits used in the pixel value SuMo-Stream to the 32 data bits for the event data SuMo-Stream is necessary, because the event coordinate has to be encoded into the SuMo-Stream entry.

In addition, the PCore generates an output SuMo-Stream, in which all extracted and encoded detector pixels are marked by a flag. This SuMo-Stream enables the event correction in the offset and noise maps calculated by the DOE and DNE algorithms (Subsection 5.3.1 and 5.3.2). For all discarded pixels, the data value is set to zero in this SuMo-Stream, while encoded pixels are marked with a data value of one.

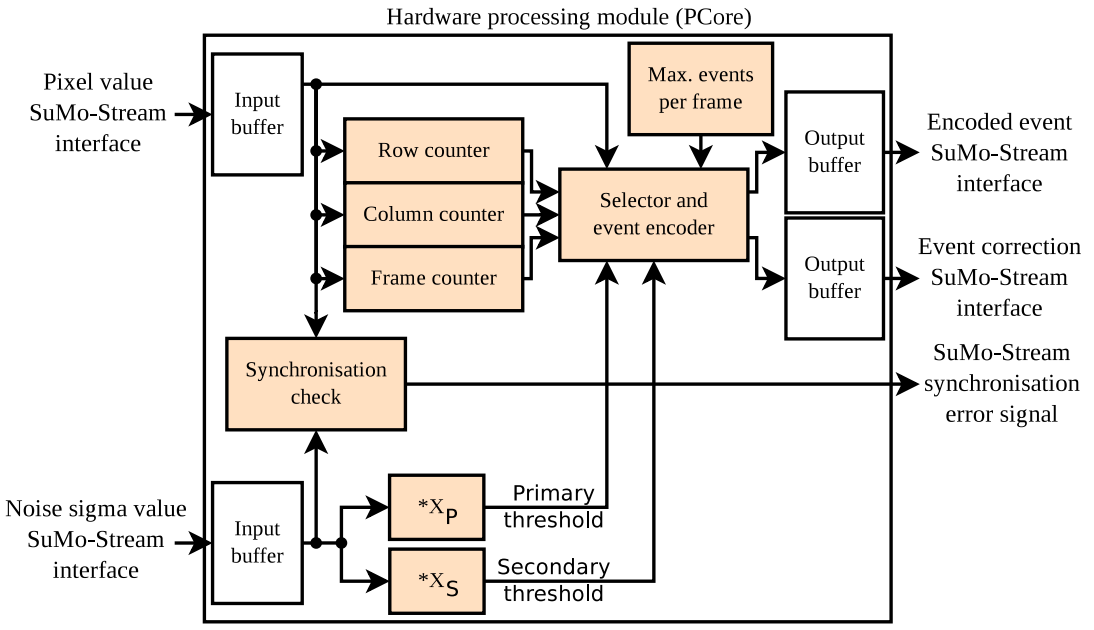


Figure 5.39: Hardware structure of the event extraction PCore. Detector row, column, and frame counter derive for the encoding necessary pixel information from the incoming SuMo-Stream. The pixel-individual primary and secondary event extraction thresholds are calculated from the provided pixel noise sigma values by multiplying with X_P and X_S. The maximum number of encoded events per frame is limited for safety reasons by a constant number. For the encoding of extracted events into the output SuMo-Stream a fixed scheme is used. To enable the event correction for the offset and noise maps, an additional SuMo-Stream is created for the DOE and DNE correction module.

Slice registers	LUTs	BRAMs	Max. implementation clock frequency [MHz]
724	513	0	149.7

Table 5.11: Hardware resources required for the event extraction PCore. The synthesis and implementation was made with the XST version 13.1.

Y in	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Y out	31	32	30	33	29	34	28	35	27	36	26	37	25	38	24	39
Y in	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Y out	23	40	22	41	21	42	20	43	19	44	18	45	17	46	16	47
Y in	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
Y out	15	48	14	49	13	50	12	51	11	52	10	53	9	54	8	55
Y in	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
Y out	7	56	6	57	5	58	4	59	3	60	2	61	1	62	0	63

Table 5.12: **L**ook**u**p-**T**able (LUT) used in the event transformation for the MIXS detector system to convert the relative Y-coordinates into absolute Y-coordinates of the detector.

Event transformation

The event transformation PCore allows converting the raw event coordinate derived from the SuMo-Stream into the absolute coordinate on the detector matrix. This can be used to offload processing power from the software post-processing part into the hardware pre-processing part. Furthermore, this task is also necessary to enable the implementation of an optional event clustering functionality in hardware, which is independent from the detector system.

The realization of the coordinate transformation can be realized universally by a **L**ook**u**p-**T**able (LUT) or with an arithmetic transformation function. For the hardware realization of the PCore, a mixed approach was chosen for a good balance between the hardware resource utilization and the system flexibility. The utilization of LUTs uses more expensive FFs and BRAMs on the FPGA than simple arithmetic functions, but enables a fast reconfiguration by loading a new transformation table into the memory. For the X-coordinate transformation, where the ASTEROID mounting position defines if the row readout direction is from left to right or vice versa, an arithmetic transformation function is used. The Y-coordinate transformation is defined by the programmable i-Seq readout sequence and can be a very complex transformation, if special readout schemes such as the window mode are used. The window mode allows for reading out a region of interest on the detector matrix with a higher rate, while the rest of the detector matrix is skipped. Therefore, the row selection is not necessarily sequential and can contain jumps, which are sequence-dependent. For this reason, for the Y-coordinate transformation a LUT-based approach is used to enable the fast reconfiguration of the transformation. In case of the MIXS detector system with two ADC input channels and a bidirectional detector readout configuration, the X-coordinate transformation is described by Equation 5.35, while the LUT content is shown in Table 5.12. The implemented hardware structure of the PCore is depicted in Figure 5.40. The necessary hardware resources for this mixed implementation style are shown in Table 5.13.

$$f_x(x) = \begin{cases} 63 - \text{X\_Coordinate} & \text{if Y\_Coordinate} > 31 \\ \text{X\_Coordinate} & \text{if Y\_Coordinate} \leq 31 \end{cases} \tag{5.35}$$

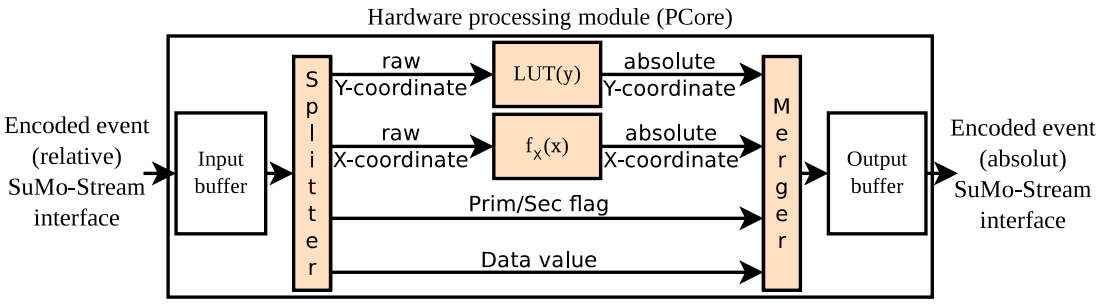


Figure 5.40: Hardware structure of the event transformation PCore. The event SuMo-Stream is split up into the individual stream parts directly behind the input buffer. The Y- and X-coordinates are then modified and transformed into absolute detector coordinates. The individual stream parts are finally merged together into a single SuMo-Stream in front of the output buffer.

Slice registers	LUTs	BRAMs	Max. implementation clock frequency [MHz]
398	206	1	197.3

Table 5.13: Hardware resources required for the event transformation PCore. The synthesis and implementation was made with the XST version 13.1.

Event correction

The PCore increases the calculation precision of the DOE (Subchapter 5.3.1) and DNE (Subchapter 5.3.2) algorithm by eliminating false value adaptations created by X-ray photon events in the data stream. Especially for measurements with a high event rate this is an important feature. Events are only extracted if the pixel signal level is above a certain threshold and, therefore, the values for the DOE and DNE algorithm are always incremented in such a situation. The adjustment error cannot be avoided in the DOE or DNE algorithm itself, because the information, which pixel contains an event signal, is not available at the time of calculation. To remove the influence of events on the calculated values, the adjustment has to be reversed later for all the affected pixels. The adjustment increments for both algorithms are known and can, under these conditions, be corrected by a separate PCore. Figure 5.41 shows the hardware structure of the PCore. The information, which pixels have to be corrected, is provided directly from the event extraction PCore via a dedicated SuMo-Stream, where the pixels with an event signal are marked. The calculated parameter value (offset or noise) is decremented for each pixel, for which an event was extracted if the correction is enabled. The ability to disable the correction is necessary to allow for a reliable system initialization. During the initialization phase of both algorithms, parameter constellations are possible, where otherwise a continuous correction is made and the parameter values will never settle at the right level. The enable signal is controlled by the SuMo-DAQ control software and typically set after a short system stabilization phase. An indicator can be calculated to make the automatic decision if the system is in a steady state or, respectively, to provide the information to the system operator. Various approaches are possible to calculate such an indicator. During the MIXS long-term measurement campaign, the mean parameter change over the entire matrix was calculated in certain time intervals. If the value was below the threshold of 8,192 ADUs, the system was considered to be in steady-state condition.

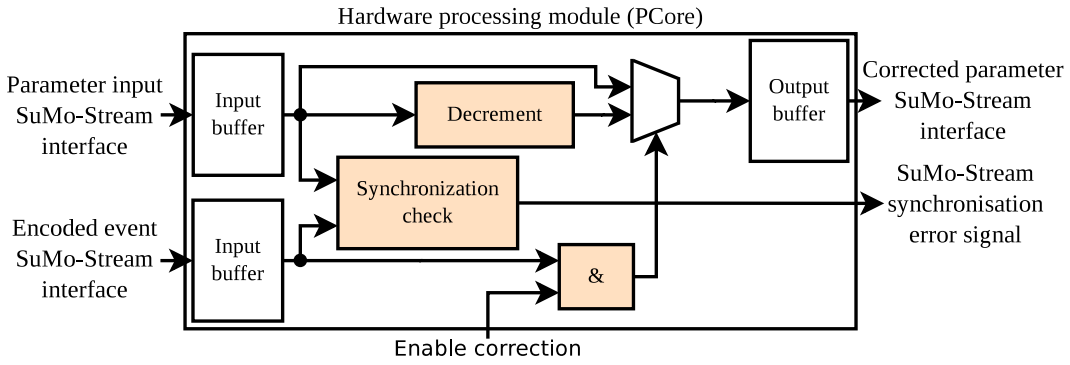


Figure 5.41: Hardware structure of the event correction PCore. The PCore is usable for the DOE as well as for the DNE-calculated parameters. The adaptation error of the parameter for pixels, where an event was detected, is corrected by this PCore. The correction can be enabled by the DPB control software via the enable signal.

5.4.7 Dynamic parameter calculation

Dynamic offset estimation (DOE)

This PCore implements the newly developed DOE algorithm for the dynamic pixel-individual offset calculation, which was described in Subsection 5.3.1. The DOE algorithm can be implemented efficiently in hardware and is realized as shown in Figure 5.42. Directly after the input buffers the synchronization of both SuMo-Streams is checked. If a desynchronization is detected, an error signal is generated. Before the value comparison the old offset value is divided by the chosen M parameter to realize the filter option in the DOE algorithm. The standard filter parameter $M = 256$ for the SuMo-DAQ system is selected as a power-of-two value. This enables to realize the divider unit in the PCore by a resource-efficient shift operation instead of a full integer divider. The comparison result directly controls the offset adaptation by the increment and decrement unit. The new offset value calculated is provided to the output SuMo-Stream buffer. This stream is connected via an external feedback system to the old offset input SuMo-Stream for the calculation of the next detector frame. The feedback system contains a buffer structure, where all individual pixel values are stored up to the next processing cycle. The buffer structure is usually realized with the DPB on-board memory, but for small detectors it can also be realized with BRAMs. The hardware resources required by the PCore are depicted in Table 5.14.

Slice registers	LUTs	BRAMs	Max. implementation clock frequency [MHz]
292	254	0	139.1

Table 5.14: Hardware resources required for the DOE PCore. The synthesis and implementation was made with the XST version 13.1.

Dynamic noise estimation (DNE)

The PCore implements the DNE algorithm for the pixel-individual noise calculation as described in Subsection 5.3.2. This processing step is one of the most important in the SuMo-DAQ processing chain and very critical for the achievable measurement precision. The pixel noise value is crucial in the X-ray data processing and used as a base value for the event detection. The simplicity and the special design of the DNE algorithm allows for a hardware-efficient implementation on the FPGA. The hardware structure of the PCore is shown in Figure 5.43.

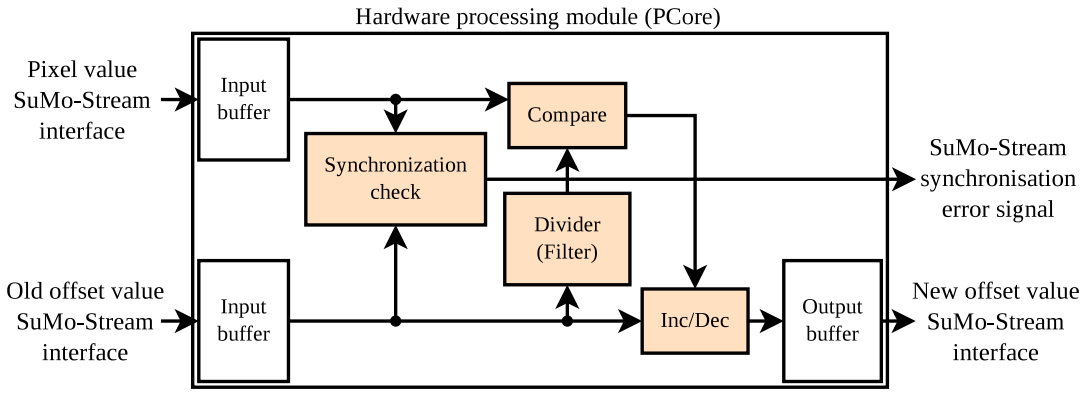


Figure 5.42: Hardware structure of the **D**ynamic **O**ffset **E**stimation (DOE) PCore. The synchronization of both input SuMo-Streams is continuously checked in the PCore to avoid miscalculations. The divider in this PCore is realized with hardware resource-efficient shift, adding and subtracting operations instead of a real hardware divider unit. The calculated new offset output stream is fed back via an external buffer system to the old offset input SuMo-Stream for the processing of the next detector frame.

The synchronization of the pixel input and the old noise value ($N_i(n)$) SuMo-Stream required for the DNE algorithm is checked continuously. In case that a desynchronization is detected, an error signal is generated. The algorithm compares the absolute value of the incoming offset-corrected pixel data value with the noise value ($N_i(n)$) divided by filter parameter D. The D parameter in the DNE algorithm is typically chosen as a power-of-two value to enable the approximation of the integer division by a resource-efficient shift operation. With the result of the comparison the adjustment of the noise value ($N_i(n)$) for the particular pixel i is controlled to calculate the new noise value ($N_i(n+1)$). This new noise value is directly forwarded to an output SuMo-Stream, which is provided to the input noise SuMo-Stream via a feedback buffer system for the calculation of the next frame. The buffer structure is able to store the pixel-individual noise values ($N_i(n)$) up to the next processing cycle. The buffering is typically realized in the DPB on-board memory, but for small detectors it can also be realized with BRAMs. The Memory_Write_Interface and Memory_Read_Interface blocks are utilized to create the feedback buffer system via the memory as shown in Figure 5.51. For the noise sigma output the new noise value ($N_i(n+1)$) needs first to be transformed into the noise sigma value. This transformation is realized by shift, adding, and subtraction operations to approximate the conversion factor of $\mathrm{erf}^{-1}\left(\frac{1}{2}\right)\sqrt{2}D \approx 0.67449 * D$ with less hardware resources. Table 5.15 shows the synthesis and implementation result for the DNE PCore realization.

Slice registers	LUTs	BRAMs	Max. implementation clock frequency [MHz]
459	334	0	204.7

Table 5.15: Hardware resources required for the DNE PCore. The synthesis and implementation was made with the XST version 13.1.

5.4.8 Integration of the complete SuMo-DAQ hardware processing chain

The PCores described in the last subsections are interconnected to a complete hardware processing chain to achieve the desired data handling functionality for the SuMo-DAQ system. The essential hardware processing chain structure on the FPGA for a detector system with a

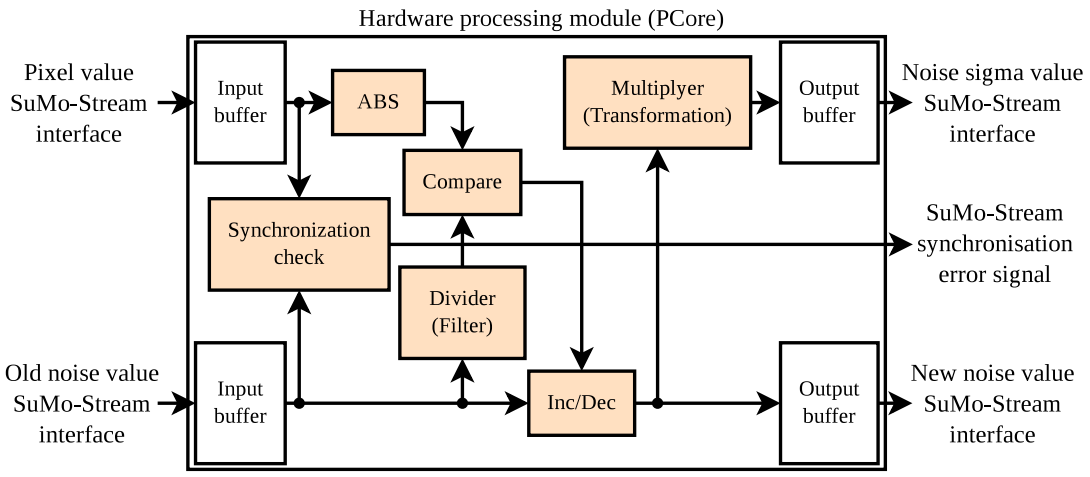


Figure 5.43: Hardware structure of the **D**ynamic **N**oise **E**stimation (DNE) PCore. The synchronization of both input SuMo-Streams is continuously checked in the PCore. The divider and the multipliers in this PCore are realized with hardware resource-efficient shift, add, and subtract operations instead of real hardware divider or multiplier units. The calculated new noise output stream is fed back to the old noise input SuMo-Stream via an external buffer system for processing the next detector frame.

single analog input channel is shown in Figure 5.44. The PCores are grouped by their base functionality into seven categories, which are indicated by different colors in this figure.

The input data stream is generated by the ADC interface and is processed in the different PCores. Finally, the data stream is handed over to the DHC system by the UDP network interface. The sequence of the different data processing PCores is similar to the processing order in the ROAn analysis software. However, there are some hardware implementation-specific components in between to optimize the processing performance of the SuMo-DAQ chain.

In the configuration depicted in Figure 5.44, after the ADC_Interface a Stream_Buffer is placed to enable the compensation of short processing delays in the chain. The dynamically calculated offset value is loaded with a Memory_Read_Interface from the DPB memory and distributed by a Stream_Splitter to the Offset_Correction and the Dynamic_Offset_Calculation PCore. This stream splitting avoids multiple reads of the same value from the memory and allows for minimizing the necessary memory bandwidth. The offset corrected stream is then distributed to the Common_Mode_Correction and the Dynamic_Offset_Estimation PCore via a second Stream_Splitter. The offset values adapted by the Dynamic_Offset_Calculation PCore are written back to the memory, where they were read from. On the other branch of the offset corrected pixel stream, the Common_Mode_Correction PCore is applied to the stream data. The resulting SuMo-Stream is again split and distributed to the Dynamic_Noise_Estimation and the Event_Extraction PCore. The adapted noise values are directly written back to the memory place they were read from and the calculated noise sigma values are provided to the Event_Extraction PCore. The detected and extracted events are forwarded by the Event_Extraction PCore to the UDP_Packet_Generator PCore to hand them over to the DHC system for the software post-processing.

The data taking is started from the DPB control software by activating the ADC_Interface, which then starts to inject the sampled pixel values with the beginning of the next detector frame. With the self-controlling SuMo-Streams, no further global control logic instance is necessary to guide the data streams through the processing chain. To stop the data capturing, the ADC_Interface is deactivated and it stops the data injection after completing the currently processed detector frame.

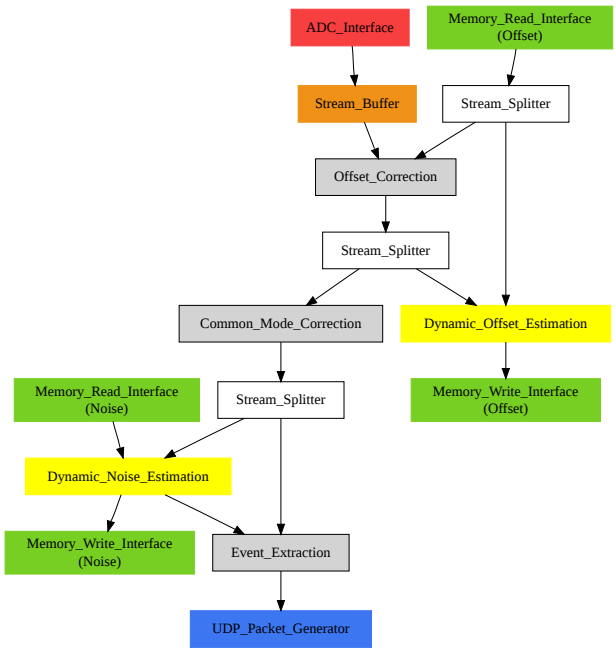


Figure 5.44: Data flow inside the SuMo-DAQ data processing chain on the FPGA for a minimal single analog channel configuration. The block color corresponds to the functional groups to which they belong. The data stream flows from the ADC_Interface at the top down to the network interface at the bottom, where the data is handed over to the DHC system for the software post-processing.

A significant configuration detail for all SuMo-DAQ hardware processing chains is the arrangement of the Common_Mode_Correction and the Dynamic_Offset_Estimation PCore. While all other processing PCores obtain offset and common-mode corrected pixel data values, the data stream for the Dynamic_Offset_Estimation PCore is not common mode-corrected. This increases the noise level for the offset calculation slightly due to the additional common-mode noise fraction, but with an appropriate setting of the filter parameter M this does not influence the calculation precision. The advantage is the access to raw and unchanged pixel data values with the ability to calculate the true pixel offset reliably in any situation. In case the correction has already been made on the input data this can lead to an error in the offset calculation due to partial pre-correction of the pixel offset by the common-mode correction. The calculated offset map would then not represent the true pixel offset.

For the SuMo-DAQ hardware processing chain configuration currently used, the essential configuration depicted in Figure 5.44 is extended by several supplementary PCores, which increase the chain functionality in terms of e.g. debugging and housekeeping. The common implementation of a SuMo-DAQ processing chain for a single analog channel detector system is typically used as shown in Figure 5.45. At several positions in the chain Debug_Interface, Stream_analyzer, and Stream_Statistic PCores are inserted. The added PCores do not change the processing; they only provide a better stream and processing monitoring, allow for more efficient system control and provide supplementary functionality. The processing chain configuration can be extended even further to debug or investigate certain aspects of the hardware processing chain. The Old_ADC_Sequence_Adapter PCore is added to enable direct performance comparison measurements between the SuMo-DAQ system and the old PCI-based DAQ system (see Section 6.2). To maximize the dynamic adaptation range of the processing part, the Data_Shift PCore is inserted into the data stream after the separation of the raw data stream. Bad_Pixel_Reset PCores are added after the Offset_Correction PCore and the Common_Mode_Correction PCore to suppress the influence of bad pixels in the event

Part	Slice registers	LUTs	BRAMs	Max. implementation clock frequency [MHz]
Essential SuMo-DAQ processing chain for one analog channel	9,931 (15.5 %)	8,824 (13.8 %)	42 (18.4 %)	154.8
Full SuMo-DAQ system on the FPGA for one analog channel	29,629 (46.3 %)	28,088 (43.9 %)	114 (50.0 %)	125.5

Table 5.16: Hardware resource requirements of a single analog channel SuMo-DAQ system on a Xilinx Virtex 5 FPGA. The synthesis and implementation was made with the XST version 13.1. The full SuMo-DAQ system on the FPGA includes, besides the hardware processing chain, all further necessary system components (MPMC, PLB, and so on) to operate the SuMo-DAQ system on the DPB. The values in parentheses are referred to the Virtex5-LX100T FPGA from Xilinx.

output stream as well as in the dynamically calculated offset and noise maps. For the data transmission in total three UDP_Packet_Generator PCores are used in the standard system configuration. The UDP_Packet_Generator PCore at the bottom of the chain is still used to hand over the extracted events to the software post-processing on the DHC. The additional UDP_Packet_Generator PCore in subblock (3) is used for the transmission of the complete full-frame detector system data stream without any processing. In this configuration the SuMo-DAQ full-frame data capturing mode can be used in parallel to the hardware data stream processing. This enables precise performance comparisons between both processing variants, because the two branches become exactly the same ADC data values. The three added PCores in the subblock (1) (upper right corner), which are decoupled from the main processing chain, are steered by the DPB control software and used for fast hardware-controlled data transfers to the DHC system without direct CPU interaction. These components are e.g. used to provide updates of the dynamically adaptive offset and noise maps to the system operator. The hardware-controlled data transfer allows for reducing the load on the CPU used for system control tasks in the FPGA. The control software on the DPB only configures the correspondent UDP_Packet_Generator for the data transfer, sets the read start address of the Memory_Read_Interface and initializes the reading. The complete data transfer of the specified memory area is then carried out autonomously by the hardware. To efficiently extend the range of information available for the system operator, an additional Memory_Write_Interface (system) is added by the subblock (2) into the chain. This allows for extracting processed full frames from the data stream and store them at a certain memory area. For the extraction of a new frame from the data stream the Stream-Switch PCore is activated for a single frame capturing by the DPB control software. The two consecutive Stream_Buffers in subblock (2) are to avoid system processing slowdowns due to congestions on the memory interface. Via the software-controlled UDP_Packet_Generator the stored full frame can then be sent to the real-time detector monitoring system of the operator, where also the dynamically calculated offset and noise maps are visible. Using this method to extract and transmit full frames instead of adding an additional UDP_Packet_Generator PCore to the processing chain reduces the necessary hardware resources for the system implementation.

Table 5.16 shows the resource requirements of the SuMo-DAQ hardware processing chain as well as for the complete SuMo-DAQ system on the FPGA. The complete SuMo-DAQ system contains, besides the hardware processing chain, all further necessary components to operate the SuMo-DAQ system on the DPB. This includes the clock and reset signal generation, the two memory controllers, the TEMAC and LL_FIFOs for the network interfaces and the FPGA-internal control signal generation. Furthermore, the FPGA contains various **S**erial **P**eripheral **I**nterfaces (SPI) [100] and **I**nter-**I**ntegrated **C**ircuit (IIC) [101] interfaces to enable the configuration and control of the DPB on-board components by the DPB control software running

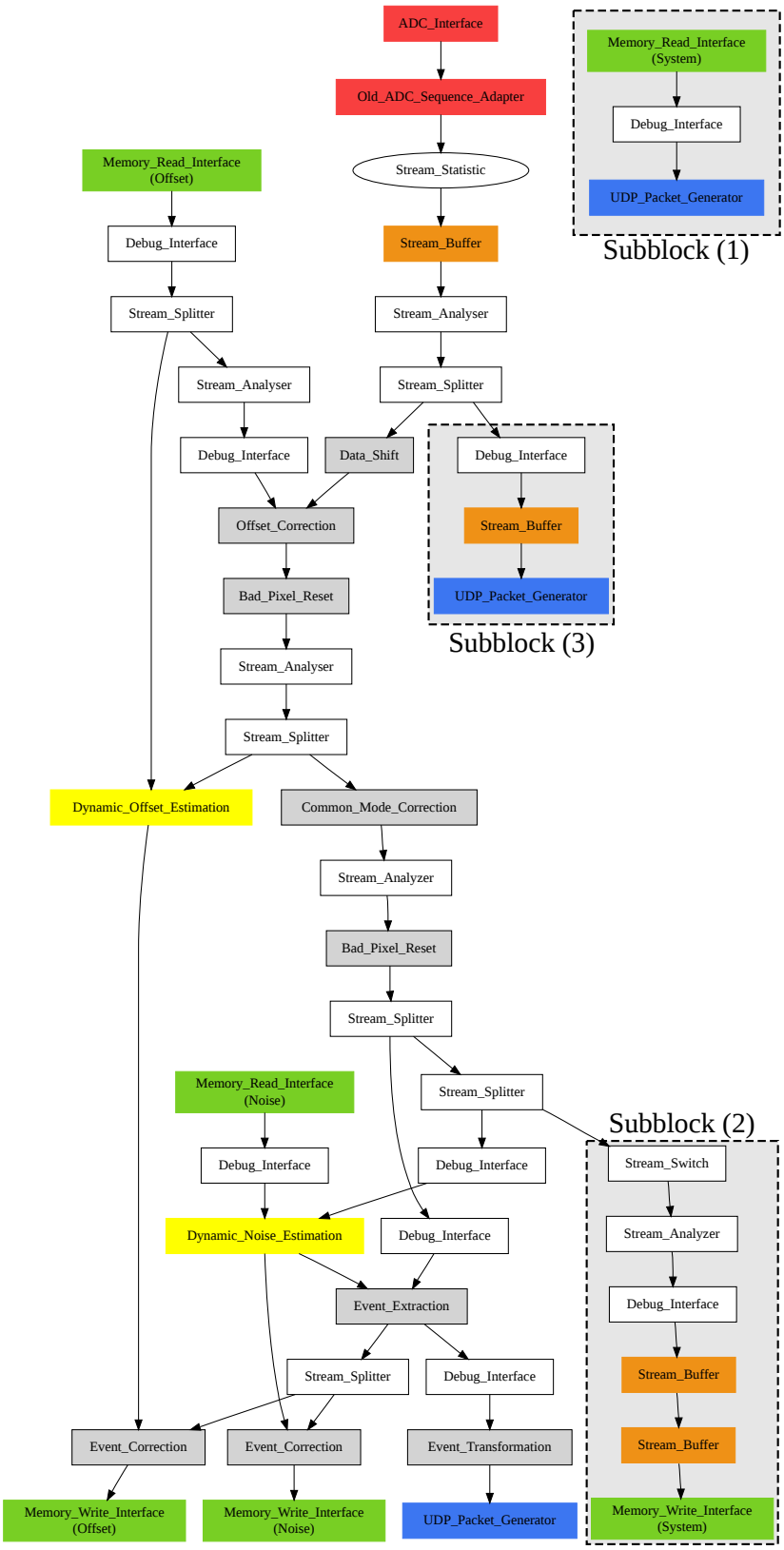


Figure 5.45: Data flow inside the SuMo-DAQ data processing chain on the FPGA for the standard single channel configuration. The block color corresponds to the functional groups to which they belong. The subblock (1) in the upper right corner, which is not connected to the processing chain, is controlled by the SuMo-DAQ control software and used for fast data transfers to the DHC system.

on the PPC cores. The results in Table 5.16 show that only $\sim$ 33.5 % of the hardware resources utilized in the complete SuMo-DAQ system are needed for the data processing chain. The processing chain also requires only $\sim$ 15.5 % of the total available hardware resources in the mid-size Virtex5-LX100T FPGA device. This demonstrates the efficiency of the realized implementation of the processing chain and gives head room for modifications, extensions, and scaling of the SuMo-DAQ system with a single DPB board.

The implementation of all PCores are designed to enable a minimum clock frequency of at least 125 MHz on a Virtex-5 FPGA from Xilinx. This results in a theoretical peak pixel processing rate of 125 MPixels/s for one processing chain. For the measurement of the real processing chain throughput on the FPGA, a dedicated chain configuration, where the ADC_Interface PCore is replaced by a Memory_Read_Interface to generate the maximum chain load, is utilized. All dynamically calculated parameters and the test input values are stored in the DPB on-board memory. The additional loading of the test input values from the memory during the measurement leads to a higher memory bandwidth requirement than for the standard data processing operation. The measured processing performance in this worst-case scenario can, therefore, reliably be achieved under normal conditions. With the Common_Mode_Correction PCore in the balanced implementation version optimized for low hardware resource utilization, the chain has an average throughput of $\sim 0.496 \frac{\text{Pixels}}{\text{clock cycle}}$, which corresponds to $\sim$ 62.0 MPixels/s at a clock frequency of 125 MHz. For the performance implementation of the Common_Mode_Correction PCore the throughput rises to $\sim 0.895 \frac{\text{Pixels}}{\text{clock cycle}}$, which corresponds to $\sim$ 111.8 MPixels/s at a clock frequency of 125 MHz. This is very close to the theoretical peak performance of 125 MPixels/s and demonstrates the high data throughput capability of the hardware processing concept.

5.5 SuMo-DAQ software processing system

The software processing part in the SuMo-DAQ system is separated in three parts. The first part is the data stream post-processing software, where the UDP packets with the extracted events from the DPB systems are handled. The second part is the real-time monitoring and control software, where the status of the detector system and the SuMo-DAQ data processing system is continuously updated. The third part is for the generation of the detailed analysis report of a recorded data set.

5.5.1 Data stream post-processing software

The internal structure of the SuMo-DAQ **d**ata stream **p**ost-**p**rocessing **s**oftware (DPPS) is shown in Figure 5.46. The software is written on the base of the Qt [19] cross-platform application development framework to enable an operating system-independent implementation. For reliable and high-performance data processing even under high work load, the software uses multi-tasking techniques and is separated into three parts: first, the Control (main) process, which initializes and controls the entire data processing and, furthermore, realizes the external communication; secondly, the dedicated network interface thread, which is essential to avoid packet loss under high work load; thirdly, the data post-processing threads, which calculate the results and store the events in the database.

The Control (main) process is the core process of the software and creates all other threads during the start-up process. To avoid continuous time-consuming memory allocation and deallocation operations during the data processing, the software uses a fixed memory pool. This pool is allocated by the software Control process at the program launch and is shared with all the processing threads. Free memory blocks in this pool are organized in the empty list, from which the network interface thread obtains the memory space to store the incoming event packets. The memory blocks filled with new event packets from the DPB systems are moved over into the new event memory block list, where they wait for the software post-processing by one of the various worker threads. After the memory block is processed and stored in the

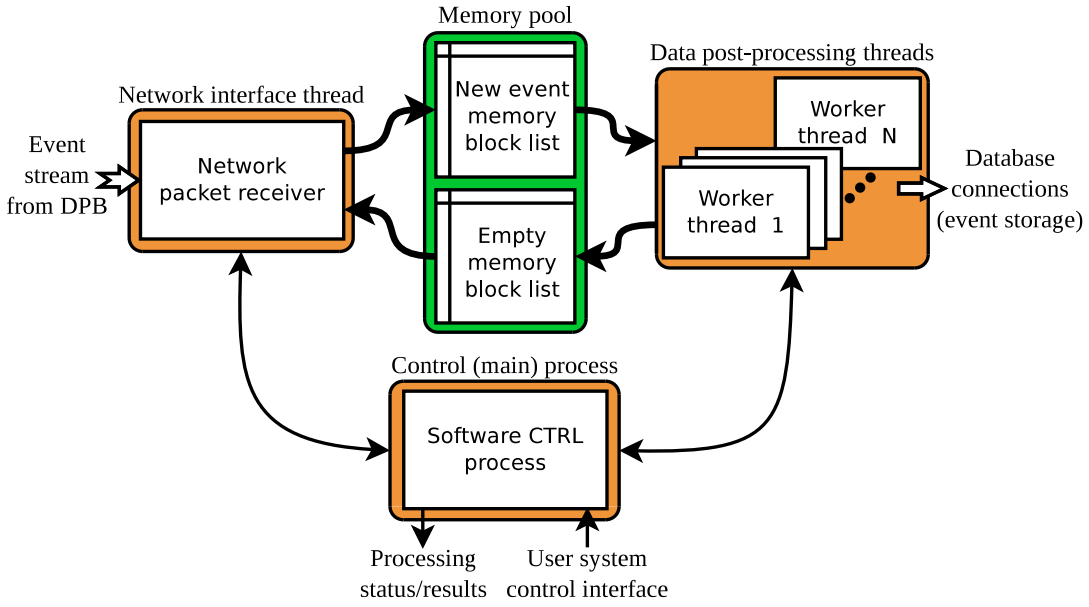


Figure 5.46: Internal structure of the SuMo-DAQ **d**ata stream **p**ost-**p**rocessing **s**oftware (DPPS). The memory pool is filled with new event packets received from the DPBs by the network interface thread. These new events wait in a list to be processed by one of the worker threads. The memory space is returned to the memory pool for reuse after processing and storing of the events is finished by the worker thread. The memory pool and all threads are initialized and controlled by the dedicated software control process.

DB, it is moved back into the empty memory block list to be reused. The dedicated network interface thread is necessary to enable reliable reception of data without packet loss due to buffer overflows in the network stack because of non-continuous packet acceptance. The thread retrieves the packets from the network interface, includes a time stamp and adds the packets to the "New event memory block list". The size of the fixed memory pool and the number of worker threads can be adapted to the operation environment. The number of UDP packets in the memory pool is typically set to 125,000. This provides sufficient buffer capacity to overcome short processing and storage delays in the system even for high-speed detector systems. For a higher data handling efficiency, the event packets are processed blockwise. If the "New event memory block list" contains sufficient packets, the next free worker thread takes a block of 5,000 packets for processing and storing. This block processing reduces the memory access synchronization overhead and the DB insert performance can be increased significantly with multiple row inserts, where many events are collected in a single DB command. The multiple row insert technique is widely used to increase the DB storage performance under high system load. The precise order, in which the events are added into the DB, is not important, because for the data access the unique FrameID key word (Subsection 5.1.6) is used to address the frame content.

Typically 16 worker threads are utilized by the software to have enough processing power available for the post-processing task even under high workload. Each worker thread calculates several parameters and maps for the processed events. For the MIXS long-term irradiation campaign at the TANDEM accelerator, the most important parameters and maps calculated by the post-processing software are the totally placed irradiation dose, the distribution of the irradiation on the detector and, for the calibration measurements, the raw data spectrum. The worker threads can be extended quickly to calculate additional information of interest to the system operator if desired, and the multi-threading software approach provides enough processing power for these extensions. The Software CTRL process collects this data from all

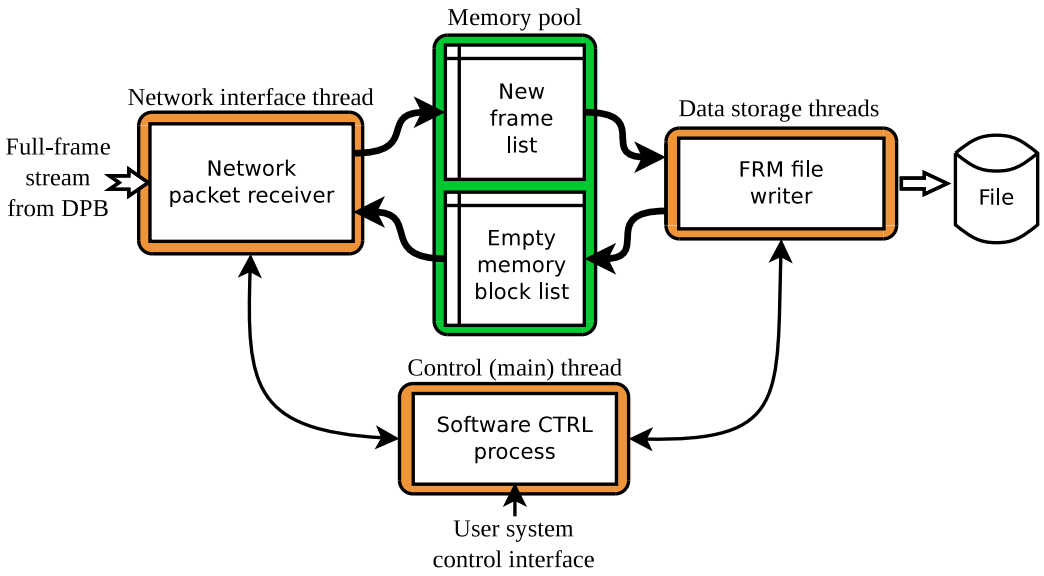


Figure 5.47: Internal structure of the SuMo-DAQ full-frame capturing software. This tool stores the full-frame data stream provided from the DPB system in the FRM file format without any data processing. For the offline data analysis task, the ROAn software is used in the same configuration as for the old PCI-based DAQ system. This capturing option enables precise performance comparisons between the SuMo-DAQ data processing and the ROAn processing on identical data sets.

worker threads and merges them together to the global real-time processing results.

The visualization of the global real-time processing results is decoupled from the data handling software. This allows for the use of the same software even for larger systems, where multiple DPBs are operated concurrently and the real-time information from the different detector sections has to be merged before they are visualized. The Control thread in the DPPS software broadcasts, therefore, the collected real-time processing results for the handled detector section on a specified network port. A visualization tool collects this information from the different detector sections and displays them together to provide an overview of the detector system.

For the creation of the detailed analysis report, a modified version of the ROAn analysis software is used, which reads the pre-processed events directly from the DB table. This enables to reduce the analysis time significantly by nearly 80 %, because the majority of the time consuming ROAn processing steps can be skipped. With its step based data processing approach the ROAn analysis software can be quickly adapted to the DB data processing environment. In this configuration the ROAn software only needs to realize the global pattern search, gain map calculation, event calibration, and the calculation of the detailed analysis results.

For the capturing and storing of the full-frame detector data stream, which is an optional recording mode of the SuMo-DAQ system, an additional stand-alone software tool is used. The internal structure of the software is shown in Figure 5.47 and almost identical to the post-processing software, besides the fact that the file I/O operation is realized by only a single processing thread. The tool only stores the full-frame data stream in the FRM file format and does not provide any data processing functionality. With the direct creation of the FRM files instead of RAW files the format conversion step in front of the data analysis by the ROAn software can be eliminated, which reduces the processing time by approximately 33 % (Table 4.1).

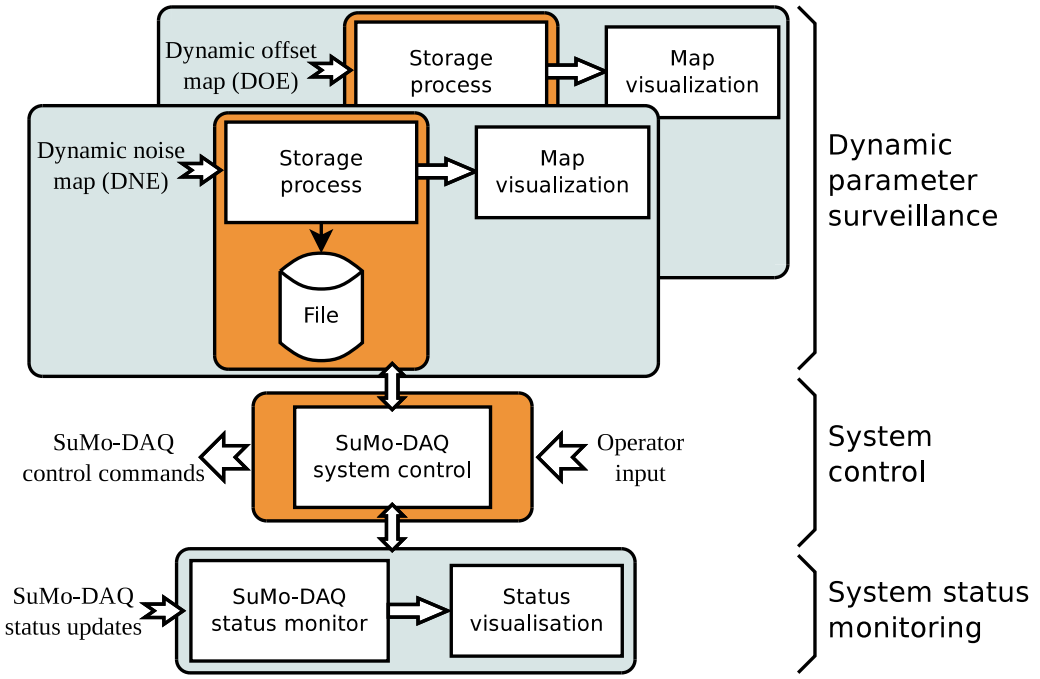


Figure 5.48: Internal structure of the SuMo-DAQ control and real-time monitoring software package. The package consists of three programs: first, the SuMo-DAQ system control, which is the core program and controls all other tools; secondly, the SuMo-DAQ status monitoring tool; thirdly, the dynamic parameter surveillance and recording software, from which two instances are used to monitor the DOE and DNE maps from the detector system.

5.5.2 Control and real-time monitoring software

The control and real-time monitoring software package is a collection of software tools. Separating the software into different stand-alone software tools enables better scalability and provides a higher flexibility. The complete tool collection shown in Figure 5.48 runs on the DCC of the SuMo-DAQ system (Figure 5.2). The main tool is the SuMo-DAQ system control interface, which configures, starts, controls, and terminates all other software tools. The SuMo-DAQ system control tool is the operator interface to communicate with the DPB on-board software and controls the complete SuMo-DAQ data processing and acquisition system. The monitoring tool for the system status enables the observation of the SuMo-DAQ data processing status and major DPB parameters such as the number of processed frames, FPGA temperature, system voltages, SuMo-DAQ hardware processing settings, and so on. For the surveillance of the dynamically calculated offset and noise parameters on the DPB a separate software tool from the package is used. This storage and visualization tool is executed twice on the DCC with different parameter configurations to handle the offset as well as the noise map update stream. The performance requirements to handle these streams are small and the low data rate enables monitoring and continuous storing of the maps even during long-term measurements. The recorded maps and parameters are important for the analysis and evaluation of the SuMo-DAQ data processing performance and stability.

5.5.3 SuMo-DAQ offline data analysis software

For the detailed offline analysis of the recorded data sets an adapted version of the ROAn (Subsection 3) software is used, which works directly on the DB tables, where the captured events from the detector system are stored in. This enables the decoupling of the data processing from the data stream to avoid the hard real-time processing requirement. Additionally, this

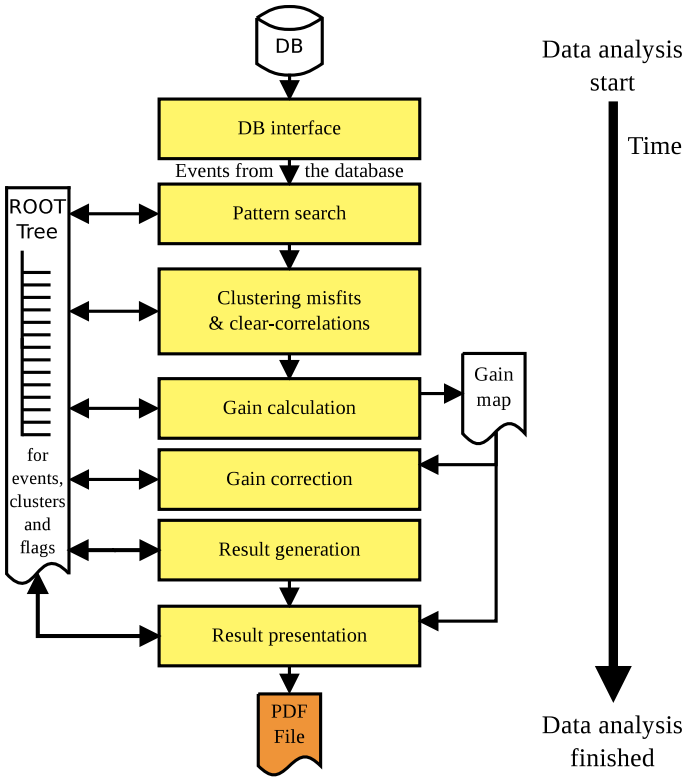


Figure 5.49: Processing sequence of the adapted ROAn version to calculate the detailed analysis report for a measurement data set captured with the SuMo-DAQ system. The direct DB interface and the SuMo-DAQ pre-processing reduce the number of processing steps and the analysis time significantly by approximately 80 %.

makes the data conversion step, which is necessary in the old PCI-based DAQ system, obsolete. Figure 5.49 shows the processing steps and their sequence. With the step-based approach, the adaptation to the new event data input from the DB and the modified processing configuration could be realized without changes in the existing ROAn processing components. Due to the data pre-processing and significant degree of data reduction by the SuMo-DAQ system, the required processing power and steps are considerably reduced compared to the ROAn offline configuration (Figure 3.2). The new DB data input interface, which allows to obtain the extracted events directly, reduces the necessary processing power for the data analysis task significantly and makes the report generation much faster. Based on the measured step processing times shown in Table 4.1, the reduction is approximately 80 % compared to the standard full-frame analysis procedure. The complete structure of a SuMo-DAQ system with the ROAn data analysis software is depicted in Figure 5.50.

In large system configurations the detailed offline analysis report is generated on a dedicated **d**ata **a**nalysis **c**omputer (DAC) to avoid any interference with the SuMo-DAQ data capturing part. For system sizes currently operated with up to 24 MPixels/s, all tasks from the three computer systems can be aggregated on a single computer system. During the TANDEM irradiation campaign, the detailed offline analysis reports for the measurements were created simultaneously to the data recording on the same DB and computer system without performance limitations.

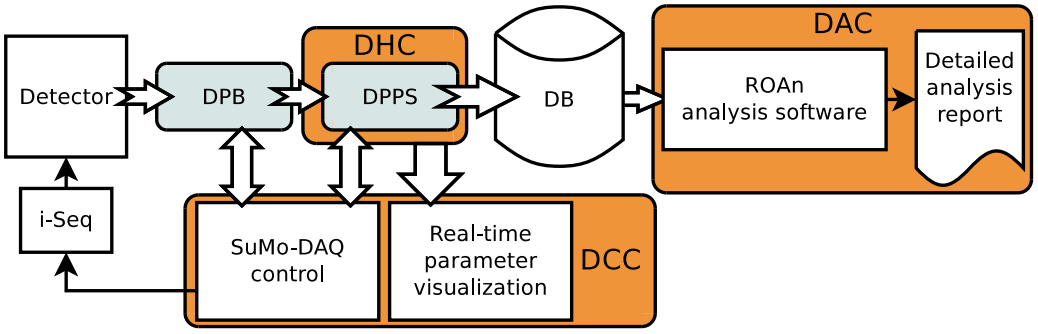


Figure 5.50: The complete SuMo-DAQ system structure including the detailed offline analysis part realized by the adapted ROAn software version on the **d**ata **a**nalysis **c**omputer (DAC). The DPB does the hardware pre-processing and hands the data stream over to the DPPS for the software post-processing. The results are stored in the DB tables to enable more detailed offline analysis cycles on the measurement if necessary. The real-time visualization and system control is realized in software on the DCC.

5.6 Scaling capabilities of the SuMo-DAQ system

For performance improvements and the scaling of the SuMo-DAQ system two levels have to be distinguished. First, the scaling to more analog input channels with a single DPB system and, secondly, the concurrent utilization of multiple DPB systems in larger SuMo-DAQ systems.

5.6.1 SuMo-DAQ scaling with a single DPB system

A system with a single DPB board is usable for mid-size detector systems. Each DPB is currently equipped with two ADC channels, which can synchronously be operated. In such a single DPB system, the data from the entire detector are available on the DPB board and can be transmitted in a single UDP packet to the DHC. The hardware processing of the different ADC data streams on the DPB can easily be realized by multiple independently working processing chains in the FPGA.

Alternatively, the different ADC data streams can also be merged together into one single SuMo-Stream and be processed consecutively in a single SuMo-DAQ hardware processing chain. The advantage of merging multiple SuMo-Streams into a single processing chain is the much better hardware resource efficiency compared to the approach with multiple processing chain instances on the FPGA. The SuMo-DAQ processing chain performance of $\sim$ 111.8 MPixels/s at a 125 MHz clock frequency (Subsection 5.4.8) allows for operating up to 11 ADC channels concurrently with a single processing chain at the typical 12 MSampels/s for an ADC channel in a DEPFET detector system. The ADC adapter cards currently available, however, only provide two analog channels and have been the main limitation factor for this type of the system scaling so far. In the current ADC configuration, 24 MPixels/s can be captured with the standard sampling rate. The rate can be increased to up to 40 MPixels/s; the spectroscopic performance, however, will be reduced. The maximum frame rate for a specific detector matrix size can be calculated by Equation 5.36 for a single DPB system in the first hardware generation with two ADC channels. The maximum achievable frame rate with a single SuMo-DAQ hardware processing chain under full load can be calculated by Equation 5.37 for a specific detector matrix size. A matrix with $64 * 64$ pixels, for example, can therewith, up to now, be operated with two ADC channels at a frame rate of $\sim 9,765$ fps and at a frame rate of $\sim 27,294$ fps if the SuMo-DAQ processing chain is fully loaded.

$$FrameRate_{2CH} = \frac{40\,MPixels/s}{\#DetectorPixels} \tag{5.36}$$

$$FrameRate_{Full} = \frac{111.8\,MPixels/s}{\#DetectorPixels} \tag{5.37}$$

To operate a detector system with two analog input channels, such as the MIXS detector system, the processing chain only has to be extended by a second ADC_Interface with the corresponding Old_ADC_Sequence_Adapter and an ADC_Data_Stream_Merger PCore. The standard two-channel SuMo-DAQ hardware configuration is depicted in Figure 5.51. From the two ADC input streams, the ADC_Data_Stream_Merger generates a single data stream, where the detector rows appear in an alternating manner. This has no influence on the SuMo-DAQ data processing itself; it only changes the pixel order inside the data stream. To hide this from the subsequent software post-processing system on the DHC, the event transformation PCore (Subsection 5.4.6) has to be reconfigured for the particular detector readout and processing scheme. The hardware resource requirements for the two-channel implementation of the SuMo-DAQ system is shown in Table 5.17. The small increase in used hardware resources from the single to the two-channel configuration demonstrates the SuMo-DAQ scaling efficiency (reference data in Table 5.17 for a single-channel system is taken from Table 5.16). This scaling efficiency makes it possible to use the full processing performance of the SuMo-DAQ chain on the FPGA either by speeding up the ADC or increasing the number of concurrently operated ADC channels. The estimated hardware resource requirements for a ten ADC channel SuMo-DAQ system are depicted in Table 5.18. Even such a large processing system fits with the SuMo-DAQ concept on a mid-size FPGA.

Part	Slice registers	LUTs	BRAMs	Max. implementation clock frequency [MHz]
SuMo-DAQ processing chain for two analog channels	10,679 (16.7 %)	9,108 (14.2 %)	46 (20.2 %)	137.0
Full SuMo-DAQ system on the FPGA for two analog channels	31,022 (48.5 %)	28,749 (44.9 %)	121 (53.1 %)	125.5
Full SuMo-DAQ system on the FPGA for one analog channel (reference data taken from Table 5.16)	29,629 (46.3 %)	28,088 (43.9 %)	114 (50.0 %)	125.5

Table 5.17: Hardware resource requirements of a two-analog channel SuMo-DAQ system on a Xilinx Virtex 5 FPGA. The synthesis and implementation was made with the XST version 13.1. The full SuMo-DAQ system on the FPGA includes, besides the hardware processing chain, all further necessary system components (MPMC, PLB, and so on) to operate the SuMo-DAQ system on the DPB. The values in parentheses are referred to the Virtex5-LX100T FPGA from Xilinx.

Part	Slice registers	LUTs	BRAMs	Max. implementation clock frequency [MHz]
Full SuMo-DAQ system on the FPGA for ten analog channels	42,166 (65.9 %)	34,037 (53.2 %)	177 (77.6 %)	125.5

Table 5.18: Estimation of the SuMo-DAQ hardware resource requirements for the concurrent processing of ten ADC channels on a Xilinx Virtex 5 FPGA. The values in parentheses are referred to the Virtex5-LX100T FPGA from Xilinx.

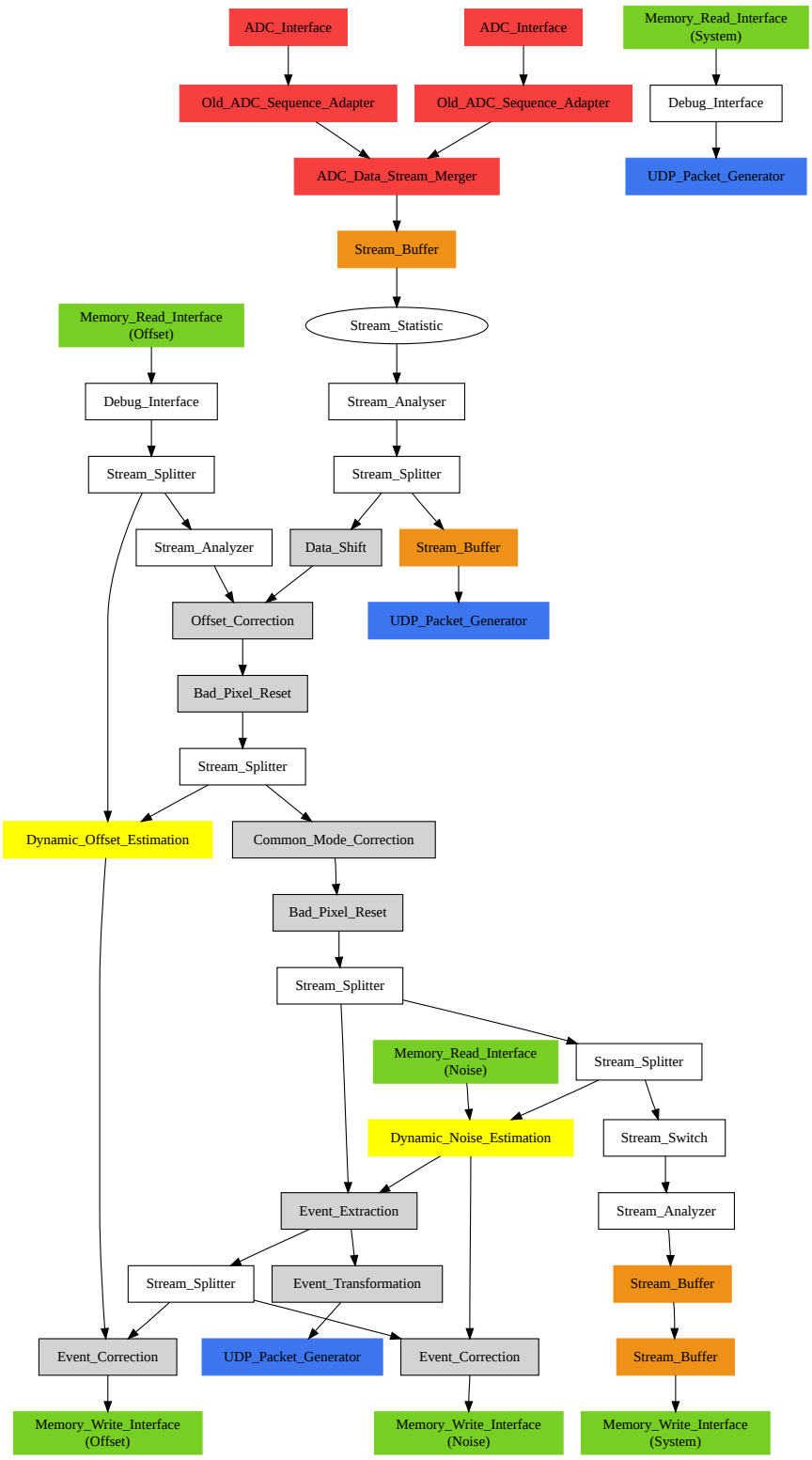


Figure 5.51: Data flow inside the SuMo-DAQ data processing chain on the FPGA for the standard two-channel configuration. The block colors correspond to the functional groups to which they belong. The three blocks in the upper right corner, which are not connected to the processing chain, are steered by the control software and used for fast data transfers to the DHC system.

5.6.2 SuMo-DAQ scaling with multiple DPB systems

For even larger detector systems, the analog channels can be distributed over multiple DPB systems. The SuMo-DAQ system structure for a configuration with multiple DPB systems is depicted in Figure 5.52.

The key task for the operation of large-scale SuMo-DAQ systems is the synchronous operation of all DPB and DPPS sub-systems. If the different DPB systems run asynchronously, the FrameID numbers generated on each DPB sub-system do no longer refer to the same detector frame, which would lead to problems during the offline data analysis. The DPPS systems are synchronized via the corresponding DPB system, because they only process the input data stream from the dedicated DPB system and are idle if no data are transmitted. For a synchronous operation of all DPB systems the timing bus signals generated by the i-Seq have to be distributed to each DPB.

In such a configuration, each DPB system covers only a section of the detector and processes the data streams independently from other detector sections. The SuMo-DAQ hardware processing concept has been designed to enable this parallel scaling approach without a direct communication between the different DPB systems. This simplifies the SuMo-DAQ system structure for large detector systems and allows for a better scalability. In this configuration the complete detector information from the different sections is assembled in software. The use of a DB to store the various event data streams (Subsection 5.1.6) makes the back-end synchronization easier and allows for compensating variations in the processing delay. The data stream from each of the DPB systems is post-processed by a dedicated DPPS system and stored in its own DB table. This separation allows for distinguishing the events from the different DPB systems for the data analysis. Utilizing multiple tables, furthermore, improves the DB performance because of the reduced number of table locks for write operations and enables to distribute the data storage over multiple computer systems if necessary to avoid storage bottlenecks in large-scaled SuMo-DAQ systems. The relative event position inside the detector section handled by the particular DPB system is stored in the DB tables. This avoids an individual adaptation of each DPB system for processing a specific detector section. Furthermore, the relative X/Y position encoding requires less bits in the data stream than the global coordinates, which allows saving bandwidth and storage space. In the SuMo-DAQ system used for the MIXS detector system, a transformation from relative to global coordinates was included directly on the DPB system to minimize the necessary processing power on the DHC system. The global merging and clustering of the events from the different detector sections has to be made by ROAn software before the detailed data analysis results can be calculated. The detailed offline analysis report is generated with an adapted version of the ROAn software on the **d**ata **a**nalysis **c**omputer (DAC). Dependent on the system performance requirements, the functionality of the DHC, DCC and DAC systems can be aggregated on a single computer system. For the systems operated until now with up to 24 MPixels/s, the aggregation on a single computer was possible without performance limitations.

5.7 Super-Modular-DAQ system realization for the evaluation measurements

For the operation of the MIXS detector system, the two analog input channels were operated by a single DPB system. This single DPB scaling option presented in Subsection 5.6.1 enables a compact system configuration and requires for the operation of the MIXS detector systems only a single DAQ computer system to run all software tasks as well as the DB for the event storage. Figure 5.53 depicts the complete two-channel configuration used for all the evaluation measurements as well as for the ESA qualification of the MIXS detector modules in the long-term irradiation campaign at the TANDEM facility. The i-Seq sequencer on the X-Board, which generates the timing-bus signals and controls the detector system operation, is controlled from a dedicated computer system by the same software already used in the old DAQ system.

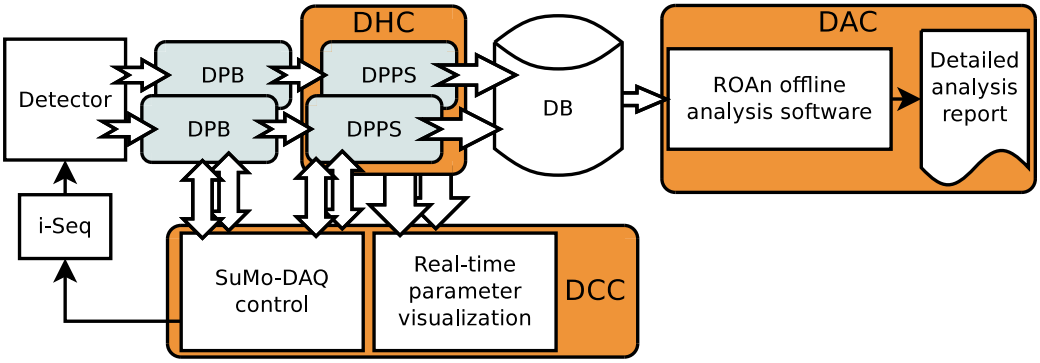


Figure 5.52: The SuMo-DAQ system structure for a large-scale configuration, where the analog channels are distributed over multiple DPB systems. Each DPB realizes the hardware pre-processing for a certain detector section and is linked to a dedicated software post-processing DPPS instance. The global SuMo-DAQ system control and real-time monitoring software package is utilized on the DCC as a system operator interface. The detailed offline analysis report is generated with a modified version of the ROAn software on the **d**ata **a**nalysis **c**omputer (DAC).

This enables an independent DAQ and detector system operation and allows switching between the old PCI-based DAQ system and the new SuMo-DAQ system by simply reconnecting the analog channels and the timing bus to the particular DAQ system. Fast exchanges of the old and new DAQ system without changing anything else in the system makes precise comparison measurements possible under equal system conditions. This separation of the DAQ system and the detector control system provides, furthermore, a higher level of operation safety during the test phase. In case of a serious failure in the DHC, DCC or DAC software the detector system is still controllable and could be shutdown in a safe way; if necessary independently from the DAQ system state. After the finishing of the SuMo-DAQ evaluation and the test phase, the detector system control software on the **d**etector **s**ystem **c**ontrol **c**omputer (DSCC) will be fully integrated into the DCC system with the launch of the next SuMo-DAQ hardware generation. The remaining SuMo-DAQ system configuration is realized exactly as described before in this chapter. The measurement and evaluation results presented in Chapter 6 are, therefore, fully representative and comparable to the achievable SuMo-DAQ system performance in a productive environment.

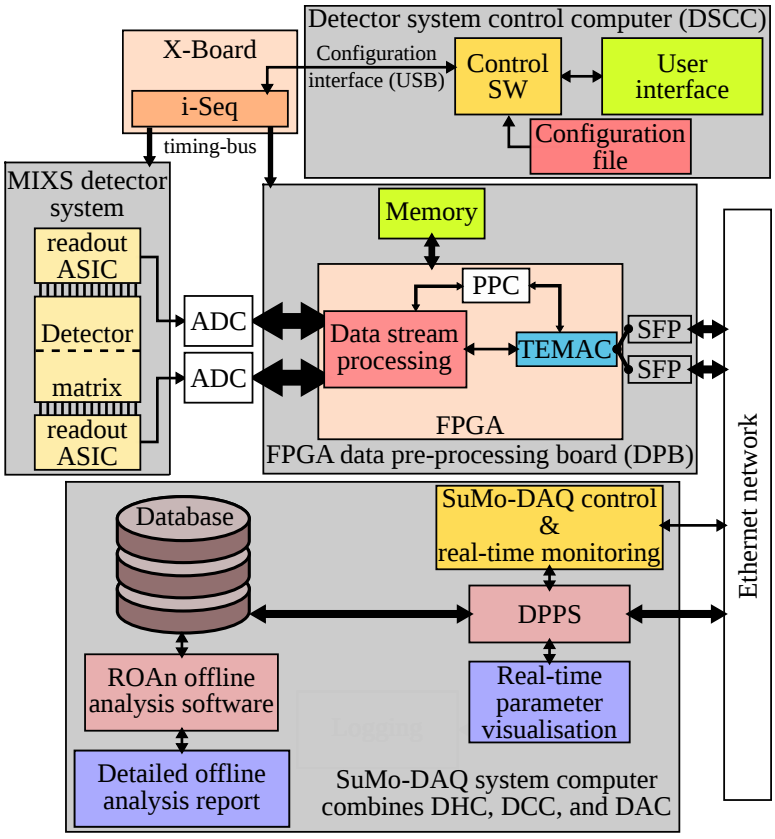


Figure 5.53: SuMo-DAQ system architecture for the MIXS detector system used during the irradiation test campaign at the TANDEM facility. In this configuration the DAQ system is completely decoupled from the detector system control to enable a fast exchange between the old and new DAQ system.

Chapter 6

Super-Modular-DAQ system measurement and evaluation results

In this chapter, the SuMo-DAQ measurement and evaluation results with the MIXS detector systems are presented. In addition, the results from the SuMo-DAQ hardware-in-the-loop simulations are introduced in Subsection 6.1. The hardware-in-the-loop simulations where made to check and verify the base system and processing functionalities before the first measurements with a real DEPFET detector system. Measurement results with a DEPFET detector system are then discussed in Subsection 6.2, while in Subsection 6.3 the SuMo-DAQ performance under changing environmental conditions is investigated. The results from the long-term irradiation campaign to calibrate the MIXS detector system are finally presented in Subsection 6.4.

6.1 Hardware-in-the-loop simulation and evaluation of the Super-Modular-DAQ system

The hardware-in-the-loop operation mode was crucial to enable a fast check and verification of the SuMo-DAQ system functionality. During the hardware-in-the-loop operation, known data values are transmitted from the simulation control computer to a DPB board, where the data values are processed and finally sent back to the computer. In contrast to the direct processing of ADC values, the processing of known simulation data values enables the precise verification of the calculation results from the SuMo-DAQ processing system. The SuMo-DAQ system behavior can therewith be checked faster than in pure software simulations and directly on the real hardware. With the ability to process recoded data sets as well as synthetically generated data sets, the processing system behavior can be analyzed for any detector input.

Figure 6.1 shows the setup used for the SuMo-DAQ hardware-in-the-loop test. In this configuration the DPB board is used as a data co-processor to analyze the data from a virtual detector system on the SuMo-DAQ hardware. The simulation input data values are transferred from the simulation control computer via Ethernet to the DPB board, where they are stored in the on-board memory. If a data block transfer is completed the data processing of this block on the DPB is initialized by the simulation computer. This software-controlled data processing on the SuMo-DAQ hardware system allows for extracting intermediate processing results, which cannot be accessed during the normal data processing. The ADC interface in the hardware pre-processing chain on the DPB for the hardware-in-the-loop operation is replaced by a memory read interface PCore to load the simulation data values into the processing chain. The events extracted from the simulation data on the DPB system are transmitted exactly in the same way as in the normal SuMo-DAQ operation mode. This also enables to check all other system components and the software tools of the SuMo-DAQ system. Besides the extracted events the simulation computer can request the dynamically calculated offset and noise maps as well

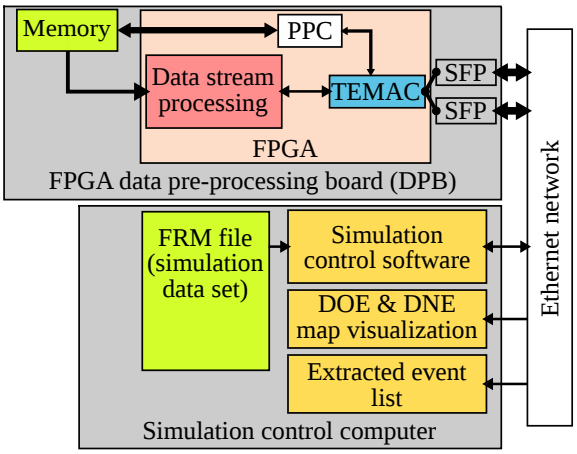


Figure 6.1: Setup for the SuMo-DAQ hardware-in-the-loop simulation. The simulation input data sent from the simulation control computer to the DPB. The data is processed by the SuMo-DAQ hardware processing chain and the results are sent back to the simulation control computer for verification. This enables to test the entire SuMo-DAQ processing system directly on the hardware and provides, furthermore, a higher simulation performance than a pure software simulation.

as the statistical parameters processed by the SuMo-DAQ hardware processing chain. All this information provides a detailed picture of the SuMo-DAQ system condition and behavior during the processing of a simulation data set.

The necessary communication task for the hardware-in-the-loop operation is realized on the DPB in software and runs on the PowerPC CPU. Realizing the communication in software provides a high flexibility and allows for a fast implementation, but leads to a relatively low data transfer rate and limits the simulation performance. However, this operation mode is only used for checking the SuMo-DAQ system with small test data sets and the simulation does not need to perform in real time. The DOE and DNE algorithms require, depending on the used filter parameter, a certain amount of frames after the initialization to reach a steady state. Therefore, a higher frame processing rate would still be an advantage for testing the DOE and DNE algorithms. To enable a faster investigation of the algorithm stability and performance, a block processing mode was added. For this mode a block of multiple frames is transferred and stored in the DPB on-board memory. The block is then processed in a loop without any further data transfer from the simulation control computer to the DPB, which increases the frame processing rate significantly. Stability, convergence speed and behavior of the DOE and DNE algorithms can therewith be investigated and demonstrated at full SuMo-DAQ processing speed. For transmitting the processing results between the DPB and the simulation control computer, the standard SuMo-DAQ communication components are used. This allows for utilizing the full processing performance of the SuMo-DAQ hardware processing chain for the hardware-in-the-loop simulation without limitations.

For the hardware-in-the-loop simulations, various offset and noise maps with characteristic patterns and the full DOE and DNE value range were generated synthetically. The simulation data sets processed with the SuMo-DAQ hardware-in-the-loop setup (Figure 6.1) fully confirmed the software simulation results presented in Subsection 5.3.3 for the DOE and DNE algorithms in all terms. With this achievement, a first test measurement with a real DEPFET detector system was realized in the next step to approve the SuMo-DAQ system with real data sets.

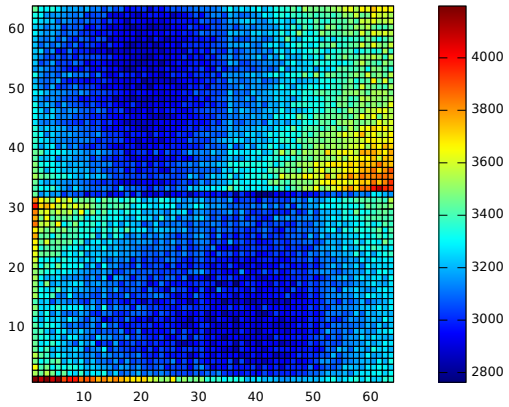
6.2 Measurements with the Super-Modular-DAQ in static environment conditions

After the verification of the SuMo-DAQ hardware processing functionality with the hardware-in-the-loop simulation, the system was connected to the MIXS detector system. The MIXS detector setup was used for evaluating the full performance of the SuMo-DAQ system with the dynamic DNE and DOE algorithms in combination with a two-analog channel detector setup. The SuMo-DAQ system was, furthermore, used in the long-term irradiation test campaign for the ESA qualification of the MIXS flight detector modules. During the preparation of this qualification campaign the SuMo-DAQ system settings were optimized for the MIXS detector setup. The achieved spectroscopic performance in this configuration is, therefore, representative and comparable to other DAQ systems.

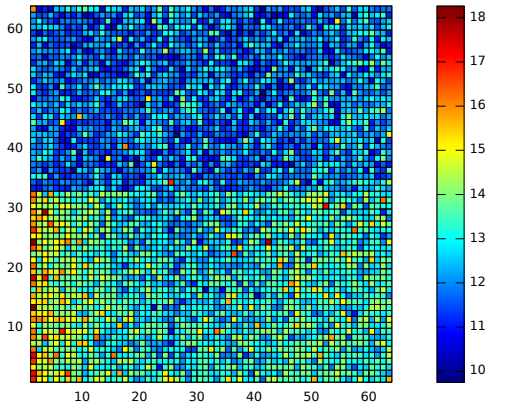
The offset and noise maps calculated dynamically by the DOE and DNE algorithms for a MIXS detector matrix are depicted in Figure 6.2a and 6.2b. The boarder between the north and south hemisphere is clearly visible in the offset and noise map at Row 32. This is a result from the two-side readout scheme of the MIXS detector matrix. The conditions differ between the two matrix hemispheres, mainly because of variations between the two analog readout ASICs. To provide a large offset calculation range (Equation 5.8) and enable the covering of the high variation in the absolute offset value over the matrix, the DOE filter parameter was set to $M = 8$. The DNE filter parameter was set to $D = 256$ in the SuMo-DAQ configuration. This gives, additionally, more headroom for parameter changes during the irradiation campaign of the detector system (Equation 5.31). The dynamically calculated maps are almost identical to the statically calculated maps from the ROAn analysis software. The difference between the SuMo-DAQ and the ROAn maps are shown in Figure 6.2e and 6.2f. For the offset difference, the standard deviation between the two calculation methods is ± 1.48 ADU; for the noise sigma difference, it is ± 0.73 ADU. Both difference maps are depicted in Figure 6.3 and are homogeneous without any characteristic value distribution. This confirms the independence of the calculation precision from the particular absolute value for the DOE and DNE algorithms.

The two difference maps shown are, of course, only snapshots of the situation, because the DOE and DNE algorithms continuously recalculate the parameters. For short reference measurements or measurements in non-changing environmental conditions, the difference between the dynamic and the static maps can be seen as quasi-constant. Figure 6.4 shows the behavior of the dynamically calculated offset and noise values of several pixels as a function of the captured frames. The parameters are extracted approximately once per second directly from the SuMo-DAQ system and are both very stable. For the display, the raw parameter values from the SuMo-DAQ system are transformed into the true offset and noise value by Equation 5.3 and 5.30 with floating point precision. This floating point transformation allows for a more precise analysis of the parameter than rounding of the raw parameter value to the nearest integer value, which is used for the data processing.

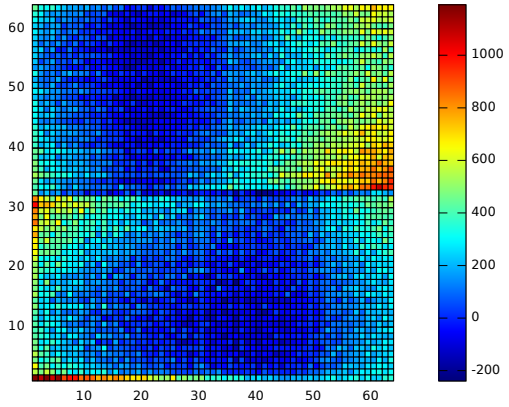
The calibrated spectra from a SuMo-DAQ measurement with a ^{55}Fe source and the MIXS detector matrix prior to the irradiation campaign are depicted in Figure 6.5. The single as well as the all-valid pattern spectra both show a good result with clearly visible and neatly separated Mn-K$_\alpha$, Mn-K$_\beta$, and Si-escape peaks. The low energy background level between $1,000$ eV and $4,000$ eV is low and the noise peak around 0 eV is slim. The optimized tab-delay setting of the SuMo-DAQ system enabled to increase the P/B ratio significantly for the MIXS detector system. For single patterns, a P/B of $1,698.93$ could be achieved with the SuMo-DAQ system, while the old PCI-based DAQ system achieved only a ratio of $1,577.78$. The P/B ratio for all valid patterns achieved with the SuMo-DAQ system was $1,914.49$ and only insignificantly lower than the P/B ratio of $1,956.18$ achieved with the old PCI-based DAQ system.



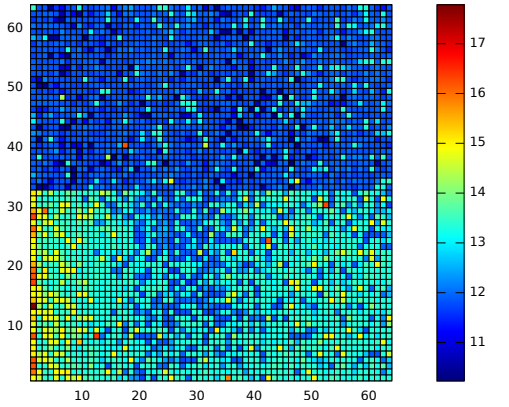
(a) Offset map calculated by the ROAn analysis software.



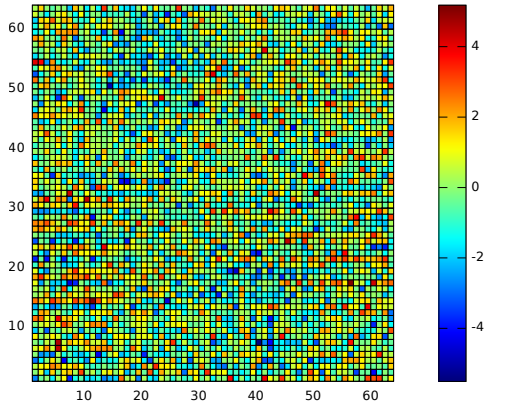
(b) Noise map calculated by the ROAn analysis software.



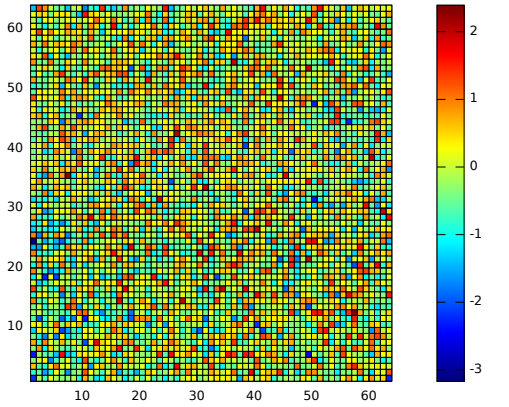
(c) Offset map calculated by the DOE algorithm.



(d) Noise map calculated by the DNE algorithm.

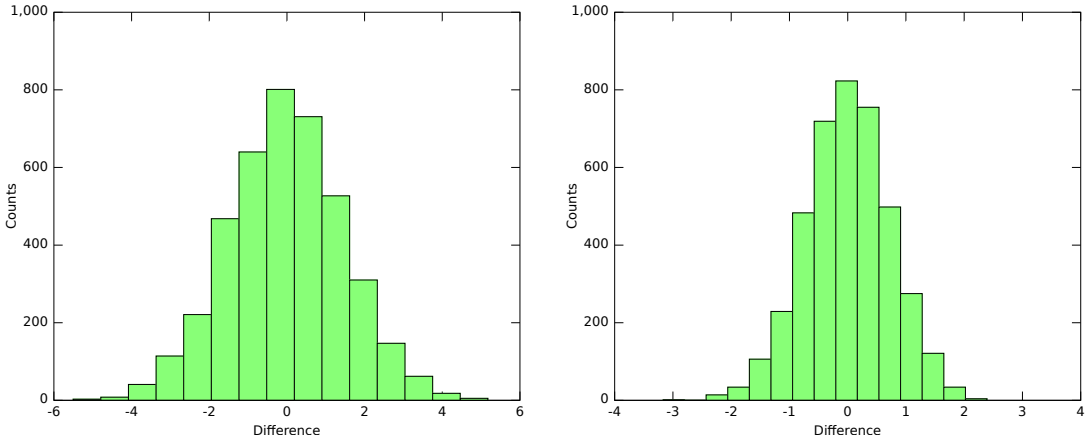


(e) Difference between the offset map calculated by the DOE algorithm and the ROAn analysis software.



(f) Difference between the noise map calculated by the DNE algorithm and the ROAn analysis software.

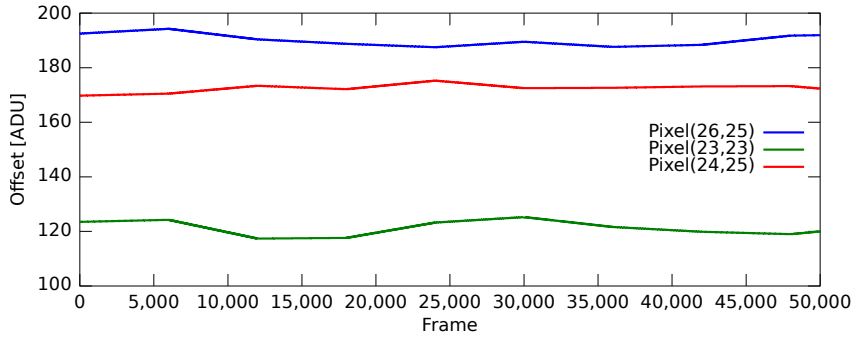
Figure 6.2: Parameter maps dynamically calculated by the SuMo-DAQ system for a MIXS detector module and their differences to the maps statical calculated by the ROAn analysis software. The boarder at Row 32 in both maps results from the two-sided detector matrix readout schemata of MIXS.



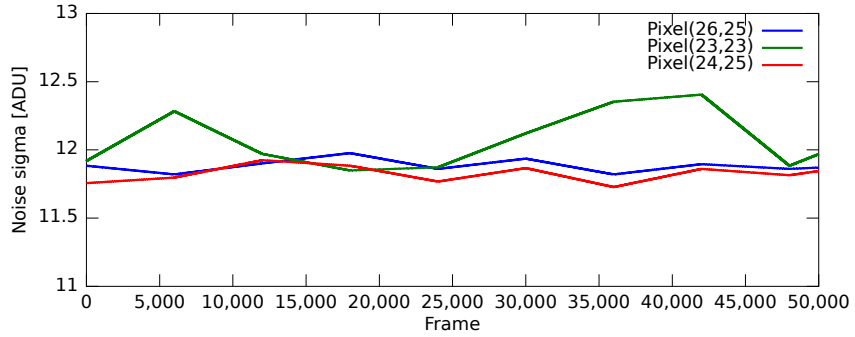
(a) Histogram of the offset differences.

(b) Histogram of the noise differences.

Figure 6.3: Histograms of the differences between the offset and noise values calculated dynamically by the SuMo-DAQ system and the statically calculated values by the ROAn analysis software.

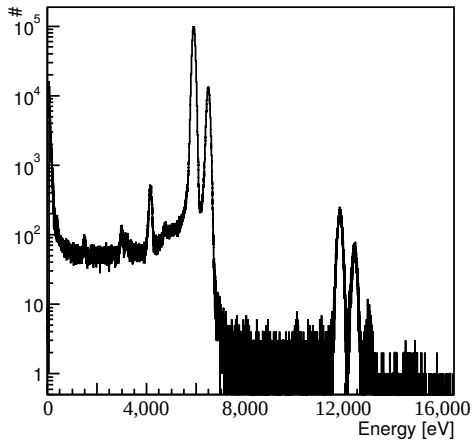


(a) Course of the offset parameter calculated by the DOE algorithm for several pixels.

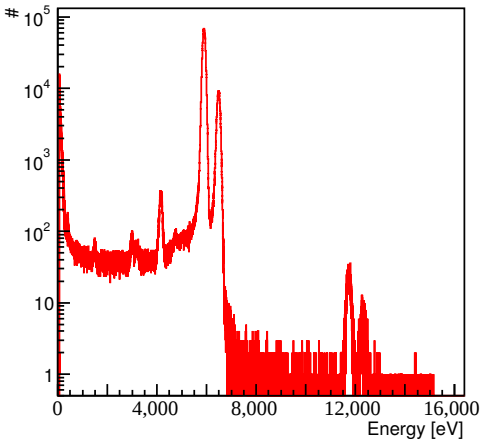


(b) Course of the noise parameter calculated by the DNE algorithm for several pixels.

Figure 6.4: The offset and noise time series for some pixels of the MIXS detector matrix during a measurement in static system conditions. The parameter values are extracted directly from the SuMo-DAQ system and transformed into the true offset and noise value with floating point precision instead of rounding to the nearest integer value.

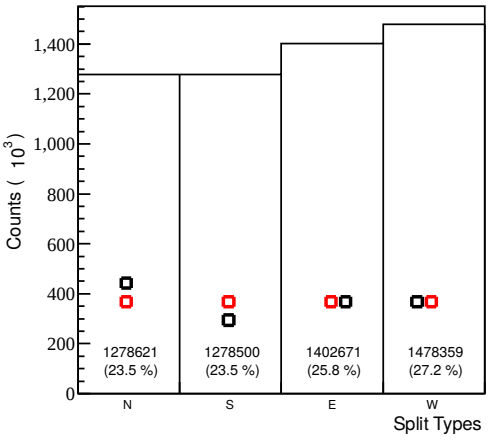


(a) The calibrated spectrum of all valid event pattern types.

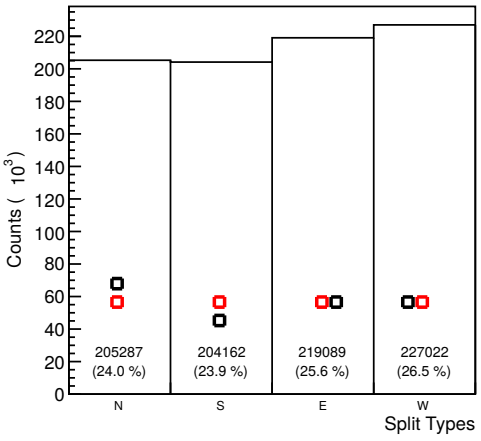


(b) The calibrated spectrum of all single (SNG) events.

Figure 6.5: Calibrated spectra from a SuMo-DAQ measurement with the MIXS detector system. Single (SNG) patterns are with 67.4 % the most common pattern type for the MIXS detector matrix.

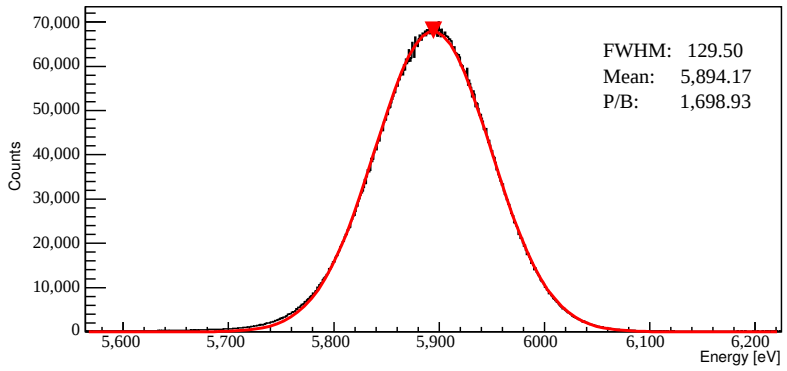


(a) Ratio of DBL pattern for a measurement with the SuMo-DAQ system.

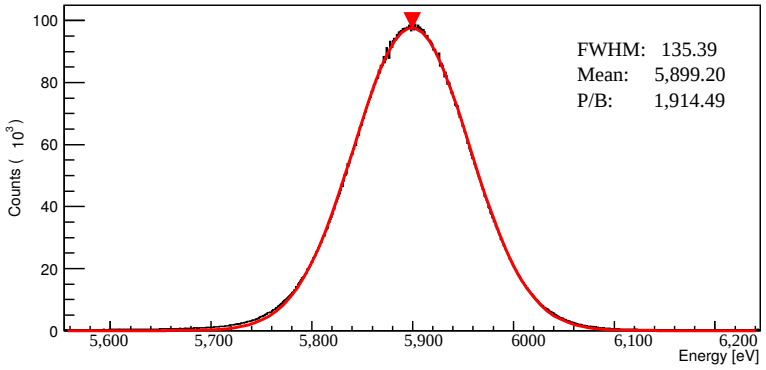


(b) Ratio of DBL pattern for a measurement with the old PCI-based DAQ system.

Figure 6.6: Comparison of the DBL split ratio between the SuMo-DAQ and the old PCI-based DAQ for the MIXS detector system. The nearly uniform pattern distribution indicates an optimal tab delay setting for both DAQ systems. The slightly increased number of East and West splits is a result of the serialized analog readout structure for DEPFET detector systems and similar for both DAQ systems.



(a) Energy resolution for single (SNG) pattern.



(b) Energy resolution for all valid patterns.

Figure 6.7: The energy resolution of the Mn-K_α peak at 5,899 eV for a SuMo-DAQ measurement with the MIXS detector system.

The DBL pattern distribution depicted in Figure 6.6a is nearly uniform, which indicates an optimal tab-delay setting. The slightly increased number of East and West splits compared to North and South splits is a result of the serialized analog readout structure for DEPFET detector systems. The ASTEROID readout ASIC serializes the 64 parallelly operated input channels to a single output (Subsection 2.3.4). Due to the limited output driver strength **i**ntersymbol **i**nterference (ISI) occurs on the transmission line to the digitization stage at the operation speed necessary for MIXS. In the double side readout configuration of MIXS, an ASTEROID is placed on the north as well as on the south side of the matrix. One matrix hemisphere is, therefore, read out from East to West and the other one from West to East. This leads to the increase of both horizontal split combinations over the two vertical split combinations as shown in Figure 6.6a for the SuMo-DAQ system. The asymmetry between the different combinations is almost identical to the old PCI-based DAQ system, which shows a similar distribution characteristic as depicted in Figure 6.6b.

The optimized tab-delay setting not only improves the pattern ratio, it also leads to a better Mn-K_α peak shape. The Gaussian fits to determine the energy resolution achieved are nicely matching the measured Mn-K_α peak for singles as well as for all valid patterns as shown in Figure 6.7. An energy resolution of 129.50 eV FWHM for single patterns and 135.39 eV for all valid patterns was achieved for the Mn-K_α peak. Compared to 130.41 eV FWHM for single patterns and 135.52 eV FWHM for all valid patterns achieved with the old PCI-based DAQ system, both SuMo-DAQ results are in very good agreement.

These results show that the measurement of both DAQ systems in static environmental conditions are mainly Fano-limited. The Fano-limit represents the lower physical limit for the

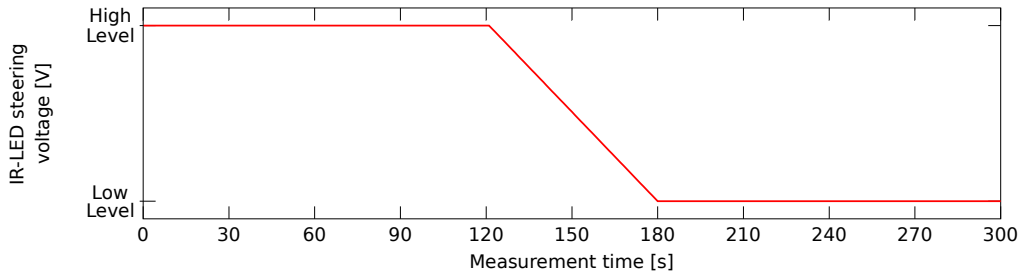


Figure 6.8: Control function of the infrared-LED (IR-LED) voltage for the Super-Modular-DAQ measurements in changing environmental conditions. For the generation of different changes during the various test measurements the high and low levels are adjustable while the measurement time of 300 s was identical for all tests.

achievable energy resolution with a silicon detector. The comparison demonstrates, furthermore, that in static system conditions the dynamic calculation of offset and noise parameters with the DOE and DNE algorithms have no negative impact on the performance. The most significant performance improvement with the SuMo-DAQ system can, however, be expected under changing environmental conditions, where the DOE and DNE algorithms can fully show their advantages.

6.3 Measurements with the Super-Modular-DAQ under changing environment conditions

The measurement results presented in the previous subsection are made in static or quasi-static system conditions. This subsection investigates the SuMo-DAQ performance under changing environmental conditions in more detail by applying synthetically generated condition changes to the system.

Dynamic algorithm evaluation in changing environmental conditions

For the controlled change of the system operation conditions an infrared-LED (IR-LED) was installed inside the vacuum tank. The infrared light generates additional charge in the detector matrix and enables to change the offset and noise level of the detector matrix. To allow for the precise control of the environment condition the IR-LED is driven by a programmable function generator. The programmed voltage function applied to the IR-LED for the generation of the synthetic environmental change is depicted in Figure 6.8. An increased IR-LED voltage leads to a higher pixel offset and simultaneously increases the pixel noise. With the high-to-low voltage difference the size of the change can be selected. To obtain reference data sets for each of the evaluated voltage combinations, additional measurements with a constant IR-LED voltage at the high and low levels are made. These two static reference measurements for a specific combination enable to determine an upper and lower performance limit under the particular environment conditions. For all evaluation measurements subsequently presented, the low level was set to 0 V, while the high level was varied to obtain the desired environmental change. The total measurement time of 300 s was for all evaluation measurements the same.

Figure 6.9 shows the achieved energy resolutions in different environmental change conditions for the old static and the new dynamic SuMo-DAQ parameter calculation methods. To allow for a better interpretation of the results, the achieved energy resolutions are plotted over the mean detector offset change instead of the high-level IR-LED voltage. In addition, the optimum performance limit corresponding to the best achievable energy resolution value is calculated from the measurements in static conditions for each combination, is shown. This

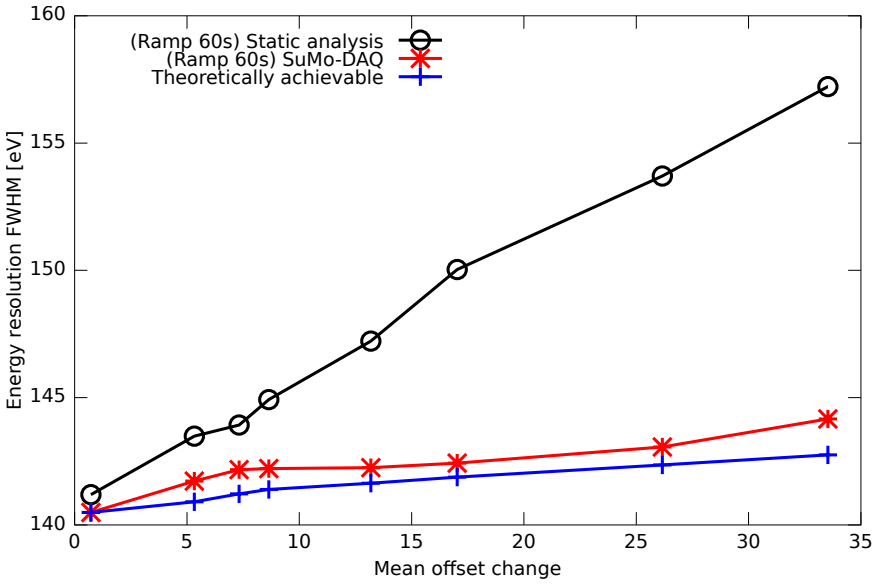


Figure 6.9: Comparison of the achieved energy resolution between the old static and the new dynamic parameter calculation methods in measurements with changing environmental conditions. The energy resolution achieved with the SuMo-DAQ system is close to the theoretical limit for any condition change, while the results calculated with the old static methods becomes disproportionately worse.

curve represents the lower boundary for the achievable energy resolution under the particular environment conditions. With the static offset and noise calculation methods used up to now, the achieved energy resolution becomes disproportionally worse compared to the theoretically achievable resolution with increased change levels. With the dynamical adaptation of the parameters by the DOE and DNE algorithms the SuMo-DAQ system achieved an energy resolution close to the theoretical limit in any change condition. The difference between the theoretical and the SuMo-DAQ result is in agreement with the non-perfect parameter readjustment during the change phase. However, this slight degradation of the energy resolution is acceptable in these extreme conditions and the results are significantly better than with the static calculation methods.

Due to fluctuations in temperature, voltage, and because of matrix degradation, changing measurement conditions occur in every detector system. In these changing environmental conditions the SuMo-DAQ system can demonstrate the full performance advantage over the old PCI-based DAQ system. The simulated condition changes are significantly faster than in real measurement environments, but the expected changes are similar in satellite environments or in long-term measurements.

6.4 SuMo-DAQ for the irradiation campaign of the Mercury Imaging X-ray Spectrometer

For the evaluation of the SuMo-DAQ system in a radiation environment typical for a space mission, the system was used during the long-term irradiation campaign for the ESA qualification of the MIXS camera modules on board the BepiColombo mission. In this campaign, the detector was irradiated with 10 MeV protons, which damage the detector matrix and generate the same degradation effects as during the BepiColombo mission. To simulate the complete BepiColombo mission, a total irradiation time of one week was necessary for each of the two detector modules to achieve the expected $1.5 * 10^{10}$ protons.

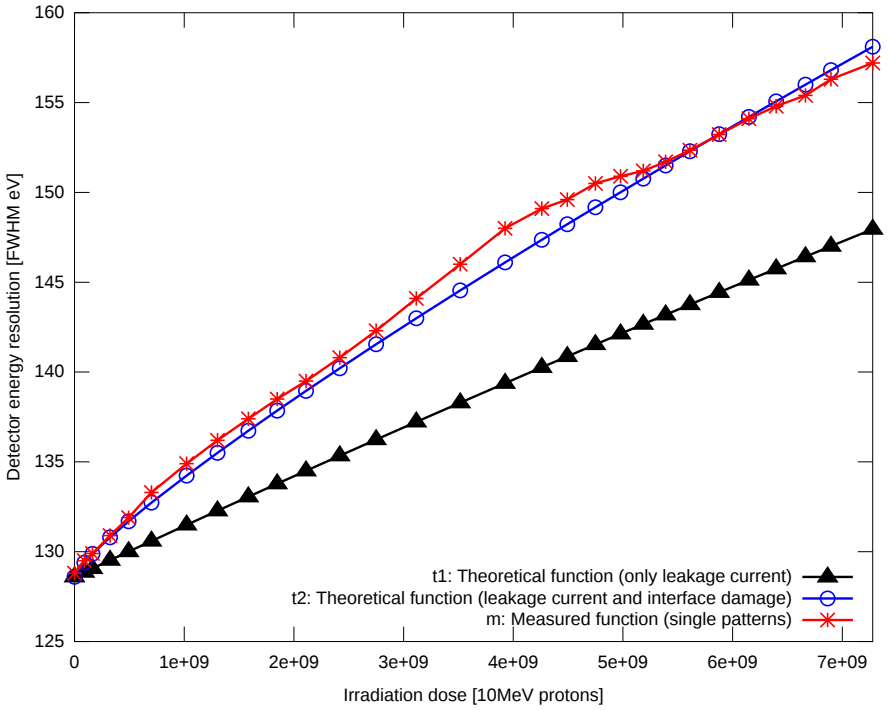
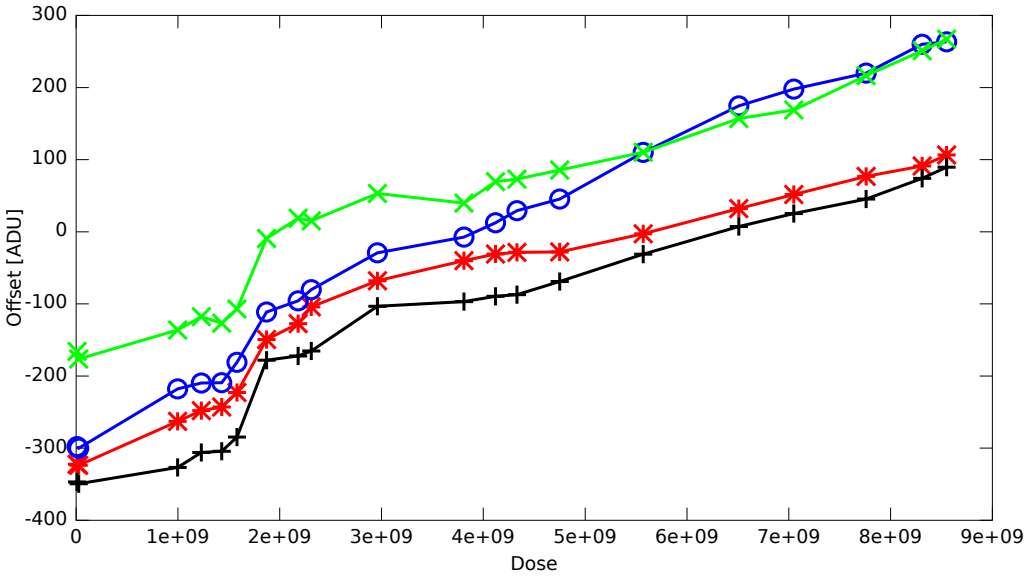


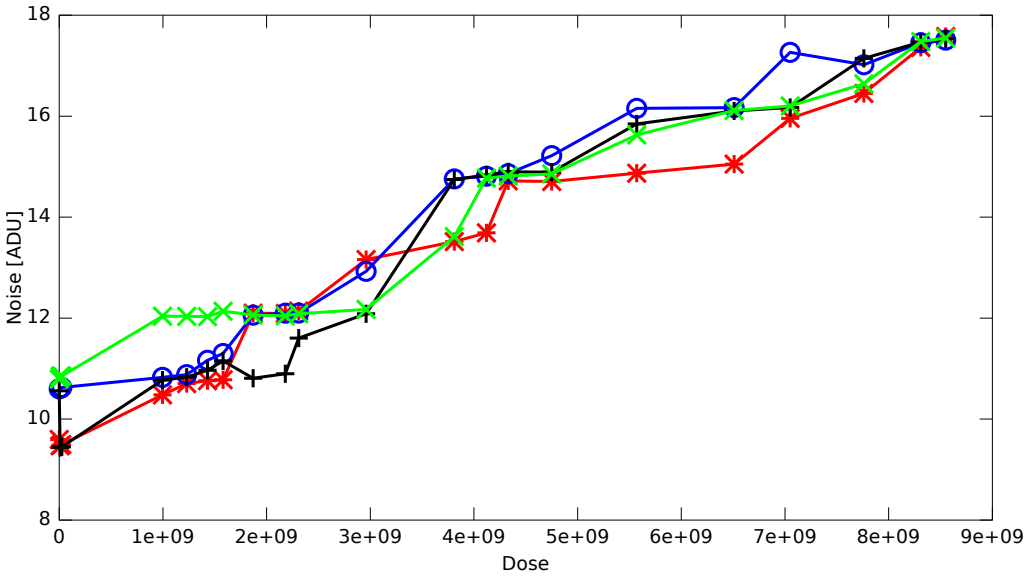
Figure 6.10: Function of the energy resolution in dependency of the irradiation dose for the MIXS detector system. With the increase of the irradiation dose the measured energy resolution capability becomes continuously worse because of the physical damages in the detector matrix. For reference measurements a ^{55}Fe source was used.

The change of the detector conditions by radiation damage allows demonstrating the performance and behavior of the SuMo-DAQ system in realistic mission conditions. Due to the physical damage of the used detector modules the ability for comparison measurements are limited in contrast to the measurements with the IR-LEDs. Therefore, the detector irradiation was interrupted multiple times during the campaign at certain radiation levels to check and verify the energy resolution of the detector system with a reference measurement. The measured energy resolutions for one of the detector modules is depicted in Figure 6.10 as a function of the irradiation dose. The theoretical curves shown in Figure 6.10 are calculated on the base of the results from [73]. The difference between the theoretical function (t1), where only the additional leakage current created by radiation damage is considered and the measured function (m) is generated by interface damages in the silicon detector. Up to now no precise theoretical model is available for silicon interface damages. Usually a rough estimation for these interface damages is made by a $\triangle N \propto Dose^{\frac{2}{3}}$ function [59]. This approximation is considered in the theoretical function (t2).

The stable trend and close relation of the measured energy resolution (m) to the theoretical function (t2) in Figure 6.10 demonstrates the performance of the dynamic system adaptation by the DOE and DNE algorithms in combination with the detector system in real applications. In Figure 6.11, the trend of the offset and noise values of four pixels are plotted as a function of the irradiation dose during the TANDEM campaign. All four pixels show a similar behavior and, as expected, a nearly linear increase of the offset and noise values over the irradiation dose. The irradiation campaign was a good test environment for the DOE and DNE algorithm because of continuous and realistic changes in the detector parameters due to the radiation damage. The reliable adaptation of the data processing to the detector is a key feature for autonomous system operation.



(a) Offset trend of four pixels as a function of the irradiation dose during the TANDEM irradiation campaign.



(b) Noise trend of four pixels as a function of the irradiation dose during the TANDEM irradiation campaign.

Figure 6.11: Offset and noise trend of four pixels as a function of the irradiation dose during the TANDEM irradiation campaign. The pixels show a similar behavior and, as expected, a nearly linear increase over the irradiation dose.

The system stability and reliability of the SuMo-DAQ could also be demonstrated during the two weeks of continuous measurements in the long-term campaign. The SuMo-DAQ system was operated without any incident or system failure during the entire measurement campaign. In addition, the significant data reduction achieved with the real-time data processing by the SuMo-DAQ system could be shown. The MIXS detector is read out with 6,000 frames per second and has 64x64 pixels, which are digitized with an 14-bit ADC. The raw data rate produced by the detector system is, therefore, approximately $\frac{64*64*2*6{,}000+FrameHeader}{1{,}024^2} = 47.1$ MB/s. For the complete 128 hours of recorded measurement during the irradiation campaign roughly 20.8 TB of data would be required to be processed and stored. Due to the real-time data analysis, where only pixels hit by a proton are stored instead of full frames, the amount of data required to be stored could be reduced to approximately 0.567 TB. During the entire irradiation campaign, a mean number of ∅226 Bytes was necessary to store a frame with the SuMo-DAQ system. Even with the high detector occupancy during the irradiation campaign this is a substantial reduction compared to the ∅8,259 Bytes/Frame required with the old full frame DAQ system. Under these conditions, the achieved data reduction factor of 36.5 without removing relevant information for further data analysis is significant. This is a major advantage for all continuously operated detector systems in accelerator facilities, where large quantities of data storage space have been necessary up to now for the operation as well as on satellites, where high-speed downlinks to the ground station would otherwise be required.

6.5 Measurement result summary

The SuMo-DAQ system could demonstrate an excellent performance throughout the evaluation and reference measurements with different detector systems and environmental conditions. In static measurement environments the achieved energy resolution could be slightly improved compared to the old PCI-based DAQ system. The full potential and performance advantage of the SuMo-DAQ system compared to the old PCI-based DAQ system was demonstrated under changing measurement conditions. Such changes occur in many applications due to voltage or temperature drifts, disturbing IR-light sources or radiation damages of the detector matrix. In simulated environmental changes, the achieved measurement performance was always close to the theoretical optimal performance value and much better than the results from the old PCI-based DAQ system. Furthermore, the dynamic real-time parameter adaptation enables the autonomous operation of the detector system with the SuMo-DAQ. This makes the concept also interesting for satellite applications, where autonomous system operation is a key feature. With the real-time data processing capability of the SuMo-DAQ system, the produced output data rate and the necessary storage capacity could also be reduced significantly without any performance reduction. This was demonstrated during the long-term irradiation test campaign for the MIXS camera modules on board the BepiColombo mission, where a data reduction factor of 36.5 was achieved with the SuMo-DAQ system in comparison to the old PCI-based DAQ system.

Chapter 7

Conclusions and outlook

The **Su**per-**Mo**dular-DAQ (SuMo-DAQ) concept developed during this thesis enables the real-time data handling of large and high-speed X-Ray detector systems. With the two novel **D**ynamic **O**ffset **E**stimation (DOE) and **D**ynamic **N**oise **E**stimation (DNE) algorithms the SuMo-DAQ real-time data processing system is able to automatically and continuously adapt the processing optimally to the connected detector system.

The SuMo-DAQ system uses a two-level processing architecture to enable a simple scalability and real-time data processing capability. The first level is the data pre-processing of the complete detector output in hardware on FPGA-based **d**ata pre-**p**rocessing **b**oards (DPBs). This hardware pre-processing enables the dynamic system adaptation in real time. Additionally, this hardware pre-processing allows for the significant reduction of the output data rate and the reduction of the required processing power for the second processing level. The second level is the software data post-processing of the reduced data stream. The software post-processing realizes the high level analysis functionalities of the SuMo-DAQ system.

The next generation of DEPFET detector systems will utilize multiple concurrently operated read-out channels to achieve the desired frame rates. For the proposed **A**dvanced **T**elescope for **H**igh **EN**ergy **A**strophysics (ATHENA+) with a detector matrix size of 640×640 pixels, up to 16 channels have to be operated simultaneously. The operation of such large X-ray detectors with the SuMo-DAQ system can be realized by concurrent utilization of multiple DPB systems. Since no communication between the different DPB systems is needed, adding new input channels to scale the system is only a minor change. The data streams from the different DPBs are collected, merged, and analyzed by the software post-processing system of the SuMo-DAQ.

The two-level processing architecture allows for an optimal distribution of the various X-ray data processing tasks between hardware and software. Hardware and software co-design increases the system complexity, but allows using the advantages from both fields. Highly optimized hardware structures enable high-speed data processing and high data throughputs, while very complex algorithm parts can be implemented more efficiently in software. The SuMo-DAQ concept enables to fulfill the requirements on a DAQ system for the next generation of high-speed DEPFET detector systems:

- **Real-time data processing** capability has to be provided by the system. This includes even large and fast detector systems with output rates of up to $100 * 10^6$ pixels per second.

- **High flexibility** of the system is needed to enable the operation of a wide variety of different detector types and systems. This is a key feature for laboratory operation during the detector development, where different detector systems have to be evaluated.

- **Scalability** to allow also for the operation of the next generation of detector systems with multiple output channels and composite detector modules.

- **Data reduction** without loss of relevant information for more detailed data analysis tasks.
- **Autonomous system operation and self-adapting algorithms** to enable a stand-alone operation of the system on board of satellites and to allow for precise measurements under changing conditions.

With the presented and evaluated realization of the SuMo-DAQ system the advantages and performance of the concept could be demonstrated in combination with different detector systems and system operation conditions. The direct performance comparison between the old PCI-based DAQ system and the SuMo-DAQ system where made during the first measurements in static environmental conditions. With regard to the two used detector systems, the SuMo-DAQ results with the dynamic processing adaptation were better than with the old PCI-based DAQ system even in static conditions. The full performance and main advantage over the old PCI-based DAQ system could, however, be shown in changing environmental conditions. These changes, for example, occur during normal long-term measurements, in satellite environments and due to radiation damages of the detector matrix. Under these realistic environmental conditions the SuMo-DAQ performance has always been close to the theoretical achievable performance and significantly better than the old DAQ system.

The SuMo-DAQ concept with the hardware pre-processing and the two dynamic adaptation algorithms is, furthermore, directly usable in satellite applications. The dynamic processing adaptation in real time and the ability of autonomous system operation in changing environmental conditions provide significant advantages on satellites. The high-level analysis tasks in the software post-processing have, however, to be adapted to the specific mission requirements. These tasks are highly dependent on the mission tasks and objectives and must be adapted to the storage and data transmission constraints of the satellite. Due to the modularity of the data processing in the SuMo-DAQ concept further application-specific processing steps can directly be added to the SuMo-DAQ system.

A new hardware platform is currently being developed for the SuMo-DAQ system at the Max Planck Institute for Extraterrestrial Physics. The new hardware is based on Spartan-6 FPGAs for the pre-processing and combines up to four ADC channels on each DPB pre-processing system. In the new version, only the hardware components are optimized and the cheaper Spartan-6 FPGA platform is used to enable a cost-efficient building of scaled SuMo-DAQ systems. The SuMo-DAQ concept presented in this thesis still remains the same with the new hardware.

This new SuMo-DAQ hardware will be used as a base platform for the development of the **w**ide-**f**ield **i**mager (WFI) on board the **A**dvanced **T**elescope for **H**igh **EN**ergy **A**strophysics (ATHENA+) mission. ATHENA+ is likely to be selected for the L2 mission – the second 'Large-class' mission in ESA's Cosmic Vision [22] Science Programme – with a launch date foreseen for 2028. By promoting the SuMo-DAQ concept during the ATHENA+ development phase, there is a good chance that the concept is also utilized for flight electronic due to the excellent performance and significant advantages of the SuMo-DAQ concept.

Appendix A

Appendix

A.1 SuMo-Stream I/O interface

For the interconnection of the different processing units on the FPGA the SuMo-Stream protocol is used. The SuMo-Stream protocol is a slightly modified version of the **A**dvanced e**X**tensible **I**nterface (AXI) [102, 7] stream protocol. AXI is part of the **A**dvanced **M**icrocontroller **B**us **A**rchitecture (AMBA®) from ARM®. The second generation of AXI was released in 2010 in AMBA 4.0 standard as AXI4. Three types of AXI4 interfaces are defined in the standard. The AXI4-Stream [6] protocol has been specially designed for high-speed streaming applications and is optimal for the X-ray data stream processing application in the SuMo-DAQ. The simplicity of the AXI4-Stream protocol enables an implementation with very low hardware resource requirements.

Xilinx fully supports the protocol for their newest FPGA. The first version of the FPGA processing board used for the SuMo-DAQ system is equipped with a Virtex-5 FPGA. On the Virtex-5 the AXI protocol is, unfortunately, not supported by Xilinx. Nevertheless, custom implementation and utilizing of the AXI protocol is possible without any limitation.

The AXI4-Stream protocol exclusively supports data transfers without an address phase and allows for an unlimited data burst size. This is exactly what is needed for the on-chip interconnection of the various processing units. In case of the DEPFET X-ray data processing each pixel value is encoded in 16 bits. For the processing of the detector data stream, additional pixel position information is necessary. To allow for a throughput of one pixel per clock cycle, this information is encoded in two auxiliary data lines. Therefore, the pixel data SuMo-Stream has a width of 18 bits and deviates from the original specification of the AXI-Stream. This data bus width allows for a better hardware resource utilization, because the **B**lock **RAM** (BRAM) [104] on the FPGA is also 18 bits wide. A standard-compliant data bus width would lead to unnecessary interconnection signals in the hardware and waste of valuable BRAM resources.

Figure A.1 shows an example for the interconnection of two processing units with the pixel data SuMo-Stream interface. The bus signals have the same names and functions as in the standard. A description of the individual signal function is given in Table A.1 and a timing diagram of the communication is shown in Figure A.2.

The event data SuMo-Stream is an extension of the described pixel data SuMo-Stream used for the pixel data transfer. This version is currently used after the event extraction, where the pixel position on the detector can no longer be determined from the position inside the data stream. For the data transfer on the event data SuMo-Stream the same protocol is used as in the pixel data SuMo-Stream; only the width of the TData signal is increased by 16 bits. The meaning of the individual bus signals of the event data SuMo-Stream are described in Table A.2.

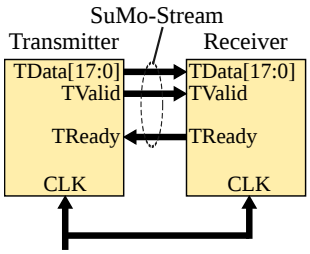


Figure A.1: Processing unit interconnection with the SuMo-Stream interface.

Signal	Bits	Description
TValid	1	Indicates that the data values provided on the TData lines are valid
TData[17]	1	Flag for the last pixel of a detector frame.
TData[16]	1	Flag for the last pixel of a detector row.
TData[15:0]	16	Pixel data value
TReady	1	Indicates that the receiver can accept data in the current clock cycle

Table A.1: Signal summary of the pixel data SuMo-Stream version used for the hardware unit interconnection in the SuMo-DAQ.

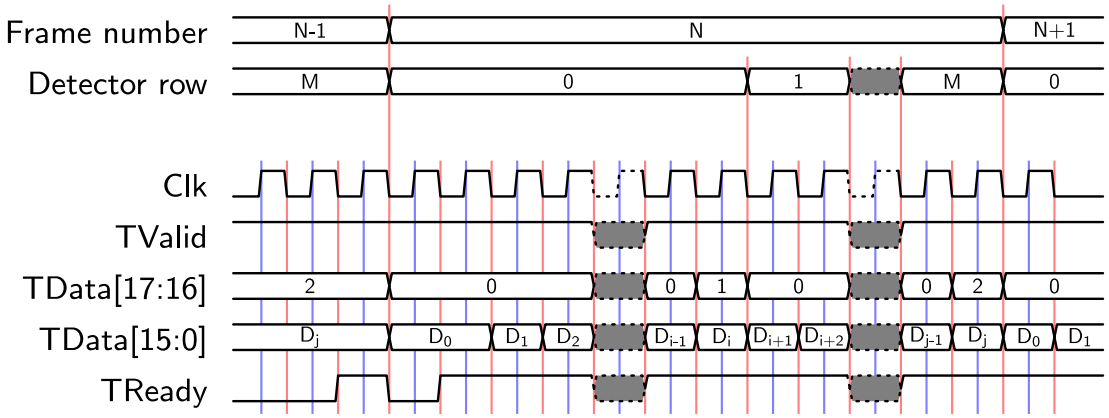


Figure A.2: Timing diagram of the pixel data transfer between two processing units. The diagram start with the transfer of the last readout detector pixel value D_j in Frame $N-1$. After this pixel has been transmitted from the detector row M, the first readout pixel value from the next frame N is transferred. The last pixel of a detector row and frame is marked with the two added signal lines TData[17:16]. The data flow is controlled by the TValid signal from the transmitter unit and the TReady signal from the receiver unit.

Signal	Bits	Description
TValid	1	Indicates that the data values provided on the TData lines are valid.
TData[33]	1	Flag for the last pixel of a detector frame.
TData[32]	1	Flag for the last pixel of a detector row.
TData[31]	1	This bit is currently not used and only for debugging purposes.
TData[30:24]	7	Y-Coordinate of the event.
TData[23:17]	7	X-Coordinate of the event.
TData[16]	1	Primary/Secondary flag. Pixels above the primary threshold are indicated by '1'.
TData[15:0]	16	Pixel data value.
TReady	1	Indicates that the receiver can accept data in the current clock cycle.

Table A.2: Signal summary of the event data SuMo-Stream version used for the hardware unit interconnection in the SuMo-DAQ.

A.2 Achievable data reduction with the event extraction and clustering

The first step in the X-ray data processing chain, where the data rate can be reduced significantly, is the event extraction. Figure A.3 shows the simulated data rate reduction factor for the event extraction as well as the additionally clustered data stream compared to the full frame data stream over the detector occupancy. The detector occupancy describes how many pixels per frame are hit by an X-ray photon and contains relevant information for the data processing. The maximum occupancy typically used for the design of DEPFET detector systems in satellites and in laboratory applications is about 1 %. The reason for this relatively low occupancy is the desired single photon energy resolution. If the occupancy is higher, the probability that two photons hit the detector close together or even with overlap in a single frame, is high. Resolving the individual energy is not possible for these photons.

The clustering of the extracted events with suitable neighboring events should further decrease the data rate. The achievable improvement by the clustering over the pure event extraction depends on the pattern type ratio and, therefore, also on the used detector system. For a typical DEPFET detector system, approximately 70 % of all events are singles, 28 % are doubles and all other patterns are combined in the remaining 2 %. With this distribution the achieved improvement by the clustering is not significant as depicted in Figure A.3.

The majority of the processing power has, up to now been spent for the parts, where the full frame data stream has to be processed, as Table 4.1 shows. For an occupancy of 1 % the data rate is reduced by a factor of 43 with the event extraction. This significantly reduces the processing power necessary to handle the data stream.

To enable the necessary data rate reduction by the event extraction step on the FPGA, all processing steps in front of this task also needed to be implemented on the FPGA. For the design of this processing system the limited storage capacity on the FPGA has to be considered. Storing a complete history of frames on the FPGA is not possible, but for a few full detector frames such as the offset and noise maps enough storage space is available. The key algorithm parts for the hardware implementation are sorting in the common-mode correction and the continuous offset and noise estimation directly on the data stream.

In the contemplation of MIXS with 4,096 pixels, the peak value of relevant detector pixels per frame is roughly 40. In case that the event extraction is the last processing step on the FPGA gain correction, clustering and histogram building remain for the software processing part.

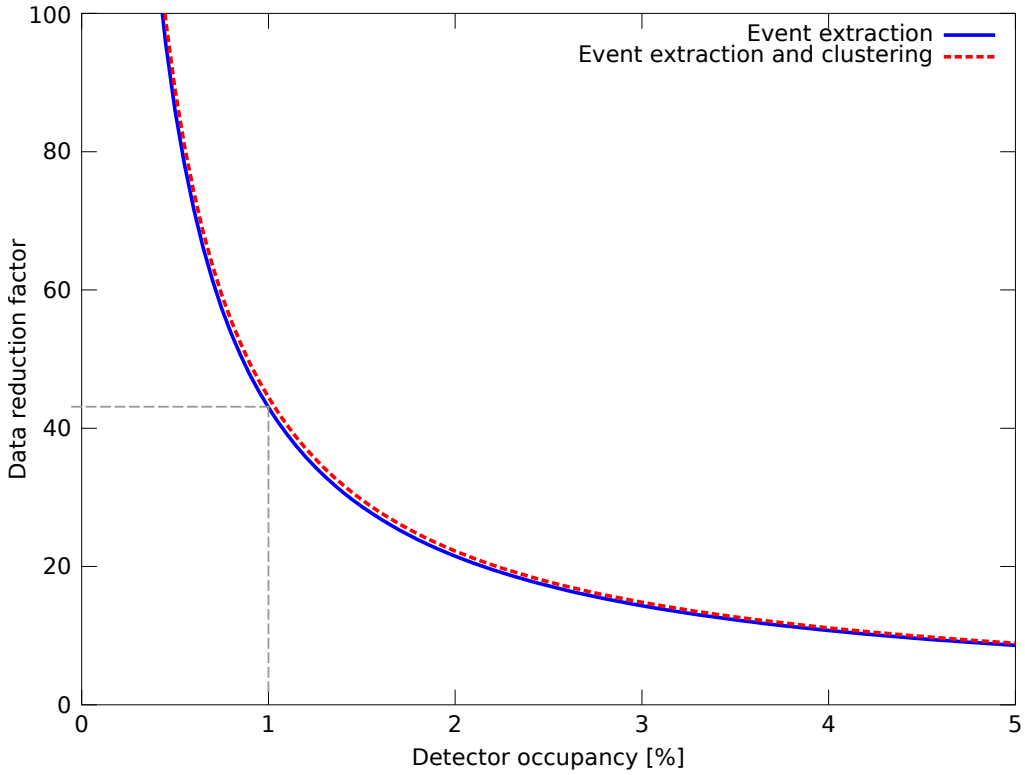


Figure A.3: Estimated data rate reduction factor for the event extraction and the additional event clustering over the detector occupancy. The detector occupancy for DEPFET detector systems is usually around 1 %.

A.3 Calculation of the pixel offset with moving average filters

Continuous offset calculation on the raw detector data stream can be realized by a moving average filter as described by Equation A.1. This equation calculates the offset value for the frame $n+1$ and the detector pixel i from the previous input values $I_i(n)$ of this pixel. In this equation, M is the selectable filter length of the moving average filter. If we assume a low photon rate compared to the filter length even under the presence of photon signals, the error is small.

$$O_i(n+1) = \frac{I_i(n) + I_i(n-1) + \cdots + I_i(n-(M-1))}{M} \tag{A.1}$$

The main limitation of this approach is the storage space required for the data history and the necessary memory bandwidth for the calculation. Each detector pixel in this application represents an individual filter channel, where the specific pixel history has to be considered. Implementing a filter in hardware for each pixel requires too many hardware resources on the FPGA. Furthermore, this approach would limit the hardware design to a detector system with a specific number of pixels. Using a single filter hardware and continuously loading all filter parameters for the currently processed pixel leads to a huge memory bandwidth requirement.

The moving average filter requires n memory read operations and one write operation; in total, $n+1$ memory transactions are necessary for each pixel. The memory bandwidth for the moving average filter can be calculated by Equation A.2 in dependency of the detector size, the frame rate f and the history length n.

$$B(n) = Detector_{Width} * Detector_{Height} * f * 2 * (n+1) \tag{A.2}$$

A.3.1 Disturbance of moving average filters by photon signals

By extending Equation A.1 with the disturbing photon signal $S_i(j)$ for pixel i in frame j as shown in Equation A.3, the estimation error in presence of photon signals can be calculated for moving average filters. The estimation error $e_i(n+1)$ depicted in Equation A.4 can be simplified for an assessment of the error, where $\bar{S}$ is the mean signal value of the occurring photon events and $\bar{R}_i$ the mean frequency of photon event occurrence in the filter window. This value can be derived from the detector occupancy. To achieve an acceptable calculation error with this approach the required history length does not allow implementing this efficiently on an FPGA. A more general approach for the estimation filter can be realized with **F**inite **I**mpulse **R**esponse (FIR) filters. FIR filters provide a higher design flexibility and can be optimized for different system requirements, but have an even more complex hardware structure.

$$\begin{aligned} \tilde{O}_i(n+1) &= \frac{\sum_{j=n}^{n-(M-1)} (I_i(j) + S_i(j))}{M} \\ &= \frac{\sum_{j=n}^{n-(M-1)} I_i(j) + \sum_{j=n}^{n-(M-1)} S_i(j)}{M} \\ &= O_i(n+1) + e_i(n+1) \end{aligned} \tag{A.3}$$

$$e_i(n+1) = \frac{\sum_{j=n}^{n-(M-1)} S_i(j)}{M} \approx \frac{\bar{R}_i * \bar{S}}{M} \tag{A.4}$$

A.4 Mathematical aspects of the Super-Modular-DAQ system

A.4.1 Full width at half maximum (FWHM) derivation

The **p**robability **d**ensity **f**unction (PDF) of a Gaussian function is given by Equation A.5.

$$f(x;\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} * e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{A.5}$$

The **f**ull **w**idth at **h**alf **m**aximum (FWHM) for a Gaussian distribution is calculated by determining the half-maximum points x_0. Therefore, Equation A.6 has to be solved.

$$\frac{1}{\sigma\sqrt{2\pi}} * e^{-\frac{1}{2}\left(\frac{x_0-\mu}{\sigma}\right)^2} = \frac{1}{2} f(x_{max}) \tag{A.6}$$

The $f(x_{max})$ value occurs at $x_{max} = \mu$ for a Gaussian distribution. This leads to Equation A.7. If this equation is resolved to x_0, which is realized in Equation A.9, the value x_0 can be calculated in dependency of σ and μ. For the case of the standard normal distribution ($\mu = 0$) Equation A.10 follows.

$$\frac{1}{\sigma\sqrt{2\pi}} * e^{-\frac{1}{2}\left(\frac{x_0-\mu}{\sigma}\right)^2} = \frac{1}{2} f(\mu) = \frac{1}{2}\frac{1}{\sigma\sqrt{2\pi}} * 1 \tag{A.7}$$

$$e^{-\frac{1}{2}\left(\frac{x_0-\mu}{\sigma}\right)^2} = 2^{-1} \tag{A.8}$$

$$x_0 = \pm\sigma\sqrt{2\ln 2} + \mu \tag{A.9}$$

$$\text{FWHM} \equiv x_+ - x_- = 2\sqrt{2\ln 2}\sigma \approx 2.3548\sigma \tag{A.10}$$

A.4.2 Median definition

The statistical median of random samples is defined by Equation A.11 (Hogg and Craig 1995, p. 152)[40], where Y is the given order statistics $Y_1 = min_j X_j, Y_2, Y_3 ... Y_{N-1}, Y_n = max_j X_j$. The median value is commonly denoted as $\mu_{1/2}$ or $\tilde{x}$ [40].

$$\tilde{x} \equiv \begin{cases} Y_{(N+1)/2} & \text{if } N \text{ is odd} \\ \frac{1}{2}\left(Y_{N/2} + Y_{1+N/2}\right) & \text{if } N \text{ is even} \end{cases} \tag{A.11}$$

For continuous statistical distributions the median value is defined as $\tilde{x}$ value, where the corresponding **c**umulative **d**istribution **f**unction (CDF) $F(x)$ is equal to $F(\tilde{x}) = \frac{1}{2}$ [10]. The $\tilde{x}$ value is equal to the mean value for a symmetric distribution function $F(x)$.

A.4.3 Dynamic noise estimation precision

A.4.3.1 Calculation of the transformation factor for the DNE

The transformation factor for the DNE algorithm depends on the expected noise distribution function. The calculation of this factor for the normal distribution can be carried out as follows. The PDF of the normal distribution is given by Equation A.12. The CDF for the normal distribution is derived by the integration of this equation and given by Equation A.14. With the error function $\text{erf}(x)$ defined by Equation A.13 the CDF can be simplified as shown in Equation A.14.

$$f(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} * e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{A.12}$$

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \tag{A.13}$$

$$F(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{(t-\mu)^2}{2\sigma^2}} dt = \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right] \quad , x \in \mathbb{R} \tag{A.14}$$

A.5 Simulation results

A.5.1 VHDL simulations of the sorting chain

Figure A.4 shows a VHDL simulation of a sorting chain with eight elements in the hardware resource-optimized ("Size") version. The chain is used by the common-mode PCore described in Subsection 5.4.6 for the median calculation. The sorting chain input is shown as "In SuMo-Stream". The sorting is organized in two stages. The first stage is the "LoadLine" (from clock cycle 3 to 17 in this simulation). In this stage, the line data values are alternately loaded into the chain and presorted. During the second "Processing" stage, the final value sorting is realized. This sorting strategy requires only a single data storage place in each element of the sorting chain and is, therefore, resource-efficient. The drawback of this realization is the low throughput of the chain. In the worst-case sorting scenario, which is shown in this simulation for the complete sorting, $N * 3$ clock cycles are required. The first of the eight data values is loaded into the chain during clock cycle 3, while the sorting is completed in clock cycle 26. The worst-case sorting situation is when the last data value loaded into the sorting chain (clock cycle 17 in this simulation) is the biggest data value (0x80 in this simulation). In this case, the largest value has to travel through the entire sorting chain. The data value internally stored in each chain element is depicted in Figure A.4 in the StoredValue signal vector. The data values in these elements are set to the highest positive value (0x7FFF for 16-bit data width) during the chain reset to initialize the chain for the sorting. The CmpResults vector is the collection of the compare results from the individual Sort/Store element (Figure 5.34) of the chain. This signal vector is used by the Sort/Store elements to control the data value swapping of neighboring elements during the sorting process. The calculated median value is provided to the output SuMo-Stream during the last processing clock cycle for each row (clock cycle 26 in this simulation).

Figure A.5 shows a simulation of the throughput-optimized "Performance" version, where two sorting chains are alternately used. The Sort/Store element used in these sorting chains is depicted in Figure 5.37. It has two data value storage places and a more complex stand-alone control logic. This requires more hardware resources than the "Size" version, but enables faster sorting and is more suitable for larger sorting chains. The data values stored in the storage space 1 (Figure 5.37) of the elements are, for both chains, depicted in the two signal vectors SortChain1 and SortChain2. In this simulation the first detector row is loaded from clock cycle 3 to 10 into SortChain1, while the second row is loaded into SortChain2 from clock cycle 12 to 19. The sorting of the first chain is finished in clock cycle 17, where the calculated median value is provided to the output SuMo-Stream. At this time, the second chain is still in the loading phase. The median value for the second detector row is provided to the output SuMo-Stream during clock cycle 26, while the first sorting chain is already in the next loading phase.

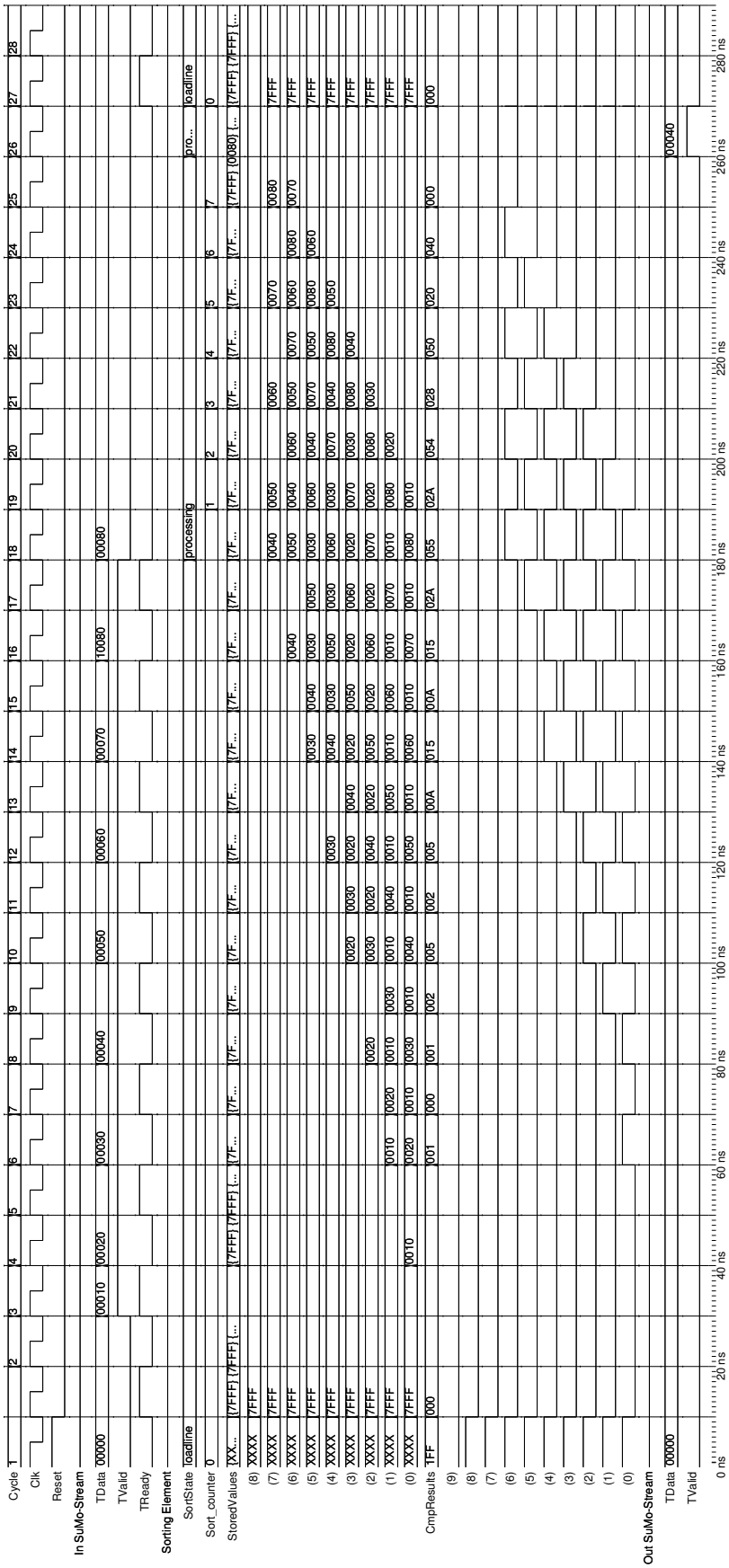


Figure A.4: VHDL simulation results for the hardware resource-optimized ("Size") implementation of the Sort/Store element chain with eight elements. The input value order represents the worst-case sorting scenario.

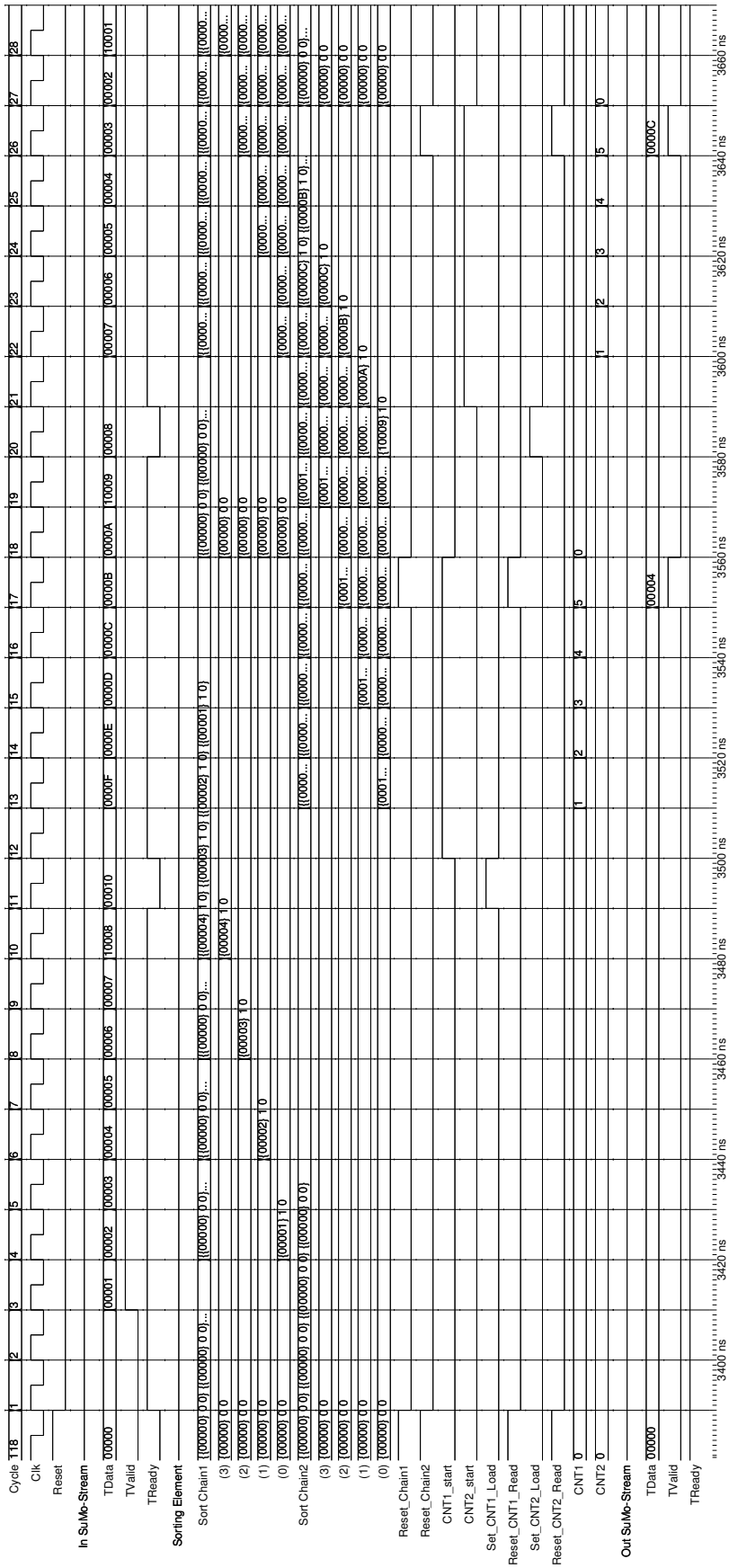


Figure A.5: VHDL simulation results for the performance-optimized implementation of the Sort/Store element chain with eight elements. The input value order represents the worst-case sorting scenario.

A.6 Data packet, data stream and event encoding formats for the SuMo-DAQ

A.6.1 Pre-processed event data stream

The structure of the pre-processed data packets is depicted in Figure A.6. Each data entry D_i has a fixed data size in this structure. This reduces the necessary processing power to extract the information on the software post-processing system and enables fast information extractions. The encoding structure for the data entries D_i is described in Table A.3.

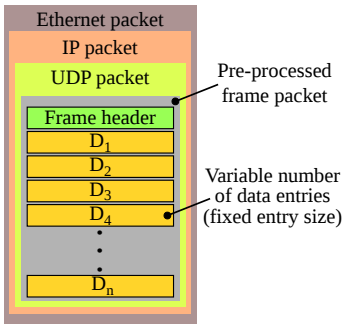


Figure A.6: Structure of the pre-processed data packets with a variable number of fixed size data entries.

Field name	Bits	Description
FrameID	1	The bit toggle from frame to frame
PosY	7	Y position of the extracted event
PosX	7	X position of the extracted event
PrimSec	1	Primary/Secondary flag
DataValue	16	Pixel data value

Table A.3: Data entry composition in the SuMo-DAQ pre-processed event data stream.

A.6.2 Command and status data stream

The structure of the command and status data packets is depicted in Figure A.6. Each data entry D_i has a variable data size in this structure. The available SuMo-DAQ commands are described in Table A.5 and the status key words in Table A.4.

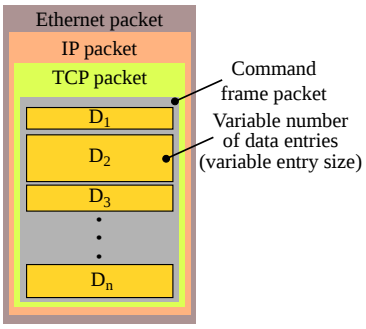


Figure A.7: Structure of the command and status frame data packets with a variable number of data entries and a variable entry size.

Status key string	Data type	Size	Description
BoardID	Int	4 Bytes	The configured ID of the DPB board
Temp	String	Variable	The measured FPGA temperature. (Format: %0d.%03d)
VCCINT	String	Variable	The measured VCCINT voltage on the FPGA. (Format: %0d.%03d)
VCCAUX	String	Variable	The measured VCCAUX voltage on the FPGA. (Format: %0d.%03d)
SystemTime	Unsigned long	8 Bytes	Current run time of the DPB board in milliseconds.
TotalProcessed-Frames	Int	4 Bytes	Number of frames completely processed by the DPB board.
Counter2	Int	4 Bytes	Idle bus cycles in the processing chain during the last measurement block.
Counter3	Int	4 Bytes	Busy bus cycles in the processing chain during the last measurement block.
TotalOffsetSum	Unsigned int	4 Bytes	The sum value of all pixels in the offset map. (This value is used for the stability control)
TotalNoiseSum	Unsigned int	4 Bytes	The sum value of all pixels in the noise map. (This value is used for the stability control)
TotalFps	Float	Variable	The average frame rate since the last reset of the DPB board.
CurrentFps	Float	Variable	The current frame rate of the DPB board.
SystemCtrlReg0	Unsigned int	4 Bytes	Summary of all FPGA-internal SuMo-DAQ configuration flags.
DaqOn	Unsigned int	4 Bytes	Status is '1' if the DPB board captures frames.
EventCorrectionOn	Unsigned int	4 Bytes	Status is '1' if the event correction is enabled.
DAQ_Status	Int	4 Bytes	The SuMo-DAQ capturing status.

Table A.4: List of available status key words sent by the SuMo-DAQ DPB systems.

Command	Key word	Bytes	Byte function description
Dummy command to test the communication.	0	4	Dummy command key word
Write a variable amount of data to specific memory address.	1	4	Write data to memory command key word
		4	Write start address
		n*(4)	Data in consecutive order
Read data from a memory address.	2	4	Command key word
		4	Read start address
		4	Number of bytes to read
Set control signals on the DPB board	3	4	Command key word
		4	Register offset value to select the channel bank
		4	New register value
Request to update all maps	4	4	Command key word
Stop cont. DAQ (CPU driven)	5	4	Command key word
Start cont. DAQ (CPU driven)	6	4	Command key word
Single frame DAQ	7	4	Command key word
Reset maps	8	4	Command key word
Set/update the bad pixel map	9	4	Command key word
		4	Offset position inside the bad pixel map
		4	New bad pixel flags
Toggle block processing	10	4	Command key word
	11	4	Command key word
Toggle event correction	12	4	Command key word
Stop cont. DAQ (clk driven)	13	4	Command key word
Start cont. DAQ (clk driven)	14	4	Command key word
Reset IGEL board	15	4	Command key word
Set data shift value	16	4	Update the data shift value in the processing chain
		4	Data value
Toggle cont. DAQ (clk driven)	17	4	Command key word
Set DACs	18	4	Command key word
		2	DAC channel
		2	New DAC value
Set tab delay	19	4	Command key word
		4	New tab delay value
Set sigma level	20	4	Command key word
		4	New sigma level
Initialize memory area	253	4	Command key word
		4	Memory start address
		4	Length
		4	Value

Table A.5: List of available SuMo-DAQ commands implemented on the DPB systems.

Index

Bibliography

[1] R. C. Alig, S. Bloom, and C. W. Struck. Scattering by ionization and phonon emission in semiconductors. *Phys. Rev. B*, 22:5565–5582, Dec 1980.

[2] M. Allman, V. Paxson, and W. Stevens. TCP Congestion Control. :FC 2581 (Proposed Standard), April 1999. Obsoleted by RFC 5681, updated by RFC 3390.

[3] Gene M Amdahl. Validity of the single processor approach to achieving large scale computing capabilities. In *Proceedings of the April 18-20, 1967, spring joint computer conference*, pages 483–485. ACM, 1967.

[4] C.S. Anderson, A.E. Mossman, D.W. Kim, G.E. Allen, K.J. Glotfelty, and G. Fabbiano. Acis sub-pixel resolution: Improvement in point source detection. In *Astronomical Data Analysis Software and Systems XX*, volume 442, page 139, 2011.

[5] Alberto Anselmi and George E.N. Scoon. Bepicolombo, esa's mercury cornerstone mission. *Planetary and Space Science*, 49(14-15):1409–1420, 2001. Returns to Mercury.

[6] ARM. Amba 4 axi4-stream protocol specification (v1.00), March 2010.

[7] ARM. Amba axi and ace protocol specification, October 2011.

[8] L. Bombelli, C. Fiorini, A. Marone, M. Facchinetti, S. andPorro, J. Treis, S. Herrmann, and A. Wassatsch. A new readout method based on source-current readout for depfet-based imagers. In *Nuclear Science Symposium Conference Record (NSS/MIC), 2010 IEEE*, pages 131–134. IEEE, 2010.

[9] L. Bombelli, C. Fiorini, M. Porro, S. Herrmann, and S. Wolfel. Vela: the cmos circuit based on fast current read-out for x-ray spectroscopy with depmos. *Nuclear Science, IEEE Transactions on*, 54(4):1359–1366, 2007.

[10] Musiol Mühlig Bronstein, Semendjajew, editor. *Taschenbuch der Mathematik*. Deutsch, Frankfurt am Main [u.a.], 5., überarb. und erw. aufl. edition, 2000. Parallele CD-ROM-Ausg. unter gleichem Titel.

[11] R. Brun and F. Rademakers. Root–an object oriented data analysis framework. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 389(1):81–86, 1997.

[12] Rene Brun. ROOT TTree documentation. http://root.cern.ch/root/html/TTree.html, Feb. 2013.

[13] V. Cerf, Y. Dalal, and C. Sunshine. Specification of Internet Transmission Control Program. RFC 675, December 1974.

[14] CERN. Cern website. http://home.web.cern.ch/, Jan. 2013.

[15] William E Cohen. Tuning programs with oprofile. 2004.

[16] Clifford E Cummings. Clock domain crossing (cdc) design & verification techniques using system verilog. *SNUG-2008, Boston*, 2008.

[17] A. Decourchelle, JL Sauvageot, M. Audard, B. Aschenbach, S. Sembay, R. Rothenflug, J. Ballet, T. Stadlbauer, and RG West. Xmm-newton observation of the tycho supernova remnant. *Arxiv preprint astro-ph/0012288*, 2000.

[18] Maier-Leibnitz-Laboratorium der Universität und der Technischen Universität München. Website maier-leibnitz-laboratorium der universität münchen und der technischen universität münchen. http://www.bl.physik.uni-muenchen.de/, Feb. 2013.

[19] Digia. Qt project website. https://qt-project.org/, October 2013.

[20] E Knuth Donald. The art of computer programming. *Sorting and searching*, 3, 1999.

[21] Paul Drumm. Personal e-mail communication, July 2012.

[22] ESA. Esa cosmic vision website. http://sci.esa.int/cosmicvision/, Feb. 2013.

[23] ESA. Esa website for xmm-newton. http://xmm.esac.esa.int/, Feb. 2013.

[24] ESA. Esa website for xmm-newton. http://sci.esa.int/science-e/www/object/index.cfm?fobjectid=24599, Jan. 2013.

[25] Bier J. et al. Evaluating dsp processor performance. Berkeley Design Technology, Inc., 1996.

[26] Rene Brun et al. Root–an object-oriented data analysis framework, version 5.26. URL: http://root.cern.ch/download/doc/ROOTUsersGuide.pdf, Feb. 2011.

[27] F. Jansen, D. Lumb, B. Altieri, J. Clavel, M. Ehle, C. Erd, C. Gabriel, M. Guainazzi, P. Gondoin, R. Much, R. Munoz, M. Santos, N. Schartel, D. Texier, and G. Vacanti. Xmm-newton observatory. *A&A*, 365(1):L1–L6, 2001.

[28] U. Fano. Ionization yield of radiations. ii. the fluctuations of the number of ions. *Phys. Rev.*, 72:26–29, Jul 1947.

[29] Peter Fischer. *Switcher-S, a HV Switch ASIC for DEPFET Matrix Control*. Steinbeis-Transferzentrum Microelectronics and Sensor Systems, 3.2 edition, 2009-2010. Chip Manual, Version 3.2.

[30] Max-Planck-Institute for Extraterrestrial Physics (MPE). Max-planck-institute for extraterrestrial physics (mpe) website. http://www.mpe.mpg.de/main.html, Feb. 2013.

[31] Max-Planck-Institute for Physics (MPP). Max-planck-institute for physics (mpp) website. http://www.mpp.mpg.de/english/index.html, Feb. 2013.

[32] GW Fraser, JD Carpenter, DA Rothery, JF Pearson, A. Martindale, J. Huovelin, J. Treis, M. Anand, M. Anttila, M. Ashcroft, et al. The mercury imaging x-ray spectrometer (mixs) on bepicolombo. *Planetary and Space Science*, 58(1):79–95, 2010.

[33] Khronos Group. Khronos group a not-for-profit industry consortium focused on creation of open standards such as opencl. URL: http://www.khronos.org/, 2013.

[34] Tianyi David Han and Tarek S Abdelrahman. Reducing branch divergence in gpu programs. In *Proceedings of the Fourth Workshop on General Purpose Processing on Graphics Processing Units*, page 3. ACM, 2011.

[35] Sven Herrmann. *ASTEROID Interface Dokument*. Semiconductor laboratory (HLL) Max-Planck-Institute, 1.0 edition, 2010.

[36] Sven Herrmann. *ISEQ programming instructions V 2.0*. Semiconductor laboratory (HLL) Max-Planck-Institut, 2011.

[37] Sven Herrmann. *XBOARD MC SCIO (slow control I/O interface) V0.95*. Semiconductor laboratory (HLL) Max-Planck-Institut, 2011.

[38] M.D. Hill and M.R. Marty. Amdahl's law in the multicore era. *Computer*, 41(7):33–38, 2008.

[39] Max-Planck-Institute Semiconductor Laboratory (HLL). Max-planck-institute semiconductor laboratory (hll) website. http://www.hll.mpg.de/, Feb. 2013.

[40] R.V. Hogg, J.W. McKean, and A.T. Craig. *Introduction to mathematical statistics*. Pearson education international. Pearson Education, 2005.

[41] Charles Hornig. A Standard for the Transmission of IP Datagrams over Ethernet Networks. RFC 894 (INTERNET STANDARD), April 1984.

[42] IEEE. Ieee standard for information technology–telecommunications and information exchange between systems–local and metropolitan area networks–specific requirements part 3: Carrier sense multiple access with collision detection (csma/cd) access method and physical layer specifications. *IEEE Std 802.3-2008 (Revision of IEEE Std 802.3-2005)*, pages 1–2977, 2008.

[43] Texas Instruments. Optimized sort algorithms for DSP. http://processors.wiki.ti.com/index.php/Optimized_Sort_Algorithms_For_DSP, Apr. 2013 2013.

[44] Kamran Karimi, Neil G Dickson, and Firas Hamze. A performance comparison of cuda and opencl. *arXiv preprint arXiv:1005.2581*, 2010.

[45] J. Kemmer and G. Lutz. New detector concepts. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 253(3):365–377, 1987.

[46] J. Kemmer, G. Lutz, U. Prechtel, K. Schuster, M. Sterzik, L. Struder, and T. Ziemann. Experimental confirmation of a new semiconductor detector principle. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 288(1):92–98, 1990.

[47] Niels Kimmel. *Analysis of the charge collection process in solid state X-ray detectors*. Dissertation, Universität Siegen, Siegen, 2008.

[48] G.F. Knoll. *Radiation detection and measurement*. Wiley, 2010.

[49] Paul Larson, Nigel Hinds, Rajan Ravindran, and Hubertus Franke. Improving the linux test project with kernel code coverage analysis. In *Proceedings of the 2003 Ottawa Linux Symposium*, pages 260–275. Citeseer, 2003.

[50] Thomas Lauf. ROAn–A ROOT based Offline Analysis tool. URL: http://www.hll.mpg.de/~pixana/, 2010.

[51] Thomas Lauf. *Analysis and Operation of DePFET X-ray Imaging Detectors*. Dissertation, Technische Universität München, München, 2011.

[52] Peter Lechner. Personal e-mail communication, July 2012.

[53] Victor W Lee, Changkyu Kim, Jatin Chhugani, Michael Deisher, Daehyun Kim, Anthony D Nguyen, Nadathur Satish, Mikhail Smelyanskiy, Srinivas Chennupaty, Per Hammarlund, et al. Debunking the 100x gpu vs. cpu myth: an evaluation of throughput computing on cpu and gpu. In *ACM SIGARCH Computer Architecture News*, volume 38, pages 451–460. ACM, 2010.

[54] Jingqiang Li, Joel H. Kastner, Gregory Y. Prigozhin, Norbert S. Schulz, Eric D. Feigelson, and Konstantin V. Getman. Chandra acis subpixel event repositioning: Further refinements and comparison between backside- and frontside-illuminated x-ray ccds. *The Astrophysical Journal*, 610(2):1204, 2004.

[55] Y.T. Liang, W.Y. Lu, Y.K. Zhu, and R. Pang. Sub-block size on impact of fundus image noise estimate. *Advances in Multimedia, Software Engineering and Computing Vol. 2*, pages 413–417, 2012.

[56] G. Lutz. *Semiconductor radiation detectors: device physics*. Springer Verlag, 1999.

[57] G. Lutz, S. Herrmann, P. Lechner, M. Porro, RH Richter, L. Strüder, and J. Treis. New depfet structures: concepts, simulations and experimental results. In *Proceedings of SPIE, the International Society for Optical Engineering*, pages 70210Y–1. Society of Photo-Optical Instrumentation Engineers, 2008.

[58] Licheng Ma. Analyse von Messdaten aus Strahlungsdetektoren mit einem Grafikprozessor. Diplomarbeit, Technische Universität München, 2011.

[59] Tso-Ping Ma and Paul V Dressendorfer. Ionizing radiation effects in mos devices and circuits. 1989.

[60] A. Meuris, F. Aschauer, S. Herrmann, T. Lauf, P. Lechner, G. Lutz, P. Majewski, D. Miessner, M. Porro, J. Reiffers, et al. Development and characterization of new 256× 256 pixel depfet detectors for x-ray astronomy. In *Nuclear Science Symposium Conference Record (NSS/MIC), 2010 IEEE*, pages 38–42. IEEE, 2010.

[61] David R. Musser. Introspective sorting and selection algorithms. *Software Practice and Experience*, 27(8):983–993, 1997.

[62] AB MySQL. Mysql homepage. 2013.

[63] NASA. Nasa website for the einstein observatory (heao-2). http://heasarc.gsfc.nasa.gov/docs/einstein/heao2.html, Feb. 2013.

[64] Information Sciences Institute University of Southern California. INTERNET PROTOCOL. RFC 894 (INTERNET STANDARD), September 1981.

[65] James B. Rawlings (University of Wisconsin-Madison) and John G. Eherdt (University of Texas). Gnu octave. http://www.gnu.org/software/octave/, September 2013.

[66] M. Porro, G. De Vita, S. Herrmann, E.L. Vaquero, P. Lechner, G. Lutz, R.H. Richter, L. Striider, J. Treis, S. Wolfel, et al. A new silicon detector system for optical and low light imaging, based on depfet rndr. In *Nuclear Science Symposium Conference Record, 2007. NSS'07. IEEE*, volume 3, pages 1942–1949. IEEE, 2007.

[67] M. Porro, S. Herrmann, and N. Hornel. Multi correlated double sampling with exponential reset. In *Nuclear Science Symposium Conference Record, 2007. NSS'07. IEEE*, volume 1, pages 291–298. IEEE, 2007.

[68] J. Postel. User Datagram Protocol. RFC 768 (INTERNET STANDARD), August 1980.

[69] J. Postel. Transmission Control Protocol. RFC 793 (INTERNET STANDARD), September 1981. Updated by RFCs 1122, 3168, 6093, 6528.

[70] K. Rank, M. Lendl, and R. Unbehauen. Estimation of image noise variance. In *Vision, Image and Signal Processing, IEE Proceedings-*, volume 146, pages 80–84. IET, 1999.

[71] Nadathur Satish, Changkyu Kim, Jatin Chhugani, Anthony D Nguyen, Victor W Lee, Daehyun Kim, and Pradeep Dubey. Fast sort on cpus and gpus: a case for bandwidth oblivious simd sort. In *Proceedings of the 2010 international conference on Management of data*, pages 351–362. ACM, 2010.

[72] Nadathur Satish, Changkyu Kim, Jatin Chhugani, Anthony D Nguyen, Victor W Lee, Daehyun Kim, and Pradeep Dubey. Fast sort on cpus, gpus and intel mic architectures. Technical report, Technical report, Intel, 2010.

[73] Gabriele Segneri, Craig Brown, J-D Carpenter, Bernd Kuhnle, Thomas Lauf, Peter Lechner, Gerhard Lutz, Stefan Rummel, Lothar Struder, Johannes Treis, et al. Measurement of the current related damage rate at $-50°$C and consequences on macropixel detector operation in space experiments. *Nuclear Science, IEEE Transactions on*, 56(6):3734–3742, 2009.

[74] Kyle Siegrist. The folded normal distribution, 2013. University of Alabama in Huntsville Department of Mathematical Sciences [online]
Available: http://www.math.uah.edu/stat/special/FoldedNormal.html (accessed 16 August 2013).

[75] Jasprit Singh. *Semiconductor devices: basic principles*. Wiley, 2001.

[76] Bjarne Stroustrup. *Programming: Principles and Practice Using C++*. Addison-Wesley Professional, 1st edition, 2008.

[77] S.M. Sze. *Semiconductor devices, physics and technology*. Wiley, 1985.

[78] The ROOT Team. Root website at cern. http://root.cern.ch/drupal/, Jan. 2013.

[79] A.C. Thompson, D. Vaughan, et al. *X-ray data booklet*. Lawrence Berkeley National Laboratory, University of California Berkeley, CA, 2001.

[80] J. Treis, L. Andricek, F. Aschauer, G. De Vita, S. Herrmann, K. Heinzinger, T. Lauf, P. Lechner, G. Lutz, M. Porro, et al. Depfet macropixel arrays as focal plane instrumentation for simbol-x and mixs on bepicolombo. In *Nuclear Science Symposium Conference Record, 2008. NSS'08. IEEE*, pages 1778–1788. IEEE, 2008.

[81] J. Treis, L. Andricek, F. Aschauer, K. Heinzinger, S. Herrmann, M. Hilchenbach, T. Lauf, P. Lechner, G. Lutz, P. Majewski, et al. Mixs on bepicolombo and its depfet based focal plane instrumentation. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 624(2):540–547, 2010.

[82] Richard M Wallace, Bogdan Vacaliuc, Dwight A Clayton, Bruce R Chaffins, Olaf O Storaasli, Dave Strenski, and Dan Poznanovic. Consideration of the tms320c6678 multicore dsp for power efficient high performance computing.

[83] E. W. Weisstein. Mathworld: Gaussian function, 2012. Wolfram Research, Inc. [online]
Available: http://mathworld.wolfram.com/GaussianFunction.html (accessed 17 October 2012).

[84] Eric W. Weisstein. Half-normaldistribution From MathWorld—A Wolfram Web Resource. Last visited on 1/3/2013.

[85] Eric W. Weisstein. Inverse erf. From MathWorld—A Wolfram Web Resource. Last visited on 8/24/2013.

[86] S. Wölfel. *Neuartige DEPFET-RNDR-Detektoren im experimentellen Betrieb*. PhD thesis, Universitätsbibliothek Siegen, 2007.

[87] S. Wölfel, S. Herrmann, P. Lechner, G. Lutz, M. Porro, R.H. Richter, L. Struder, and J. Treis. A novel way of single optical photon detection: beating the 1/f noise limit with ultra high resolution depfet-rndr devices. *Nuclear Science, IEEE Transactions on*, 54(4):1311–1318, 2007.

[88] Röbbe Wünschiers. Relational databases with mysql. In *Computational Biology*, pages 295–315. Springer, 2013.

[89] XILINX. Locallink interface specification sp006 (v2.0). Xilinx Inc. http://www.xilinx.com/aurora/aurora_member/sp006.pdf, July 2005.

[90] XILINX. lwip library (v2.00a). Xilinx Inc. http://www.xilinx.com/ise/embedded/edk91i_docs/lwip_v2_00_a.pdf, January 2007.

[91] XILINX. Virtex-5 family overview (v5.0). Xilinx Inc. http://www.xilinx.com/support/documentation/data_sheets/ds100.pdf, February 2009.

[92] XILINX. Chipscope pro 12.3 software and cores (user guide). Xilinx Inc. http://www.xilinx.com/support/documentation/sw_manuals/xilinx12_4/chipscope_pro_sw_cores_ug029.pdf, September 2010.

[93] XILINX. Embedded system tools reference manual (edk v12.4). Xilinx Inc. http://www.xilinx.com/support/documentation/sw_manuals/xilinx12_4/est_rm.pdf, December 2010.

[94] XILINX. Logicore ip processor local bus (plv) v4.6 (v1.05.a). Xilinx Inc. http://www.xilinx.com/support/documentation/ip_documentation/plb_v46.pdf, September 2010.

[95] XILINX. Logicore ip xps ll temac (v2.03.a). Xilinx Inc. http://www.xilinx.com/support/documentation/ip_documentation/xps_ll_temac.pdf, December 2010.

[96] XILINX. Platform specification format reference manual embedded development kit (edk) 12.3. Xilinx Inc. http://www.xilinx.com/support/documentation/sw_manuals/xilinx12_4/psf_rm.pdf, September 2010.

[97] XILINX. Command line tools user guide (v13.1). Xilinx Inc. http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_1/devref.pdf, March 2011.

[98] XILINX. Logicore ip multi-port memory controller (mpmc) (v6.03.a). Xilinx Inc. http://www.xilinx.com/support/documentation/ip_documentation/mpmc.pdf, March 2011.

[99] XILINX. Logicore ip xps ll fifo (v1.02a). Xilinx Inc. http://www.xilinx.com/support/documentation/ip_documentation/xps_ll_fifo.pdf, March 2011.

[100] XILINX. Logicore ip xps serial peripheral interface (spi) (v2.02a). Xilinx Inc. http://www.xilinx.com/support/documentation/ip_documentation/xps_spi.pdf, June 2011.

[101] XILINX. Xps iic bus interface (v2.03a). Xilinx Inc. http://www.xilinx.com/support/documentation/ip_documentation/xps_iic.pdf, June 2011.

[102] XILINX. Axi reference guide (v14.3). Xilinx Inc. http://www.xilinx.com/support/documentation/ip_documentation/axi_ref_guide/latest/ug761_axi_reference_guide.pdf, November 2012.

[103] XILINX. Lightweight ip (lwip) application examples xapp1026 (v3.2). Xilinx Inc. http://www.xilinx.com/support/documentation/application_notes/xapp1026.pdf, October 2012.

[104] XILINX. Virtex-5 fpga user guide (v5.4). Xilinx Inc. http://www.xilinx.com/support/documentation/user_guides/ug190.pdf, March 2012.

[105] Johannes Zeppenfeld. Lis-ipif specification, July 2006.

[106] S. Zhong and A. Macinnis. Method and system for video noise filtering, February 2 2012. US Patent 20,120,026,402.