OBSERVATOIRE DE PARIS

ÉCOLE DOCTORALE
ASTRONOMIE ET ASTROPHYSIQUE D'ÎLE-DE-FRANCE

Thesis
ASTRONOMY AND ASTROPHYSICS
Instrumentation

Shan Mignot

Towards a demonstrator for autonomous object detection on board Gaia

(Vers un demonstrateur pour la détection autonome des objets à bord de Gaia)

Thesis directed by Jean Lacroix then Albert Bijaoui

Presented on January 10th 2008 to a jury composed of:

Ana Gómez	GÉPI - Observatoire de Paris	President
Bertrand Granado	ETIS - ENSEA	Reviewer
Michael Perryman	ESTEC - Agence Spatiale Européenne	Reviewer
Daniel Gaffé	LEAT - Université de Nice Sophia-Antipolis	Examiner
Michel Paindavoine	LE2I - Université de Bourgogne	Examiner
Albert Bijaoui	Cassiopée - Observatoire de la Côte d'Azur	Director
Gregory Flandin	EADS Astrium SAS	Guest
Jean Lacroix	LPMA - Université Paris 6	Guest
Gilles Moury	Centre National d'Études Spatiales	Guest

Acknowledgements

The journey to the stars is a long one. If the poet may with one swift movement of his imagination sweep the reader off his feet to embark at their summons on a voyage to the blinking stars, for the multitude, the path remains undecipherable. Many an intriguing stratagems have been devised by generations of men yearning for such freedom and, alas, of those who have succeeded little remains but awe for their genius or for their folly. By no means do we hope to compare with them. Instead, with the sand at our feet to the stars above our heads, we venture to construct a ladder, grain per grain, and with infinite patience and method, eventually, step by step. May the readers forgive the means, overlook the ticking of the inflexible clock and bear with us long enough to either discard our laborious undertaking with evidence or to contribute their own grain. . . for such an endeavour cannot be a solitary one.

And, indeed, a solitary one it has not been. From the start, the buzz of the contradictors and of the collaborators was heard surrounding our molehill. Certainly, the improbable balance in which it now stands owes much to the varied views of the scientists, of the engineers, of the agencies as of industry. Pray, accept our gratitude for doubts, good counsels, militant beliefs, patient contributions, precise criticisms have shaped the edifice, sometimes like a wind bringing down feverish protrusions, sometimes like a lever. Of all, special praise must be given to my directors and my proofreaders, and among them to Philippe Laporte who has not only groped for viable solutions in the realm of hardware in our company, but has also bravely ventured in inspecting every single portion of this text where many dangers lay in hidding.

Family, friends and our sweetheart have also played a decisive role. Some with earthly, and others with not so earthly means. Let them not be forgotten, for if we have come within reach of the stars it is certainly because they have made us stand as on the shoulders of giants.

Contents

Acronyms

AA	Available Addresses
AC	ACross scan
ACI	Action Concertée Incitative
ACP	ASIC and FPGA Control Plan
ADC	Analog to Digital Conversion
ADP	ASIC and FPGA Development Plan
ADU	Analog Digital Unit
AF	Astrometric Field
AF1	First AF CCD
AGIS	Astrometric Global Iterative Solution
AL	ALong scan
ALMA	Atacama Large Millimeter Array
AOCS	Attitude and Orbit Control System
API	Application Programming Interface
APM	Automatic Plate Measuring
AR	Acceptance Review
ARS	ASIC and FPGA Requirement Specification
ASIC	Application Specific Integrated Circuit
BB	Bounding Box
BER	Bit-Error Rate
BP	Blue Photometer
CC	Connected-Component
CCC	Clock Conditioning Circuit
CCD	Charge-Coupled Device
CDMU	Command and Data Management Unit
CDR	Critical Design Review
CE	Chip Enable
CECC	CENELEC electronic components committee
CENELEC	Comité Européen de Normalisation Electrotechnique
CI	Charge Injection
CISC	Complex Instruction Set Computer
CL	Continued Labels
CLK	CLocK
CNES	Centre National d'Études Spatiales
COTS	Commercial-Off-The-Shelf
CPLD	Complex Programmable Logic Device
CPU	Central Processing Unit
CS	Chip Select
CTI	Charge Transfer Inefficiency
CTU	Central Terminal Unit (master on the OBDH bus)
CU	Coordination Unit
DCLK	Data CLocK
DDR	Detailed Design Review
DFT	Design For Test
DI	Data In (from the PC's standpoint)
DMS	Double and Multiple Star
DO	Data Out (from the PC's standpoint)
DPAC	Data Processing and Analysis Consortium
DPAS	Dual Port Asynchronous SRAM
DPU	Digital Processing Unit
DPSS	Dual Port Synchronous SRAM
DRAM	Dynamic RAM
DSNU	Dark Signal Non Uniformity
DU	Development Unit
E	Integer part of (by truncation)
ECSS	European Cooperation for Space Standardisation
EDA	Electronic Design Automation
EDAC	Error Detection And Correction
EEPROM	Electrically Erasable Programmable Read-Only Memory

EPROM Erasable Programmable Read-Only Memory

ESA European Space Agency

ESCC European Space Component Coordination

ESD ElectroStatic Discharge

ESOC European Space Operation Center

FAME Full-sky Astrometric Mapping Explorer

FAST Five-hundred-meter Aperture Spherical Telescope

FDIR Fault Detection, Isolation and Recovery

FET Field Effect Transistor

FIFO First In First Out queue

FL Freed Labels

FM Flight Model

FOM Fine Object Model

FOV Field Of View

FPA Focal Plane Assembly

FPASS Focal Plane Assembly Static Simulator

FPGA Field Programmable Gate Array

FPU Floating Point Unit

FRA Feasibility and Risk Analysis report

FSM Finite State Machine

FWC Full Well Capacity

GD Gaia Detect (part of Pyxis)

GEPI Galaxie, Étoile, Physique et Instrumentation laboratory (division of OBSPM, UMR 8111)

Gibis Gaia image and basic instrument simulator

GNU GNU's Not Unix

GPL General Public License

GSFC Goddard Space Flight Center specification (NASA)

GW Global Write

HiPParCoS High Precision Parallax Collecting Satellite

HQ Hierarchical Queue

IAU International Astronomical Union

ID IDentifier

IDE Integrated Design Environment

IEEE Institute of Electrical and Electronics Engineers

IM Inter-Module data structure

IO Input/Output

ITAR International Traffic in Arms Regulations

ITT Invitation To Tender

ITU Intelligent Terminal Unit (RTU with decentralised processing capabilities)

JASMINE Japan Astrometry Satellite Mission for INfrared Exploration

JTAG Joint Test Action Group (usual name used for the IEEE 1149.1 standard entitled Standard Test Access Port)

LA Label Addresses

LaMI Laboratoire de Méthodes Informatiques (UMR 8042)

LET Linear Energy Transfer

LSB Least Significant Bit

LSF Line-Spread Function

LUT Look-Up Table

LV Label Values

MIPS Million Instructions Per Second

MoM Minutes of Meeting

MOS Metal-Oxide-Semiconductor

MRD Mission Requirement Document

MSB Most Significant Bit

MTF Modulation Transfer Function

NASA National Aeronautics and Space Administration

NIEL Non-Ionizing Energy Loss

NGST Next Generation Space Telescope

OBDH On-Board Data Handling

OBSPM OBServatoire de Paris Meudon

OF Object Feature

OLMC Output Logic Macro Cell

OS Operating Software

PC Personal Computer

PCB Printed Circuit Board

PCDU Power Control and Distribution Unit

PDF Probability Density Function

PDHE Payload Data Handling Electronics

PDHS	Payload Data Handling Subsystem
PDH-S	Payload Data Handling Support contract
PDHU	Payload Data Handling Unit (supervisor)
PDR	Preliminary Design Review
PEM	Proximity Electronic Module
PIA	Programmable Interconnect Array
PLD	Programmable Logic Device
PLL	Phase-Locked Loop
PLM	PayLoad Module
PPE	Prompt Particle Event
PPN	Parameterized Post-Newtonian
PRiSM	Parallélisme Réseaux Systèmes Modélisation (UMR 8144)
PSF	Point-Spread Function
PRNU	Pixel Response Non Uniformity
PSRAM	Pseudo-Static RAM
QA	Quality Assurance
QML	Qualified Manufacturing Line
QPL	Qualified Part List
QR	Qualification Review
RAM	Random Access Memory
RISC	Reduced Instruction Set Computer
RM	Relabeling Map
RMS	Root Mean Square
ROM	Read-Only Memory
RON	Read-Out Noise
RP	Red Photometer
RPM	Relabelling Paths Memory
RTL	Register Transfer Level
RTU	Remote Terminal Unit (slave on the OBDH bus)
RVS	Radial Velocity Spectrometer
SCL	Static Chained List
SCLK	Sram CLocK
SEL	Single Event Latch-up
SET	Single Event Transient
SEU	Single Event Upset
SKA	Square Kilometer Array
SM	Sky Mapper
SMP	Symmetric MultiProcessing
SNR	Signal to Noise Ratio
SOM	Simple Object Model
SPARC	Scalable Processor Architecture
SPAS	Single Port Asynchronous SRAM
SPC	Science Programme Committee (ESA)
SPSS	Single Port Synchronous SRAM
SRAM	Static RAM
SREM	Standard Radiation Environment Monitor
SRR	System Requirement Review
SSMM	Solid State Mass Memory
SSO	Simultaneous Switching Outputs
SVM	SerVice Module
SWA	Sliding Window Algorithm
SWG	Simulation Working Group
TBC	To Be Confirmed
TBD	To Be Defined
TC	TeleCommand
TDA	Technical Definition Assistance
TDI	Time-Delay Integration
TID	Total Ionizing Dose
TM	TeleMetry
TMR	Triple Module Redundancy
TTL	Transistor-Transitor Level
TTVS	Techniques et Technologies des Véhicules Spatiaux (a course on spaceborne technology organised by CNES)
URD	User Requirement Document
VERA	VLBI Exploration of Radio Astronomy
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuits
VLBA	Very Long Baseline Array
VPU	Video Processing Unit
WD	Window Data
WE	Write Enable

WED	White Electronics Designs
WFE	WaveFront Error
XNOR	Negated eXclusive OR
XOR	eXclusive OR

Foreword

This thesis presents an analysis of the problem of detecting objects of interest on board ESA's cornerstone mission Gaia. Based on an understanding of the scientific context, an algorithmic formulation is proposed which is compatible with operation in space while being coherent with the use intended for the data. To further ascertain this conclusion, steps are taken for the elaboration of a hardware demonstrator as part of the mixed architecture which is envisaged. This offers an opportunity to evaluate performances altogether scientifically and technically and conclude on the R&D effort.

The work summarised in these pages originates in the phase A studies carried out by the OBDH working group to assess achievable performance and assist in the production of specifications. As such it is deeply rooted in the preparation of the Gaia mission. As a consequence, the role it has assumed in this frame, while considerably enriching our reflection, has also imposed restraints on our search for solutions. To increase the probability of success within the project's time line, previously successful approaches were selected as starting point and have led, because of the imperatives of consistency in our collaborations, in algorithms which are not wholly original in themselves. Nevertheless, we believe that their expressions in the system devised, the end-to-end view to the question addressed and the solutions and trade-offs upon which the technical realisation builds remain worthy of interest, especially considering how Gaia's processing requirements are orders of magnitude above other existing or planned systems.

Conversely, the underlying engineering has been of relevance to the project in return and is an endeavour to contribute to the field of space technology, however modestly. During phase A, beside providing grounds for establishing scientific requirements, the full data acquisition model built by the OBDH working group (Pyxis) was placed at the heart of the PDHE TDA study (2004-2005) to dimension the on-board electronics and collaboration with the contractor was made official by the PDH-S contract between OBSPM and ESA (2004-2005). When the industrial phases began, this contract was prolonged (2005-2007) to support the prime contractor with some of the expertise gathered and explanations on the need, algorithmic discussions and document and specification reviews have been corollaries of our developments. In parallel, the design of the demonstrator has been conducted within the ALGOL ACI (2004-2007) which, with computer scientists (PRiSM and LaMI), has sought to reconcile the masses of data generated on board satellites with the telemetry limitations in a more general sense. Accordingly, the results collected in this volume are not the result of individual isolated work but proceed from all these different frames. Tab. 1 recalls the contributions of my collaborators, may they be wholeheartedly thanked for them.

In March 2006, the design of the on-board processing subsystem became the responsibility of the prime contractor for Gaia (EADS Astrium SAS). As a consequence, Pyxis, and the detection scheme described in this text in particular, are only considered as reference work since then. Although comparison with the industrial application would be highly enlightening, it is proprietary information covered by a non-disclosure agreement so that the corresponding strategies are not further mentioned in this document.

After a first part introducing the concept and needs of Gaia (chapter 1), then the constraints applicable to on-board processing (chapter 2), the second focuses on the algorithms proper (chapter 3). To this end, approaches to the calibration of the CCD data (chapter 4), the background estimation (chapter 5) and finally the identification (chapter 6) and the measurement of objects (chapter 7) are discussed in the light of the specifications. High level descriptions are given in this part corresponding to the development of a software prototype (GD part of Pyxis) partly intended to evaluate functional performance and partly useful as a reference model. Finally, a third part refines the sample-based operations in the process of porting them to a test platform equipped with an FPGA. After presenting the technology and methodology for hardware implantation (chapter 8), an architecture is selected which facilitates testing (chapter 9) and provides facilities to the processing module, notably in terms of access to external memory (chapter 10). After translating floating-point formulae to fixed-point equivalents for compatibility with the hardware (chapter 11), the core calculations can then be derived from part II (chapter 12) before an overall inspection of the demonstrator is carried out in a concluding chapter (chapter 13). Chapters 10 and 12 are notably technical because of the intricacies of hardware implementation and are hence easier to follow for those familiar with electronics.

Contributions

3 Introduction	S. Mignot, F. Chéreau, based on APM [1] & [2] and Sextractor [3]
4 Pre-calibration	S. Mignot, P. Laporte
5 Background	S. Mignot, F. Arenou, with interpolation based on [4] by F. Patat
7 Fine object model	S. Mignot, based on the watershed transform by F. Meyer [5] and by J.C. Klein [6]
8 Platform	S. Mignot, F. Rigaud, G. Fasola
11 Fixed-point	S. Mignot, F. Arenou
12 Pipeline	S. Mignot, P. Laporte
A Platform	F. Rigaud, G. Fasola, S. Mignot
C Simulations	F. Chéreau, C. Babusiaux

PDH-S

Management	S. Mignot
Team	S. Mignot, F. Chéreau, C. Macabiau, F. Arenou, C. Babusiaux

Pyxis

Integration	F. Chéreau, S. Mignot, F. Arenou
Validation	F. Arenou, S. Mignot
Pre-calibration	S. Mignot
Detection	S. Mignot, F. Chéreau
Propagation	F. Chéreau
Selection	J. Chaussard, F. Chéreau, F. Arenou
Confirmation	S. Mignot
Assembly	S. Mignot, F. Chéreau
Sensitivity	C. Macabiau
Simulations	F. Arenou, F. Chéreau, S. Mignot

ALGOL ACI

Laboratories	PRiSM, GEPI, LaMI,
GEPI team	S. Mignot, P. Laporte, F. Arenou, F. Rigaud

Table 1: *Cuique suum* (let each have his own).

Part I

Introduction

Chapter 1

The Gaia mission

Contents

1.1 Introduction

Concept

The GAIA mission was first proposed in response to a call for ideas for ESA's next cornerstone missions (Horizon 2000+ program) in 1993 by Lindegren et al. [7] [8] to build on the success of HiPParCoS [9]. Beside achieving sub-milliarcsecond (mas) astrometry[1] in visible light, the latter's intent was extended to perform a survey of the Galaxy to tackle questions relative to its origin, structure and evolution. To this end, following the principle established by HiPParCoS, a largely over-determined set of high precision observations is to be collected from two distant FOVs[2] at different epochs to not only allow the determination of the astrometric parameters but also, simultaneously, permit the calibration of the instrument. Photometric and spectroscopic data are to be collected jointly to enrich the resulting phase-space map of the Galaxy with astrophysical information in a manner resembling the Tycho catalogue[3]. With a sample amounting to approximately 1% of the Galaxy[4], of the order of one billion objects are observed on average during 80 transits and in 12 to 15 CCDs each time. As one may easily imagine, this wealth of data, with its volume and diversity, proposes scientific and technical challenges to both the ground and the space segments.

This latter aspect is made all the more critical by the decision to place Gaia on a small amplitude Lissajous[5] orbit around L2, the second co-linear libration point in the Earth-Sun system about 1.5×10^6 km away from the Earth[6] (Fig. 1.1). Communications over such distances are more difficult due to the propagation delay and the attenuation. Besides implying that the spacecraft and the payload must be highly autonomous, this significantly reduces the available bandwidth for transmissions to ground, as compared to geostationary orbits for instance. With the intermittent visibility from ground due to the cost of operating antennas (currently two, respectively in Spain (Cebreros) and Australia (New Norcia)), it is obvious that HiPParCoS's strategy of piping data to ground directly

[1]Astrometry is the discipline in astronomy concerned with measuring positions and motions. Positions are classically determined as two angles on the celestial sphere of reference (right ascension and declination) complemented by the parallax. The parallax corresponds to the angular extension of the semi-major axis of the Earth's revolution around the Sun, which knowing the Earth's orbit and the direction of the star, provides a measure of distance expressed in parsec. One parsec is the distance at which this semi-major axis represents one arcsecond (as), ie. 2.78×10^{-4} degrees and corresponds to 3.262 light years (3.086×10^{13} km).

[2]ie. separated by a large angle on the sphere (106, 5 deg).

[3]An auxiliary star mapper on board HiPParCoS was used to produce a second catalogue, the "Tycho catalogue", with reduced astrometric precision but with photometry.

[4]The 1% figure is the one advertised by [10] but the total visible mass is likely above 10^9 solar masses with stars less massive than the Sun in average.

[5]Lissajous curves, of the general form ($x = a.sin(nt + c)$, $y = b.sin(t)$), describe harmonic motions resulting from the combination of several harmonic oscillators and can be periodic or quasi-periodic.

[6] 1% of the Earth-Sun distance.

after acquisition is not realistic for Gaia. Instead, the constraints call for sophisticated processing and storage on board to first reduce the volume then manage the flow through the downlink facility.

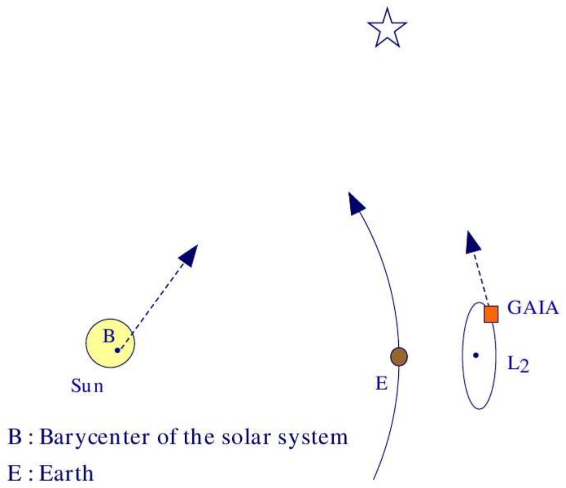


Figure 1.1: The Sun, the Earth and Gaia's Lissajous orbit in the reference system co-rotating with the Earth around the Sun (courtesy of ESA [10], scales are not respected).

A bit of history. . .

The first satellite concept which emerged relied on interferometry as a natural means of improving the precision of individual measurements [12] [13]. Interestingly enough, the "Concept and Technology Study", begun in July 1997, examined the corresponding constraints in terms of pixel size, alignment of mirrors, photon counts and resulting data rate and found that the astrometric accuracy could be improved, completeness obtained at a fainter limit and the overall design simplified by merging the two halves of each primary mirror. Hence, the interferometric option was discarded and the preference given to a "better, cheaper, faster" direct imaging system [10]. After this founding principle was adopted[7] several successive configurations were envisaged for the payload: [10] proposed three telescopes[8] and two focal planes mounted on a hexagonal optical bench. Overall size reduction was later carried out (2002) to fit the satellite within the Soyouz-Fregat fairing in a global cost reduction endeavour. Finally, in response to ESA's ITT, EADS Astrium SAS's[9] proposal was accepted in 2006 featuring only two telescopes and a single focal plane for further reduction of the complexity of the payload.

This series of designs has largely affected the payload and the imaging properties but the key elements of on-board operation have remained applicable throughout. From a high level point of view, these are relevant to commanding the acquisition of snapshots around the objects of interest, managing the resulting data on board before it can be transmitted to ground and providing estimates for the attitude control loop based on the apparent motion of objects. As a consequence, detecting the objects is of critical importance not only for the quality of the scientific return of the mission but also, quite simply, for nominal operation of the instruments.

Schedule

Fig. 1.2 illustrates ESA's schedule for Gaia. The project is currently in phase B2, generally denominated "preliminary definition" in ECSS standards, but referred to as "detailed design" for Gaia, and negotiations for the transition to phase C are on-going between Astrium and ESA. With a recommendation for being no later than 2012 by the SPC, launch is planned for December 2011 from Kourou in French Guiana with a Soyouz-Fregat launcher. After six months of transit to L2, the satellite would remain in nominal operation for 5 year (+1 extended time) with an estimated 10% system, satellite or ground segment dead time. Accordingly, accounting for some margin for the determination of parameters depending on the complete data set, the overall refinements and verifications, the final catalogue is expected to be published around 2020. Considering that HiPParCoS's results, for 118 218 stars, fill 16 volumes, R&D will be also be necessary for the former's format and distribution. . .

[7]The GAIA acronym for "Global Astrometric Interferometer for Astrophysics", then became simply a name "Gaia" as will be spelled hereafter.
[8]Two for the astrometric measurements and one for the medium band photometric and spectroscopic ones.
[9]Hereafter Astrium.

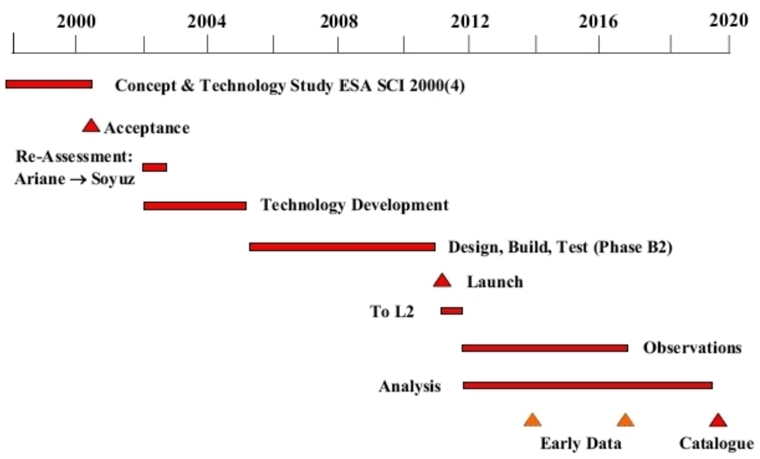


Figure 1.2: Gaia's official schedule (courtesy of ESA (website)).

1.2 Scientific return

Although only a "small" percent of the Galaxy, Gaia's sample will be of remarkable relevance to many fields of astronomy and astrophysics thanks to its high precision, bulk (20 Terabytes of compressed raw science data with a database exceeding the petabyte) and completeness. Beside the main aims of the mission emphasising astrometry and stellar populations, many by-products can be derived by data mining in the Gaia catalogue as Tab. 1.1 illustrates.

A crucial aspect of the catalogue for epoch-dependent studies and for statistical inference is Gaia's selection function. The MRD [14] has three specifications constraining object selection on a transit basis:

Specification 1 (SCI-220) *More than 95% per FOV of the transits of single objects and multiple systems over the magnitude*[10] *range specified in SCI-200 and SCI-210 shall be observed. Missing transits shall be accounted for in the overall error budget.*

Specification 2 (SCI-200) *The bright object completeness limit shall be not fainter than*[11]*:*

- $V = 6.0$ *for unreddened B1V stars,*
- $V = 6.2$ *for unreddened G2V stars, and*
- $V = 8.4$ *for unreddened M6V stars.*

Specification 3 (SCI-210) *The faint object completeness limit shall be not brighter than:*

- $V = 20.0$ *for unreddened B1V stars,*
- $V = 20.2$ *for unreddened G2V stars, and*
- $V = 22.4$ *for unreddened M6V stars.*

While predictable causes of heterogeneity[12] can be accounted for, those related to failed detection for instance are more problematic. Although complex due to the size of the database, random effects can be identified with the help of cross-transit matching. On the contrary, systematic ones will distort the resulting picture of the Galaxy. Meeting all the expectations listed in Tab. 1.1 then calls for a detection mechanism both highly reliable and sufficiently versatile to accommodate the variety of observing conditions for the entire spectrum of objects of interest.

[10] Magnitude relates to flux according to $m = -2.5log(\frac{f}{f_0}) = m_0 - 2.5log(f)$ and hence decreases when the flux increases.

[11] V designates the magnitude measured in one of the three bands of the standardised Johnson-Morgan photometric system. It is often termed the "visual" magnitude because this band is in the visible domain. B1V, G2V etc. designate stellar classes in the Harvard spectral classification and correspond essentially to different colours of main sequence stars as a result of different effective surface temperatures which are connected to the astrophysical nature of the stars (mass & radius mainly).

[12] For instance fluctuations in photometric or radial velocity precision due to the fact that not all detected objects are imaged in the spectro-photometers (BP & RP) or in the RVS.

Domain	Contribution	Description
Galactic physics	Structure of the Galaxy	Positions and kinematics for stars in all populations and structures (thin disk, thick disk, halo, bulge, spiral arms, bar, globular clusters, star-forming regions etc.).
	Age and metallicity	For the characterisation of populations together with the detection of remarkably metal-poor/rich or old objects.
	Tracers of evolution	Tidally-disrupted accretion debris.
	Dark matter	
Extra-galactic physics	Unresolved galaxies	Photometry for more than 10^6.
	Local Group	Dynamics and parallaxes (direct measurements for the brightest objects).
Stellar physics	Stellar types	Populations at the different stages of evolution (including pre-main sequence and rapid evolutionary phases), Hertzprung-Russel diagram.
	Astrophysical parameters	Mass, radius, luminosity, temperature, chemical composition for the brightest stars.
	Binary stars	10^7 resolved binaries expected within 250 pc.
	Variable stars	1.8×10^7 expected.
Distances & Reference systems	Distances	In the Galaxy and the Local Group.
	Quasars	5×10^5 expected for the construction of a reference system.
	Cepheids & RR Lyrae	Several 10^3 & 7.4×10^4 expected for recalibrating the "distance ladder" in the Universe.
Solar system	Asteroids	5×10^5 expected (from the Main Belt essentially but also Trojan, trans-Neptunian & Plutinos) with masses and rotation for part of them.
	Near-earth objects	10^5 expected.
	Orbits	Improvement by a factor 30.
	Sun	Improvement on the measurement of the solar quadrupole moment J_2.
Exotic objects	Burst sources	
	Extra-solar planets	3×10^4 expected with Jupiter-like masses up to 200 pc, some with orbits, some only based on transits.
	Brown & white dwarves	Several 10^4 expected.
	Supernovae	10^5 expected.
	Micro-lensing	
Fundamental physics	General relativity	Assess need for scalar correction, high precision PPN γ and β parameters.
	Gravitational waves	Energy determination over a certain frequency range.
	Variability of constants	Rate of change of G.

Table 1.1: Expected contributions from the Gaia data (as per [10])

1.3 The satellite

1.3.1 Measurement principle

Scanning law

Gaia's measurement principle is analogous to HiPParCoS's except in respect to technology, to orbit and to the introduction of spectro-photometry and a RVS complementing the five-dimensional astrometric parameters into a complete phase-space map. Although relying on a pre-selected list of 118 218 stars, pointed operation was made possible, a scanning strategy could be devised for HiPParCoS thanks to the distribution of targets over the celestial sphere imposed by the purpose of building a reference system. With Gaia's homogeneous survey objective, this argument applies twofold, with the corresponding benefits of reduced needs in terms of propellant and higher precision attitude reconstruction. As a consequence, Gaia's observing principle could reduce to: "measure everything transiting across the detectors". Technical considerations make this undesirable however, as discussed below, so the concept should rather be restated as: "measure everything *in the magnitude range of interest* transiting across the detectors". It is worth stressing that this restriction simply corresponds to acknowledging that precision is fundamentally limited by the photon statistics and that it remains fully in line with Gaia's completeness aims provided a sufficient number of photons are received for all objects of interest. Nevertheless, the scanning concept combined with the magnitude-driven object selection have far-reaching consequences concerning the design of the spacecraft and the payload.

Gaia's motion corresponds to a free-fall of its centre of gravity on the selected quasi-periodic Lissajous orbit[13] (Fig. 1.3) combined with spin and precession movements of the spacecraft around it (Fig. 1.4). The corresponding rates, along with the initial conditions resulting from the injection on the orbit, impact on the coverage of the sphere, particularly as regards homogeneity and the frequency of great circles[14] running parallel to the galactic plane or crossing the galactic centre. With stellar densities fluctuating between an average 25 000 stars/deg^2 over the entire sky, a typical 150 000 stars/deg^2 in the disk and a maximum of 3 000 000 stars/deg^2 in the low extinction zones of the galactic centre (Baade's windows), these events are of special relevance for on-board operation, be it processing or data storage. Besides, one of the properties of the coverages obtained is that the variable number of observations for stars at different galactic coordinates (80 transits on average) tends to be unfavourable to dense areas. As one may guess, this is rather fortunate as far as dimensioning the processing resources is concerned. It is, however, problematic scientifically speaking because these areas suffer from increased crowding: the ratio between the number of parameters to be estimated and the volume of data collected increases because multiple objects fall in the same window, thereby affecting the final precision. Hence, stringent reliability expectations result on the on-board processing for such configurations in spite of their being intrinsically more difficult to handle.

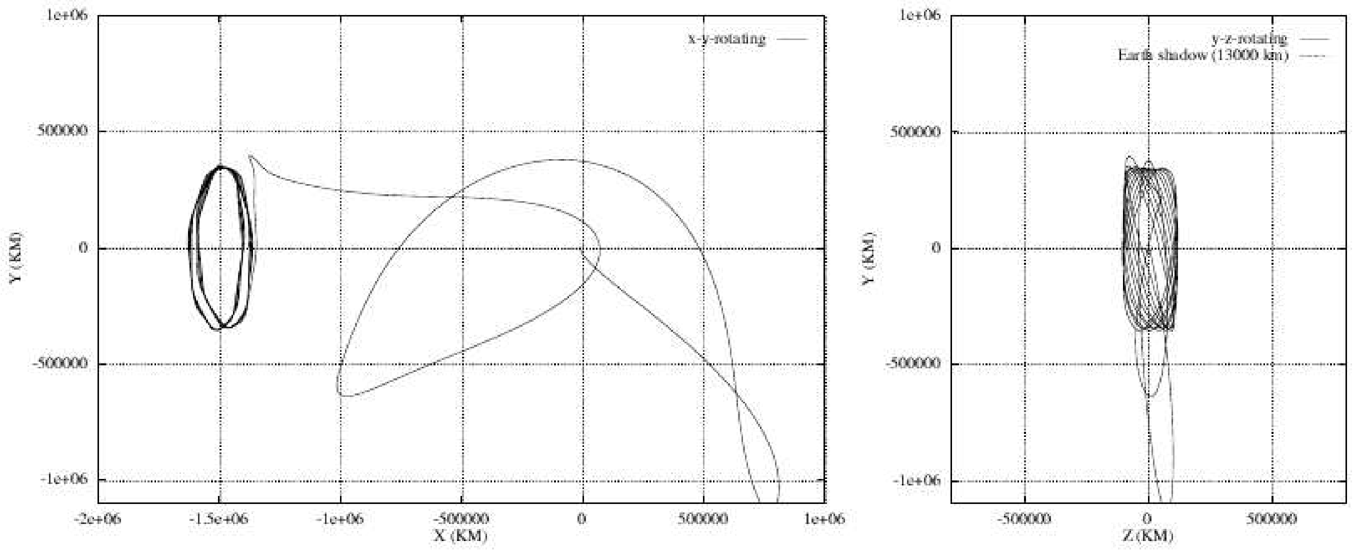


Figure 1.3: A typical one-manoeuvre transfer and Lissajous orbit for Gaia with a launch on 9/04/2009 as seen from two different viewpoints (courtesy of M. Hechler et al. [15])

[13]With maintenance manoeuvres expected about once per month to counteract the instability [15].
[14]A "great circle" denotes the trace of the FOVs during one spin period.

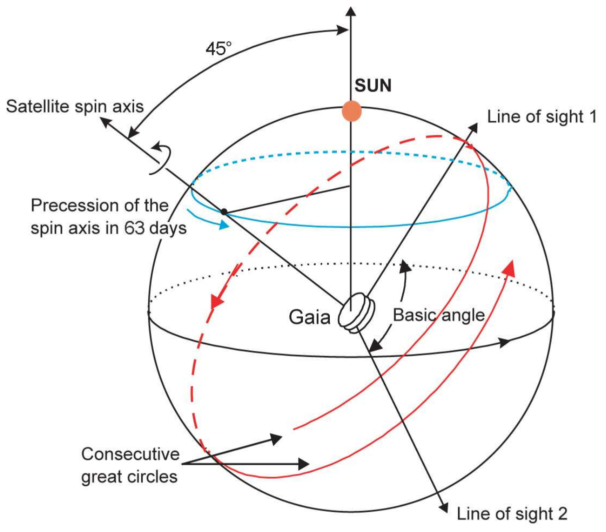


Figure 1.4: Parameters of Gaia's attitude (spin rate: 60 arcsec/s, period: 6h) and scanning law (courtesy of DPAC, [16])

Astrometry

Just like HiPParCoS, Gaia's astrometric measurements are performed in two well separated FOVs and only one-dimensionally. The former relates to the desire to obtain absolute positioning of the objects as opposed to a relative one. With the two FOVs, each object is referenced against all those of the other field, instead of only versus the neighbouring ones. The corresponding large scale constraints translate into a form of "rigidity" in the set of measurements, as opposed to summing small distortions when integrating between one line of sight and the next. Technically speaking, it implies that two FOVs must be observed in comparable conditions at all times, thus doubling the load on average.

Collecting one-dimensional data corresponds to measuring only the objects' longitude (abscissa) on the great circles. It yields precision improvements on individual measurements due to a higher SNR and the determination of fewer parameters based on a given data set. Design-wise, although the entire payload can hardly be simplified to the extent of working with only one dimension because of the need to estimate two rates for attitude control (for spin and precession with a precision of the order of the mas/s), it nevertheless leads to significant simplifications as regards the volume of scientific data per observation and the different angular resolution and stability constraints in the two directions. Conversely, the degraded information calls for collecting more data since two "missing" dimensions need to be inferred from the data set during the analysis. This is possible by globally fitting the stellar motion model to the set of abscissa on the great circles acquired during the entire mission, provided objects can be identified by inter-transit cross-matching. Fig. 1.5 illustrates how the parallax and proper motion can be jointly estimated in this process.

While the astrometric data acquired on a long time base are valuable for the determination of the parallax and proper motion, shorter term ones are relevant, for example, to the study of binary systems. As Fig. 1.5 illustrates, the presence of a companion object perturbs the motion of the primary object around the system's centre of gravity which may only be characterised if the angular and temporal resolutions are sufficiently high.

Photometry and spectroscopy

Like the astrometric data above, "instantaneous" photometry and spectroscopy are relevant for the brighter objects since they provide valuable multi-epoch data for studies related to variability. Conversely, fainter ones are affected by the degradation of the photon statistics which results from dispersing the light and require combining measurements to increase the SNR before any interpretation can become possible. With the intent to cover a large spectrum of astrophysical investigations, spectro-photometry is carried out instead of relying on distinct multiple photometric bands and a moderate resolution was selected for the RVS ($\lambda/\Delta\lambda = 11500$). With Astrium's decision to rely on a single focal plane, acquisition of this data follows from collecting the astrometric one (Fig. 1.8) so that, from an on-board operation point of view, it calls for a limited amount of additional processing, while the data represents the most significant contribution to the problem of data management (storage and telemetry).

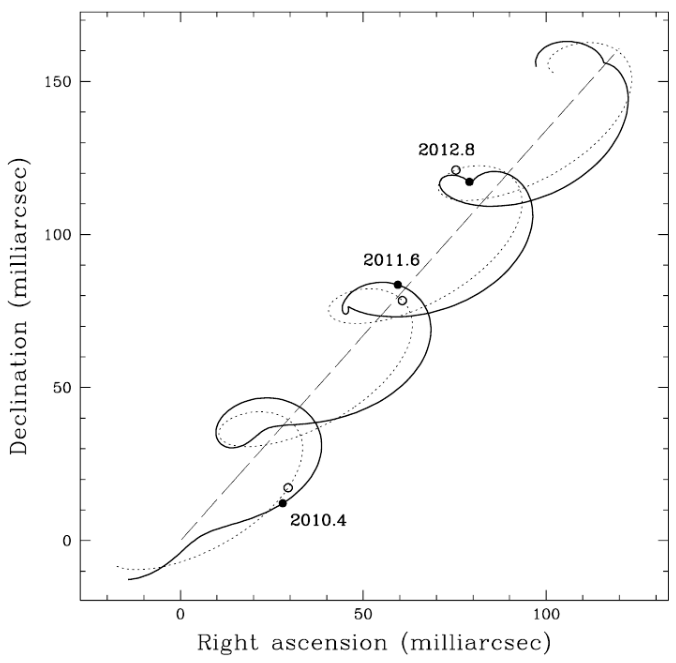


Figure 1.5: Principle of the determination of the parallax and proper motion of stars (courtesy of ESA, [10]). The apparent motion is the result of the projection on the celestial sphere of the composition of the star's motion in the Galaxy (proper motion) with the reflex motion induced by the revolution of the Earth-satellite system around the Sun (parallax). The proper motion may be coarsely estimated by linear regression, while the parallax corresponds to the semi-major axis of the ellipse recovered after compensation of the proper motion (although in practice all parameters are estimated jointly). Additionally on this figure, the perturbation resulting from the presence of an undetected companion object is illustrated by the difference between the plain and the dotted lines (isolated star model).

Data reduction

Fig. 1.6 illustrates the dependencies existing between the various types of data acquired and the diversity of products to be extracted through the different analyses. Although complex, this representation is highly simplified compared to the intrinsically convoluted problem of disentangling the multiplicity of scientific and instrumental effects.

Beside the need for numerous observations, attribution of data to physical entities ("Object matching" in Fig. 1.6) relies on the capability to compare several transits and cross-match observations. This is only possible if preliminary attitude predictions are available between each of the observing instants to determine the directions of the two lines of sight while, conversely, attitude reconstruction itself relies on the identification of the contents of the FOVs. To solve this problem in spite of the coupling between the two, a global iterative procedure (AGIS) should jointly estimate attitude, calibration and astrometric parameters for a subset of well-behaved stars. It is worth noting that its convergence depends crucially on the quality of the data received from the satellite and on the reliability of on-board detection in particular because of the need to predict observations rapidly after a first round of great circles covering the celestial sphere.

1.3.2 Overview of operation

Orbit

As for HiPParCoS, the decision to make Gaia a space mission results from the large time base required for astrometry. The very definition of the parallax implies that several measurements from significantly different positions of the Earth on its orbit around the Sun are necessary. Similarly, proper motion estimates, as first order models of the projection of the displacement of stars on the sphere, call for multiple points with large separations in time. In both cases, the quantity of interest is intrinsically of relative nature which calls for ensuring that data are comparable in spite of their belonging to different epochs. On ground, calibration issues resulting from short term variability of the instrument parameters – mainly due to thermal and gravity deformation – introduce systematic effects which limit the achievable precision in a manner incompatible with seeking parallaxes, at least based on existing technology. Conversely, in space, the absence of atmosphere and of gravity together with the possibility to achieve highly precise thermal control

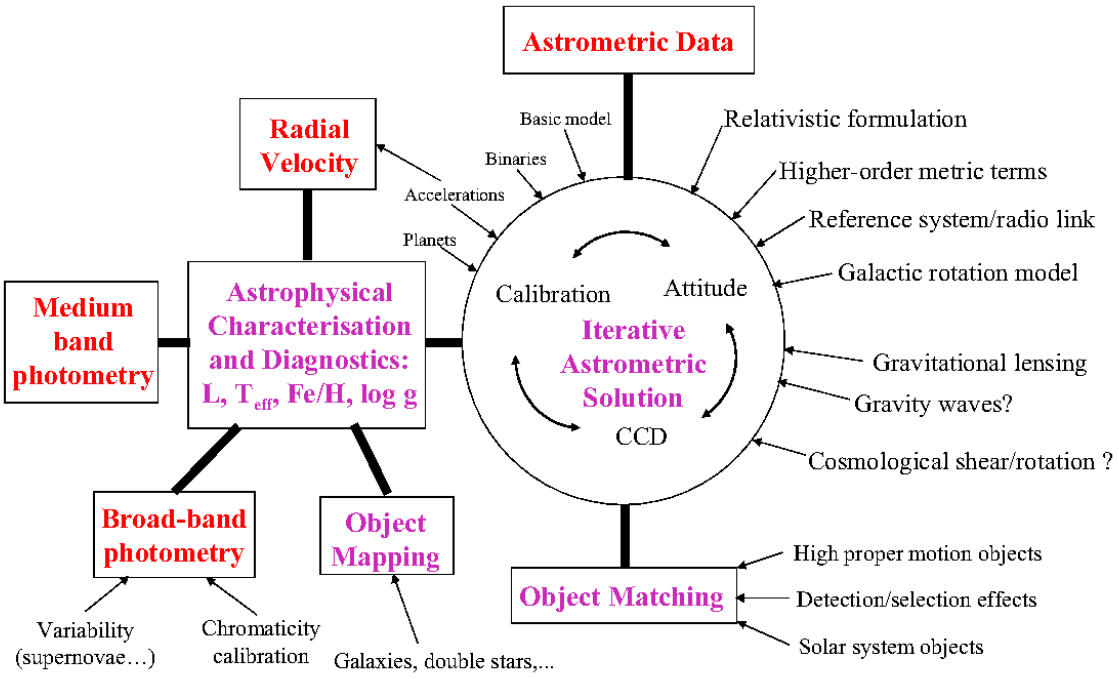

Astrometric Data
Radial Velocity
Basic model
Binaries
Accelerations
Planets
Relativistic formulation
Higher-order metric terms
Reference system/radio link
Galactic rotation model
Calibration
Attitude
Iterative Astrometric Solution
CCD
Gravitational lensing
Gravity waves?
Cosmological shear/rotation ?
Medium band photometry
Astrophysical Characterisation and Diagnostics: L, T$_{eff}$, Fe/H, log g
Broad-band photometry
Object Mapping
Object Matching
High proper motion objects
Detection/selection effects
Solar system objects
Variability (supernovae...)
Chromaticity calibration
Galaxies, double stars,...

Figure 1.6: Overview of the data reduction processes for the interpretation of Gaia's data (with acquired data in red, models and products in black and processes in purple). This figure corresponds to the 2002 baseline design in which separate broad and medium band photometers where planned. These have been replaced by the BP and RP spectro-photometers (section 1.3.2). Courtesy of M. A. C. Perryman.

of the payload provide more favourable circumstances for high precision, systematic and homogeneous measurements collected with a single well-calibrated instrument.

Although HiPParCoS, intended to be geostationary, did operate successfully from a highly elliptical orbit between 507 and 35 888 km, orbit is a key design driver. From a technical point of view, it impacts on mass mainly, as a consequence of the propellant budget for manoeuvres during transfer and for attitude and orbit control. Mass, in turn, impacts on the selection of the launcher which places constraints on cost and volume... Tab. 1.2 recalls the main elements in the trade-off performed between possible orbits for Gaia. This table lacks an indication of weights attributed to the criteria, but following the argument above, the decisive concern has been to favour stability. Lissajous orbits around L2 offer a very stable thermal environment (for the payload), means of altogether avoiding eclipses (for the power subsystem), the absence of Earth or Moon occultations (for observations) and reduced dynamical perturbations (for pointing accuracy and maintenance).

As already mentioned, the principal downside to this option is communication with ground because of partial visibility from a single station and reduced transmission rates whatever the transmitter and antenna technology used. This calls for autonomous operation and large and intelligent data management and storage on board. These are the main requirements applicable to the on-board computing. It is worth noting that while the geostationary alternative, considered feasible in [10][15], would induce simplifications as regards the general data management, the need for on-board detection and object characterisation would remain nevertheless.

Spacecraft and the payload

The Gaia satellite is built upon a custom platform. As illustrated Fig. 1.7a, the satellite is segmented in two parts. Away from the Sun, the thermal tent houses a hexagonal optical bench (3 m in diameter) supporting the telescopes corresponding to the two FOVs, the 9 mirrors used to fold the two 35 m focal lengths, the dispersive optics for the spectro-photometers, the RVS and the FPA. This part of the satellite should be a region of passive thermal stability of the order of 0.1 millikelvin. Additionally, silicon carbide (SiC) material is used for the optical bench for its very

[15]Quoting: "The study showed that the main problem in geostationary orbit is the presence of eclipses (a standard feature of geostationary orbits). These are however not a problem in terms of thermal control if the astronomical observations are performed also during the eclipse: the main driver in the thermal control is in fact not the external solar radiation, but rather the internal power dissipated by the electronics. Thus, if sufficiently large on-board batteries are included observations can be performed into eclipse. Thus the study showed that either orbit is suitable for GAIA, if necessary."

Parameter	Low Earth	Earth-Moon (L4/L5)	Earth-Sun (L2)	Geostationary
Thermal environment	–	*	***	–
Radiation environment	**	***	***	–
Optical environment (occultations)	–	*	***	*
Eclipse avoidance	–	**	***	*
Dynamic environment	–	**	***	**
Injection Δv	***	***	***	**
Maintenance Δv	–	***	***	***
Communications	**	*	**	***
Launch mass	***	**	***	*
Transfer duration	***	**	*	***
Operations	**	**	**	***
Ground facilities	***	***	***	***
Lifetime	*	***	***	*
Overall complexity/cost	–	*	***	*
Recommendation	rejected	rejected	baseline	rejected

Table 1.2: Elements of the orbit trade-off conducted by ESOC studies for Gaia as recalled in [10] (with '-', '*', '**' and '***' in increasing preference order) and leading to the selection of an orbit around L2.

low coefficient of thermal expansion. The main reason for such stringent requirements is the need to tightly monitor fluctuations on the angle between the two FOVs, the "basic angle" (106.5 deg), to obtain absolute astrometry without the need for external references for calibration.

On the other side, a service module structure, thermally insulated from the bench, contains the propellant reservoirs as well as all the equipment susceptible of variable thermal dissipation: the star trackers necessary for the acquisition of the nominal scanning law, the Li-Ion battery, the X band transponders, a particle detector SREM measuring the particle fluences, the phased-array antenna and all hardware for the processing (chapter 2). Attached to this module, a large deployable sun shield, 10 m in diameter, protects the telescopes from the Sun's unwanted light and contributes to the thermal control. Solar panels are fixed on this shield for a power generation with high efficiency due to the constant orientation to the Sun (Fig. 1.4).

Focal plane assembly

As already mentioned and visible on Fig. 1.7c, the two FOVs are combined by the M4 and M'4 mirrors and imaged by a single focal plane. The design of the telescopes and of the focal plane is driven by the photon statistics which ultimately limits the final precision. An aperture of 1.45×0.5 m^2 has been selected for the telescopes, leading to a focal plane extending over approximately 1.0×0.5 m^2. The needs to image the data at a very high angular resolution (the reason for the 35 m focal length) and to have long exposures are antagonistic in the current and projected states of technology for detectors. The former indeed calls for a small pixel size and the latter implies detectors of large dimensions while state-of-the-art CCDs are "only" of the order of 4000×4000 pixels. As a trade-off, the focal plane is tiled with large area full-frame CCDs which yield segmented integration[16].

102 CCDs dedicated to scientific observations are organised in 7 rows[17] and 16 columns to form the FPA illustrated Fig. 1.8a. The rows function independently from one another to minimise data loss in case of malfunction of one of them, thereby avoiding the need for any further redundancy. Different tasks are attributed to the CCDs column-wise because the objects traverse the focal plane from left to right:

1. **SM1 & SM2.** These two series of CCDs image only objects from one of the FOVs[18] (the other one being

[16]Conversely, were it possible to image the entire focal plane with a single CCD, it would not be desirable due to AC and AL smearing and PPEs in spite of significant improvements in the focal plane's complexity and reduction of the data flow.

[17]The FPA also houses an additional four used for the monitoring of the basic angle and the measurement of the wavefront errors.

[18]Knowing from which FOV objects originate is a considerable simplification:

- for tracking objects across the focal plane (since objects from the two FOVs have different apparent AC velocities),
- for entering observations in the database and for cross-matching objects between transits,
- for unambiguously determining the attitude of the satellite at all times.

.

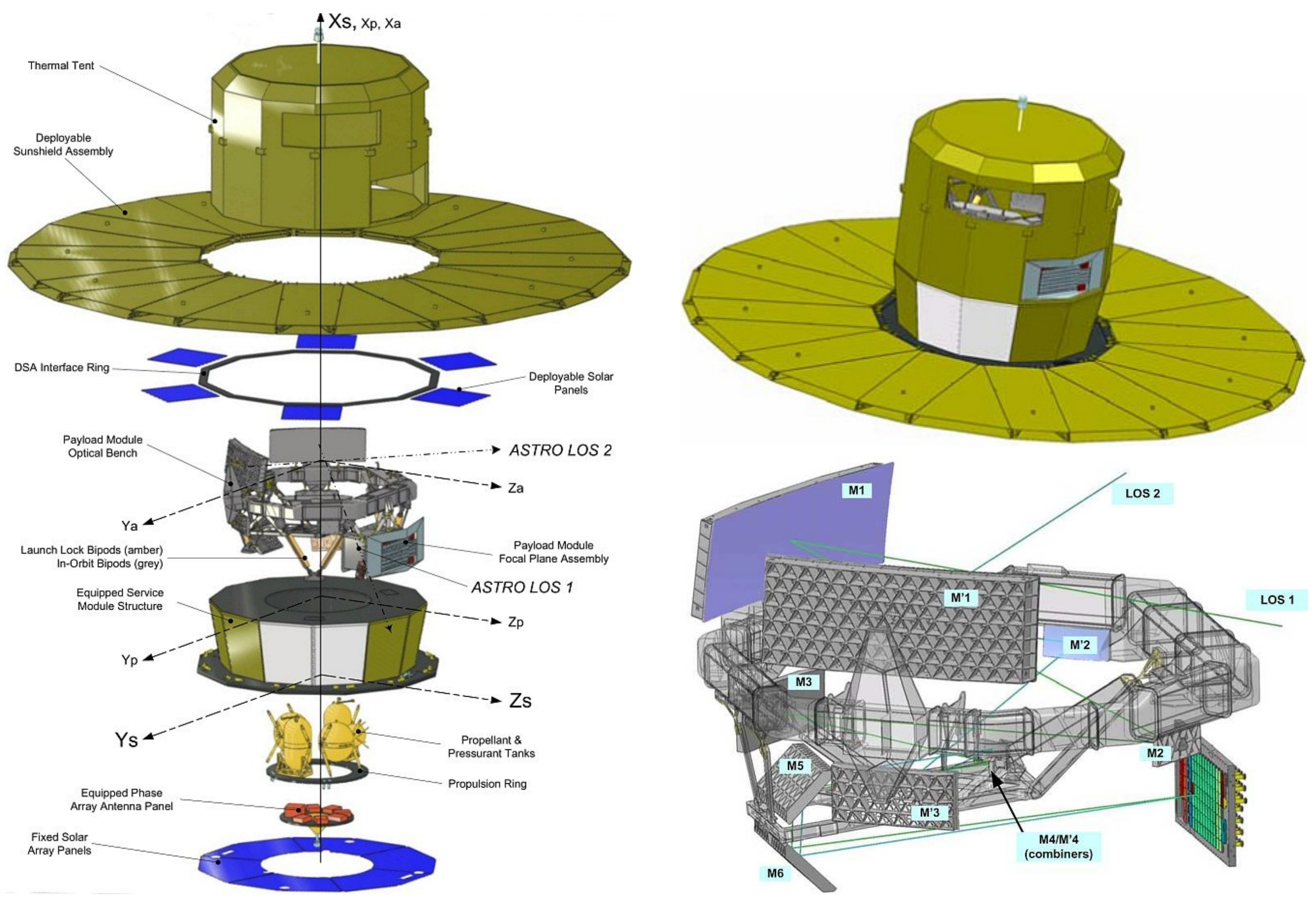

Xs, Xp, Xa
Thermal Tent
Deployable Sunshield Assembly
DSA Interface Ring
Deployable Solar Panels
Payload Module Optical Bench
ASTRO LOS 2
Za
Ya
Launch Lock Bipods (amber) In-Orbit Bipods (grey)
Payload Module Focal Plane Assembly
ASTRO LOS 1
Equipped Service Module Structure
Zp
Yp
Zs
Ys
Propellant & Pressurant Tanks
Propulsion Ring
Equipped Phase Array Antenna Panel
Fixed Solar Array Panels
M1
LOS 2
M'1
LOS 1
M'2
M3
M2
M5
M'3
M4/M'4 (combiners)
M6

Figure 1.7: The Gaia satellite as planned in phase B2 (courtesy of Astrium). Left (a): exploded view of the satellite. Top right (b): external view of the satellite with the thermal tent, the apertures of the two telescopes and the deployable sun-shield. Bottom right (c): the hexagonal optical bench supporting the two telescopes with the intermediate mirrors and the focal plane.

masked). They are entirely read-out for the purpose of detecting the objects entering the focal plane and to be subsequently observed. To increase the SNR and reduce the processing load, pixels are not read one by one but in samples: 2 pixels AL by 2 pixels AC (2×2).

2. **AF1.** The objects identified in SM1 or SM2 are confirmed in AF1 to ensure they correspond to astrophysical sources and not to noise or PPEs. Additionally, this second two-dimensional observation allows for estimating attitude parameters as part of the corresponding control loop. To make the AF1 data also relevant for on-ground astrometric exploitation, the CCD is not entirely read-out. Instead, only rectangular windows are read-out for better noise performance (section 1.3.3) based on the objects' previously estimated locations in SM and extrapolated using the nominal attitude parameters.

3. **AF2-AF9.** The observations performed in AF1 are repeated between AF2 and AF9 to improve the transit's location estimate versus photon statistics. One-dimensional data are acquired[19] in each and larger regions (in AL) are imaged in AF2, AF5 and AF8 for the purpose of searching the vicinity of objects of interest for companion stars fainter than $V = 20$ which might perturb the analysis.

4. **BP & RP.** Low resolution spectra are produced in BP and RP with the main intent of providing multiple band photometry with two CCDs: the first optimised towards the blue end of the visible domain (BP), the other towards the red (RP). This approach has the particular advantage of solving the problem of accommodating a series of filters without multiplying the number of CCDs and degrading optical quality[20]. Beside the astrophysical value of this information, measuring the chromaticity of the objects is required for full exploitation of the

[19]For bright objects, two-dimensional data is acquired in AF, BP/RP and RVS CCDs because of the need for finer calibrations.

[20]As was the case in the baseline design of 2002, where one CCD was allocated for each band – which in turn called for a dedicated telescope and focal plane.

astrometric data since the PSF depends on the objects' colour. One-dimensional data is acquired in both, with pixels corresponding to a mixture of photometric bands depending on the sub-pixel location of the object and its apparent AL velocity compared to the TDI.

5. **RVS1-RVS3.** Average resolution spectra (11 500) are acquired in the RVS CCDs to estimate the velocities of the objects along the line of sight via shifts in the absorption lines (in particular the CaII lines) resulting from the Doppler effect. For objects with higher SNRs, the spectra also provide estimates of astrophysical parameters (the effective temperature, chemical composition, the surface gravity etc.).

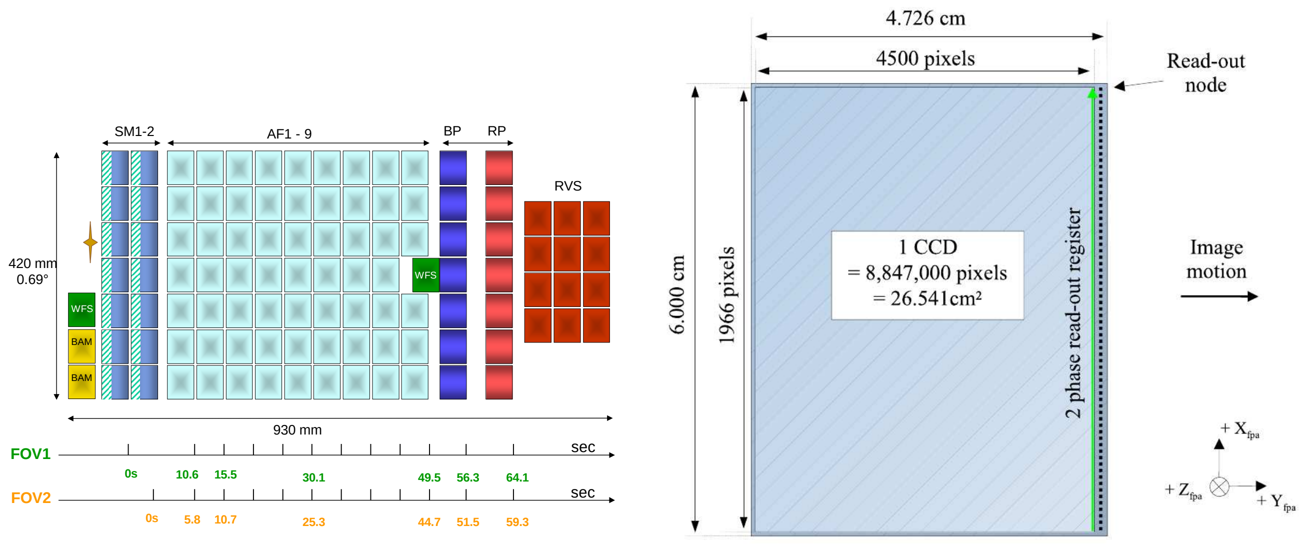


Figure 1.8: Organisation of Gaia's focal plane as planned in phase B2. Left (a): focal plane assembly (courtesy of DPAC [16]). Right (b): dimensions of the individual CCDs (courtesy of A. Short et al. [17]). In spite of the orientation on this figure, Gaia terminology designates with "TDI lines" the sets of pixels parallel to the read-out register (vertically) and as columns those traversed by the charged in the course of the TDI (horizontally).

1.3.3 CCD technology

Time Delay Integration

As a result of the satellite's spin, the stars transit across the FPA and suggest to define two directions: the along-scan (AL) one parallel to the overall movement of the objects (the abscissa in Fig. 1.8a) and the across scan (AC) one orthogonally. Conventional exposures are not possible because the stars' movement is two dimensional: due to the precession of the spin axis and to the projection of the image sphere onto the focal plane, smearing would result in the two directions. This is incompatible with the astrometric need for very high angular resolution in at least one. Instead of "static" exposures, "dynamic" ones are carried out by operating the CCDs in a TDI mode instead. This mode shifts the charges collected in the pixels across the CCD matrix at a rate selected to exactly compensate the displacement of the objects. As the combined spin and precession movements lead to both higher and constant AL velocities versus varying AC ones, the AL axis is preferred for this to allow longer integration.

Read-out

As a consequence, the image data is not acquired as a series of snapshots to be processed individually but rather as a single continuous travelling shot stretching over the five years of the mission. It cannot, hence, be read in a single pass yielding a complete image but is, instead, read column per column[21] as on Fig. 1.8b, coherently with the movement of charges induced by the TDI. The read stage then "simply" consists in shifting the charges into a parallel-in serial-out register used for read-out instead of the next line of pixels.

The CCDs, as usual, comprise two-dimensional arrays of pixels. Although, one might think of collapsing the AC resolution entirely (ie. to a single pixel) to acquire data only one-dimensionally, this is not possible since objects should be observed independently as far as feasible (otherwise additional mixing parameters will need to be estimated at the

[21]Line by line in Gaia's official nomenclature and in the following in spite of the vertical orientation in most figures.

expense of astrometric precision). The read-out registers implanted on Gaia's CCDs offer a simple and efficient way of reducing the AC resolution locally around each object. Intuitively, the manner in which charges are shifted into the amplifier allows for summing adjacent AC pixels. With interest in some pixels only (where the objects are) and not others, summation can be extended over entire intervals either to form larger data units or to efficiently flush the content of the register.

Structurally speaking it would be possible to also use the read-out register to merge photo-electrons from different pixels in the AL direction by shifting charges from the pixel matrix during two TDI periods[22] without intermediate read-out. However, to isolate the read-out register from the pixel matrix, it is preferable to insert a summing register for this purpose[23]. The two combined allow of reading $n \times m$ pixels as a single one – an operation known as "binning" and producing "samples" in Gaia terminology – with the only limitation that samples belonging to the same TDI are necessarily of the same AL dimension.

The benefit of reading samples rather than pixels wherever possible stems from the fact that for electronic noise reasons, it is desirable to read the data at the slowest possible rate. This is because estimation of the photon signal is based on repeatedly measuring both the closest available reference voltage and that of the output capacitor. Slower charge transfers translate in leaving time for the electrical state of each to stabilise before new charges are ushered in and sampling occurs. For this purpose, the read-out register is clocked at two different frequencies: a faster one for flushing a majority of pixels and a slower one for transferring the pixels of interest, all within the duration of a single TDI. The succession of regimes impacts on the stationarity of the noise statistics but an approximate Gaussian model based on average parameters can be derived.

A final but key aspect impacting on the command of the CCDs is the overall thermal equilibrium of the payload to which the FPA contributes significantly. For maximum stability, all regimes should be strictly permanent to ensure constant dissipation. This is, unfortunately, not compatible with the random arrival of objects but a possibility consists in performing a systematic read-out of the CCDs whether or not pixels of interest are present. Although, the additional data ("dummy samples") can be discarded as part of the on-board processing at a later stage this implies a constant load on the interface and hence management-wise (chapter 9).

Pixels

One of the decisive parameters evaluated in [10] concerning feasibility was the physical pixel size required to achieve measurements compatible with the micro-arcsecond target and one of the strengths of the proposed baseline was to rely on existing technology. Gaia's CCDs have, accordingly, pixels of a reasonable 10 $\mu m \times 30$ μm size, corresponding, angularly, to 58.9 $mas \times 176.8$ mas via the telescopes' focal length. With the need to cover the focal plane, large CCDs are used with 4500 TDI stages in AL and 1966 pixels in AC for a total area measuring 4.726 $cm \times 6.000$ cm (with the read-out register and bounding).

In spite of the TDI, smearing results from the difference between the continuous movement of the sources and the discrete nature of the TDI charge transfer. This degradation, characterised by the detector's MTF can be reduced by subdividing the transfer into phases, themselves matching more closely the movement. These phases, or rather the corresponding electrodes are, in fact, the only physical structure existing in the AL direction and pixels are only virtual with the TDI[24]. Four phases of successive sizes 2, 3, 2 and 3 μm, are used for Gaia for quality reasons as illustrated in Fig. 1.9.

Dynamics

Although based on an assessment of existing technology, Gaia's FPA represents quite a technological challenge. While the various characteristics discussed above already accumulate into a fair set of specifications, as regards TDI operation, pixel and CCD size and noise control, they remain incomplete without the three corresponding to efficiency, accommodation and dynamics.

The first two relate directly to maximising the level of signal collected for each object. Achieving efficient conversion between photons and electrons in the visible band, with particular attention to the red end because of interstellar reddening, is a prerequisite for this. To this end, the telescope transmission has been optimised for several spectral types and back-illuminated thinned devices are used to the limit of the thinning process imposed by their large dimensions (16 μm) [17]. Besides, not only should the focal plane be covered with CCDs but the dead zones between them should remain minimum. The CCDs' packaging has been optimised in this sense to reduce the constraints imposed by bounding and to populate the FPA more densely.

[22]Hereafter simply TDIs for brevity.

[23]In fact several lines are masked between the last "active" pixel and the read-out register. Besides insulation, these serve for maintaining the full charge capacity in spite of the transfer for read-out.

[24]In AC, on the opposite, the pixels are materialised by the anti-blooming drain structure.

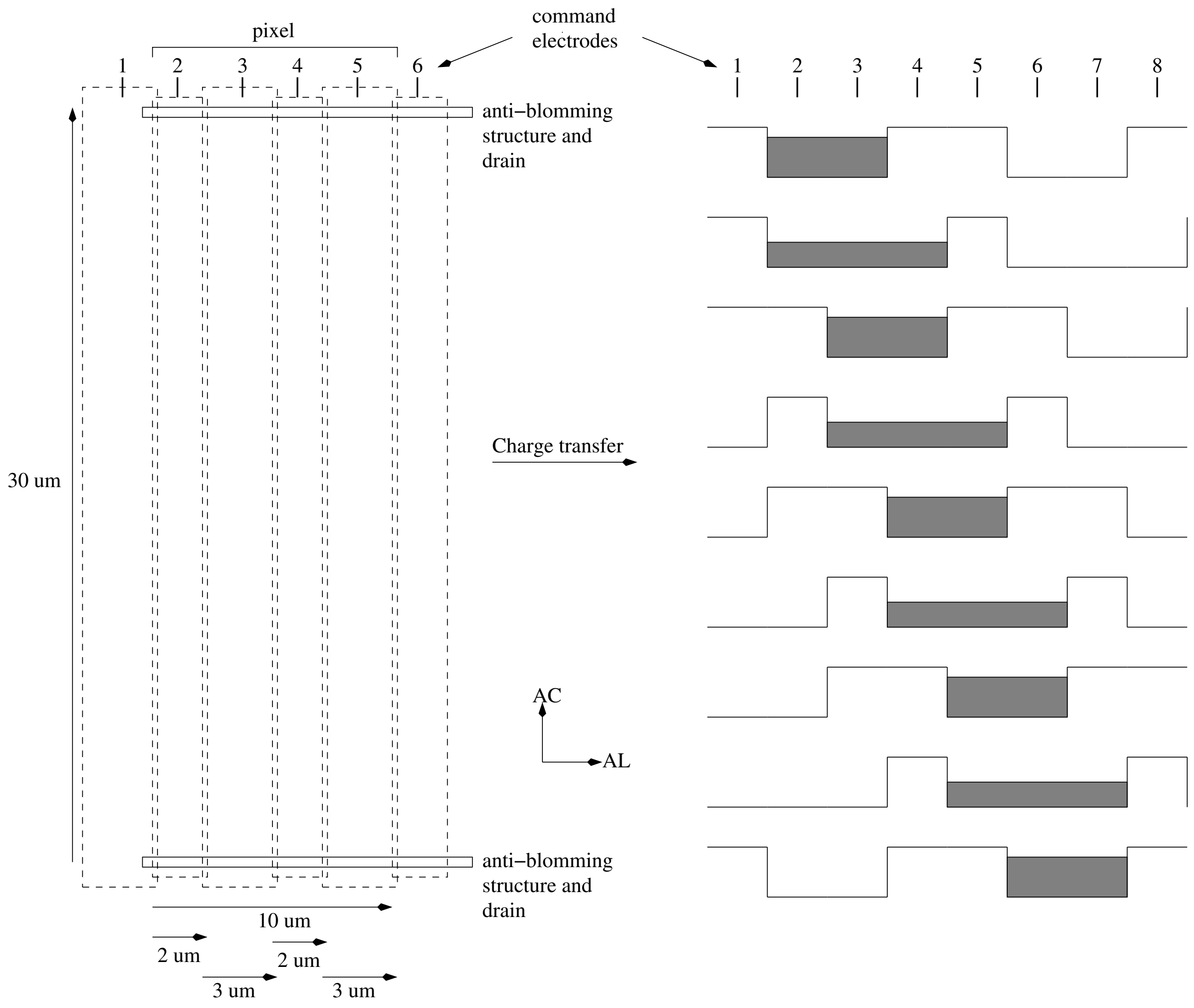


Figure 1.9: Representation of charge transfer in a four-phase TDI CCD (courtesy of CNES [18])

With targets differing by up to 14 magnitudes, that is with fluxes in ratios 1 to 400 000, the CCDs' dynamics should be extremely large. The afore-mentioned electronic noise reduction is a key step to extending the sensitivity at the faint end, around $V = 20$. Conversely, for bright stars, a trade-off should be found between maximising the total integrated flux for noise purposes while minimising the number of saturated values which are not exploitable. Notwithstanding the final sampling strategy adopted for these objects, the CCDs are equipped with gates allowing for reducing the exposures by approximate power-of-two factors. Another aspect related to the existence of bright sources is the risk of blooming. Charge diffusion is naturally limited by the thinning of the CCDs but drains are introduced to mitigate overflows of the pixels' FWC.

Device

The clock determination and routing together with control logic for binning and read-out represent a significant amount of electronics. Necessarily implanted close to the detectors themselves, these PEMs are submitted to the strict thermal constraint of constant dissipation. As a consequence, requests and transmission of samples are carried out in hard real-time. Three conditions are imposed on the former by the architecture described above: samples should be ordered in AC (due to the serial output of the register), samples should not overlap (as the individual charge packets can only be read-out once) and should come in constant numbers (ie. with additional "dummy samples" if necessary for thermal reasons).

Tab. 1.3 summarises the technology and procurement specifications for the FM. Beside the features presented above, the required quality grade must be balanced against cost, but most of all against the manufacturing yield (of order of several percents) which impacts on the availability of the 106 devices and hence on the project's schedule.

Technology	
Manufacturer	E2V
Operation	TDI
Back-illuminated	16 μm-thick
Supplementary channel	✓
Charge injection	✓
Gates	at columns 5, 9, 14, 22, 38, 70, 134, 262, 518, 1030, 2054, 2906

Pixels	
Total size	4500 $\times$ 1966 pixels
Pixel size	10 $\mu m \times 30\ \mu m$
	58.9 $mas \times 176.8\ mas$
TDI period	0.9828 ms
Exposure	0.002, 0.004, 0.008, 0.016, 0.031, 0.063, 0.126, 0.252,
	0.503, 1.006, 2.013, 2.850, 4.416 (no gate)
Pixel FWC	190 000 e^-
Summing register FWC	380 000 e^-
Read-out register FWC	475 000 e^-

Encoding	
Data type	16-bit unsigned (ADU)
Gain	4.07 e^-/ADU
Offset	1000 ADU
Saturation	262657 e^- (65535 ADU)
Zero point	26.96 mag

Noise	
Dark	$1.71 \times 10^{-2}\ e^-$/pixel
Read-out (SM)	$\sigma_{RON} = 10.78$ ADU (modelled as Gaussian)
Total	$\sigma = 10.85\ e^-$ (read-out + ADC + Dark, modelled as Gaussian))

Defects (FM procurement specification [19])	
DSNU	less than 0.5 e^-/pixel/s at -80 ^oC
PRNU	3-5 % RMS (per pixel & per band)
Traps	less than 10 greater than 200 e^- and one maximum per column
Black pixels	less than 300 lower than 80% of the local mean responsivity
White pixels	less than 15 generating more than $20e^-/s$ at -80 ^oC
Column defects	less than 7 containing 100 black or white pixels

Table 1.3: Summary of CCD properties for Gaia.

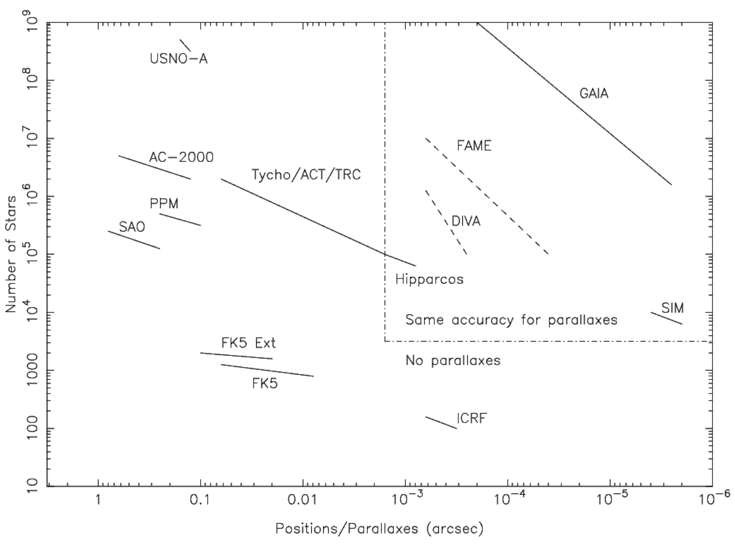


Figure 1.10: Overview of existing and intended astrometric catalogues (courtesy of ESA, [10]).

1.4 Conclusion

The Gaia mission poses scientific and technical challenges entirely coherent with the innovation to be introduced by a cornerstone mission according to ESA's criteria. With a total budget amounting to 550 million euros, including a 317 million contract with Astrium for building the satellite itself, the project involves a large community on both the industrial and scientific sides. During phase A, the latter was organised in Working Groups at the European level (eg. the photometry, RVS, DMS or OBDH working groups) to define the need and elaborate the specification for the satellite, but also to evaluate the complexity of the data reduction to be carried out on ground. Approximately one year after the prime contractor was selected, the data processing consortium (DPAC) was setup by the SPC following the publication of an "announcement of opportunity" in November 2006. It is structured in CUs, themselves composed of DUs, one for each key aspect of the data management and analysis on ground[25], and represents approximately 300 scientists all over Europe.

As Fig. 1.10 illustrates, other space missions for astrometry were planned back in 2000 for the 2010-2020 horizon. Two projects emanated from the USA: FAME with 40 million stars and precision better than 50 μas and SIM PlanetQuest with 10^4 preselected targets and precision between 1 and 4 μas. While the first is currently suspended due to the withdrawal of the NASA sponsorship, the second is currently "reconsidering" its schedule. Japan's JASMINE, inspired by Gaia in a number of respects, plans to extends the map to the infrared domain with a precision estimated around 14 μas for 10^7 stars brighter that magnitude 14 (in the z band). Launch is currently planned[26] in 2014.

Although we have argued in section 1.3.2 that high precision astrometry is near to impossible on ground, our arguments are essentially applicable in the visible domain. Ground very-large-base interferometry is however possible with radio telescopes and capable of yielding accuracies comparable to Gaia but not for such a large number of objects due to the very nature of the technique. Astrometric projects have been reported during the latest IAU symposium (248) entitled "A Giant Step : from Milli- to Micro-arcsec Astrometry" [11] for large equipments such as VLBA (USA), VERA (Japan), SKA (Australia), ALMA (USA, Japan, Europe, Chile) and FAST (China).

[25]The complete list of CUs is:

1. System architecture
2. Data simulation
3. Core processing
4. Object processing
5. Photometric processing
6. Spectroscopic processing
7. Variability processing
8. Astrophysical parameters
9. Catalogue access

[26]According to the website.

Chapter 2

On-board processing

Contents

2.1 Introduction

The on-board processing subsystem is in charge of both the operation of the payload and the surveillance of the overall state of the satellite. The past generations of satellites have essentially implemented this latter aspect, thus focusing primarily on satellite safety while limiting the former to merely relaying data from the instruments to the ground and vice versa. More ambitious systems are gradually emerging trading complexity against operational facilities and autonomy together with reduced TM/TC for monitoring against ground segment costs.

Telecommunication satellites provide a particularly relevant example of this. Such systems must balance very stringent objectives of availability, transmission capabilities and quality, on the technical side, with cost and profit on the managerial one. Two main paradigms consist in relying either on a transparent or a regenerative transponder and need to be compared as regards complexity and performance. The first, while significantly simpler in terms of design, suffers from lower SNR and higher BER because the input's noise is amplified along with the signal and re-emitted towards the ground. Conversely, the second relies on on-board decoding followed by re-encoding yielding greater transmitting capabilities and quality improvements, provided the on-board processing architecture is properly dimensioned. Choosing between either is essentially a matter of risk management and, with the maturity of on-board processing techniques and hardware, the current trend is at accepting greater complexity and investment for greater return. To some extent, science missions follow a similar evolution. In the case of Gaia, for instance, capability translates into the multiplicity of instruments and quality into a distant orbit for better thermal stability and into collecting masses of high resolution data.

On-board processing tasks are very diverse as they relate to supervision of the spacecraft and the payload altogether in terms of hardware, of operation and of data – as Tab. 2.1 illustrates in the case of Gaia[1]. Due to the overall

[1]The three different redundancy schemes correspond respectively to:

complexity, a subclass of tasks is often identified restricted to the management of the payload data. Although frequently not physically distinct from the rest, OBDH systems can be studied for themselves. The on-board detection scheme which is the object of this text belongs to this category.

Function	Sub-function	Description
TC management	Validation	verification of validity and coherence of commands received
	Data upload	update of parameters, work plan and software
	Differed execution	time-tagged TC execution for flexibility vs. ground visibility
TM acquisition	House-Keeping	typically temperatures, voltages and currents from the hardware with logs from software
	Scientific	scientific data multiplexed with the rest of TM
Control	Mode management	mode transitions to or from nominal operation, safe mode, reconfigurations from the equipment level to the entire on-board processing architecture
	Redundancies	activation of cold, lukewarm and hot redundancies
Supervision	Internal communications	management of the data buses (SpaceWire, MIL-1553B etc.)
	Synchronisation	generation of on-board time, distribution of clocks,
& scheduling	synchronisation with ground	
Data processing	Communication (OBDH)	coding/compression of data packets, communication protocol
	FDIR	surveillance plan and analysis for autonomous reconfiguration
	AOCS	management of the attitude control loop
	Thermal control	(if active)
	Scientific (OBDH)	operation of instruments
On-board storage (OBDH)	SSMM	management of the mass memory for storage of data prior to transmission
	Software & parameters	for autonomous reconfiguration

Table 2.1: Overview of on-board processing functions (from [18]).

2.2 Functional needs

2.2.1 Processing flow

We focus in the following sections on the aspects of OBDH relying on the detection of objects. Three main users can be identified: the data acquisition proper which requires predicting the location of the objects of interest on the CCDs for optimal windowed read-out (as presented in section 1.3.3), the attitude control loop based on the comparison of the effective displacements between SM1 or SM2 and AF1 with theoretical ones and the management the data packets which calls for some amount of object characterisation.

The complete data acquisition sequence on board Gaia functions as follows:

1. **Detection**. The SM1 and SM2 CCDs, at the entrance of the FPA, are entirely read-out for the purpose of detecting objects. A first calibration step ("pre-calibration") provides control over the errors induced by the detectors' imperfections and ageing: white and black pixels, heterogeneous sensitivity (intra and inter-CCDs) are handled at this level to enforce basic assumptions and maintain a stable image model throughout the mission. Objects of interest are then identified in the stream of samples and at least roughly characterised ("detection").

- hot: the redundant module is permanently powered-up and operating to allow switching between one and the other on-the-fly (only for modules which could not be power-on from ground, eg. transmitters).
- cold: the redundant module is powered-off by default and switching between both occurs upon reception of a TC from ground.
- lukewarm: the redundant module is permanently powered-up but inactive unless the need arises (only for critical modules with long boot sequences, eg. the redundant on-board supervisor is ready for operation with OS and software loaded).

2. **Propagation.** Following detection, the operation of the FPA relies on predicting object positions in the subsequent CCDs. To this end, "propagation" extrapolates positions determined in a given CCD for objects or windows to another one by taking into account the effects contributing to the apparent movement of sources on the detectors [20] (attitude, FPA geometry, aberrations, etc.).

3. **Selection.** The technical constraints governing the CCD read-out[2] calls for optimising the selection of object and the placement of windows. This is managed by "selection" which allocates resources to detected objects and solves conflicts in preparation for the CCD commands [21].

4. **Confirmation.** Detected objects are re-examined in AF1 based on the windows read-out to discard false detections and PPEs. As a by-product the SM and AF1 location estimates of well-behaved objects are combined to provide the AOCS with effective attitude parameters.

5. **Observation.** Confirmed objects are observed through a sequence of "propagation" (to predict their positions in the target CCD) and "selection" (to refine the list of windows to be observed based on the specific constraints which apply in the target CCD). Taking into account the fact that the same constraints apply in AF1 through AF9, BP and RP and RVS1 through RVS3, this sequence is optimised by simply propagating the windows commanded in the first CCD of each series to the subsequent ones for the benefit of reduced processing.

6. **CCD interfaces.** Management of the output CCD interface calls for formatting SpaceWire packets with configuration words (to define the state of charge injection and gates) and a list of samples (positions and AC sizes) to be read during a given TDI period. Conversely, on the input side, the packets must be decoded and the data routed to the appropriate WD packets.

One possible implementation (Pyxis) for this flow is illustrated in Fig. 2.1 and 2.2 and Tab. 2.2[3].

Structure	Content	Origin	Purpose
Science buffer	pre/post-scan samples	pre-calibration	determination of calibration coefficients
	raw window data	assembly	scientific data (WD packets in AF1-RVS3)
	object features	confirmation	auxiliary data (OF packets)
SM-FIFO	raw SM data	pre-calibration	window confirmed objects in SM1/SM2
FOV-BKGD	background estimates	detection	extrapolate SM backgrounds to AF1
LUT0	object features	detection	list of detected objects (in TDI order)
SCL1	selected objects	selection	objects to be observed in AF1
SLC2	windows for AF1	selection	windows to be read-out in AF1
AF1-FIFO	raw AF1 data	pre-calibration	window confirmed objects in AF1
AOCS buffer	SM & AF1 location estimates	confirmation	input to AOCS control loop
SCL3	selected objects & windows	selection	allow observations in BP
SCL4	selected objects & windows	selection	allow observations in RVS1
Programming buffer	CCD commands	read-out command	CCD configuration & requested samples (communication buffer)
Video buffer	raw samples pre/post-scan samples	FPA	video data from the FPA (communication buffer)

Table 2.2: Main data structures implemented in Pyxis.

2.2.2 Specifications

Although the processing described above is complex, Gaia's MRD [14] contains only high level and QA requirements (SFTW-010 to SFTW-280 and references to applicable ECSS standards). Although this complexity leads to much risk, it is, seemingly and paradoxically, because of this that detailed performance specifications are not given. Instead, they are left to be derived from the items documenting the intended use by the prime contractor: Spec. 1, 2 and 3 given in section 1.2 complemented with the end-of-chain standard error (Spec. 4) below and the pointing and rate errors (SCI-730 & SCI-735).

[2]ie. fixed number of read-out samples per line and absence of overlap between samples (section 1.3.3).

[3]Pyxis, as in Fig. 2.1 introduces two distinct functional blocks corresponding respectively to "pre-calibration" and "detection" to clarify the specification. Nevertheless, for increased integration, the two have been combined in the FPGA implementation presented in part III (chapter 12).

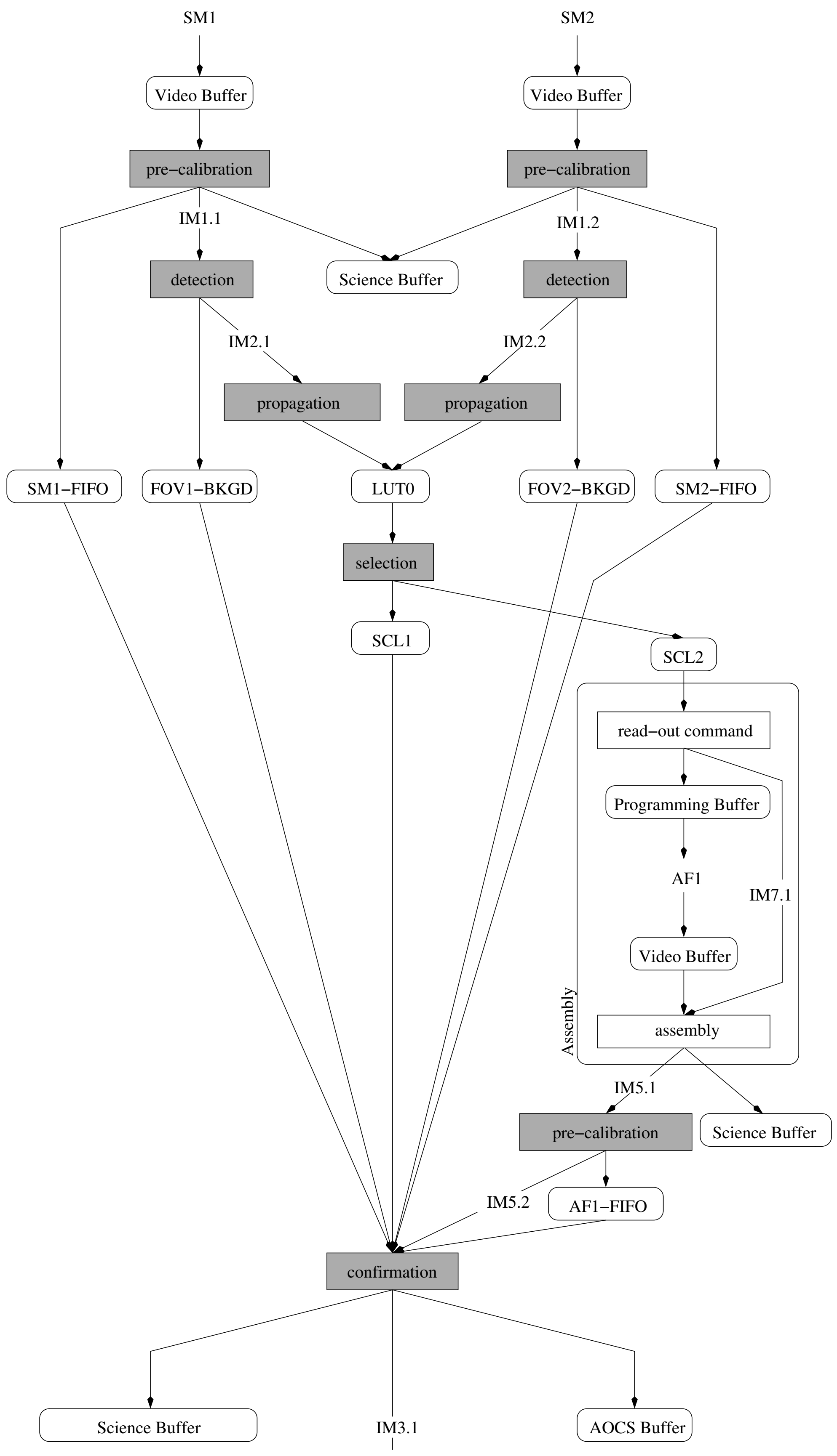


Figure 2.1: Processing flow up to and including AF1 as implemented in Pyxis. The grey rectangular boxes are the processing tasks themselves while the rounded ones represent the data structures of significant size (introduced for data exchanges Tab. 2.2) and the IM structures are small data buffers (introduced for synchronising between modules). “Assembly” is represented differently from other tasks because it is strongly dependent on the implementation of the interface with the FPA and was, as a consequence, rather integrated to Pyxis’s simulation framework.

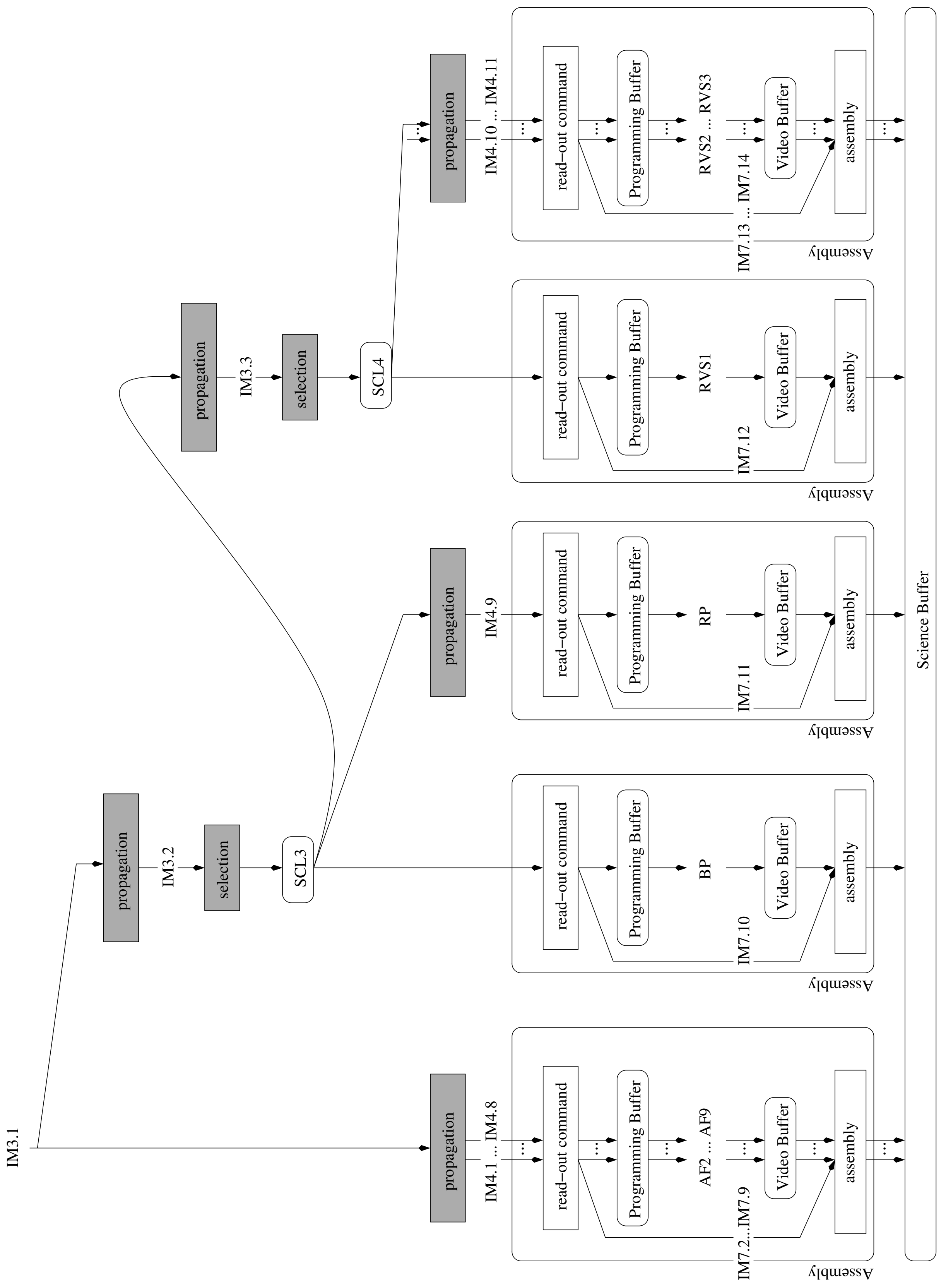


Figure 2.2: Processing from AF2 to RVS3 as implemented in Pyxis.

Specification 4 (SCI-250) *The end-of-mission statistical (parallax) standard errors (averaged over the sky) for unreddened B1V, G2V, and M6V stars shall be as specified in the following table:*

	B1V	*G2V*	*M6V*
$V < 10$	$< 7\ \mu as$	$< 7\ \mu as$	$< 7\ \mu as$
$V = 15$	$< 25\ \mu as$	$< 24\ \mu as$	$< 12\ \mu as$
$V = 20$	$< 300\ \mu as$	$< 300\ \mu as$	$< 100\ \mu as$

As a consequence, solutions are then only weakly constrained, thereby granting much freedom to the prime contractor for the technical design of the satellite and favouring the emergence of original solutions. However, if one considers the difficulty of verifying that detailed specifications are met, the decision to rely only on the quality of final products (Spec. 4) makes things even worse since they happen to be under the responsibility of the DPAC, not of the prime contractor. Furthermore, the scarcity of guidelines implies the obligation of very finely understanding the intended use which represents a difficulty from industry's standpoint and a risk for the project. As mentioned in the foreword, the two combined have motivated our involvement in the PDHE TDA and through the PDH-S activity.

Science application functional requirements

The specifications for the "science application" are copied exhaustively from [14] in this subsection as a basis for the discussions of part II where algorithms will be derived.

Spec. 5 provides a list of operational modes to be supported by the science application. Beside the nominal one which is further described in Spec. 6, calibration and diagnostic capabilities are aggregated, a transient phase is introduced at the beginning of the mission to assess the precise state of the satellite once in orbit ("commissioning") and a degraded mode is planned as part of the control of failure propagation (FDIR).

Specification 5 (SCI-680) *The Science Application shall allow the following operation of the Gaia payload:*

- *nominal science acquisition*
- *calibration during ground and in flight*
- *commissioning (in L2 and during cruise to L2)*
- *reduced capability commanded and after failure.*

The items of Spec. 6 correspond approximately to the description given in section 2.2.1 above and apply to the categories listed in Spec. 7.

Specification 6 (SCI-650) *The Science Application shall be able to:*

- *detect all objects (in the spatial domain) of interest;*
- *select objects of interest in order to discard false detection (e.g. cosmic rays);*
- *provide data necessary for the AOCS;*
- *follow all objects of interest as they cross the focal plane (considering the apparent movement of the object on the detectors);*
- *acquire astrometric, photometric and spectrometric data for the selected objects;*
- *provide data necessary to perform instrument calibration during ground testing and in flight;*
- *use uploaded calibration parameters to perform on-board corrections needed to perform on-board operations;*
- *provide instrument(s) monitoring.*

Specification 7 (SCI-660) *The Science Application shall be able to deal with the following type of objects:*

- *single objects including saturated objects*
- *blended multiple objects*
- *solar system objects*
- *extended objects*

The attributes listed in Spec. 8 are intended to serve a number of purposes on ground ranging from cross-matching objects between transits (2, 3, 8), to deriving the location, photometry and spectroscopy estimates proper (1, 2, 3) using values determined values as initial conditions (4, 6, 7, 9). Data management is also part of the concerns (4, 5, 8, 9). Beside ground use, they are also of relevance on board for tracking the objects through the focal plane (2, 3, 8) as for storage in the SSMM and the priority-driven transmission to ground (2, 4, 7, 8, 9).

Specification 8 (SCI-665) *For each object, at least the following category of information shall be transmitted to ground:*

1. *object samples*
2. *time information*
3. *position information*
4. *object type*
5. *special processing information*
6. *background*
7. *flux intensity*
8. *field information*
9. *extended objects*

Observing conditions

One of the key difficulties for the design of the on-board processing as for verifying its conformance versus the specifications is the extraordinarily large number of combinations of observing conditions. Although conceptually simple, the underlying parameter space is a large product of universe (object magnitude, type, vicinity etc.) and instrument-related ones (attitude, radiation, ageing, etc.). While a number of particularly difficult configurations are of high scientific interest – eg. observations towards the galactic centre, close DMS, star-forming regions, globular clusters etc. – they cannot be assumed to be representative. Whereas dimensioning resources on average figures would leave out many such "outliers", sustaining permanent operation in the corresponding worst-case conditions would endanger the project with over-design impacting on mass, cost and delay.

As a result of the PDHE TDA conducted in phase A, the impact of the stellar density was confirmed as being significant, hence, the MRD defined four reference ones :

- **Average: 25 000 stars/deg^2.** This value corresponds to the all-sky average. Due to the structure of the Galaxy it is not representative of observing conditions at any given moment but is indicative of a lower bound to the amount of data collected over long periods of time, for example to estimate the smoothed bandwidth requirements after buffering in the SSMM.
- **Typical: 150 000 stars/deg^2.** This one is typical of the stellar density in the disk. It is relevant both for great circles running parallel to the galactic plane and as a worst-case average in one FOV when the other line of sight points towards the bulge.
- **Design: 600 000 stars/deg^2.** This value has no direct physical meaning but is introduced as a factor for dimensioning the permanent regime capabilities of the on-board processing subsystem – based on the expectation that multiple scans at this performance level over regions at the maximum density would still yield complete observations.
- **Maximum: 3 000 000 stars/deg^2.** This is the worst case of figure for which completeness in the magnitude range of interest should be obtained, typically corresponding to densities found in Baade's windows (towards the galactic centre in regions of low extinction). Although higher densities exist on the sky in the cores of globular clusters for instance, and will be observed due to the complete coverage of the sphere achieved by the scanning law, completeness at the faint end is not required beyond this limit.

The following two items then specify the performances of the permanent and of potential special regimes density-wise.

Specification 9 (SCI-150) *Nominal observations and all astrometric requirements shall be achieved in the two superimposed fields computed using the design density in one instrument field plus the typical density in the other instrument field.*

Specification 10 (SCI-160) *A strategy to observe high-density sky regions (e.g. Baade's windows, Omega Centauri, etc.), with stellar densities up to the indicated maximum, shall be proposed. If higher densities than the stated maximum are encountered, the brightest objects up to the maximum density given shall be observed.*

2.2.3 Processing capabilities

A fundamental specificity of Gaia as compared to previous missions – one for which it certainly deserves its status as a cornerstone mission – is the significant increase in the processing. The PDHE TDA, with its full software real-time implementation of Pyxis identified that even the projected resources advertised by Maxwell for its SCS750 board ($\simeq$ 1400 MIPS at 800 MHz with L2 cache enabled) would scarcely be sufficient and recommended the use of dedicated hardware to alleviate the load on the processor. The issue is not only with the complexity of the tasks but also with their multiplicity. This latter aspect, together with the volume of data, indeed implies both context switches and frequent cache misses so that performances on the PDHE breadboard could be shown to be limited by the slow memory bus[4] on Maxwell's "proto-engineering module".

The figure above is orders of magnitude above other spaceborne computer systems. As Tab. 2.3 shows, emphasis was previously rather placed on low operating frequencies for savings in power consumption and sensitivity to radiation – some devices even advertise an absence of minimum frequency. Even the Mars Reconnaissance Orbiter with a need for approximately 300 MIPS clearly belongs to a different generation of systems compared to Gaia.

Processor	Data	Address	Devices	Mission
ERC32	32	32	Atmel TSC695F: RISC, 20 MIPS at 25 MHz, radhard, FPU, SPARC V7, 5V	Cryosat, Pleiades
Leon	32	32	LEON3FT-RTAX: RISC, 25 MIPS at 25 MHz, radhard, FPU, SPARC V8, 1.3 or 3.3V	still under development
RCA 1802	8	16	low power, silicon-on-sapphire (rad-tolerant), no minimum frequency	Voyager (3), Viking, Galileo
HM6100	12	12	Harris semiconductors: 2's complement arithmetics, no mimimum frequency	HiPParCoS
RAD 600	32	32	IBM: RISC, 35 MIPS at 33 MHz, radhard, EDAC RAM	Spirit, Opportunity etc.
RAD 750	32	32	IBM: RISC, based on PPC750, $<$ 300 MIPS at 133-166 MHz, radhard, $< 5W$	Mars reconnaissance orbiter
IMA31750	16	16	CISC, 1 MIPS at 8 MHz, radhard	Protéus

Table 2.3: Overview of the processors used in space (data and address correspond to word sizes in bits).

2.3 Constraints

2.3.1 Space

Operation in space is exacting due to the attenuated atmosphere and geomagnetic field. As a result, systems are directly exposed to the Sun's magnetic field, plasmas, micro-meteoroids (or debris) and intense radiation. The precise nature of the environment varies greatly depending on the altitude of the orbit. Low earth orbits, for instance, benefit from the proximity of the earth for some protection against these effects, but the remnants of atmosphere pose orbit maintenance difficulties and, combined with the presence of highly reactive constituents (eg. atomic oxygen), cause intense erosion of covering materials. Conversely, interplanetary missions like Gaia are submitted to complex gravitational, electromagnetic and radiation environments with contributions from the Sun, the Earth and the Galaxy. Paradoxically, because of their being isolated, self-contamination becomes a key concern. As an example, without the atmosphere to sweep them away, they are subject to the formation of clouds of active constituents and low energy plasma surrounding the satellite and susceptible of altering optical surfaces or affecting the efficiency of solar panels.

Design recommendations as regards operation in space can be found as part of the ECSS standards [22] or in CNES's TTVS [18]. The subject underlies all aspects of satellite design, from general considerations relative to the choice of materials (eg. to ensure they will only "outgas" neutral constituents) or to making the satellite conductive (to avoid the accumulation of charges leading to damaging electrostatic discharges), all the way to component selection. Some of these measures sometimes lead to seemingly incomprehensible design choices without a global view to the problem. A somewhat surprising consequence of protecting the instruments against contamination in the case of Gaia lies in the installation of heaters close to the focal plane (Fig. 2.5) to force outgassing during the satellite's transfer to L2 (with temperature well above the nominal range) before passive microkelvin stability is sought. Tab. 2.4 provides some examples of considerations applying to component selection.

[4]Partly related to resynchronisation between the processors and EDAC protection of the RAM.

Category	Constraint
Packaging	Outgass only neutral constituents
	Toxic materials should not be used
Mechanical	Withstand launch stress (pull/adhesion strength, warp & twist, harness)
	Minimum impact on inertia (moving parts are critical)
Thermal	Operate over a large temperature range (-55 to $+125$ ^oC)
	Properties should not vary with temperature
	Withstand temperature cycling due to eclipses (cracking of package)
Radiation	TID, SEU performance, absence of latchup (components)
	Resistance to short-circuits (spacing between conductors, insulation resistance, high destructive currents on PCBs)

Table 2.4: Examples of constraints for selecting components.

2.3.2 Radiations

The satellites orbiting around L2 evolve in a mixture of three categories of particles. The first is the sum of the black-body photons emitted by the Sun and those diffused (albedo) or originating from the Earth. While by a long way dominant in numbers, these fluences are only a concern regarding thermal equilibrium, the radiation pressure applied to the satellite and stray light. The second, in numbers, consists of the solar wind plasma composed essentially of protons and He^{2+} ions with energies in the keV range. It represents an electrostatic threat because of surface charge and erodes covering with sputtering. Last, but not least, come high energy particles (from several keV to GeV): electrons, protons and ions of solar or galactic origins and secondary particles produced through their interaction with the satellite's structure (mainly electron-induced bremsstrahlung radiation[5]). Although in comparatively small numbers, these are responsible for most of the damage to electronic components and are, as Tab. 2.5 shows, the main cause of anomalies in orbit.

Cause	Frequency (%)
Radiation	51
Plasmas	27
Thermal	10
Debris	7
Other	5

Table 2.5: Main causes of anomalies in operation (from [18]).

A variety of processes take place depending on the incident particle's nature and energy. Electrons can mix with electronic signal or build up static charge resulting in discharges (ESD) which cause interferences or damage depending on the currents involved. Ions tend to deposit energy on their tracks resulting in ionisation of materials or the formation of electron/positron pairs (LET). Finally, energetic protons break nuclei and lead to highly-localised ionisation. These ionising interactions are typically the cause of the spurious signal on CCD detectors known as PPEs (section 7.2.2) or of changes in the electrical state of semi-conductors (SEUs for direct effects of each interaction or SET for those which propagate through the logic). Except in cases when these lead to destructive short-circuits (latchup: SEL), they represent “only” transient perturbations to normal operation to which the design can be made robust, for instance through TMR[6]. Conversely, another class of interactions builds up lasting damage. Elastic collisions (NIEL) with atoms can cause “displacement damage” by removing them from their original sites. This alters the very structure of the material and results in a modification of its electrical, mechanical or optical properties. Detectors, such as solar cells and CCDs, but also semi-conductors are notably affected by this mechanism characterised by a TID measure[7]. A summary of all these effects together with ground test approaches is presented in Tab. 2.6.

[5]Bremsstrahlung designates the radiation emitted by a charged particle when it is submitted to an acceleration.

[6]A design approach consisting in implementing a given function thrice together with a voting logic to protect the system against the consequences of SEUs.

[7]It is measured in rads (obsolete unit of absorbed radiation): the dose causing 0.01 joule of energy to be absorbed per kilogram of matter). For programmable electronic devices TDI translates into leakage currents impacting on the power consumption or altered timing building up to functional failure.

Radiation effect	Parameter	Test means
Electronic component degradation	TID	radioactive sources (e.g. ^{60}Co)
		particle beams (e^-, proton)
Material degradation	TID	radioactive sources (e.g. ^{60}Co)
		particle beams (e^-, proton)
Material degradation (bulk damage)	NIEL	proton beams
CCD and sensor degradation	NIEL	proton beams
Solar cell degradation	NIEL	proton beams (low energy)
	equivalent fluence	
SEU and latch-up	LET spectra (ions)	heavy ion particle beams
	proton energy spectra	proton particle beams
	explicit SEU/SEL rate of devices	
Sensor interference (background signals)	flux above above energy threshold	radioactive sources
	flux threshold	particle beams
	explicit background rate	
Internal electrostatic charging	electron flux and fluence	electron beams
	dielectric E-field	discharge characterisation

Table 2.6: Principal effects induced by radiation and parameters used for quantification (from [22]).

2.4 Quality assurance

With the operating conditions presented in sections 2.3.1 and 2.3.2, the project's stringent functional and performance specifications (section 2.5.2) are, unfortunately, only part of the criteria for selecting a particular device. Foremost among all is the concern for reliability: not only should devices be designed to withstand the harshness of launch and space but the manufactured parts must themselves be flawless since intervention in orbit is either risky and incredibly expensive or quite simply impossible. This is followed by considerations on availability. While these are related to project management, they are not less important for the success of the mission because of their impact on the schedule, on hence, on the testability of the successive models – not to mention missions with limited opportunities for launch, like Rosetta, for which readiness at the precise date is of paramount importance. Last, but not least, comes cost. In spite of the substantial budgets for space missions, it is not infrequent for entire subsystems or instruments to be cancelled at advanced stages in the development, or even during fabrication, due to cost problems. Part of the reason for this is that late changes (in the specifications or in the design) subsequent to the discovery of issues during or after the integration of subsystems can represent expenses up to several million euros. Hence, the importance of long term planning, phases, reviews and multiple models representative of various aspects of the system (mechanical, thermal, functional etc.).

2.4.1 Component policy

As far as parts are concerned, the ECSS standards[8] specify detailed procedures for selection and procurement ([23] and [24]) but agencies, such as CNES, push the logic a step further with true "component policies". The fundamental problem of procurement is to ensure that the one part which will be assembled on the FM meets its own specifications. Given the targeted application, it is not sufficient to rely on the manufacturers' advertisement only because, in this case, quality is not a matter of statistics but of units.

The parts' rating is, accordingly, only the starting point. Four main normative systems coexist for part qualification: the "MIL" standards originally devised for military systems, ESCC or GSFC for space and CECC for the commercial and industrial grade (COTS). Manufacturers are, as a matter of fact, also subject to certification[9] in an attempt to enforce adequate practices for production, testing and storage to obtain maximum confidence as to quality. Finally, experience databases are maintained and shared by the clients as a basis for decision-making.

In spite of all this, the clients very often proceed to detailed inspection either of the FM units themselves or, if potentially harmful, with parts from the same lot. This is especially true when space qualification has not been carried out by the manufacturer. As an example, CNES allocates from 3 to 9 months of labour and 15 to 150 kilo-euros per project for electronic microscopy, thermal imagery with scan laser heating or micro-volt and micro-ampere testing [18]. It is worth stressing how difficult this task necessarily is without any detailed design information from the manufacturer concerning the parts. Although space-qualified technology is notably backward as compared to COTS

[8]See also documents and procedures issued by the ESCC.

[9]"Agréement de savoir-faire" for CNES, QML class V for NASA.

and very significantly more expensive, the difficulty and cost of these verifications and the level of risk which inevitably remains makes COTS solutions undesirable unless unavoidable due to the required level of performance.

2.4.2 Availability

As developments progress, a number of models are produced to validate the design progressively and identify issues early. Demonstrators first serve for establishing feasibility before tolerance to the expected mechanical, thermal and radiation environments is gradually addressed. Parts with the required level of performance and reliability must be available for each of these phases. Fig. 2.3 shows Maxwell's development plan for their single-board space-qualified computer which illustrates a possible sequence of models – although modern mission management strives to decrease the number of models for cost and time-saving reasons. This figure dates back to the end of the PDHE TDA which implies that its demonstrator could only be representative of final performances to the extent that Maxwell's "proto-engineering module" was of the corresponding FM – something not as simple to assess as one could think.

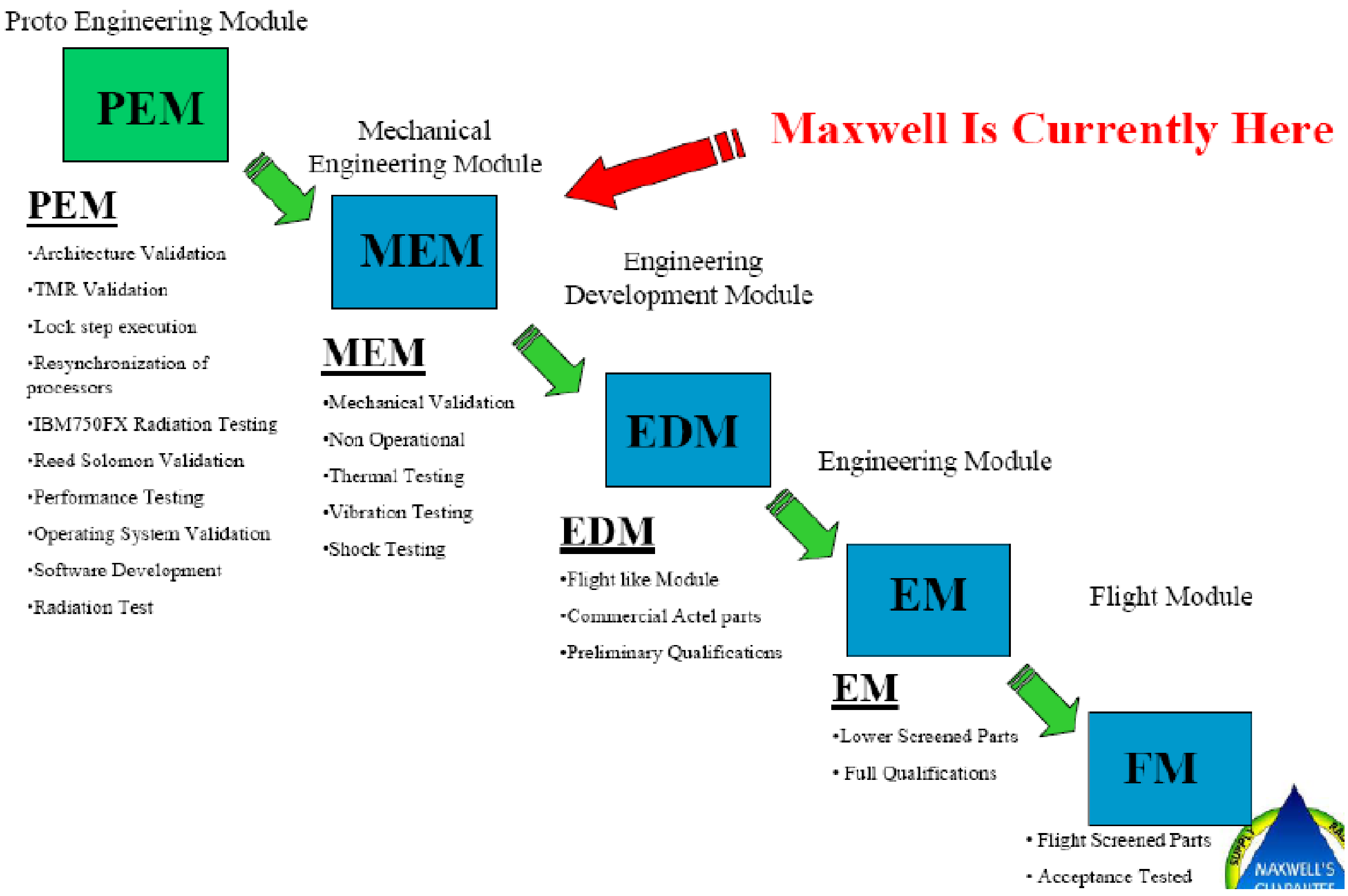


Figure 2.3: Maxwell's development plan for the single board PPC750FX-based space computer (SCS750) and status at the end of the PDHE TDA (courtesy of Astrium GmbH [25]).

Another apparently marginal constraint stems from export and import regulations. This is specific to the restrictions the USA (ITAR) place on items believed by the Department of State to represent a strategic advantage. Sharing such technologies with non-US persons is then submitted to prior approval. Beside the procurement uncertainty and delay incurred as a result of such demands, constraints may be imposed as to the nationality of the personnel manipulating the satellite, even only for launch. Unfortunately most of the high performance space-qualified electronics manufactured in the USA are submitted to ITAR: eg. Maxwell's SCS750, Actel's large RTAX-S dies etc.

2.4.3 Qualification

At the end of the development and manufacturing processes, the proposed design must undergo electromagnetic compatibility testing and qualification (QR) before being accepted (AR). The first is relevant to ensuring the device will integrate properly in the complete system. To this end, it is necessary to verify that it is neither sensitive to the natural and satellite-induced electromagnetic environments nor that it will generate perturbations on the other equipments. With very high precision instrumentation, it is not infrequent for instance for cross-talk between CCDs or the proximity electronics to occur and result in patterns superimposed on the expected image data.

As per [26], qualification is a combination of verification and validation, that is, an assessment of how the design respectively meets its own specifications and how these fulfil the intended use. Formally speaking, the former consists in correctness, consistency, technical and normative considerations while the latter should establish that performance, reliability and availability objectives are met.

2.5 Architecture

2.5.1 Paradigms

Historically, a class of star-shaped architectures handling limited data flows for TM/TC has first prevailed based on hardware with modest capabilities. In such systems, the TC decoder and the TM encoder are directly connected to the various equipments via point-to-point links. Although simple in theory, this scheme has faced increasing difficulties in providing the required level of reliability for more complex systems. The need for centralised management (to switch to a safe mode or to perform overall reconfiguration) together with the obligation of preventing single-point failures have called for significant increases in the number of connections to the point of making the approach quite simply impracticable. As an example, Intelsat VI did embark a total of 13 km of cable for a total of 132 kg. Accordingly, nowadays, this approach is used only for micro or mini satellites.

As a solution, modular architectures equipped with data buses were then introduced in which a CTU controls a set of RTUs and ITUs themselves connected to the equipments. Several norms compete for the intra-satellite transfers performed through this bus: MIL-std-1553 is the defacto standard but ESA promotes alternatives like the OBDH (low data rates) or the SpaceWire technologies (high data rates). Although the presence of such buses opens the possibility for widely interconnected nodes, the legacy of star-shaped systems and the necessity of isolating faults still favour a segmentation in subsystems communicating through the bus only as a pipeline instead of with full flexibility (Fig. 2.4).

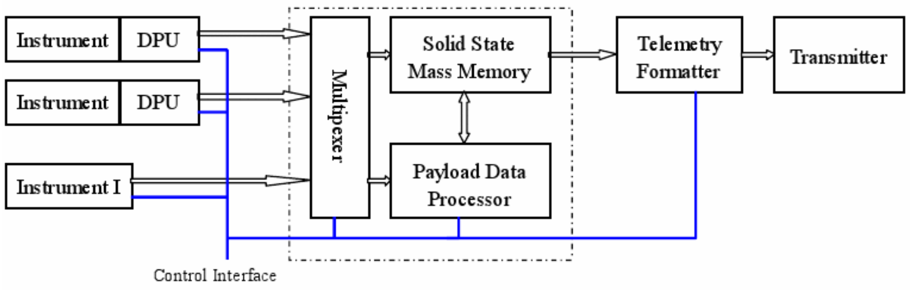


Figure 2.4: Advanced OBDH with decentralised processing units (DPU) and intelligent mass memory (courtesy of P. Armbruster et al. [27]).

Missions like XMM, Integral [28], Rosetta or the NGST have adopted this paradigm. It is also the case for Gaia as Fig. 2.5 shows. For Gaia, the VPUs attached to each row of the FPA in Fig 2.4 are the DPUs which communicate with the SSMM through a supervisor embedded in the CDMU. Multiple buses are planned to support point-to-point communications. Additionally, the VPUs are responsible for the processing of section 2.2 and kept independent for redundancy reasons in case of loss of a CCD row or of a VPU.

An overhead is associated to such an intermediate architecture with the coexistence of multiple buses: the VPU PCBs in Fig. 2.5 are necessarily equipped with two components, one for the SpaceWire interface, the other for the 1553 bus. Beside the design and verification simplifications which using the same technology would yield, connecting everything physically to a single bus would also offer advantages as to dynamic reconfiguration of the payload processing resources, for instance for reallocating a VPU to a different CCD row after failure of a VPU to benefit from better optical quality or for the management of multiple failures. This is probably the future of spaceborne PDHE systems as [27] or [29] point out.

2.5.2 PDHS

Architectural optimisations are all the more relevant for mutualizing resources or providing reconfiguration capabilities as the corresponding electronics, with its reduced degree of integration and its redundancies due to radiations, should

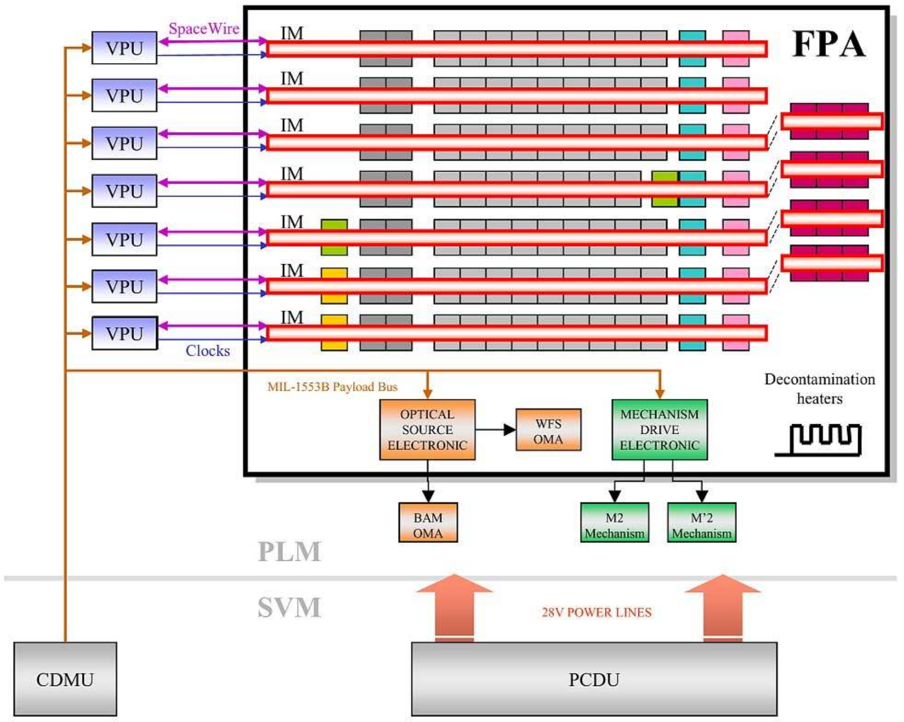


Figure 2.5: Foreseen architecture for payload data handling on board Gaia (courtesy of Astrium).

fit into the allocated mass, volume, power and thermal budgets. Although one might think the first two to be negligible compared to other subsystems in the satellite, tight constraints impose strict optimisation of all levels. With flight-qualified components, the design proposed by Laben as a conclusion for the PDHE TDA for the supervisor and the SSMM totalled 14 PCBs, weighing 18 kg and occupying a volume of 274,5 mm (width) x 300mm (depth) x 250mm (height) for a total power consumption estimated between 47 and 58 W [25]. Although based on different technologies, Fig. 2.6 illustrates an SSMM developed for Cluster of comparable bulk.

The design of the VPU is no less constrained since it is to be instantiated 7 times. Assuming a mixed architecture is selected (as became the baseline after the conclusions of the PDHE TDA) and each VPU is composed of one Maxwell SCS750 board and a secondary PCB supporting an FPGA, 14 boards will again need to be accommodated with emphasis, this time, on harness because of the multiple SpaceWire links to the FPA and power as a result of the 21 PPC750FX processors.

2.6 Conclusion

Gaia's PDHS, introduced as vital for the scientific return of the mission in chapter 1, represents a challenge, not only because of the innovation of performing so much processing on board, but also because of the intrinsic complexity and variety of the tasks to be performed. This chapter has additionally shown that the environment and the technology are also key elements in the problem because of the additional constraints they impose.

Having thus fully presented the context, both scientifically and technically, part II will proceed with the derivation of algorithms to identify the objects of interest. After what has been said, particular attention will be devoted to proposing a generic approach compatible with the intricacies of the real sky and to efficiently mapping it to the intended architecture. Implementation considerations will then follow in part III securing the approach with the elaboration of a hardware demonstrator.

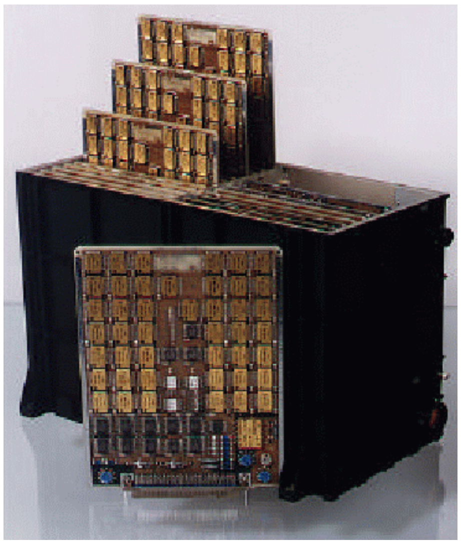


Figure 2.6: SSMM developed for Cluster (1994) by Astrium GmbH (courtesy of Astrium [30]).

Part II

Algorithms

Chapter 3

Algorithmic framework

Contents

3.1 Introduction

After the presentation in Part I of the Gaia mission with its on-board processing needs (chapter 1) and of the general context in which this can be carried out (chapter 2), this part focuses on the algorithms employed to this end. This first chapter derives lower level requirements and discusses some possible approaches with a view to implementation. Then the successive steps are derived in the subsequent chapters. In this part, emphasis is placed on the scientific performances, although care is taken to adopt a formulation compatible with the intended platform. As a result, the descriptions in this part are not more detailed than is typically necessary for building a software prototype – they are further refined in the course of the port to hardware which is the object of part III.

3.1.1 A bit of history. . .

Thanks to an early start, to our collaboration in the PDHE TDA and to the PDH-S activity, the rationale underlying this part and the next has participated in shaping the evolutions of the design – or at least, better said, the latter have not been orthogonal to the conclusions we have drawn. As a consequence, our endeavours have not been made obsolete by its evolutions.

For QA reasons, the preparation of space missions is divided into phases concluded by major project assessment procedures. After an informal phase during which a mission's concept emerges and is proposed to a space agency based essentially on its return, it evolves into phase A[1] whose objective is to formalise specifications and establish feasibility by both scientific and technical studies. As a result, an ITT is emitted by the agency for the selection of a prime contractor for the subsequent phases consisting in preliminary definition (phase B), detailed definition (phase C), production (phase D) before utilisation itself (phase E).

[1]Feasibility phase according to [31].

The algorithms presented in this part were accordingly drafted by the OBDH working group in the frame of phase A, mainly to assist in the preparation of the corresponding specifications for Gaia. When a parallel TDA contract was started to dimension the on-board electronic subsystem, the contractor (Astrium GmbH) decided to rely on Pyxis after some discussion. A scheme was proposed based on a mixed architecture after their first round of analysis but the corresponding "rough order of magnitude" estimates for computing resources were found to be incompatible with the architecture supported by ESA (including a Leon processor developed in collaboration with Gaisler research). When the decision was taken to alleviate this constraint and open the possibility of using computing devices available in the USA, Maxwell's SCS750 board replaced the Leon for a substantial increase in the expected MIPS. Accordingly, the architecture for running Pyxis evolved to an all-software one operating under VxWorks and a soft[2] real time port was carried out.

The conclusions of this breadboarding activity are collected in [32] and confirm the need for a mixed architecture due to the important number of tasks slowing down operation due to slow memory accesses [3] and frequent context switches impeding cache operation. Besides, the management of the interface with the FPA[4] was shown to amount to 50% of a TDI period (0.736 ms at the time). A mixed architecture was hence recommended, using dedicated hardware to alleviate the load on the processor by handling systematic and high data-rate tasks.

During the period which ensued the on-board processing became the focus of much interest from the industrial teams for the preparation of their respective responses to the ITT. A variety of prototypes were then evaluated, some with assembly language, others with VHDL, to come up with an original answer to this challenging question and a competitive advantage. Eventually, Astrium was selected as prime contractor with VPUs comprising Maxwell's SCS750 and secondary boards for one or more FPGAs but based on "in-house" algorithms.

We have collaborated with Astrium GmbH in support of the PDHE TDA in the frame of the PDH-S contract. Beside document reviews, explanations on the algorithms, assistance for porting and participation to evaluation and the preparation of conclusions, the algorithms also underwent in-depth modifications, first for the mixed implantation:

- threaded implementation to model the components and their interfaces,
- line-based processing respecting the ordering of samples resulting from the read-out of the TDI-operated CCDs (raster order),
- introduction of fixed-point arithmetics in replacement of floating-point for portability to FPGAs,

then for the purely software one:

- purely sequential implementation,
- bit-wise comparable results across different platforms,
- platform-independent algorithmic optimisations,
- static memory allocation for performance and minimum intervention of the OS.

As a consequence, when Astrium abandoned the algorithmic framework proposed, it was decided to pursue the migration towards a demonstrator which had already begun since the ALGOL ACI to which we participated offered an opportunity for this kind of R&D.

3.1.2 Models

Several different implementations have been conducted during these successive phases. The GD piece of software, coded in C as early as 2002, was primarily intended as a test means to draft the scientific specifications. Retained as the basis for the PDHE TDA, several versions followed, gradually narrowing the first naive approach to one hopefully more realistic. In this course, the decision was taken to rely on a mixed architecture and an entirely recoded Matlab version of the detection was then written by P. Laporte [33] to prepare the migration to hardware and address some of the specific intricacies which are the object of part III. These models have been of considerable and varied use. They have permitted testing the strategies against simulated data to improve the algorithms as regard the needs. They have allowed pinpointing key difficulties which have been reported in [SM20] as a complement to ESA's specifications. They have provided figures for the architectural design and have, hopefully, inspired Astrium for their own developments through the PDH-S activity. And last, but not least, they have served as reference for debugging the hardware and verifying it at the bit level.

[2]As opposed to one in which the different tasks would be assigned strictly enforced time slots (known as "hard" real time).

[3]With the SCS750, exchanges with the memory are subject to EDAC and inspection by the FPGA responsible for synchronising the processors functioning in TMR. With the FPGA included in the "proto engineering module" used, exchanges were then limited to 50 MHz.

[4]The necessity to format commands and decode data packets from 15 CCDs (with the baseline design of phase A) even in the absence of objects of interest due to the systematic read-out of a fixed number of sample for thermal reasons.

3.2 Image model

This section builds a model of the different steps involved in forming the images to be analysed. Central properties verified by the images are derived in this course to provide grounds for discussing the image processing approaches in section 3.4 below. A simple model of the generation of noise during these processes is also built to assess the impact of pre-calibration on the data (section 4.3.1) and the reliability of background estimates (section 5.2.3).

3.2.1 Acquisition

Incoming photons

The objects of interest are, in the majority, thermal sources and secondary ones diffusing photons (solar system objects, gas, dust etc.). In both cases, the integration time for each phase of the TDI is much longer than the coherence time so that the arrivals of photons can be assumed uncorrelated. As a consequence, the photons incident on the CCDs follow a Poisson statistic.

Optical image sphere

As a general rule, the sources being located at infinity, the statistic above can be considered as weighing a distribution of Diracs representing the positions of the point-like objects on the celestial sphere. Extended objects are an exception to this rule but may be approximated with distributions of Diracs corresponding to elementary emitting surfaces. Parallel rays are, accordingly, input to Gaia's optical system and the resulting image can be simply predicted by means of the latter's impulse response, or PSF as it is generally denominated in astronomy. This quantity characterises the deformation of the optical surface both in terms of diffraction occurring due to the finite aperture of the telescope and in terms of WFE. Gaia's rectangular mirrors lead to cross-shaped images while the WFE imply that images differ depending on the location of the Diracs on the pupil.

Additionally, a geometric mapping transforms the input fraction of the celestial sphere into a region on the FPA (the projection of the image focal surface onto the physical plane of the FPA). This effect affects the location of the PSF in the AL and AC directions before it is sampled by the detectors. Due to the apparent movement of stars across the FOVs in the case of Gaia and the TDI operation of the CCDs, the observed PSFs result from the integration of all instantaneous PSFs at the positions occupied by the object during the exposure. Accordingly, the profile is dependent both on the location of the object within the aperture and on the satellite's attitude.

Integration

The sampling of the optical signal is subject to a trade-off between the pixel size, the value of data and the resolution obtained. With a feasibility constraint affecting the physical size of pixels in state-of-the-art CCD technology (chapter 1), the sampling and optical design need to be optimised jointly. As finer sampling leads to more data, optimality is sought in the sense of minimising this amount while allowing to retrieve a number of moments describing the profile without bias. The decision to perform only one-dimensional measurements implies that the AC resolution is only useful insofar as separation of objects is concerned to allow individual observations to be made in a majority of cases (section 1.3.3). Hence, the use of rectangular mirrors which favour the AL resolution within the predetermined aperture. As the sampling needs to be adjusted accordingly, rectangular pixels are used with the same proportions as the mirrors'. This compensates the difference in dispersion in the two directions and, with pixel values represented on a square grid, results in balanced profiles as is represented in Fig. 3.1 and in all images in this text.

The TDI operation of the CCDs implies that a read-out value does not correspond to the response of a single location on the detector but rather to the sum of all phases (four per pixel) within the AL column. Let each of the four phases corresponding to the 4500 pixels be labelled with $i \in I = \{0, 4500 \times 4 - 1\}$ and $\{p_i \mid i \in I\}$ denote the number of incident photons on each phase in a given column (Poisson variables).

Considering the entire band in which Gaia's CCDs are sensitive, conversion of the signal from photons to electrons is governed by the detector's quantum efficiency. As sensitivity cannot be assumed uniform over the detector due to PRNU (black pixels in Tab. 1.3), a set of coefficients $\{0 \leq e_i \leq 1 \mid i \in I\}$ must be introduced instead of a single multiplicative constant to model this efficiency. Additionally, charge generation occurs due to thermal noise (DSNU) and white pixel defects translating in additional Poisson contributions $\{d_i \mid i \in I\}$. The total electronic signal reaching the read-out register is then:

$$S_{e^-} = \sum_{i=0}^{i=17999} e_i . p_i + d_i \tag{3.1}$$

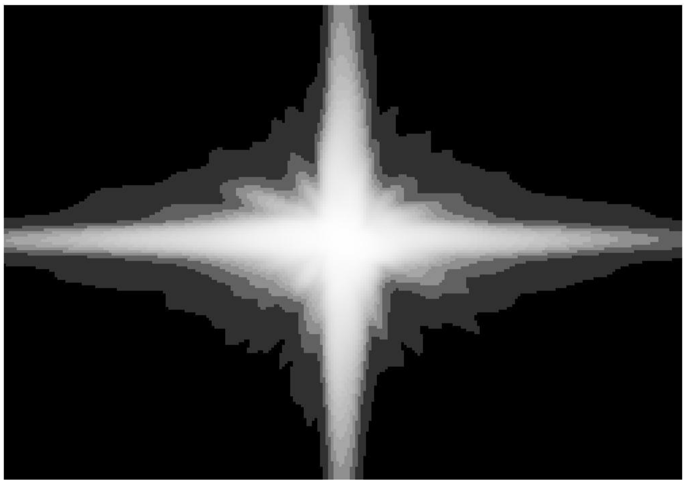


Figure 3.1: Typical noiseless effective "static" PSF in one of Gaia's SM CCD as simulated by Gibis 3.1 ("static" refers to integration by the rectangular pixel grid in the absence of attitude-related apparent velocity (in AL and AC)).

Assuming an idealised case in which the movement of a source perfectly matches the charge displacement in the matrix, all p_i share the same expectation λ (and may be written as p simply). The charge generation may be modelled globally to transform the sum of d_i into a single term D variable from column to column because of thermal gradients on the CCDs and of "white" pixel defects.

$$S_{e^-} = (\sum_{i=0}^{i=17999} e_i.p) + D \tag{3.2}$$

D is a Poisson variable[5] but not $(\sum_{i=0}^{i=17999} e_i.p)$. The probability mass function for each $e_i.p$ term is given by:

$$g_i(n) = \frac{1}{e_i} f_\lambda(\frac{n}{e_i}) \qquad \text{if} \qquad f_\lambda(n) = \frac{e^{-\lambda}\lambda^n}{n!} \tag{3.3}$$

is the one associated to p. Hence, excluding the 6 pixels which are masked ($e_i = 0$), the equivalent for $(\sum_{i=0}^{i=17999} e_i.p)$ is

$$G(n) = \frac{1}{\prod_{i=0}^{17975} e_i}.f_{17975.\lambda}(\frac{n}{\prod_{i=0}^{17975} e_i}) \tag{3.4}$$

assuming the acquisition during each of the phases to be independent from the other ones.

Read-out

With the decision to bin SM pixels in 2×2 samples which will be justified in section 3.5.2 below, the data is not read pixel per pixel but in groups of four in a grid of fixed geometry (samples hereafter). Using superscripts to denote the two contiguous columns from which the charges are collected and subscripts for the two TDI periods, $S_{e^-}{}^1_1$, $S_{e^-}{}^2_1$ $S_{e^-}{}^1_2$ and $S_{e^-}{}^2_2$ represent the four electron-count variables for the pixels merged into the single SM sample.

The CCD read-out is a complex chain involving a variety of processes contributing to the noise:

- an electronic noise term corresponding to the operation of the shift register (the one dependent on the frequency in section 1.3.3) [6],
- the KTC noise[7],
- the ADC noise including quantisation (by round-off)

[5] As a sum of Poisson variables.

[6] Strictly speaking, this term also includes the jitter during the TDI charge shift within the CCD.

[7] Noise associated with the gate capacitor of an FET with RMS value $\sigma = \sqrt{\frac{kT}{C}}$ where k is Boltzmann's constant and C is the FET gate switch capacitance.

- and various coupling and electromagnetic compatibility noises (cross-talk etc.)[8]

For Gaia, these are globally modelled as a centred Gaussian additive contribution with the standard deviation σ_{RON} provided in Tab. 1.3. The model applicable to the resulting electron count is then:

$$\begin{aligned}\tilde{S}_{e^-} &= S_{e^-}{}^1_1 + S_{e^-}{}^2_1 + S_{e^-}{}^1_2 + S_{e^-}{}^2_2 + N(0,\sigma_{RON}) && (3.5)\\ &= (\sum_{i=0}^{i=17999} e^1_i.(p^1_1+p^1_2)) + (\sum_{i=0}^{i=17999} e^2_i.(p^2_1+p^2_2)) + D^1_1 + D^1_2 + D^1_1 + D^2_2 + N(0,\sigma_{RON}) && (3.6)\end{aligned}$$

although it is rather encoded in ADUs according to:

$$\tilde{S}_{ADU} = \frac{\tilde{S}_{e^-} + offset}{gain} \text{ with } offset = 1000 \times gain \text{ and } gain = 4.07 \tag{3.7}$$

3.2.2 Simulation

An acquisition model following the guidelines above exists as an image simulator within the Gaia project. It corresponds to an effort begun during phase A by the SWG to provide test data coherent with the latest assumptions[9] to the community for a variety of studies related to the instrument and the scientific return. The general developments and tests of Pyxis have been based on data produced by several versions of the Gibis simulator [34]. As it assumes ideal CCDs as of this writing, all e_i reduce to 1 and all D terms share the same parameter value. Hence the model becomes simply:

$$S_{e^-} = 17975.(p^1_1 + p^2_1 + p^1_2 + p^2_2) + D^1_1 + D^2_1 + D^1_2 + D^2_2 + N(0,\sigma_{RON}) \tag{3.8}$$

Beside improvements in the simulator itself, significant updates have taken place to follow the evolving instrument concept (chapter 1). Two versions deserve to be mentioned here (Gibis 2 and 3), one for the baseline design applicable during phase A (2002), the other for Astrium's final proposal (2006). As Tab. 3.1 shows, the main changes as far as SM imaging is concerned are related to a longer TDI period and a shorter focal length. The former should have led to increased SNRs but the decision to activate the first gate permanently for optical quality reasons and to reduce the sensitivity to PPEs cut down exposure time by 14 %. This is somewhat compensated by the reduction of the gain which allows for more finely representing the signal at the low end and decreases the quantification noise (see RON budgets in Tab. 3.1). The latter leads to a lower angular resolution in both AL and AC by approximately 35 % which, itself, results in increased crowding.

Parameter	2002 Baseline	2006 Design
focal length	47 m	35 m
pixel angular size AL	44.1 mas	59.8 mas
pixel angular size AC	132.5 mas	176.8 mas
TDI	0.7361 ms	0.9828 ms
gates	none	gate 12
SM exposure	3.31 s	2.85 s (gated) but 4.42 s (total)
zero point	25.6 mag (unchanged)	
gain	5.92	4.07
offset	1000 ADU (unchanged)	
saturation	58122 ADU	65535 ADU
RON (σ_{RON})	14.36 e^-	10.85 e^-
CCD properties	unchanged	
WFE	modified	

Table 3.1: Key differences in the design parameters impacting on the SM imaging.

Visual comparison is possible in Fig. 3.2 based on a high density case simulated in the two configurations. Although quantitatively, the two are measurably different, they are qualitatively comparable. Because the instrument and Gibis

[8] Whereas this latter class may be considered as corresponding to self-correlated noise, the realisations are merely modelled as independent in what follows. This is partly due to the difficulty of determining the precise nature of the correlations without measurements on a representative experimental setup.

[9] Beside the centralised simulation developments, ESA has set up a parameter database covering the instrument, the mission, the Universe as well as general physics to further promote consistency within the project.

are in active development and, hence, undergoing frequent change since the beginning of phase B, it has proven difficult to present a homogeneous set of results relevant to a single stable and up-to-date version. Accordingly, most of the following chapters' performance measurements are based on data corresponding to the outdated 2002 baseline except where explicitly indicated. As expected, the main noticeable differences concern:

- the spatial distribution of objects (which are more crowded as a result of the shorter focal length in the 2007 design),
- the noise level (which is slightly lower in the 2006 design),
- the histograms of sample values (which is more spread-out in the 2006 design because of the lower gain value: 4.07 ADU/e^- instead of 5.92 ADU/e^-).

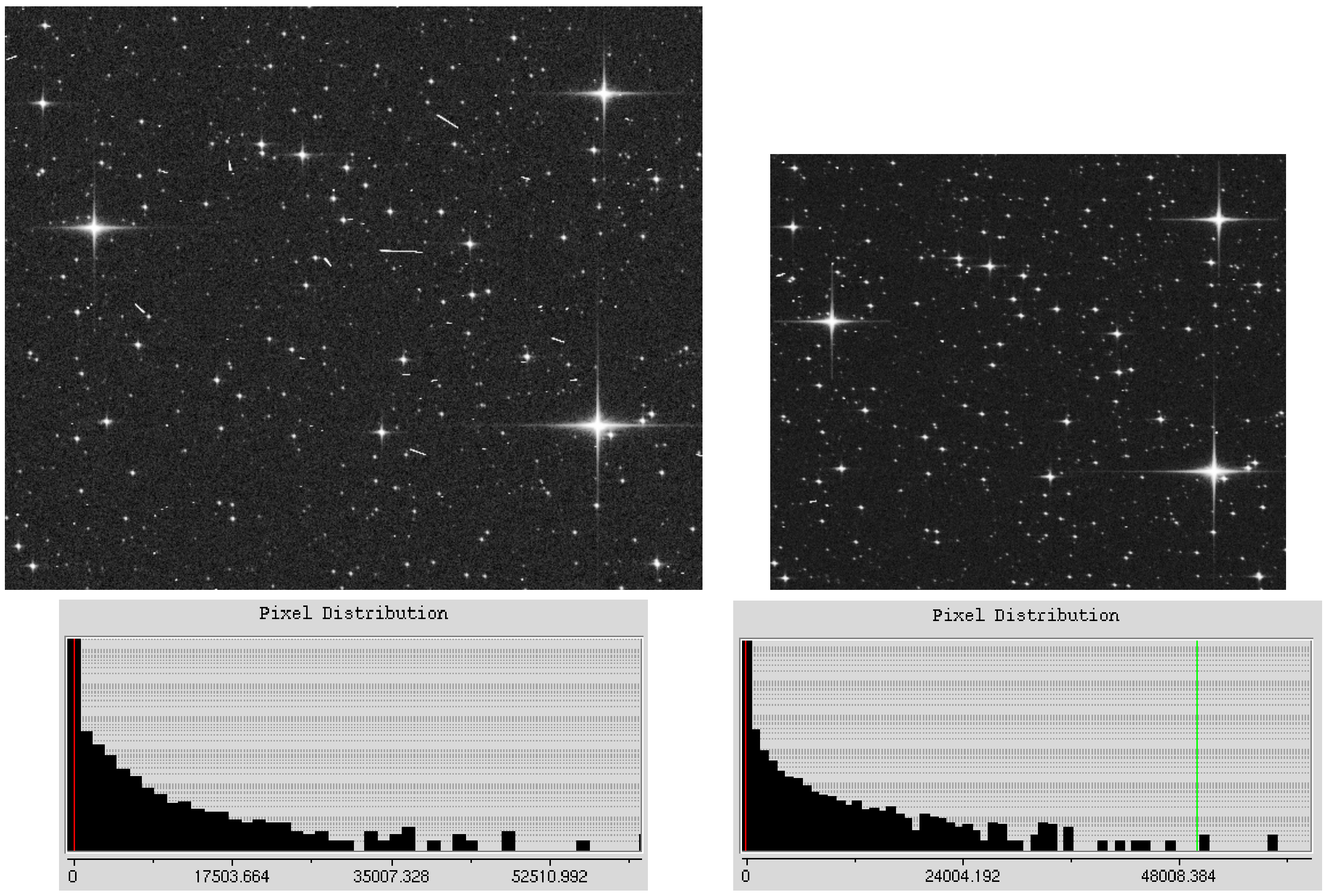


Figure 3.2: Illustration of the impact of the design changes on SM imaging with a high density synthetics example (left: 2002 baseline, right: 2006 design). The two images share the same scale and dynamics.

3.3 Needs

As discussed in chapter 2, the specifications published in the MRD are notably too high level to provide any real guideline for the design of the "Science application". Instead, the prime contractor, based on his design and understanding of the intended use, is responsible for elaborating an URD submitted to ESA as part of the phase B reviews. This document collects a number of functional and performance requirements which then form the backbone of the developments. In ECSS terms [26], "validation"[10] will then establish that these will ensure that the higher level spec-

[10]"Confirmation by examination and provision of objective evidence that the particular requirements for a specific intended use are fulfilled (ISO 8402:1994). The validation process (for software): to ensure that the requirements baseline functions and performances are correctly and completely implemented in the final product."

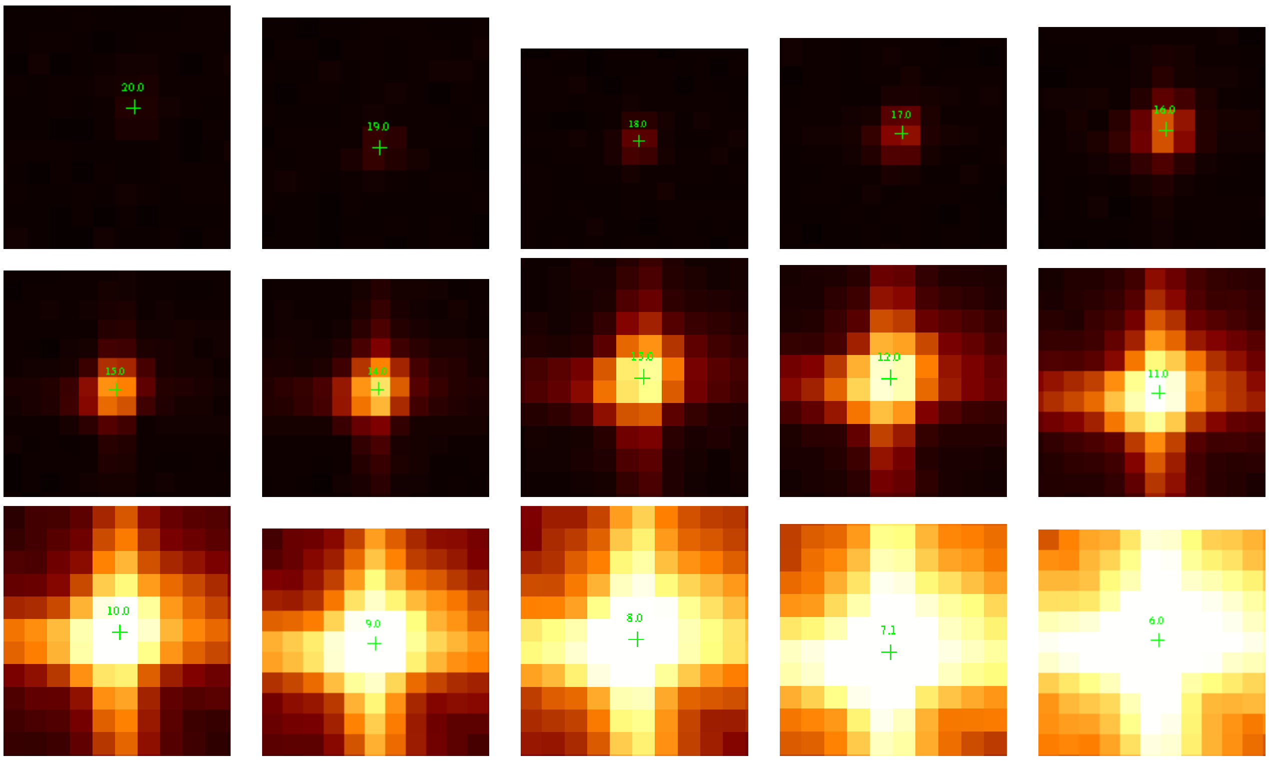


Figure 3.3: Two dimensional snapshots of objects in SM (in 2×2 samples) from magnitude 20 to 6 per steps of one magnitude from left to right and top to bottom (images use a logarithmic colour scale).

ifications of the MRD will be met, while "verification"[11] will assess the extent to which the final product fulfils the URD requirements. The combination of the two, known as "qualification"[12] is the object of the QR prior to acceptance (AR).

As part of the phase A conclusions for the PDH-S activity, ESA requested that a design-independent set of inputs for this URD be produced to facilitate the transmission of the expertise gathered to the prime contractor [SM20]. These inputs result from an interpretation of the specifications (Spec.) already cited in this document but have only existed for reference at the project level – that is for information instead of being applicable. We briefly present items connected with the detection of objects in this section as a basis for the discussion on methods in section 3.4 and to introduce the rationale behind the algorithms described in this part.

Requirement 1 *Detection shall be an autonomous process.*

Requirement 2 *Detection shall ensure that the sources of interest can be observed and should minimise the number of false detections and objects outside the magnitude range (Spec. 2 & 3).*

Although these two top-level requirements are only natural expressions of the basic functional specification for detection, the preoccupation related to making the best possible use of the transmission capabilities is placed at the heart of the design by Req. 2. Acknowledging that Gaia will not offer the opportunity to observe everything and more, Req. 3 below refines this further in the sense of controlling selection biases.

Requirement 3 *The probability of detecting an isolated source (i.e. not part of a DMS) of a given magnitude shall be uniform (i.e. AC) and quasi-stationary (i.e. AL).*

Req. 4 follows from 3: to stand a chance of finding objects anywhere and at any time, the complete content from the two FOVs must be accessible. Although compatible with the thermal stability objective for the payload as the number of samples read remains constant, full read-out is undesirable for the scientific exploitation of the data due to the significantly higher electronic noise. Accordingly, the SMs are dedicated to detection, which in turn opens the possibility to degrade the resolution through binning (section 1.3.3) to reduce the amount of processing and enhance the detection capabilities.

[11]"Confirmation by examination and provision of objective evidence that specified requirements have been fulfilled (ISO8402:1994). The verification process (for software): to establish that adequate specifications and inputs exist for any activity, and that the outputs of the activities are correct and consistent with the specifications and input."

[12]"'Qualification engineering' is used in this Standard to denote 'the total set of verification and validation activities'."

Requirement 4 *Objects shall be formed as sources enter the focal plane using the data from SM. For this reason SM shall be entirely read-out, though the AL & AC resolution may be decreased via specific sampling.*

The concerns expressed in Req. 2 & 3 imply the need for monitoring the quality of raw CCD data on input, not only within each CCD to enforce "local" uniformity but also between SM1 and SM2 to balance the two FOVs and, at a larger scale, between CCD rows for the overall management of data on board (selection, storage and telemetry). An on-board calibration engine results, described by Req. 5 and 6.

Requirement 5 *Cosmetic defects (blemishes, dead pixels) shall be corrected to avoid irrelevant sample values.*

Requirement 6 *The intra and inter-CCD photo-response non-uniformity shall be corrected as a prerequisite for uniform detection and confirmation probabilities.*

Beside the detectors' noise addressed by the read-out of a constant number of samples and Req. 5 & 6, the signal is subject to contamination by other noises (section 3.2.1 above). Stationary ones impact mainly on the precision and accuracy of measurements and hence on the possibility to compare them among themselves or against the bounds of the magnitude range of interest. Conversely, non-stationary ones, like the local background contribution, introduces spatial heterogeneity resulting in an excess or depleted amounts of data in certain regions. Robustness to noise and "environmental" parameters is simply addressed by Req. 7.

Requirement 7 *For each sample, the signal shall be estimated versus the total noise. The local sky background contribution shall hence be estimated.*

Although highly homogeneous and predictable as a consequence of the previous requirements, the performances of detection remain to be specified. Spec. 1 sets a lower bound as undetected objects will automatically be missing for the transit. Additionally, margin must be taken to account for failures at the selection or confirmation levels, so a 98% rate is proposed in Req. 8. It is worth noting that Spec. 1 imposes constraints on observation and not directly on on-board detection. One case in which the two differ corresponds to DMS for which the secondary is increasingly difficult to detect as it becomes fainter and closer to the primary but is, conversely, more easily observed because it fits inside the latter's window.

Requirement 8 *The detection probability shall be at least 98% over the magnitude range (Spec. 2 & 3) up to and including the design sky density (Spec. 9).*

If Spec. 1 is purely magnitude-based and accordingly applies in all object categories, two special cases deserve to be explicitly mentioned in Req. 9 & 10 to ensure object types are not forgotten.

The first corresponds to bright stars which saturate the SM detector and, as a consequence, appear differently from the calibrated PSF and may paradoxically raise detectability or accuracy concerns. One aspect calling for attention relates to the area over which saturated stars' images spread on the CCD. It translates, from a processing point of view, into potential data bursts with should not endanger the overall real-time operation. The risk associated to processing delays for these objects is to face causal problems during the progressive data acquisition in the TDI mode: provision should be made to ensure that the detection products become available sufficiently early for both propagation and selection to be carried out in time for the bright object's observation in the subsequent CCDs[13].

A second one applies to solar system objects, which do not appear as point-like sources because of their relative closeness to Gaia, and to distant large scale objects such as galaxies. Again these will be imaged differently (as the convolution between their luminosity distribution and the instrument's PSF) and differ from stellar sources.

Requirement 9 *Saturated objects shall be identified. Their detection may be adapted, e.g. to ensure that the latency requirements are met if they extend on too many TDI lines or call for exaggerate processing resources.*

Requirement 10 *Extended objects shall be identified.*

Req. 8, 9 & 10 should apply for the stellar densities defined in section 2.2.2 (Spec. 9). Spec. 10 opens the possibility of avoiding to design the system for the worst case density by allowing to manage the corresponding known regions with a special procedure. In a manner conforming with ECSS principles, it also provides a guideline for "graceful degradation" beyond the specified maximum density. Instead of relying on a special mode, Req. 11 & 12 specify a mechanism by which the best possible use is always made of the available processing time and problematic regions may be identified. As a prerequisite for this, it should be possible to form precursors to objects at all times.

[13]The tightest constraint in this respect applies to detection in SM2 and configuring the AF1 CCD. Depending on the needs related to charge injection or the state of gates, different latencies apply corresponding respectively to the physical distance between the last and first lines of SM2 and AF1 ($\simeq$ 400 TDIs) or the last line of SM1 and the first gate of AF1 ($\simeq$ 2000 TDIs).

Requirement 11 *Detection of objects shall be priority-driven. To this end, object parameters should be determined jointly with detection and allow for avoiding unnecessary processing (e.g. for identified PPEs).*

Requirement 12 *Whenever objects are known to not have been detected a flag shall be raised (Spec. 10).*

Once detected, objects should be measured according to Spec. 8 as discussed in section 2.2.2. Precision and accuracy for these are necessarily traded-off against resources since simple but insufficiently precise or accurate methods imply the introduction of margin to ensure the specifications are met nevertheless (Req. 2). One example connected to "flux intensity" is particularly relevant in this sense. Imprecise measurements call for observing all objects up to a threshold corresponding to a fainter limit than required to achieve completeness in the desired range. As star counts approximately double between every magnitude, this not only increases the load on the VPU but populates the SSMM with unnecessary data.

A final informal criterion, not to be underestimated, consists in being robust or at least easily adaptable to changes in the design or the simulator. Beside the large scale modifications pointed out in chapter 1, frequent updates to the instrument assumptions or the simulator are made as the project evolves (section 3.2.2). It is of paramount importance to be able to remain synchronised with these for the sake of drawing valid conclusions and making appropriate recommendations to the project. Yet, this should not imply redesign more often than not or else little progress can be made…

3.4 Approaches

The need described above falls in the general category of image interpretation problems. A number of techniques are considered here to tackle this question before one is selected based on criteria related to complexity and the type and amount of resources involved by implementation. Indeed, as a result of the PDHE TDA it became clear that a mixed architecture was desirable whatever the method chosen for detection because of the multiplicity of tasks and amount of data to be manipulated. Even before that, maintaining a balance between volume and complexity was already a central preoccupation for performance reasons. With the introduction of hardware offering the possibility to off-load the high volume / low complexity processing from the processor, this analysis was confirmed and became part of the criteria for evaluating the various methods.

Detecting then measuring objects conceptually involves two steps: searching where the object information is distributed in the image, then exploiting it. Depending on the exact meaning attached to each and the decision to perform them separately or jointly, three main classes of algorithms can be imagined and are discussed below.

3.4.1 Transforms

A first class can be designed by exploiting some a priori knowledge on the characteristics of the objects of interest to merge the two steps. The image can be examined conditionally to these assumptions and modified to concentrate the relevant information at predictable locations. In an ideal case, it then suffices to retrieve the data at these points to simultaneously determine that an object is present and have access to its properties. The strength of such approaches resides precisely in that the transform-space provides somewhat of a complete view of the situation instead of a piece-wise one. Additionally in some cases, using the a priori information, the transform can be optimised, not only to make the interpretation of the result optimal yet simple but also to reduce the cost of the transform itself.

Such a great variety of transforms exist that it would be impossible to list them all here. The linear ones can be generally formulated as convolution products between the image and a well-chosen kernel. Notable example of these are differential operators, inverse filtering or the Fourier transform. More sophisticated schemes use this as a basis and perform further non-linear operations on top of these, like decimation in the case of multi-scale approximations for instance.

Although widely used in image processing, these methods present a number of drawbacks which make them inadequate for our problem. First and foremost comes the fact that they are intrinsically pattern dependent, whether it be for the design of the transform itself or for the analysis of the resulting representation. From a distance, our images seem to result from the convolution of a set of Diracs with a PSF summarising the properties of the optical system so that matched filtering with a kernel resembling this PSF should restore the Diracs and make detection and characterisation extraordinarily easy. An attempt at this is reported in [35] but the method was found inappropriate due to the failing assumption that the optimal kernel is invariant. As a result the residuals adopted a structure too complex to be analysed. Several effects concur to this, in decreasing order of importance:

- the PSF varies from star to star due to chromatic effects,
- the objects have variable relative motions in AL and AC as compared to the movement imposed to the charges by the TDI electronics (due to the scanning law and the projection of the image sphere on the focal plane mainly),

- the CCD response introduces perturbations on the image which are negligible for bright stars but significant at the faint end,
- the PSF varies spatially as a result of optical aberrations.

One could think of making the transform adaptive to improve the situation but this would not be desirable because the second flaw of these approaches is that they are too computationally intensive. The core of the transform itself being generally a convolution, a minimum of one addition and two products need to be perform for every sample if the kernel contains two coefficients and the number of operations grows linearly with the size of the kernel, which is generally much larger[14].

Alternatively, relying on a less specific transform is an option but whereas it improves the worst-case behaviour, the average one becomes more complex[15]. This property is shared by most general transforms and substantial processing is then required to analyse the result in the transform-space – it is can even be necessary, in the worst case, to carry out the inverse transform at some point.

3.4.2 Local analysis

Reverting to two-stage approaches, a seductive strategy as far as complexity and data management are concerned consists in searching for objects and measuring them based only on local information. The idea then consists in only accessing a limited neighbourhood of predetermined geometry around each sample both for deciding whether an object is located there and for measurements if one is.

A prototype piece of software (SWA) was developed within the OBDH working group during phase A [37] to evaluate this concept. The key conclusion drawn from this experiment is that performances are fine for isolated and moderately bright stars. The method however faces difficulties when the thresholds used are relaxed to also encompass faint stars and more complex configurations like textured backgrounds or DMS. The limited amount of information available in the vicinity translates into a difficult trade-off between type one (false positive) and type two errors (false negative) which impacts on completeness or on the rate of false detections.

The detection of saturated stars or extended objects (Req. 9 & 10) also becomes paradoxically problematic because of the radically different profile of the PSF around the centroid due to either saturation or the angular extension of the object. There did not seem to exist any reasonable way of including these objects in SWA's main loop without setting false detections loose simply because the limits of local analysis are reached when dealing with objects imaged on a regional scale. This, in turn, called for the introduction of a separate process dedicated to the detection of such objects. Whereas it may seem natural this is something undesirable in practice for three main reasons:

- the necessity of exactly defining the limit between the two processes to ensure objects are detected once (for completeness) and only once (to avoid redundant processing and conflicts when allocating windows),
- the increased complexity of qualification (two processes to be verified and validated) and the risk associated to introducing seldom used data paths,
- the need to run two processes in parallel sharing the same base data.

3.4.3 Segmentation

Based on the shortcomings of the SWA experience, the decision to perform full-fledged segmentation was finally taken. As a first step, the image is partitioned into a set of domains, each attributed either to an object or to the background, based on which characterisation may proceed in a second phase. Compared to purely local approaches, the additional complexity is counterbalanced by the fact that data units with rich content are formed, thereby making it possible to manage all objects within the same paradigm. The complexity resides in the construction of the domains themselves, as opposed to relying on a predetermined model of the vicinity of objects in the local analysis above, but also in that more object-related data become available as a consequence. While the increase of the amount of data may be an issue for saturated stars, it will only occur after data selection has been carried out and more than 90% of the samples have been discarded as related to the background (in the worst density case). Conversely, forming the domains implies additional processing applied to the bulk of the data – something which needs to be kept in mind as regards the targeted volume/complexity balance sought.

[14]One would think that these repetitive calculations could be advantageously entrusted to the hardware. In the absence of cells specialised in arithmetics in space-qualified FPGAs, this would only be possible provided resources could be shared to a very high degree. Otherwise, given the kernel size and the need for precision, the amount of resources required would simply be overwhelming (section 8.3).

[15]One example of this principle in astronomy is the "Multi-scale Vision Model" [36] which relies on an "à-trous" algorithm to decompose the signal in the space-scale domain. While identifying objects is then reasonably simple, the information is only moderately concentrated as a counterpart for the generic nature of the method, which implies that measurements remain a complex issue.

Boundary-based segmentation algorithms are clearly inappropriate for our problem since the images formed, as the result of convolutions with the instrument's optical response, are intrinsically smooth. Although it might be possible to identify contours by tracking inflexion points in the PSF (Fig. 3.1), this would require applying a differential operator first which is incompatible with keeping the complexity low.

Abandoning edges, region-based approaches may be considered. Clustering and region-growing schemes suffer from being two pass processes[16] but remain promising because of their linear complexity. Both require a complete knowledge of the data and randomly traversing it, however. This is fine when full images are available but problematic when they need to be processed during acquisition as is the case here. An algorithm based on the watershed transform will nevertheless be introduced for identifying components in DMS but only after the segmentation – that is as soon as self-contained object entities have been formed (chapter 7).

Abandoning regions, the image may be tackled as a whole in a manner alike to clustering but at the sample level. Combining a threshold to form a binary mask with connected component labelling, it is then possible to select only the samples of interest and to structure the resulting set in a single pass. Additionally, the image may be traversed in a manner fully coherent with its natural raster order with the benefit of completely predictable data accesses, as in the local approach, in spite of our building more elaborate objects. It is also possible to treat all samples on an equal basis in a manner which not only involves limited arithmetics but is also fully systematic and thus perfectly suited to hardware. Although the resulting objects are admittedly somewhat coarse, this first pass fulfils the objective of filtering out most of the data at a low cost and the model may be refined by a subsequent characterisation step.

3.5 Processing flow

As has been mentioned in the foreword, this work began in phase A with the intent to identify key difficulties as regards object detection for the establishment of the scientific specifications – rather than as an original research project. Accordingly, to secure rapid results at a representative level of performance, an existing solution was selected as a starting point. The considerations above formed the basis for developing a scheme which builds upon the success of ideas which have long been in use in the field of automatic interpretation of astronomical images and successfully put to practice, for example, by the APM facility for large scale photographic plate analysis [2] or by the Sextractor software [3]. M. Irwin proposed a detailed analysis of the sequence in [1] whose initial flow ran:

1. estimate the sky background on a regional basis,
2. generate a two-dimension map by interpolation based on these estimates,
3. filter the samples with a kernel mimicking the PSF to enhance the signal,
4. threshold these values compared to the local background following an SNR criterion,
5. form objects based on CCs,
6. analyse the objects individually.

Fundamentally, the rationale behind this sequence is to retain only regions where there is signal, where signal is interpreted in the sense of a significant addition to the local average level. As a consequence, the procedure is highly generic and, hence, applicable to all objects whose flux is superimposed on the sky background. Gaia being interested in stars which emit light or in solar system objects which diffuse it, this simple assumption holds for all objects of interest (Spec. 1). Besides, thanks to relying on very few and simple assumptions, the processing is non-parametric which implies it is versatile, or rather in our case, easily adaptable to various image models through only a few parameters: mainly the size of the regions for background estimation and the SNR threshold. This is to be compared with local approaches which are bound to verify significantly more constraining hypotheses corresponding to the profile of the objects to filter out false detections and are, accordingly, dependent on more complex calibration.

3.5.1 Software & hardware partition

The PDHE TDA recommended [32] introducing dedicated hardware for the management of recurring, systematic, high-throughput and hard real-time tasks, thus dedicating the processor to more complex ones in the sense of flexibility and computation. The already mentioned handling of the interface with the FPA is a clear example to which this could apply: data transfers must be performed every TDI, repeat identically in time according to the communication protocol, relate to large volumes of data and must meet hard real-time constraints because of the synchronous operation of the

[16]Clustering: a first pass to produce an over-segmented representation of the image and a second to merge regions according to resemblance criteria. Region-growing: a first pass to identify seeds and a second to propagate the seeds' labels until all samples are assigned to a region.

PEM on the FPA side. Software running on a candidate platform such as the Maxwell SCS750 board is particularly ill-adapted for this, not only because of the processing duration but also because of resynchronisation delays[17].

A possibility for distributing resources could consist in introducing a video buffer with some hardware managing the SpaceWire link with the FPA (ie. enforcing the protocol, formatting commands and decoding incoming packets). Although this buffer would decouple the VPU from the FPA to some extent, all data would still be input to be processed: some from SM1 and SM2 (15.37 Mbit/s) for detecting objects, some from AF1 (1.89 Mbit/s) for confirming previous detections and some from AF2-AF9 (2.73 Mbit/s), BP-RP (2.27 Mbit/s) and RVS (3.35 Mbit/s) for packaging and transmission to ground. With a total of 25.61 Mbit/s transiting through the buffer, its size would remain highly sensitive to processing delays.

It is natural, accordingly, to extend the hard real-time operation until the data flow is reduced. Detection and confirmation are excellent candidates for this since they correspond to the bulk of data and consist precisely in substituting an abstract description, in the form of "objects", to the exhaustive list of samples, thereby achieving significant compression. This statement may be refined by noting that object-member samples amount, in worst case stellar density conditions, to less than 10% of input data. Hence, if upstream tasks are designed to proceed identically whether or not stars are present and to handle the data in its natural raster-scan order, the separation between hard and soft real-time, as between hardware and software may be drawn when object entities are formed.

Not only does this relieve the processor from all sample-based manipulations, but an improvement in efficiency results from the hardware's fixed scheduling and finely controlled access to dedicated memory resources. The processor is left to deal with object-wise analysis, something for which it is well suited as arrival times are inherently random and as complexity[18] varies greatly with object types (ie. the number of member samples or the need to further segment components). Thus, the balance achieved does not only correspond to real-time constraints but also to a volume/flexibility trade-off.

3.5.2 Overview

This section presents an overview of the sequence of tasks proposed to perform object detection on board Gaia. It begins with a recommendation concerning SM read-out. Then the sample-level processing is introduced which aims at forming independent object entities through segmentation. Finally, these are submitted to a cascade of object-level tasks for the purpose of characterisation. As is justified above, the separation between hardware and software follows this sample / object structure so the design of the former focuses on achieving fixed complexity while the latter is intended to optimise it on average (Fig. 3.4).

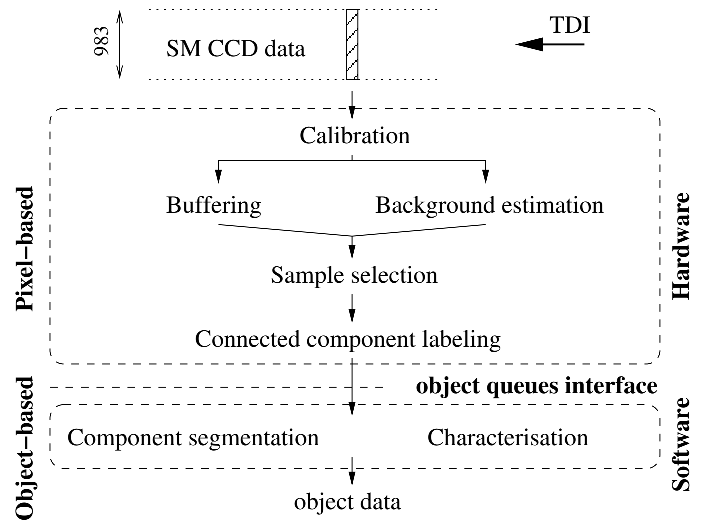


Figure 3.4: Overall organisation of tasks and target resources. "Pre-calibration" is treated in chapter 4, "background estimation" in chapter 5, "sample selection" and "connected component labelling" in chapter 6, the object-queue interface in section 6.6 and the "component segmentation" together with "characterisation" in chapter 7.

[17]This platform is equipped with three commercial PPC750FX parts running in parallel and submitted to TMR. When one of the processors is found to disagree with the others, typically after a SEUs, the resynchronisation mechanism implanted in the board is activated and suspends all processes for an entire millisecond.

[18]In the sense of the underlying number of instructions.

SM sampling

Since data from the SMs are not scientifically relevant due to the high noise related to the full read-out, operation of the CCDs can be fully optimised towards detection (Req. 4). As the primary goal consists in reliably reporting the presence of objects, and only secondarily in measuring them, it is relevant to consider the possibility of binning SM pixels together. While this naturally reduces the angular resolution, thereby impacting on the precision of the location estimate, it has two favourable effects on the data: the data flow may be reduced in terms of number of samples read at each TDI (AC binning) or in terms of how frequently samples are read (AL binning) and the read-out noise contribution diminishes because fewer samples are read. Both are highly desirable for detection as the first impacts on the amount of processing while the second makes it easier to fulfil the requirements at the faint end since the SNR is improved (Spec. 3).

The loss of precision on the SM location is to be compared to other error contributions for commanding read-out windows centred on the objects (SCI-230). Since propagation errors[19] are found to become rapidly dominant in the course of the stars' movement across the focal plane, precision of the order of half a pixel is found to be sufficient for this purpose. Another use of the SM location estimates is made by the AOCS control loop, but as this one relies on an average estimate, the degradation of the resolution can be efficiently compensated by increasing the number of objects over which it is evaluated to maintain the final precision.

Taking all aspects into consideration, an early recommendation [SM24] was made to read the SM CCDs in 2×2 samples instead of pixel per pixel. The benefit in terms of performance is illustrated Fig. 3.5 but equally important is the one affecting the data rate. The binning implies that only a total of 983 samples are read and need to be analysed every two TDIs for one SM CCD. This allows for reducing the frequency at which the hardware is run (section 9.2.3), but also, more importantly, that the SMs may be read-out alternatively. This in turn yields a significant simplification in the architecture of the VPU since it permits using the very same hardware for the two SMs (the system devised in part III switches transparently from one context to the other at every TDI).

This mode implies that only samples are visible from the standpoint of detection. Hence samples represent the only the elementary picture elements in SM in what follows and the mention of pixels should only be understood as referring to the original pixels in the frame of performance assessments where they define reference physical or angular scales.

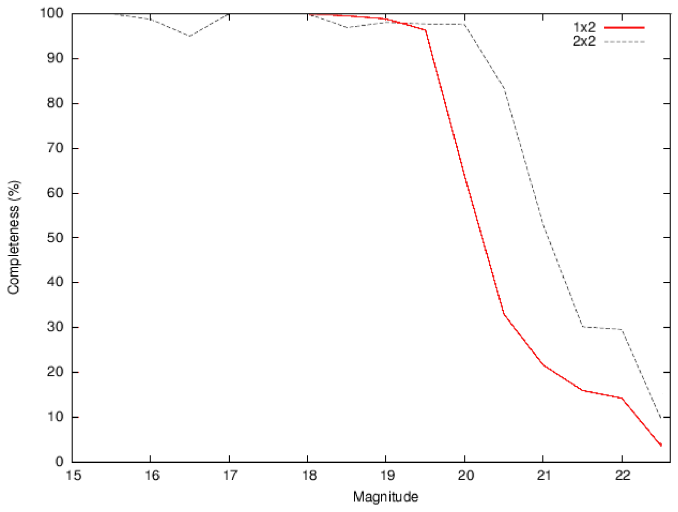


Figure 3.5: Comparison of detection performances with 1×2 and 2×2 binning in SM (from [SM24]) based on the instrument characteristics and the existing prototype software back in 2002. The 2×2 binning leads to an increased detection rate at the faint end. Conversely, the missing detections at magnitudes 16.5 and 18.5 are related to the increased crowding which leads to more visual DMSs. These are detected as single objects at this stage (ie. with the SOM (chapter 6)) but the individual components will be recovered with the finer object model introduced afterwards (ie. with the FOM (chapter 7)).

[19]Two main sources of error need to be considered as regards propagation: the difference between the real attitude and the one measured for the satellite by the AOCS and the approximate nature of the propagation algorithm as compared to the real complex movement of the sources on the focal plane.

Sample-based tasks

As mentioned above, targeting a hardware implementation for these tasks, the focus is placed on performing the same operations for all samples since fixed complexity is the best guarantee for fixed scheduling which is itself desirable for maximum performance. Besides, to minimise the amount of logical resources required to support operation, emphasis is placed on data management instead of arithmetics. "Natural" formulations for the algorithms are given in part II before they are refined in part III.

1. **Pre-calibration**
   Req. 5 & 6 call for the introduction of a pre-calibration module prior to any interpretation of the data to ensure the same assumptions can be made on all samples within a CCD but also more generally at the scale of the FPA. For practical reasons of functional specification, this module is separate from detection in the software implementation (Pyxis) but has been collocated for the hardware one for the benefit of greater integration. The corresponding rationale is the object of chapter 4 while the hardware design is described in section 12.3.2.

2. **Background estimation**
   If the instrument's noise distributions are expected to be calibrated on ground and uploaded to the satellite through parameter updates, the source-specific noise, which includes the background contribution, must necessarily be determined on board. After a brief overview of the physical origins of this background, a mechanism for estimating it at a regional scale for the sake of robustness to noise and pollution by objects (but yielding estimates at every sample location) is presented in chapter 5 and its hardware counterpart in sections 12.3.4, 12.3.5 and 12.3.6.

3. **Buffering**
   The scale at which the background is estimated introduces a delay before any decision can be made regarding the relevance of any sample. This delay corresponds to the need of waiting until the background estimate at the corresponding location becomes available. As the need for buffering is only a detail algorithmically, no further mention of it is made in part II. Conversely, it has far reaching consequences in hardware as concerns the architecture (chapters 9 and 10) and operating the underlying resources (section 12.3.3).

4. **Sample selection**

   The sample selection stage is the one at which drastic data flow reduction occurs by discarding samples associated to the background. Taking Req. 2 fully into account, this aim is somewhat extended to also suppressing those marking the presence of sources fainter than the limiting magnitude. As explained in chapter 6, the sample value, the background estimate and the noise properties calibrated on ground are brought together to apply a SNR threshold. In hardware, the level of reliability at which this test must be performed is the requirement from which the entire conversion of sample-based operations from floating to fixed point arithmetics derives (chapter 11). It is also central in terms of scheduling to ensure the various types of data can be collected and the test carried out (section 12.3.6).

5. **Connected-component labelling**
   The binary mask resulting from sample selection is then used to agglomerate samples into CCs. This first simple object model marks the transitions from the sample domain to the object one where it may be refined thanks to the availability of all the information pertaining to the object in the CC. The intended hardware implementation has necessitated algorithmic developments for building these CCs with fixed complexity (for constant time) and elementary data management. They are the major part of chapter 6. Due to development constraints discussed in section 9.2.1 it has not been possible to fully implement this tasks in hardware for the version reported in this text.

Object queues

The construction of object data packets also corresponds to the migration from the hardware to the software part of the mixed platform proposed. As the hardware engine is capable of handling arbitrarily complex configurations as a result of the fixed complexity underlying the operations in the sample domain, Req. 11 & 12 can be satisfied by the introduction of an intelligent interface (section 6.6). Priority-ordered object queues allowing the software to deal with objects in an order coherent with the scientific interest for each are defined for this purpose (Req. 11). If ever the capabilities the soft-real time engine were to be exceeded, the extraction of objects remaining in the queues after their maximum allowed latency has elapsed would allow for signalling their existence to ground based on the preliminary characterisation already available (Req. 12).

Object-based tasks

The object-based processing is composed of three essential parts organised to optimise the average computing time devoted to objects depending on their type, in accordance with the software platform's capabilities for flexible operation. It is the object of chapter 7.

1. Object filtering
   For the purpose of suppressing unwanted objects from the flow as early as possible, a cascade of simple descriptors are evaluated in a first place. These are relevant to identifying false detections related to noise and objects beyond the limiting magnitude (section 7.2.1) and PPEs (section 7.2.2).

2. DMS
   To ensure they will be correctly imaged in the windowed CCDs, the candidate DMSs then undergo component segmentation. Beside physical DMSs, optical ones due to the Poisson spatial distribution of objects or crowding in the high density regions, are submitted to this treatment. Taking this step for this last class of objects is part of an effort to ensure that every data packet sent to ground is exploitable notwithstanding the stellar density (Req. 8, extension of Req. 3 and Spec. 9 & 10). As indicated in section 3.4.3, a marker-based watershed algorithm is used for partitioning the CC into as many domains as there are components (section 7.3) which may then be considered as independent objects and re-inserted in the characterisation flow depending on their individual priorities.

3. Measurements
   Finally, the objects of confirmed interest and nature are measured in the sense of Spec. 8 to produce the outputs of detection for use both on board and on the ground (section 7.4).

3.6 Conclusion

The requirements presented in this chapter result from analysing the specifications for Gaia and understanding the intended use for the data. Additionally, considerations related to efficient implementation on the targeted technologies have led us to proposing a processing flow which, although based on APM and Sextractor, has been adapted to Gaia's needs. Each step has been thoroughly revisited to yield new algorithmic formulations at every level as will become apparent in the subsequent chapters of this part and even more in the next.

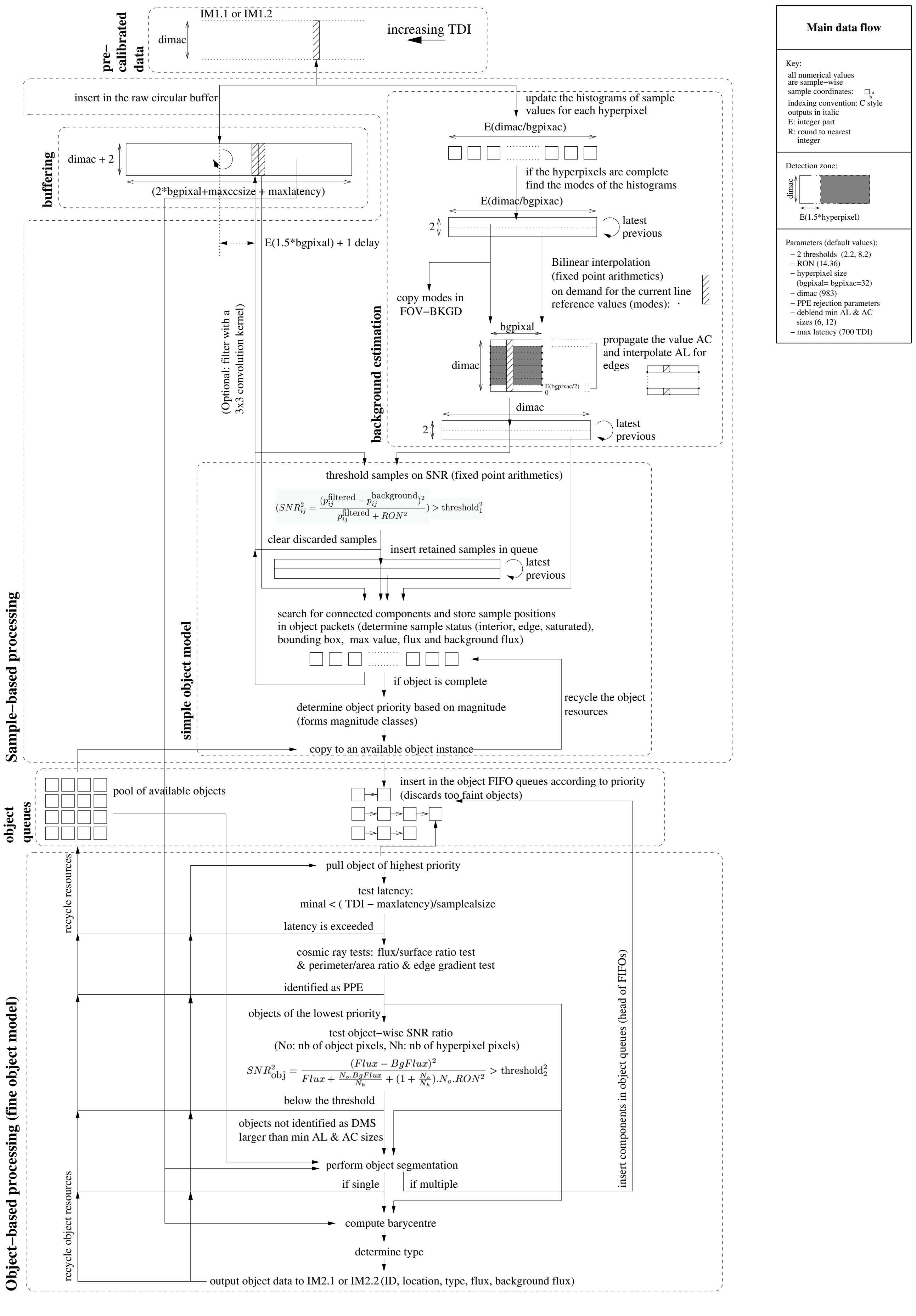

pre–calibrated data
IM1.1 or IM1.2
dimac
increasing TDI
insert in the raw circular buffer
update the histograms of sample values for each hyperpixel
E(dimac/bgpixac)
buffering
dimac + 2
(2*bgpixal+maxccsize + maxlatency)
if the hyperpixels are complete find the modes of the histograms
E(dimac/bgpixac)
2
latest
previous
E(1.5*bgpixal) + 1 delay
Bilinear interpolation (fixed point arithmetics)
on demand for the current line reference values (modes):
copy modes in FOV−BKGD
(Optional: filter with a 3x3 convolution kernel)
background estimation
bgpixal
dimac
propagate the value AC and interpolate AL for edges
E(bgpixac/2)
0
dimac
2
latest
previous
threshold samples on SNR (fixed point arithmetics)
$(SNR^2_{ij} = \frac{(p^{\text{filtered}}_{ij} - p^{\text{background}}_{ij})^2}{p^{\text{filtered}}_{ij} + RON^2}) > \text{threshold}^2_1$
clear discarded samples
insert retained samples in queue
latest
previous
search for connected components and store sample positions in object packets (determine sample status (interior, edge, saturated), bounding box, max value, flux and background flux)
if object is complete
recycle the object resources
determine object priority based on magnitude (forms magnitude classes)
copy to an available object instance
simple object model
Sample−based processing
object queues
pool of available objects
insert in the object FIFO queues according to priority (discards too faint objects)
pull object of highest priority
test latency: minal < (TDI − maxlatency)/samplealsize
latency is exceeded
cosmic ray tests: flux/surface ratio test & perimeter/area ratio & edge gradient test
identified as PPE
objects of the lowest priority
test object−wise SNR ratio (No: nb of object pixels, Nh: nb of hyperpixel pixels)
$SNR^2_{\text{obj}} = \frac{(Flux - BgFlux)^2}{Flux + \frac{N_o.BgFlux}{N_h} + (1 + \frac{N_o}{N_h}).N_o.RON^2} > \text{threshold}^2_2$
below the threshold
objects not identified as DMS larger than min AL & AC sizes
perform object segmentation
if single
if multiple
compute barycentre
determine type
output object data to IM2.1 or IM2.2 (ID, location, type, flux, background flux)
recycle resources
recycle object resources
insert components in object queues (head of FIFOs)
Object−based processing (fine object model)
Main data flow
Key:
all numerical values are sample−wise
sample coordinates: a b
indexing convention: C style
outputs in italic
E: integer part
R: round to nearest integer
Detection zone:
dimac
E(1.5*hyperpixel)
Parameters (default values):
− 2 thresholds (2.2, 8.2)
− RON (14.36)
− hyperpixel size (bgpixal= bgpixac=32)
− dimac (983)
− PPE rejection parameters
− deblend min AL & AC sizes (6, 12)
− max latency (700 TDI)

Figure 3.6: Overview of the algorithmic data flow.

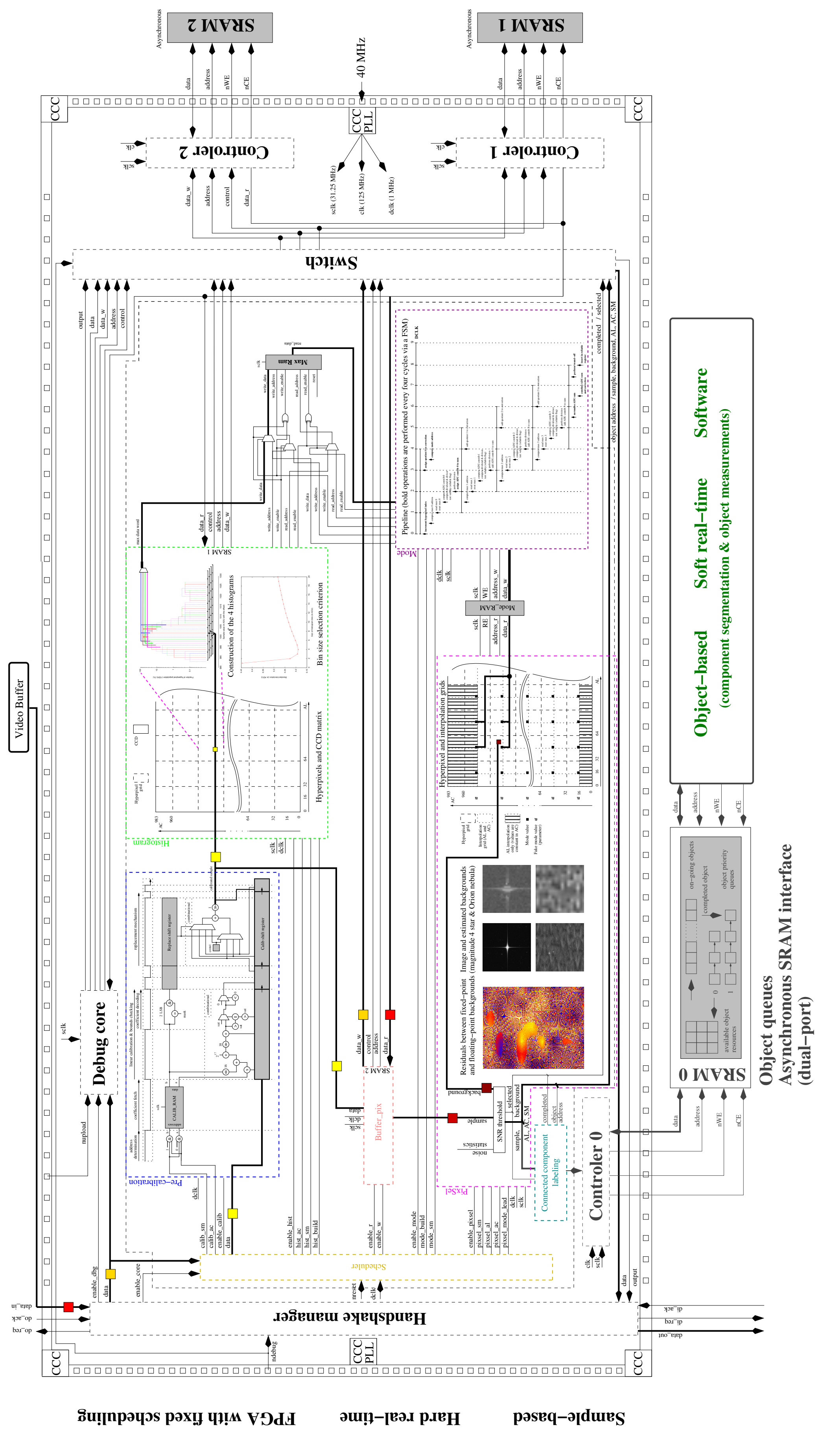

SRAM 2
SRAM 1
Asynchronous
Controler 2
Controler 1
40 MHz
CCC PLL
sclk (31.25 MHz)
clk (125 MHz)
dclk (1 MHz)
Switch
Video Buffer
Debug core
Pre-calibration
Histogram
Hyperpixels and CCD matrix
Construction of the 4 histograms
Bin size selection criterion
Max Ram
Pipeline (bold operations are performed every four cycles via a FSM)
Mode
Mode_RAM
Hyperpixel and interpolation grids
Residuals between fixed-point and floating-point backgrounds
Image and estimated backgrounds (magnitude 4 star & Orion nebula)
PixSel
Buffer_pix
Connected component labeling
Controler 0
Scheduler
Handshake manager
CCC
SRAM 0
Object queues
Asynchronous SRAM interface (dual-port)
Object-based
Soft real-time
Software
(component segmentation & object measurements)
Sample-based
Hard real-time
FPGA with fixed scheduling

Figure 3.7: Overview of the internal structure of the FPGA.

Chapter 4

Pre-calibration

Contents

4.1 Introduction

Based on the TDI operation of the CCD detectors (section 1.3.3), this chapter expands on the needs expressed in section 3.3 to define the nature and organisation of the corrections to be performed on the raw data. Precise descriptions of the numerical performance and of the implementation are then given respectively in sections 11.3.1 and 12.3.2.

Calibrating the response of the detectors is one of the key issues faced by Gaia. As may be seen by comparing the target micro-arcsecond accuracy in the final catalogue (Spec. 4) with the pixels' AL angular resolution ("only" of the order of 60 milli-arcsecond in Tab. 1.3), data analysis should work at the level of the thousandth of pixels when not limited by the SNR. The problem of evaluating the biases introduced by the TDI operation, the read-out or the defects resulting from ageing and radiation has triggered interest in the detailed characterisation of the CCDs and, because of E2V's low production yield calling for early fabrication of the devices, parts comparable to the FM are already available and test campaigns are being carried out as of this writing. Very high precision is mostly a concern related to the AF CCDs, but those used in SM employ the same technology. The requirements for on-board operation are significantly less stringent than for the astrometric reduction on ground but are nevertheless central to an homogeneous and unbiased survey.

A pre-calibration module was included early in Pyxis, much before the need were quantitatively assessed in the project, first essentially as a dummy module to stress the need for maintaining a stable image model and to estimate the associated processing load as part of the PDHE TDA activity. As expressed by the requirements of section 3.3, the corrections to be applied to the raw CCD data have two main objectives: permit the normal operation of the on-board software and ensure the scientific validity of its products by enforcing the underlying assumptions at all times. As such, it applies exclusively to the data used by the on-board processing and none of the pre-calibrated data is ever downloaded to ground for scientific purposes – raw data is always preferred.

Under the assumption that the state of CCDs degrades but slowly, to keep the conditions of operation of the on-board software traceable and transfer the processing for determining the corrections to be applied to ground, a quasi-stationary approximation is made. It consists in downloading raw data[1] to determine the parameters of the

[1]Nominal science data, special science data (not binned) and possibly additional calibration data.

corrections to be applied on ground, via some procedure which is *not* discussed here[2], then uploading them for the corrections to take place on board. The downside is that transient states exist between the moment when the defects first appear and their being taken into account and compensated. Our assumption is that these have limited impact on the on-board operation and the quality of the data generally speaking, provided new "white pixels" do not induce to catastrophic perturbations (section 4.3.2).

4.2 Motivation

4.2.1 Effects

The fundamentals concerning the CCDs have been presented in section 1.3.3 and need only be complemented here by a brief description of the effects modulating the response.

1. **PRNU.** The variable photo-sensitive properties of the samples over the area of the CCD implies that the conversion rate from photons to electrons is not uniform. Section 3.2.1 has introduced $\{e_i\}$ coefficients to account for this at the phase level. The TDI smoothes these fluctuations along the column but instead of the ideal 17976.λ count, an expected $(\sum_{i=0}^{i=17975} e_i).\lambda$ results which varies from column to column.

2. **DSNU.** Quantum thermal charge generation is susceptible of varying with position due to temperature gradients on the FPA ($\{d_i\}$ in section 3.2.1). Together with offset noise during ADC, they correspond to additive contributions to the count of relevance.

3. **Dead columns.** When PRNU or DSNU significantly affect the reliability of the values read, the column is declared "dead" and the information is considered beyond recovery.

4. **Non linearity.** The overall relation linking digital values output and photons should remain approximately linear to allow for interpreting each ADU in the dynamics as a constant amount of incident energy. Detectors are known to deviate from linearity at both ends of the dynamics with "S"-shaped curves corresponding to responses exceeding or short of the expected levels for low signals or near saturation respectively.

5. **CTI.** For detectors operated in TDI mode, the conservation of the electronic signal from phase to phase is of paramount importance. The tunnel effect causes particles to change phase in spite of the voltage barriers and, hence, some amount of diffusion of charges from one pixel to surrounding ones in a manner similar to the diffusion which occurs before the photo-electrons are collected in the pixels (between the photosensitive surface and the potential well). More significantly, traps exist in the CCD matrix which retain charges when they should be shifted to the next phase and release them randomly afterwards, although recent tests have shown that the total charge is not necessarily conserved. Various types exist which increase in numbers with ageing and radiation and differ in terms of the retention delays.

6. **Cross-talk** The proximity of electronic devices in the FPA introduces a risk of cross-talk between them. The resulting coupling classically results in spurious signal structured in patterns which repeat over time and are superimposed to the signal of interest.

7. **Vignetting and stray light.** The CCDs can only be expected to yield homogeneous outputs if their illumination is itself uniform. The distribution of photons on the FPA is the consequence of the transfer function of the optics and heterogeneity in this distribution impacts on the flux measurements and translates in apparent magnitude differences.

4.2.2 Policy

Fulfilling the main scientific objectives of on-board detection calls for a coarse correction of major anomalies to avoid detection failures and produce exploitable measurements. The first refers to the completeness aim and the second to selection biases.

Dead columns (item 3) are the most important perturbation to be mitigated although it is not part of classical calibration. The reason for this is that such columns introduce two types of artifacts which lead to an artificial increase in the number of detected objects. Dead columns with high signal would trigger the systematic detection of objects at these locations. Although these might well be filtered by detection itself on the grounds that their linear

[2]This should be part of the ground processing and is, hence, outside the scope of this thesis although the model built in section 4.3.1 might prove useful for it.

shapes resemble PPEs or would anyway not be observed because they would fail confirmation in AF1[3], they would nevertheless unnecessarily load the processing subsystem. In the frame of the selected approach, dead columns with low signal are more of a concern since they arbitrarily split CCs in two and lead to incoherent measurements most often resulting in discarding the objects during confirmation.

Calibration of the response in the traditional sense, that is by comparison to reference data, deals with the combined effects of points 1, 2 and 7 above. While the first two call for comparing samples only within a given CCD, the last calls for extending it across CCDs, notably between SM1 and SM2 to balance the importance of the two FOVs, but also between the seven rows for a fully coherent data management on board. The linear transform described in section 4.3.1 is designed with precisely these objectives in mind.

Although non linearity correction (item 4 above) could be incorporated to this calibration, it is set apart because the procurement specifications imposed on Gaia CCDs do not impose to carry it out. Indeed, although saturation can physically occur at the phase level, at the summing register level, at the read-out register level or as part of ADC, a conventional saturation threshold may be defined to limit the dynamics and avoid deviation from linearity. In this manner, the meaning associated to saturation becomes unambiguous and all values up to saturation remain trustworthy. Conversely, quality specifications at the faint end are extremely stringent because the RVS will image spectra at very low SNRs for which stacking will be necessary prior to interpretation. Additionally, with low fluxes, overestimation by a few percents will not fundamentally alter the classification of objects and the extra undesired objects can be filtered out as part of object characterisation.

As far as on-board detection is concerned, CTI effects impact essentially on the accuracy of the object measurements. However, the history-dependent nature of the process implies that the distribution of errors introduced is not stationary and is, hence, difficult to correct in a systematic way. Should the bias on the magnitude and location estimates become problematic, charge injection could be performed in SM also, as all CCDs provide this facility. The corresponding strategy is not yet fully defined. Not being able to predict when objects enter SM in any case, low-level continuous injection would seem the only reasonable option to generate an artificial sky background filling the traps constantly. In this case, systematic removal of the corresponding constant in the signal could be entrusted to pre-calibration as part of the linear transform.

The cross-talk perturbations (item 6) represent a problem similar to that of CTI in the sense that they are instationary. Although fairly well predictable because related to the clocking of the devices, the contributions vary not only from sample to sample but also from line to line and hence do not lend themselves to systematic line-wise corrections. As such they may be corrected on ground but not so easily on board due to the need to upload a large set of coefficients representing the patterns. To reduce cross-talk between the CCDs to a minimum, all CCDs on the FPA are operated in an homogeneous manner: share the same clocks and operate identically within each of the SM1, SM2, AF1 etc. columns. Should this prove insufficient, the pre-calibration approach described here could be extended to include support for compensating the cross-talk patterns (by using line-dependent $\{b_i\}$ coefficients for instance).

As a result of the analysis above, two different types of data manipulation are required from pre-calibration on a sample per sample basis but identically from line to line:

1. normalise the response to avoid the introduction of detection and measurement biases (Req. 6),

2. replace samples whose response are not representative of the sky or of too low reliability to avoid analysing erroneous content (Req. 5),

thereby keeping in mind biases and the high statistical quality aim for the final catalogue. With the quasi-stationary approximation, these corrections can be made perfectly systematic and time-independent between parameter updates and, as such, are good candidates for the intended hardware implementation.

4.3 Algorithms

4.3.1 Linear transform

Rationale

In ground-based astronomy, calibration of CCD data is usually carried out with “dark” and “flat-field” frames acquired before the observations. Several frames are usually averaged with a procedure robust to outliers to derive estimates of the response of each pixel yet filter out PPEs from the calibration map. The “dark” frames are intended to characterise the DSNU and are normal exposures but without illumination while “flat-field” ones target the PRNU and vignetting

[3]These two mechanisms are part of the reasons justifying the quasi-static approximation made above. During the transient state following the apparition of a new dead column, they would prevent it from disrupting normal operation and excessively polluting the telemetry stream.

and are based on a uniform illumination obtained either through a dedicated lamp or by observing the sky at the beginning and end of the night. The calibration equation then reads for each AC location j:

$$corrected_j = \frac{raw_j - dark_j}{flatfield_j} \times \overline{flatfield} \tag{4.1}$$

where $\overline{flatfield}$ is the mean response value over all valid pixels of the averaged "flat-field" frame.

In our case, beside the fact that normalisation is carried out across CCDs, the correction to be applied is computed separately for improved fixed-point performances (chapter 11) and formula 4.1 is rewritten as a general linear transform:

$$corrected_j = raw_j + (raw_j \times a_j + b_j) \tag{4.2}$$

Beside expression (4.2) which gives the rationale behind the normalisation calculation, it is important to acknowledge that two special cases result from the degeneracy introduced by saturation:

- saturated raw values should remain saturated,
- the corrected values should not exceed the saturation threshold.

As a consequence of these precautions, the meaning attached to saturation is preserved and summarised in a single recognisable value. Conversely, when calibration parameters are out-of-date with the behaviour of the CCD it is possible to face underflow[4]. As for saturation, particular attention is paid to this case of figure in the algorithm presented section 4.4.1.

Implementation

The general fixed-point formulation corresponding to 4.2 is derived in section 11.3.1 where the procurement specifications for the CCDs (Tab. 1.3) are analysed in an effort to determine ranges for the a_j and b_j coefficients and optimise the word length and precision. As a result variations of the PRNU and vignetting of ±6.25% and values of the DSNU up to 2^{12} ADU can be compensated to yield values correct at the ADU level (as would an ideal CCD) in 92% of cases under worst numerical conditions. Simultaneously, for maximum flexibility in the design of the architecture, the format has been devised to allow for storing calibration parameters on external 16-bit memory modules (section 12.3.2). Given that current simulations from Gibis do not feature any defects, the identical transform is however carried out and implemented with $a_j = 0$ and $b_j = 0$ for all j.

Sample distribution

From a statistical point of view, referring to the model built in section 3.2.1, the samples input to calibration correspond to $\tilde{S}_{j_{e^-}}$ represented by equations (3.6) for the electronic signal and $\tilde{S}_{j\,ADU}$ (3.7) for the ADC[5]. This quantity is modified as follows by the linear transform:

$$\begin{aligned}
\tilde{\tilde{S}}_{j,ADU} &= \tilde{S}_{j,ADU}.(1+a_j) + b_j \qquad (4.3)\\
&= \frac{\tilde{S}_{j,e^-}.(1+a_j) + offset.(1+a_j) + b_j.gain}{gain} \qquad (4.4)
\end{aligned}$$

Hence, with the notations of section 3.2.1:

$$\begin{aligned}
\tilde{\tilde{S}}_{j,ADU} \times gain \;=\; & \underbrace{(\sum_{i=0}^{i=17999} e_i^1.(1+a_j).(p_1^1+p_2^1)) + (\sum_{i=0}^{i=17999} e_i^2.(1+a_j).(p_1^2+p_2^2))}_{A} \\
& + \underbrace{D_1^1 + D_2^1 + D_1^1 + D_2^2 + N(0,\sigma_{RON}) + offset.(1+a_j) + b_j.gain}_{B} \qquad (4.5)
\end{aligned}$$

This last expression makes it clear that calibration cannot be ideal for each of the two columns simultaneously and that some criterion will need to be defined for choosing a_j coefficients that are optimal in some sense for the two considered jointly. Let us simply note that simply trying to equate the expectation of A to that of an ideal CCD at a given level of signal is bound to fail since the effective value read-out will vary depending on the ratio of contributions from the two columns. On the opposite, once a value has been chosen for a_j, b_j can be chosen to impose that the expectation of B be equal to $offset$.

[4]The ADU values, although unsigned, can represent negative electron counts (due to the RON) with the help of the offset. With offset valued 1000, the lower bound of encodable counts amounts to $-4070\ e^-$ which implies that values below this threshold will cause underflow.

[5](3.6) and (3.7) are written for all AC locations, hence the j index has been omitted.

4.3.2 Dead samples

As previously suggested, columns containing "white" and "dark" pixels can be handled in the frame of the linear normalisation above and still yield some information, although somewhat degraded statistically. This is the favourable case. In the unfavourable one, the required correction is either unpredictable or cannot be applied with sufficient precision and the column must be declared dead, thereby calling for the replacement of the value with a more reliable one. The CCD procurement specifications [19] indicate that columns should be considered defective as soon as they feature one hundred or more "dark" or "white" pixels, which sets a reliability limit for deciding when to apply the linear correction and when to replace the value.

Rationale

If replacement is to be performed, the next question is how.

1. The substituted value should be as similar to the missing one as possible. The latter is unknown but, assuming that PSFs are correctly sampled, correlation exists between adjacent samples providing grounds for estimating the missing one from its neighbours.

2. A second criteria relates to the segmentation of components (section 7.3) which relies on local maxima. Accordingly, the method should refrain from introducing spurious ones.

3. A third one concerns applicability depending on the number of dead samples and their distribution AC. Graceful degradation of performances should be ensured as more difficult configurations will be encountered as the CCDs age. Besides, some replacement should occur at all times to mitigate the risk of systematic false detections.

4. Finally, given the volume of data for which calibration is performed and to make a hardware implementation possible, the method should remain simple and systematic.

The most natural approach consists in using a linear estimator (ie. a weighed mean) based on either one[6], two or four neighbours. These three schemes correspond analytically to interpolation with a constant, linear or cubic profile. For evaluation purposes, one might consider a given image, determine optimal weights by ordinary least square estimation and compare the estimates obtained for every sample to the real sample value. Fig. 4.1 illustrates the quality of the replacement[7] in this academic case for a high density image to evaluate the method on objects rather than on the background where noise degrades the inter-sample correlation. The correlation coefficients have been computed as a quantitative measure to compare their respective effectiveness.

Applicability

As expected, Fig. 4.1 shows an increase in performance with the number of neighbours used. Simple replacement based on a single value seems insufficiently reliable in the general case (criterion 1) so this option may only be retained as a degraded mode. The four-point average yields the best results, but poses a difficulty concerning applicability (criterion 3). A simple calculation assuming uniformly distributed dead samples on the CCD and affecting N samples over the 983 from SM provides the probability that the method be inapplicable to at least one dead sample (Tab. 4.1). Quite naturally, this probability is twice as large for the four-point average as for the two-point equivalent for small numbers of dead samples.

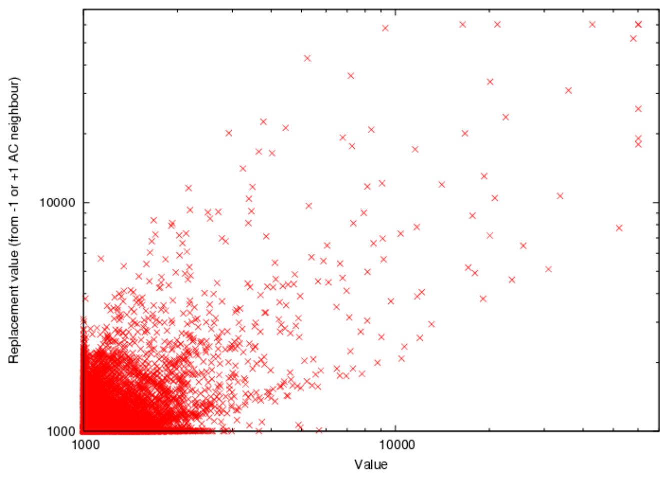
$$P_\lambda(N) = 1 - \frac{\Pi_{i=0}^{N-1}(983-\lambda.i)}{N!} / \begin{pmatrix} 983 \\ N \end{pmatrix}$$

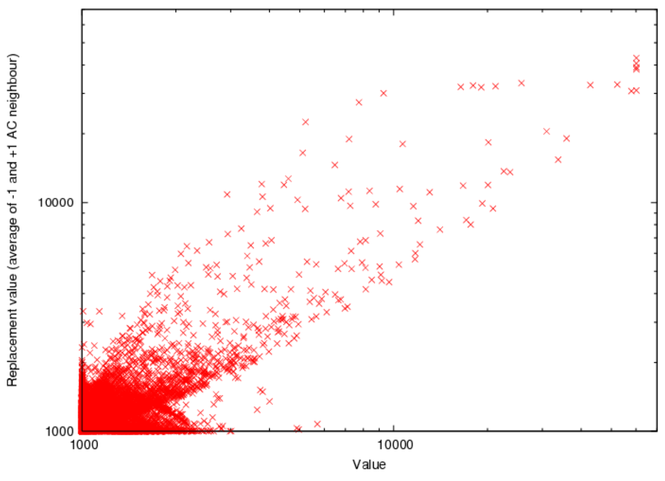
N	$P_3(N)$	$P_5(N)$
2	0.2	0.4
5	2.0	4.0
10	8.9	17.0
20	32.8	55.3
30	60.2	84.8
50	93.1	99.6

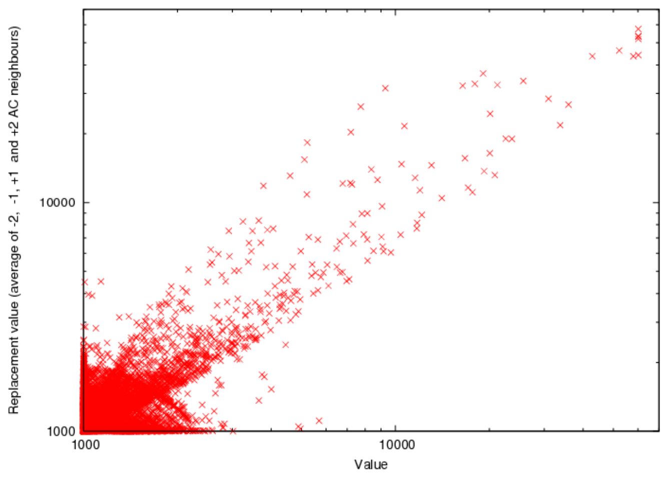
Table 4.1: Probability that the two and four point averages be inapplicable to at least one dead sample depending on the number of dead samples in SM (in % and with $\lambda = 3$ or 5 for the two-point and four-point averages respectively). CCD procurement specifications require $N \leq 7$ in beginning of life conditions.

[6]Then simply the replacement with one of the neighbours' value.

[7]The exact values for the four-point average determined on the l54b0 image are $[-0.24469, 0.74416, 0, 0.74419, -0.24475]$ resulting in a correlation coefficient valued 0.91317. Fixed-point arithmetics use $[-0.25, 0.75, 0, 0.75, -0.25]$ instead.







Number of neighbours	Correlation coefficient
1	0.68528
2	0.86316
4	0.91345

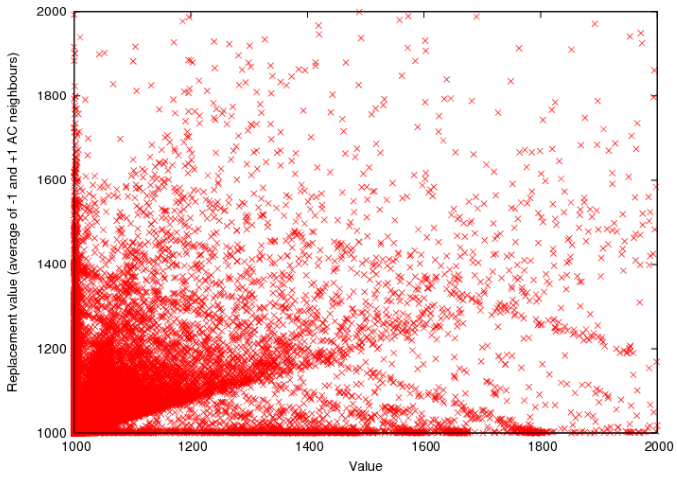
Figure 4.1: Correlations between values estimated from the AC neighbours and real sample values for a simulation at the design density in SM1. For left to right and top to bottom for 1, 2 and 4 point averages (average are computed with weights determined to maximise correlation then approximated to allow fixed-point arithmetics). Values are in ADU so 1000 corresponds to a zero electron count.

The assumption according to which a sample value can be predicted from its neighbours' (criterion 1), depends on the spatial frequencies associated to the imaging of the different kinds of objects. Although valid for optical images thanks to the properties of the PSF, it applies differently to PPEs which are imaged differently. In particular, the sharp edges introduce artifacts. Two cases of figure must be considered depending on whether the PPE fell on the dead sample or not. In the first case, because the neighbours are not necessarily affected by the PPE, it may well not be visible after the replacement. Conversely, if one of the neighbours is affected by a PPE, it ends up being extended with a smoothed edge likely to render its subsequent identification more difficult. The two cases lead to points outside of the correlation cone and close to the axes on Fig. 4.2.

Fig. 4.2 illustrates the negative performances of the two-point average in the presence of PPEs. This effect is even increased with the four-point average since the kernel extends over more samples with the additional property that isolated bright samples can be replicated.

Conclusion

Comparing applicability conditions to the increase in representativity shows that the four-point and higher-order averages are not desirable compared to more local estimates. The two-point average, although more often reliable, does not guarantee that all dead samples can always undergo replacement, in violation of item 3 above. Failing cases are of two types: first and foremost come those related to multiple consecutive dead samples, then those which do not feature valid neighbours because of border effects. Accordingly, one or more degraded modes must be defined, which implies both determining when to switch to this or these modes as well as defining which kind of replacement should



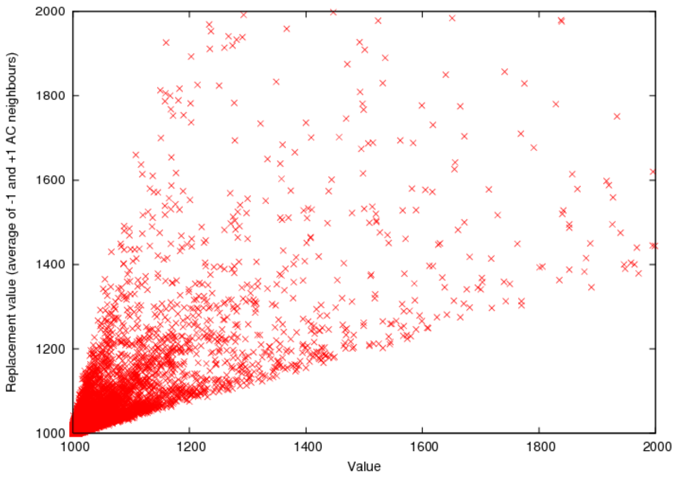


Figure 4.2: Correlations between true and replacement values (two-point average) in the presence (left) and absence (right) of PPEs (in the absence of objects).

then occur.

Since all replacement approaches relying on valid neighbours are doomed to fail at some level of degradation of the CCD or versus some specific unfavourable configurations, an ultimate backup solution is required to guarantee applicability. As it cannot rely on any knowledge of the image data to be restored, it must consist in using a pre-defined constant used to clear the dead sample's content. The objectives of avoiding spurious detections and minimising the impact on background estimation then call for using a value close to the most frequent signal, that is, the most likely background level up-to-date with the CCD's sensitivity. As such, it should be possible to update the value of this parameter (`params.DEFAULT_BKGD`).

Based on only two-point and constant replacement, pre-calibration faces a problem with consecutive dead samples. To avoid having to automatically clear these away, a convenient intermediate solution consists in also allowing the one-sample average. As the samples are processed in sequence, it offers two possibilities for correcting pairs of consecutive dead samples: the preceding valid sample may be attributed to both or to the first only, with the second then replaced by a two-point average. More generally, this mechanism makes it possible to handle entire intervals of dead samples by propagating a valid value across them.

4.4 Architecture

4.4.1 Processing flow

For each newly acquired line of data (983 unsigned 16-bit samples), the samples may be processed serially in AC-order as soon as they can be fetched from the Video Buffer. For the purpose of only relying on calibrated values, however, it is desirable to delay the replacement mechanism by exactly one sample. In software, this translates in introducing the *dead* array to optimise addressing (this array is the same as the one containing calibrated values (*calib*), only with addresses offset by one position to ensure that all values looked-at are the result of the linear transform).

Based on these data structures, the processing then runs as:

1. Get the next sample (index j and value).
2. Perform the linear normalisation: $calib_j = raw_j + (raw_j \times a_j + b_j)$.
3. Test for saturation: $raw_j \geq$ `params.NB_PIX_FWC`.
   3.1. TRUE: replace by the pre-defined saturation value $calib_j =$ `params.NB_PIX_FWC`.
4. Test for saturation: $calib_j >$ `params.NB_PIX_FWC`.
   4.1. TRUE: replace by the pre-defined saturation value $calib_j =$ `params.NB_PIX_FWC`.
5. Test versus minimum value: $calib_j <$ `params.NB_MIN_ADU`.

   5.1. TRUE: replace by the pre-defined minimum value $calib_j$ = `params.NB_MIN_ADU`.

6. Perform dead sample value replacement on the previous sample. Test the applicability conditions:
   6.1. b_{j-1} = REPLACE_AVERAGE: replace by $dead_j = E(\frac{dead_{j-1}+dead_{j+1}+1}{2})$.
   6.2. b_{j-1} = REPLACE_PREV_AC: replace by $dead_j = dead_{j-1}$.
   6.3. b_{j-1} = REPLACE_NEXT_AC: replace by $dead_j = dead_{j+1}$.
   6.4. b_{j-1} = REPLACE_CONSTANT: replace by $dead_j$ = `params.DEFAULT_BKGD`;
7. Loop to 1. or exit when all the line has been processed.

4.4.2 Sample replacement

In the frame of the targeted hardware implementation, the two point average calls for an annoying management of border samples for which only one neighbour is available. The one-point average is handy to avoid the introduction of an otherwise useless special case, without imposing to cancel the value as replacement by the constant would.

As for determining the coefficients of the linear transforms and identifying dead samples, the choice of the replacement strategy is performed on ground according to scientific performance criteria and to the map of accessible neighbours. Noting that the linear calibration stage is useless when replacement is programmed to occur, this can be carried out without the need for a dedicated LUT and multiple special values for b_i are simply used for selecting which method to apply to the jth sample. They are chosen at the extremities of the admissible range at the expense of a negligible reduction of the dynamics available for b (see Tab. 4.2).

b_j	Method	Formula for $corrected_j$
16380	constant	`params.DEFAULT_BKGD`
16381	previous neighbour	$corrected_{j-1}$
16382	next neighbour	$corrected_{j+1}$
16383	two-point average	$\frac{corrected_{j-1}+corrected_{j+1}+1}{2}$

Table 4.2: Summary of replacement strategies: indices are not necessarily AC positions but represent the order of the pre-calibration processing and the formula for averaging corresponds to also performing round-off to the nearest ADU as per (11.6).

4.5 Conclusion

The calibration procedure described in this chapter enforces a stable image model by correcting the biases introduced by the imperfections of the CCDs, thereby allowing detection to proceed on firm grounds. The correction, as it is, corresponds to shifting and scaling the CCD response distribution value-wise (as a result of the linear transform) and suppressing abnormal points (as a result of the replacement strategy). Although improving on raw values, the calibrated ones cannot be assumed to restore a behaviour comparable to an ideal CCD. The limitation comes from the insufficient amount of information available as regards the origin of the signal within a sample. A given value can result from a variety of contributions from the two columns summed in hardware. As these cannot be calibrated individually, pre-calibration must necessarily proceed based on criteria defined on average.

Chapter 5

Background estimation

Contents

5.1 Introduction

This chapter presents the background contribution present in the SM images and derives an estimation method compatible with the intended implementation. Emphasis is placed on accuracy through robustness to the pollution induced by the presence of objects. With efficient calculation in mind, a single pass approach[1] has been retained based on a statistic designed to yield a representative approximation in all cases. This results in a systematic procedure which is in turn favourable to hardware (sections 11.3.2 and 11.3.3 for the fixed-point arithmetics, 12.3.4 and 12.3.5 for the implementation).

5.1.1 Need

The data acquisition described in section 3.2.1, as it neglects the self-correlation potentially induced by cross-talk and models noise realisations as independent from sample to sample, makes noise an intrinsically local characteristic. The dead sample defects are corrected as a result of pre-calibration (chapter 4). The electronic noise is essentially a concern as regards false detections at the detection limit but may be efficiently mitigated by SNR-thresholding. The Poisson noise impacts on the precision of measurements but using an object's entire set of member-samples provides an opportunity for counterbalancing this – although only to an extent depending on the number of such samples. These successive precautions provide means to control the precision and bias in output data in a manner coherent with the need for homogeneity related to their subsequent use on board.

[1]As opposed to relying on successively trimmed statistics for instance.

Yet, this reasoning only holds if any constant contribution to the signal has been previously suppressed as it would lead to systematically preserving samples which are not otherwise of interest or would introduce a bias in measurements. Whereas a strictly constant term could be easily dispensed with as part of pre-calibration, low frequency ones call for a dedicated handling which is object of this chapter. In possible situations with textured background featuring high frequencies the scheme developed below will naturally prove ineffective but, as more objects than are really present in the scene will be expected, the astrophysical sources and the background artifacts will stand a chance of being discriminated as part of the characterisation stage (chapter 7).

Both background and noise estimates (noise will be discussed in section 6.3) are, hence, central to segmenting the CCD images in regions and the corresponding opportunity to discard the majority of data as irrelevant object-wise. The associated performance objectives relate to controlling the rate of false detections (Req. 2), forming object entities allowing to derive accurate measurements over the entire magnitude range (Req. 7) and reducing the data flow (Req. 11).

5.1.2 Contributors

With the spurious CCD signal already accounted for as part of pre-calibration, additive contributions to the signal of interest result mostly from real sky complexities and the behaviour of Gaia's optical system. We provide below a brief overview of the different sources involved, which beside making the precise nature of the perturbing signal explicit, also shows how problematic configurations as regards background are nevertheless of high scientific interest.

- **Gas & dust.**
  The Galaxy, like the solar system, is not only composed of stars but also of matter absorbing light and diffusing it. In spite of Gaia's very high angular resolution, the volumes from which each sample receives light are of considerable cross-sections due to the distances within the Galaxy. Accordingly, large gas and dust structures are the main contributors in this respect. Objects behind such structures are difficult to detect because of the absorption and the reddening which shifts the wavelengths outside of the CCDs' sensitivity range. Conversely, those in front appear on varying backgrounds at different intensity levels depending on the regions. Examples of these are the galactic centre where the dust masks a majority of objects (except for instance in Baade's windows), star-forming regions where stars are born due, precisely, to high concentrations of gas and dust, and planetary nebulae.

  Closer to Gaia, in the solar system, a similar effect is due to the zodiacal light produced by reflection on dust present around the ecliptic plane[2].

- **Faint sources.**
  If the sky is not infinitely bright as stated by Olbers's paradox[3] long exposures reveal that very faint stars can be found in all pointing directions. Stars beyond Gaia's sensitivity hence combine to produce a diffuse background.

  In addition to real faint stars, ghost images of brighter ones are formed by multiple reflexions within the optical system. Although not visible on the exploded view of the payload in Fig. 1.7, baffles are placed around the FPA to intercept these unwanted rays and limits the impact of ghost images to only high order ones. The attenuation is then very significant – as must be since stars of interest have fluxes differing by ratios up to 400 000 – so that they can be assumed to merge with the faint star background.

- **Stray light.**
  A final category contributes to the observed background. It corresponds to the ambient light within the thermal tent generated through light scattering by the particles in free-fall within it or corresponding to light entering through the openings and not focused on the FPA. Beside this light, the payload is immersed in radiation and secondary particles, generated by interaction of primary ones with the satellite's structures. Both of these, because of their low energy, interact more easily with the detectors, thereby depositing small amounts of energy which can total in a noticeable radiation background.

[2]The plane orthogonal to the solar system's angular momentum vector which passes through its barycentre. This plane may be regarded as the weighted average of all planetary orbital planes.

[3]The paradox stated by Olbers consists in the fact that the sky is dark at night instead of bright. This simple fact came in contradiction to the main cosmological hypothesis made at his time which considered the Universe infinite and homogeneous. Although light is attenuated with the square of the distance, the volume and hence the number of sources increases according to its cube so that the total amount of light received from infinity in any direction should increase to infinity. This paradox is partly solved by the fact that the observable Universe, that is the volume from which photons have had time to reach the Earth, is finite since the structures emitting light appeared in a finite past and the speed of light is also finite. Another aspect of the explanation stems from the fact that with an expanding Universe, the wavelengths are shifted towards the red end as time goes by. The cosmological background, which is a remnant of the early brightness of the Universe, now corresponds to the light which would be emitted by matte at 3K only.

5.2 Background statistic

Following the sequence outlined in section 3.5, the image is partitioned into a number of regions (hereafter hyperpixels) and the background is estimated over each independently and as a whole. The choice of a scale for these is the result of a compromise between estimating the background close enough to each object and with reasonable latency on one hand, and gathering a sufficient amount of data to limit the pollution by noise and the objects themselves on the other hand. Piece-wise bilinear interpolation is then performed between hyperpixels as an approximation to low frequency variations to yield estimates at each sample location individually.

5.2.1 Regional background

Rationale

The decision to follow [1] and [3] in collecting estimates over predetermined regions is closely connected to the desire to minimise the number of assumptions concerning the images. Local approaches, like SWA, are forced to rely on models of the intensity distribution around the centroids of objects which are fine so long as density is low and objects are isolated but face difficulties increasing with crowding since nearby objects then pollute the measurements. One of the criteria for setting the dimension of the hyperpixels consists then, naturally, in providing a sufficiently wide vision of the objects' surroundings.

One might then think that a simple and efficient way of doing this would be to treat the image as whole and apply a low-pass filter for estimating the background at any given location. While this is, indeed, the behaviour sought, the linear approach is impracticable in our case for three main reasons relative to computation, addressing and latency. All three result from the size of the corresponding kernel. Determining the background for a given object imposes to have recourse to samples for which the ratio of contributions from the object and from the background is favourable. With the extremely wide dynamics to be covered and the well-sampled PSFs, such samples are located at distances which depend on the object's magnitude, but which are generally large – even more so if robustness to companion objects should be achieved. Large kernels have a number of implications, first on the amount of computation required to yield the estimate, then on applicability because of border effects. If the TDI acquisition produces unbounded images in the AL direction, selecting it as the kernel's main axis poses addressing and latency problems because of the raster order of data incoming from the CCDs. As as consequence, a second criterion for defining our hyperpixels is then naturally to induce limited latency and border effects.

A final consideration applies to balancing sensitivity to the various spatial frequencies existing in the physical background with the resolution accessible after data acquisition. If a square region in the image features PSFs with equal extension in AL and AC, it represents a rectangular area on the sky because of the rectangular pixel geometry (section 3.2). The cut-off frequencies in AL and AC cannot then be equal simultaneously for the PSF and the underlying physical structures. With interest primarily in separating the objects' images from the background and not in observing it, the image domain may be favoured which calls for relying on square hyperpixels (composed of $n \times n$ samples).

Optimal size

The presence of objects in one of our regions introduces two types of effects which lead to over-estimating the background (Fig. 5.1). Faint stars and PPEs fitting into the hyperpixels tend to pollute the distribution of sample values with moderate values and can be modelled as an exponential contribution [38]. Hyperpixels whose size is large compared to these objects will allow for still deriving a reliable estimate in spite of these faint but numerous objects. Conversely, bright objects, as they extend over very many samples, will completely override the background contribution whatever the dimension of the hyperpixel in the practicable range imposed by latency. This calls for introducing a dedicated mechanism for the handling of these infrequent but problematic cases or the bright objects might paradoxically remain undetected (section 5.2.3).

From a physical point of view, it is advisable to ensure that the content of each regions is, at least, approximately homogeneous or else the overall value produced risks being meaningless compared to the background seen by each sample. Irwin typically recommends regions 10-30 arcsec in dimension [1].

Finally, from a purely technical point of view, to optimise the precision and the resources associated to the bilinear interpolation performed between hyperpixels (section 11.3.3), hyperpixels whose size is a power-of-two present significant advantages.

Fig. 5.2 illustrates the typical distribution of object sizes[4] as an input for selecting the optimal size. With all the previous criteria in mind, hyperpixels composed of 32 samples in AL and AC have been retained. This defines a characteristic scale which is in line with the range recommended by Irwin considering the unprecedented angular resolution of Gaia (3.772×11.315 arcsec^2).

[4]More precisely, the AL and AC dimensions of the bounding boxes for CCs formed as part of the SOM in chapter 6.

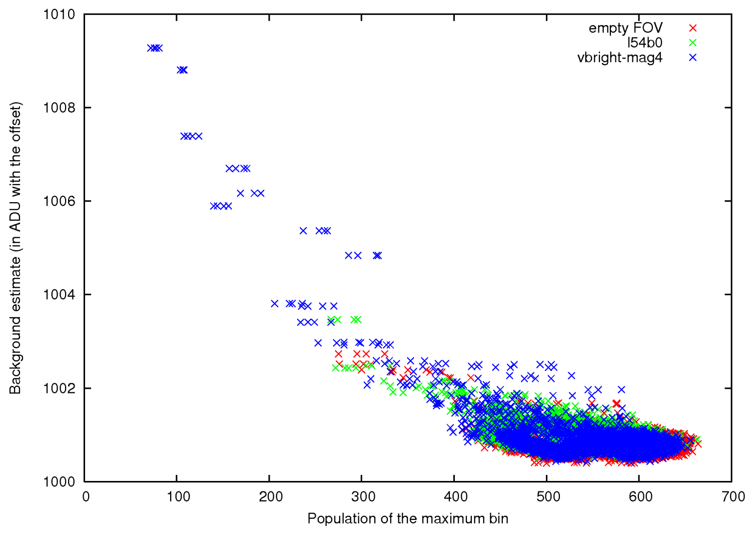


Figure 5.1: Relationship between background estimates and number of samples in the bin of maximum count (section 5.2.2) in three configurations: an almost empty FOV, pollution related to a high stellar density (600 000 stars/deg^2), pollution related the presence of a magnitude $G = 4$ star.

5.2.2 Estimator

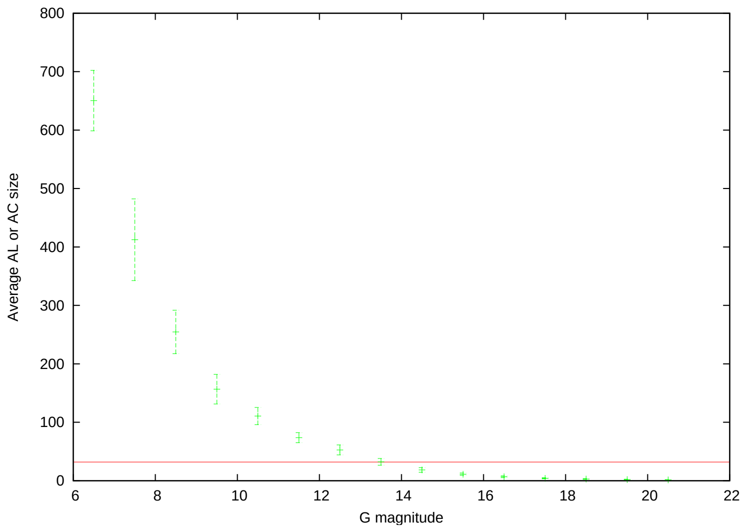
To reduce the impact of moderately bright objects and at the same time improve the efficiency of the computation, a sample value from one of the hyperpixels will only be taken into account if it is in the range corresponding to expected backgrounds. Those encountered by Gaia, although significant as compared to magnitude 20 stars to be observed, are confined to the faint end partly because of the short exposure time and partly because gas and dust re-emit light principally in the infrared domain. Accordingly, only samples in the $[936; 1959]$ ADU ($[-300.8, 4507, 3]$ e^-) range are considered, allowing for backgrounds fainter than G = 17.5, to be compared to the uniform background at G = 22.35 mag/deg^{2}assumed in Gibis by default (ie. 87 times lower).

The presence of objects alters the distribution of sample values by reinforcing the probabilities on the bright side of the interval monitored. With this type of perturbations, a mean estimate is clearly inappropriate and relying on the median is a classical solution. Two main drawbacks, however, make this alternative undesirable: the complexity of the calculation which involves partially sorting the 1024 (32×32 samples) values and the low precision resulting from the quantisation of sample values. Instead, following a maximum likelihood philosophy and noting that the sky remains mostly empty at Gaia's angular resolution and sensitivity, one might judge that the most probable source of signal for a sample is the background. Hence, the effective image background may be approximated by the mode of the distribution, itself estimated by determining the most frequent value observed. With this choice, the perturbations discussed above have no measurable effect until the total number of samples affected by the presence of objects become the majority. Robustness is, accordingly, superior to both the mean and the median by a long shot.

Although the precision of the mode, as of the median, is naturally restricted to the histogram's bin size, a profile may be locally adjusted to the observed distribution for refining it to the sub-ADU level. To this end, following [4], the bin of maximum count together with the neighbouring two are used to determine the osculating parabola before its maximum is reported as a lower-variance estimate of the mode.

With 1024 values coming from each hyperpixel, the number of samples should be large enough to populate the histogram and construct reliable statistics. Precision, however, depends on the combined SNR ratio for each sample and the sampling of the distribution achieved as a result of the bin size selected for building the histogram. The trade-off is between larger bins yielding lower resolution but more reliable counts and smaller ones leading to improved localisation of the background value but at the expense of increased dispersion. With quantised values coming from the CCDs, it is tempting to rely on the natural bins and construct histograms with ADU resolution but, as Fig. 5.3 shows, the resulting estimates are poorly distributed due to statistical noise in the histograms.

Improving the dispersion of the estimator can be achieved by decreasing the resolution of the histogram. This corresponds to re-binning the histogram and can be done in several ways. There are, indeed, N different ways to combine consecutive 1-ADU bins into N-ADU ones, each leading to a background estimate. Taking advantage of this, a final estimate can be derived by averaging these in a manner which minimises the sensitivity to noise and compensates the loss of resolution in a manner akin to super-resolution techniques. For selecting an optimal N, quality may be quantitatively expressed as the variance of estimation. Fig. 5.4 illustrates how the standard deviation varies with the



Magnitude	Star count/deg^2
6-7	1.4
7-8	2.9
8-9	5.9
9-10	12.2
10-11	25.0
11-12	36.0
12-13	98.7
13-14	235.5
14-15	523.6
15-16	1114.3
16-17	2238.7
17-18	4285.5
18-19	7888.6
19-20	14321.9
20-21	25468.3

Figure 5.2: Left: average sizes of objects as a function of magnitude in units of SM samples (the line corresponds to the hyperpixel's selected AL and AC dimension: 32 samples). Right: object count previsions averaged over the entire sky per square degree according to [39] (extrapolated to low magnitudes).

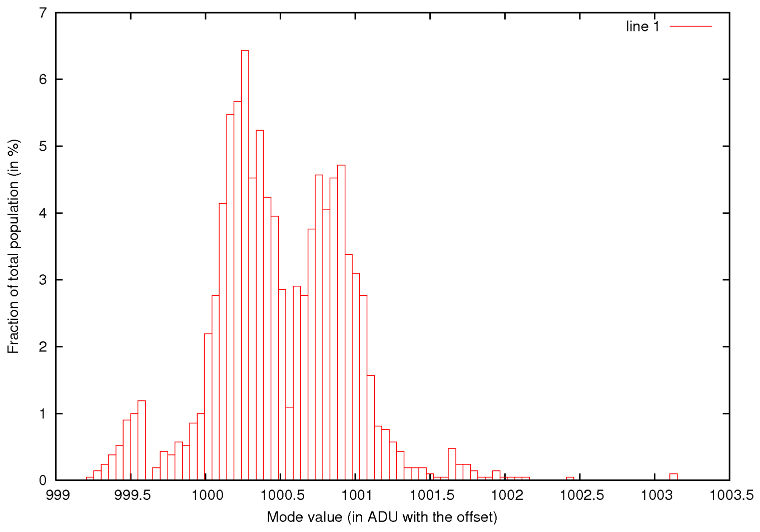


Figure 5.3: Distribution of mode values determined from histograms built with 1 ADU resolution (for the l54b0 test case).

bin size N and fully justifies forming 4-ADU bins. The distributions of mode values for 2, 3 and 4 ADU bins are shown on Fig. 5.5 and represent a significant improvement as compared to Fig. 5.3.

Let then each of the four histograms be labelled with i and $mode^i$ denote the ADU value attached to the bin of maximum count and $index^i$ its position in the ith histogram. Putting everything together, the final estimate is produced via:

$$mode = \frac{1}{4} \sum_{i \in \{0 \ldots 3\}} \left(mode^i + 2 \frac{F^i_{index^i-1} - F^i_{index^i+1}}{F^i_{index^i-1} - 2F^i_{index^i} + F^i_{index^i+1}} \right) + \texttt{params.MODE\_BIAS} \tag{5.1}$$

In addition to estimating the modes with the parabolic fit and averaging them, a term for the calibration of the method has been introduced in formula (5.1) above. As the convention for bin indices follows the addressing in C, $mode^i$ does not directly represent the most probable ADU value to be associated to the bin. Although it might be possible to transform $mode^i$ in this sense, the exact law might prove complex as it does not only depend on the way the histograms are built but also on the biases introduced throughout the acquisition chain, notably in the process of

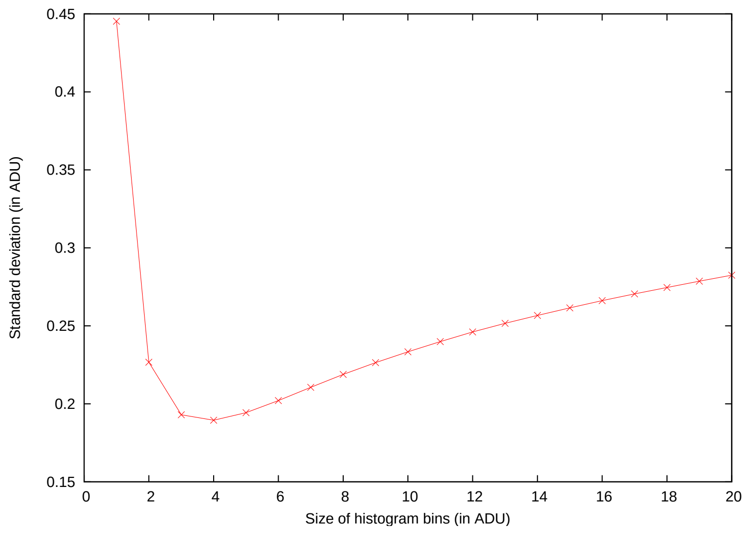


Figure 5.4: Standard deviation of the background estimator as a function of bin size (at the design density).

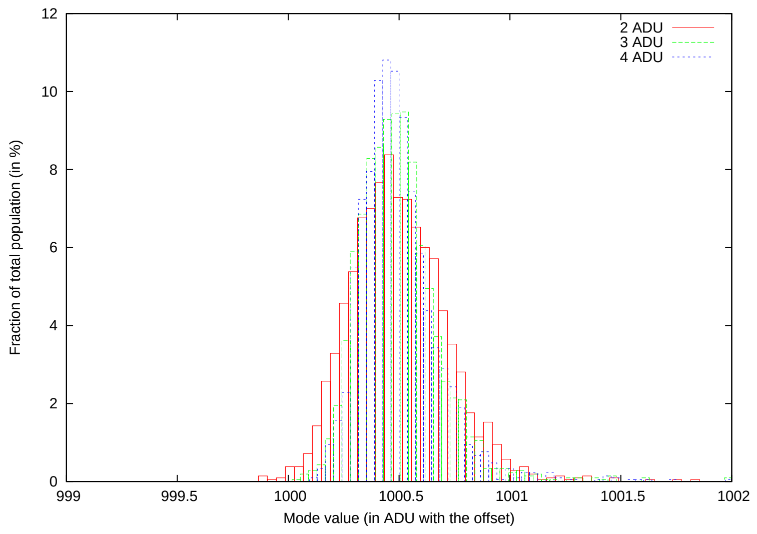


Figure 5.5: Distribution of mode values from histograms built with 2, 3 and 4 ADU resolutions (at the design density).

ADC. It is therefore preferable both as regards computation and as regards accuracy to rely on a single calibration step for all these effects.

To this end, background values may be experimentally generated for various observing conditions with the formula above but without the `MODE_BIAS` term. Fig. 5.6 compares the estimated values to the real background in a 10000 TDI simulation of an empty field with a uniform $G = 22.352$ mag/deg^2 background. This is considered as the most frequent configuration on the sky and is, accordingly, selected to calibrate our procedure, that is, to determine the bias correction term[5]: $1000.3318 - 1000.7722 = -0.4404$ ADU (encoded as `params.MODE_BIAS` $= -902$ in 0.21.11 fixed point format).

5.2.3 Replacement strategy

Bright objects with their large number of member-samples may utterly swamp the background content of hyperpixels. In these cases, there is little other possibility than to simply disregard the measured value. The regional structure imposed on the image is particularly convenient for this purpose. Based on the connectivity of hyperpixels and assuming some correlation between them as a result of the low frequencies dominant in the background signal of

[5]The 1000.3318 value is also adopted as default background value (params.DEFAULT_BKGD_VALUE) and rounded-off to the nearest ADU to yield the default replacement value for pre-calibration (params.DEFAULT_BKGD).

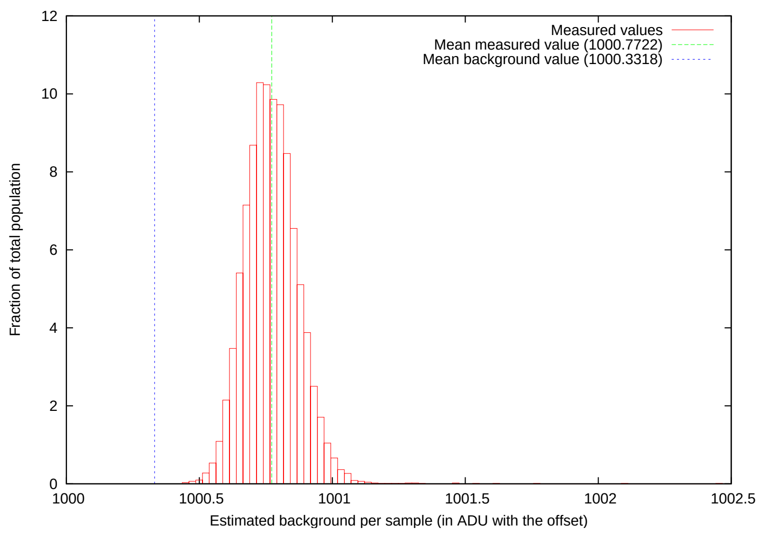


Figure 5.6: Distribution of estimated values versus mean background in a 10000-TDI empty field with a $G = 22.352$ mag/deg^2 background.

interest, a mechanism similar to that of pre-calibration for dead samples may be introduced. In an attempt to avoid catastrophic situations, it substitutes a reasonable value to the measured one provided it can be identified as unreliable on board.

Reliability test

This section derives a formal statistical criterion for deciding whether a given 4-ADU histogram can be trusted to measure the background or not. In a manner coherent with the objective defined above to identify low frequency contributions, the hypothesis $\mathcal{H}$ tested here is whether or not the bin of maximum population found corresponds to a distribution of values in the hyperpixel resulting from a uniform signal – as opposed to all circumstances which can make the estimate unreliable. Additionally, an ideal CCD is assumed, both for simplification purposes and because with existing simulations this is the only configuration which can be tested[6] (section 3.2.2). The threshold may then be somewhat relaxed, based on a representative database of experiments, to adjust it to the frequency range of interest and accommodate the necessarily imperfect pre-calibration performed in the previous chapter.

In the frame of $\mathcal{H}$ all phases share the same parameter λ and with ideal CCDs[7] the expression for the distribution of the signal (3.6) is then simply:

$$\tilde{S}_{e^-} = p_{17975\times2\times2.\lambda} + D_{2\times2} + N(0,\sigma_{RON}) \tag{5.2}$$

converted to ADUs according to expression (3.7) to yield $\tilde{S}_{ADU}$.

The population of each of the histograms' bins follows a binomial distribution because a given sample value is either in the interval covered by the bin or not and because, under $\mathcal{H}$, all samples derive from the same distribution. With the 4-ADU bins, this probability q is the sum of the four associated to each of the ADU values in the interval:

$$q = \mathcal{P}(\tilde{S}_{ADU} = k) + \mathcal{P}(\tilde{S}_{ADU} = k+1) + \mathcal{P}(\tilde{S}_{ADU} = k+2) + \mathcal{P}(\tilde{S}_{ADU} = k+3) \tag{5.3}$$

and the resulting expectation and variance are $1024q$ and $1024q.(1-q)$ respectively.

A basic test procedure is then the following:

1. Search for the bin of maximum count in the histogram.
2. Determine the electron count associated to the bin based on its index.
3. Using equations (5.2) and (5.3) determine the associated probability q under $\mathcal{H}$.
4. Compare the measured count versus probable ones at the required confidence level for a binomial distribution with parameters q and 1024.

[6]As explained in chapter 4, as a results of the correction applied at the pre-calibration level, the deviations from this ideal case will remain bounded so it may be considered an acceptable approximation.

[7]As regards PRNU and DSNU but the dark signal and read-out noise are still taken into account.

Exploiting the large number of samples inside the hyperpixels, the distribution could be approximated to perform tests on the goodness of fit to the theoretical distribution obtained under $\mathcal{H}$ (Pearson's chi-square test or a G-test). These, although more frequently used and with better properties, require to compute a quantity over all bins and represent a significantly higher computational complexity as opposed to the one proposed, which only requires to tabulate the bounds of predicted counts for each background level accessible to the method.

Fig. 5.7, which illustrates the intervals obtained by simulation, shows that it is not possible to reject unreliable hyperpixels at all levels with a single threshold due to the varying dispersion of the histograms when λ increases. However, a value corresponding to `params.HISTOGRAM_TH` = 350 counts was found, by simulation, to be appropriate for a large range of low backgrounds while preserving enough flexibility for mapping textured ones (Fig. 5.11).

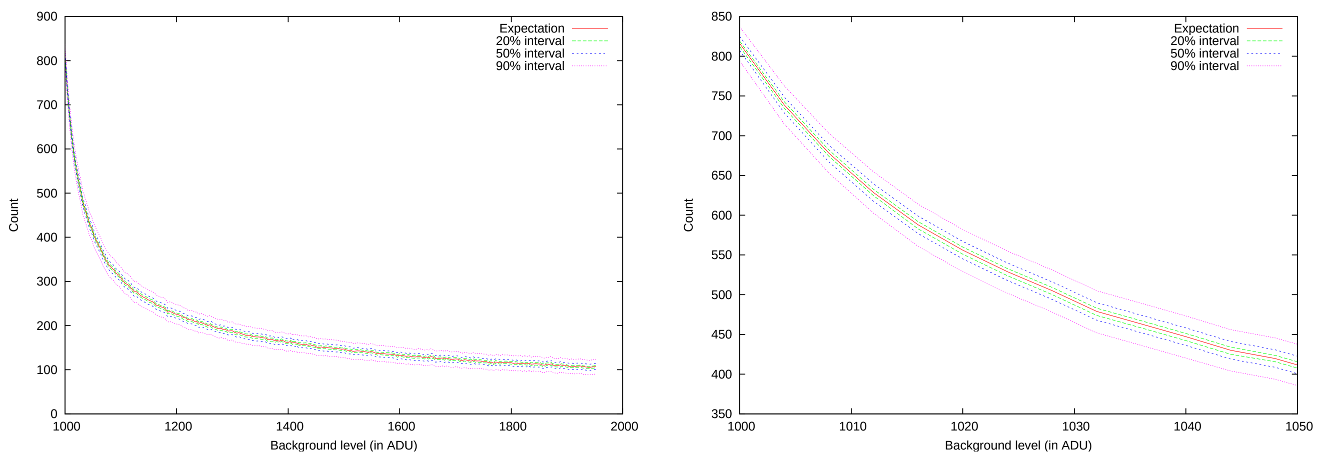


Figure 5.7: Thresholds for the reliability test at all levels accessible to the method (left) and close-up on the low backgrounds (right) for all possible 4-ADU bins. The expectation curve corresponds to: $1024.(\mathcal{P}(\tilde{S}_{ADU} = \frac{\lambda+4070}{4.07} - 2) + \mathcal{P}(\tilde{S}_{ADU} = \frac{\lambda+4070}{4.07} - 1) + \mathcal{P}(\tilde{S}_{ADU} = \frac{\lambda+4070}{4.07}) + \mathcal{P}(\tilde{S}_{ADU} = \frac{\lambda+4070}{4.07} + 1))$ while the envelope curves limit the domains covering respectively 20, 50 and 90 % of realisations under the hypothesis $\mathcal{H}$.

Procedure

Assuming the unreliable estimates can be detected by the procedure described above, the inter-hyperpixel replacement procedure itself is identical to that employed for dead sample replacement by pre-calibration (sections 4.3.2 and 4.4.2). As a consequence five possible cases of figure are possible depending on the geometry[8]. With the three reliable flags for the previous, current and next hyperpixels in AC arranged as $(previous, current, next)$, the corresponding truth table reads:

1. (-, 1, -): use measured value.
2. (1, 0, 0): propagate the preceding hyperpixel's value.
3. (0, 0, 1): propagate the next hyperpixel's value.
4. (1, 0, 1): use the average of the preceding and next values.
5. (0, 0, 0): replace by a constant (`params.DEFAULT_BKGD_VALUE`).

5.2.4 Background map

The mode estimates constructed in the previous section represent only 30 values for an area amounting to 31456 samples. Each one is by construction attached to a hyperpixel and represents its background content in the sense developed above. As it does not take into consideration the relative location of individual samples within the hyperpixel, each estimate should be considered global to its region.

A natural and simple way of translating the background map from the hyperpixels' scale to the samples' would then consist in simply attributing this value to all samples in the region. Since the SNR-threshold is applied to each

[8]As for pre-calibration, the possibility to propagate valid neighbouring values allows for handling the AC border effects without introducing any dedicated special case.

sample individually, the discontinuities of the such a piece-wise constant map do not pose any mathematical difficulty and the method might well be preferred for its simplicity. The discontinuities are, however, problematic as regards Req. 3: the decision to keep or discard a given sample might then depend on its position versus the edge of the hyperpixel. In spite of having built a map representing the low-frequency content of the image, the dependency on such thresholds would distort its interpretation and introduce undesirable high-frequency selection effects.

Instead, as a trade-off with complexity, a piece-wise continuous map is produced by two-dimensional interpolation between the hyperpixels and, since continuity is sufficient for our purpose, a bilinear scheme suffices. It is also coherent with the intent to suppress the low frequency component in the signal. In a certain sense, the map may be interpreted as a first order approximation of the slowly varying contributions allowing to suppress the linear trends they introduce in the data (section 5.3.3). The mode values are placed at the centres of the hyperpixels as the area most likely to be well represented by the estimate. Fig. 5.8 illustrates how the two grids, one for hyperpixels and one for interpolation, are related.

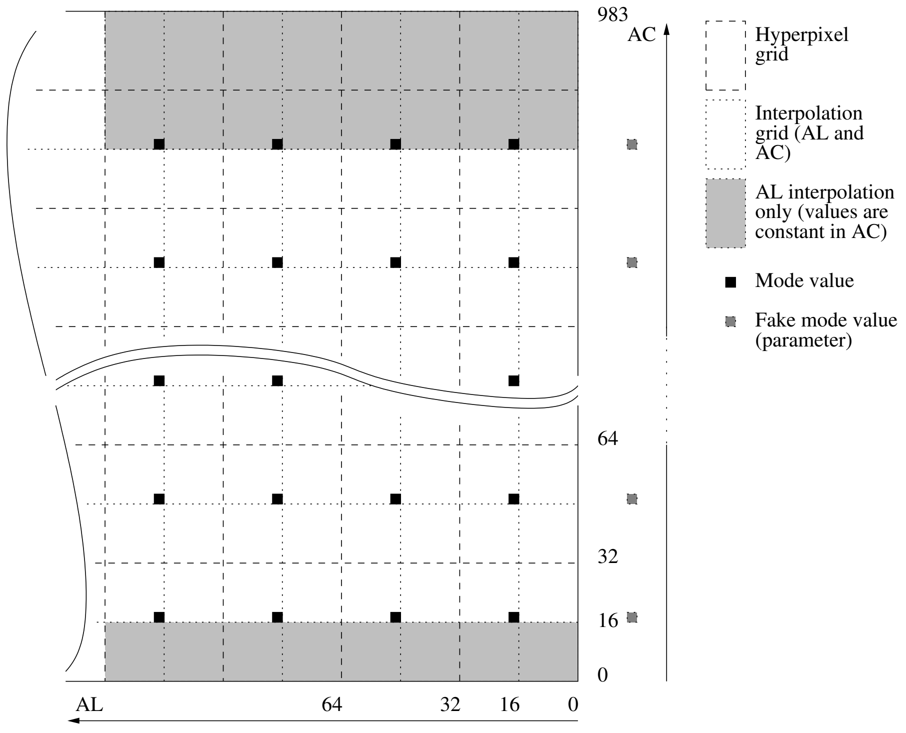

983
AC
64
32
16
0
AL
64
32
16
0
Hyperpixel grid
Interpolation grid (AL and AC)
AL interpolation only (values are constant in AC)
Mode value
Fake mode value (parameter)

Figure 5.8: Hyperpixel and interpolation grids.

As already mentioned in section 3.5.2, a delay between the latest line received from the FPA and the one to be thresholded results from the need to wait until the hyperpixels are complete for determining the background statistic. This delay is a function of the AL size of the hyperpixels and of the interpolation procedure envisaged. In our case, it amounts to $1.5 \times 32 = 48$ cycles (ie. 96 TDI periods with the 2×2 binning) and decomposes into: 32 cycles until the hyperpixels are completed and 16 cycles related to the placement of mode values inside the hyperpixels for interpolation.

Two interpolation regimes are used depending on whether the location considered is surrounded by four mode values or not. In the first case, bilinear interpolation is performed. In the second case, only AL interpolation is carried out and the resulting values are copied to all CCD columns to the border of the CCD. This latter case affects a total of 55 samples in the adopted design (greyed-out in Fig. 5.8) as a consequence of the hyperpixel AC dimension and the placement of modes for interpolation. Although background estimates are available at every location on the CCD, this special case necessarily translates in slightly different statistical properties. Although this comes in violation of Req. 3, it is only transit-specific for each of the objects potentially affected and, as experiments have shown, the additional error has limited impact on the detection rate.

5.3 Performances

Performances are regards background estimation are difficult to assess because of the diversity of observing conditions already mentioned in section 2.2.2. Whereas the specified need serves as a guideline for assessing the detection rate or the quality of measurements, the background underlies all these aspects and exists independently from the objects. Although, as is classical in estimation theory, the bias and variance of the estimator may be derived formally or measured in selected cases chosen to form a representative set, indicators allowing to predict the behaviour of the method in the general case cannot be produced. Two essential reasons concur to this: the non separability of the AL and AC directions in the adopted method and its non linearity. Accordingly, the results presented in this section can be little more than a catalogue of results, hopefully covering the main cases of interest but nevertheless fundamentally incomplete.

5.3.1 Precision

Although a mathematical model has been derived above for the purpose of testing the reliability of the estimates in each hyperpixel, it is simpler to estimate the precision of the method through simulation than to derive it formally due to the complexity of equation 5.1. As an example, Fig. 5.6 illustrates the performance in a low background case for which the read-out noise dominates. Transforming this field into a low density configuration by adding 200 objects ($\simeq 5600$ star/deg^2) and realistic PPEs, the standard deviation amounts to 0.098555 ADU. This favourable case typically corresponding to high galactic latitudes and defines a lower bound which may be adopted as an approximation of the estimator's intrinsic precision.

5.3.2 Robustness

It is of paramount importance that the method be robust to the presence of objects. Failure at this level would translate in overestimated background levels leading to fewer detections where there are more objects or where they are brighter. In the first case, as high density configurations meet with a more difficult determination of the astrometric parameters due to crowding, depleting the set of observations rapidly becomes critical for the scientific interpretation of the data. Conversely, in the second one, exclusion zones are formed around the bright objects forbidding characterisation of possible DMSs or investigation of light bending effects in the vicinity of planets for example. Beside the random effects, systematic ones are a particular concern since, by construction, they would extend over entire regions and, hence, distort the vision of the Galaxy.

Four key steps have been taken to reduce the coupling between the background and the stars:

1. the construction of trimmed histograms to avoid arbitrarily high background values,

2. the use of an estimator highly robust to outliers,

3. the introduction of a replacement mechanism to detect and suppress estimates manifestly polluted by objects,

4. the generation of the map through linear interpolation to increase the total number of samples on which each estimate is based.

Measures of the pollution by individual objects are plotted on Fig. 5.9. The correlation coefficients between the background estimates and the sample values are computed over the domains covered by the object-member samples themselves with the intent to identify the presence of trends related to the PSF in the background map. The correlation is satisfactorily low for stars between the saturation and the detection limit but increases at the low magnitudes, in spite of the replacement mechanism, because of propagating estimates slightly polluted by the spikes' extremities towards the photocentre. This is further exemplified in Fig. 5.10 which focuses on the rare observation of a magnitude $G = 4$ star and results from the trade-off on the reliability threshold (section 5.2.3): spotless backgrounds around bright stars could be achieved but at the cost of reduced representativity for textured backgrounds since both cause the histograms to be more spread-out.

With sensitivity targeted at low frequencies, the PSFs contribute to the background essentially through the extremities of the spikes where the gradient is reduced. Accordingly, sensitivity to density results not only from crowding, which tends towards regions of uniform illumination, but also from summing small contributions from a large number of objects. Tab. 5.1 collects measurements in this sense based on realistic object distributions both spatially and as regards magnitude generated with the Besançon Galaxy model [40]. The behaviour remains stable for all low density cases and begins to be affected by the stellar content only for the design density. Although the bias then increases measurably up to the worst nominal case according to Spec. 10 (Baade), it remains within reasonable bounds thereby impeding mostly the detection of faint objects.

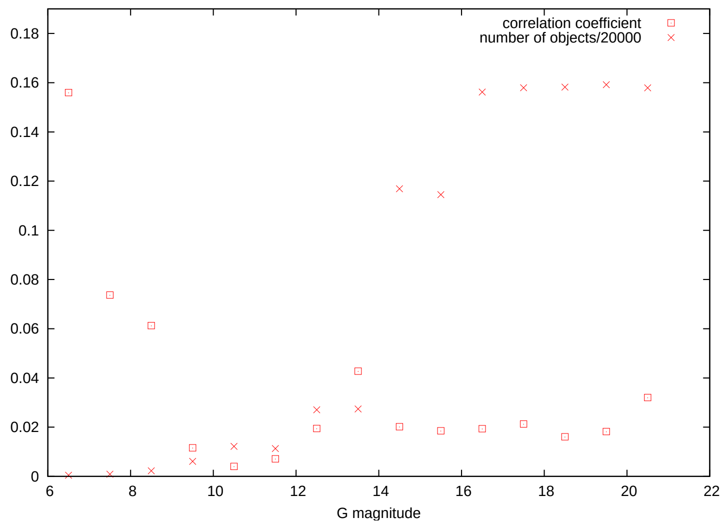


Figure 5.9: Pollution of the background map by stars (values of the correlation coefficients between the background and the sample values at the locations of object-member samples). Simulations which are all 20000 TDI long and simulated objects are placed on a grid whose cells are not commensurable with hyperpixels.

Density	Mean	Standard deviation	Description
3200	1000.31544	0.074591	lowest density close to the galactic pole (l=74, b=74) (Fig. C.1)
22250	1000.32703	0.079735	average sky density (l=74, b=15) (Fig. C.2)
195000	1000.36741	0.089175	representative density in the galaxy disk (l=74, b=0) (Fig. C.3)
609000	1000.48862	0.14340	design density case (l=54, b=0) (Fig. C.4)
$3.31.10^6$	1001.1575	0.20463	maximum density case in Baade's windows (Fig. C.5)

Table 5.1: Sensitivity to realistic object distributions at various levels of stellar density (densities in the table correspond to number stars per square degree with $G \leq 20$ but fainter objects and PPEs are present in the simulations). The mean value and standard deviation are computed based on all samples for simulations covering 5000 TDIs with uniform backgrounds at $G = 22.352$ mag/deg^2 leading to an expected 1000.3318 ADUs background as in Fig. 5.6.

5.3.3 Representativity

Beside measuring the background with sufficient precision and accurately, the representation of its variations is also of interest due to the local nature of the SNR-threshold applied for sample selection. Characterisation based on a database of relevant cases is one possibility for this. Fig. 5.11 provides one synthetic example which illustrates how the selected hyperpixel size is appropriate for structures present in the Orion nebula.

From a more general point of view, "representativity" can be analysed in terms of frequential response. Fig. 5.12 presents the frequency content of the background map obtained for backgrounds uniform in AC and monochromatic in AL (Fig. 5.13). Beside the expected low-pass behaviour corresponding to the hyperpixel resolution at which the background is estimated, three perturbing effects shape this response. Most notable are the cases where the spatial period is commensurable with the dimension of the hyperpixel: when smaller, a uniform measure results (eg. the 30-sample period case in Fig. 5.12) and when larger, energy is transferred to the harmonics. This translates the fact that, due to resonance, the frequential response is distorted – yet, the average background level remains properly evaluated. Second, as a consequence of the rigid hyperpixel grid used, the method is not translation invariant and causes aliasing (leftmost case in Fig. 5.13). Finally, due to discontinuities in the derivative, the piece-wise bilinear interpolation introduces oscillations which pollute the frequency response – although they do not significantly hinder use of the estimates for SNR-thresholding.

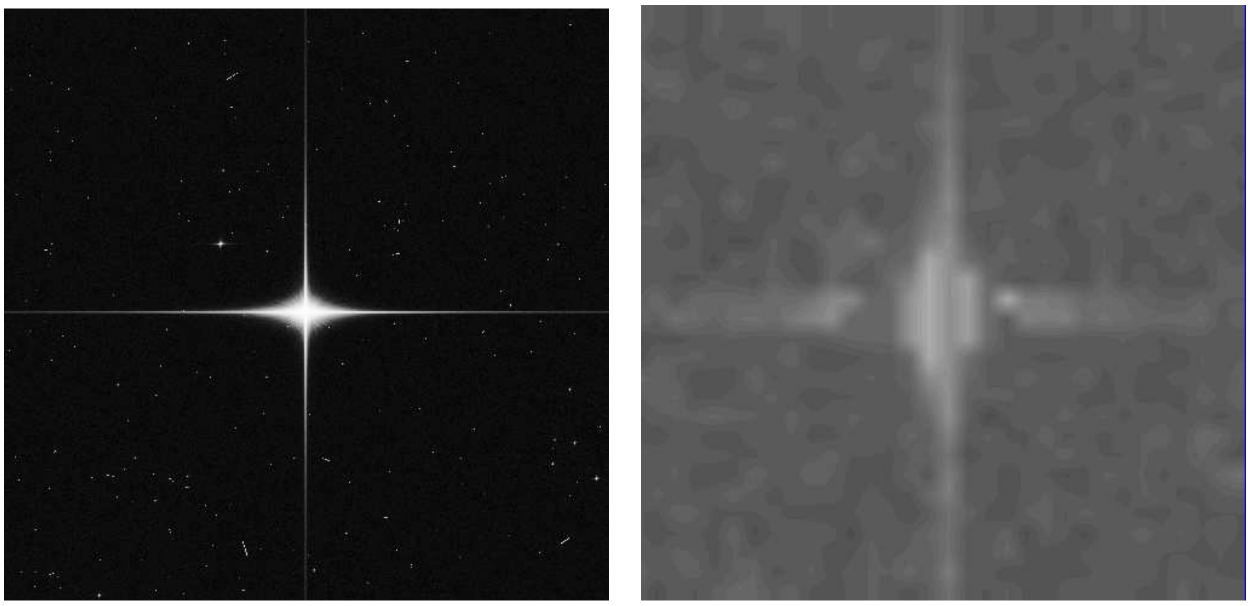


Figure 5.10: Background estimates in the vicinity of a magnitude $G = 4$ star. Values have been re-normalised in the snapshot on the right to increase the contrast but the background levels measured on the spikes are typically below 1001.5 ADUs and the peak corresponds to 1002.46 ADUs near the centroid compared to the expected 1000.3318 ADUs.

5.4 Algorithms

5.4.1 Histograms

The software-oriented description provided in this section corresponds to that of Pyxis and is rather straightforward. Its hardware counterpart is, however, significantly more intricate because of complex data management and the decision to track maxima as the histograms are built (section 12.3.4). The sequence presented below serves then not only for making the discussions of the previous sections more precise but also as a reference for disentangling the algorithmic needs from the technical constraints in chapter 12.

As the hyperpixel grid is fixed, the correspondence between the samples' positions and the hyperpixel they belong to is straightforward and allows for updating the corresponding histograms by relying on the data's serial ordering. Only samples with AC coordinates in the $[0; 959]$ interval belong to hyperpixels fully read-out and are processed below and in software[9].

The procedure, as it relies on four 4-ADU histograms offers two natural implementations:

- one working in parallel on the four histograms to produce four mode estimates,
- one building a 1-ADU histogram then re-binning it only in the neighbourhood of its global maximum but in all possible manners, to yield the four mode estimates.

The second option was preferred for the software version in Pyxis because of the possibility to optimised the search for the global maximum but is less hardware-friendly than the first, which is accordingly the one selected in section 12.3.4 and the only one made explicit below.

```
1. For each line of samples.
  1.1. For each sample position (from 0 to 959).
  1.1.1. Identify the corresponding hyperpixel (from 0 to 29).
  1.1.2. For each 4-ADU histogram (from 0 to 3).
    1.1.2.1. Test the value versus the histogram's range ([936, 1959[ ADUs with the offset).
      1.1.2.1.1. True: increment the corresponding 4-ADU histogram bin.
```

[9]For regularity's sake all samples are however processed in hardware (section 12.3.4).

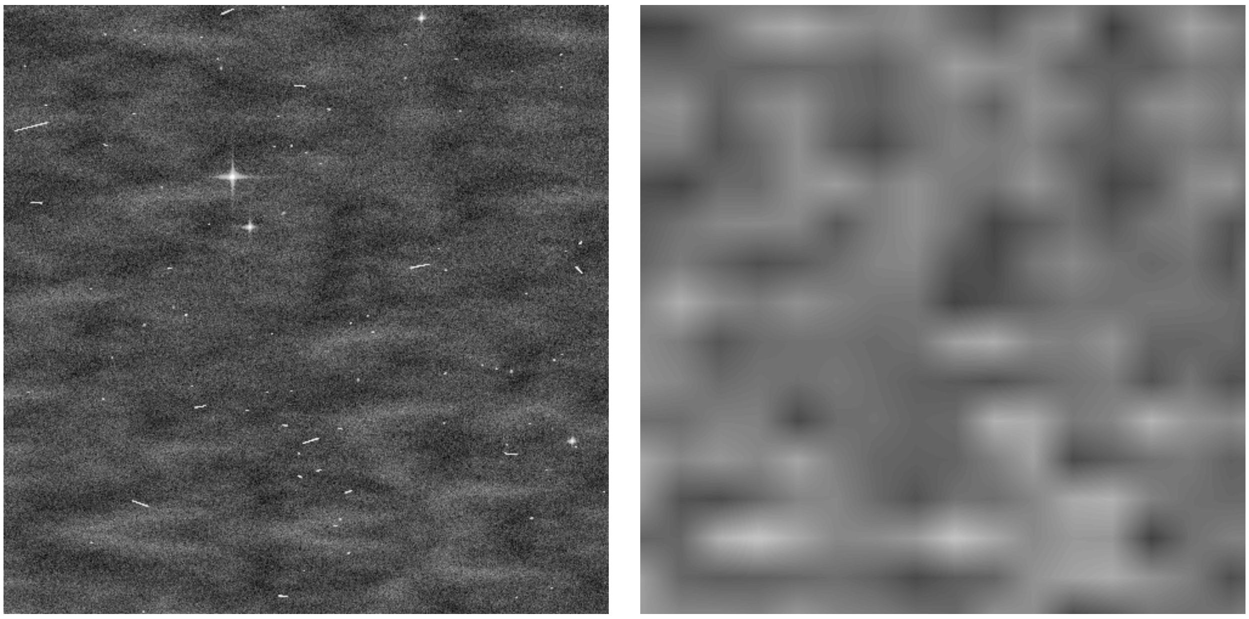


Figure 5.11: A synthetic case with textured background based on the imaging of the Orion nebula with a reliability threshold set at a minimum of 350 counts in the maximum bin (values have been re-normalised in the snapshot on the right to increase the contrast).

```
        1.1.2.2. Loop to 1.1.2.
    1.1.3. Loop to 1.1.
    1.3. Loop to 1.
```

5.4.2 Mode determination

As soon as hyperpixels and their four 4-ADU histograms are complete, the mode of the distribution of sample values can be determined. As a consequence, the mode determination is not carried out for every TDI or upon the reception of every sample, but only once every 32 TDI lines (ie. every 64 TDIs due to the $2x2$ binning).

The core of the procedure determines the bins of maximum population, checks their reliability and determines the interpolated mode values before the four estimates are combined. If one of the populations is found to be unreliable, the replacement (section 4.4.2) discards the estimate and proceeds based on those from valid neighbouring hyperpixels.

```
1. Rotate the mode buffers (two series of mode estimates are required for interpolation).
2. For each hyperpixel (from 0 to 29).
  2.1 For each re-binned histogram (from 0 to 3).
    2.1.1. For each bin (in decreasing order of value).
      2.1.1.1. Test the count against the previous bin's.
        2.1.1.1. $\geq$: Record the current bin as the maximum (mode).
      2.1.1.2. Loop to 2.1.1.
    2.1.2. Determine the ADU value corresponding to the recorded bin.
    2.1.3. Test the bin of maximum count for reliability.
      2.1.3.1. False: mark as unreliable (flag), skip to 2.
    2.1.4. Adjust the parabolic profile on the maximum and neighbouring bins.
    2.1.5. Loop to 2.1.
```

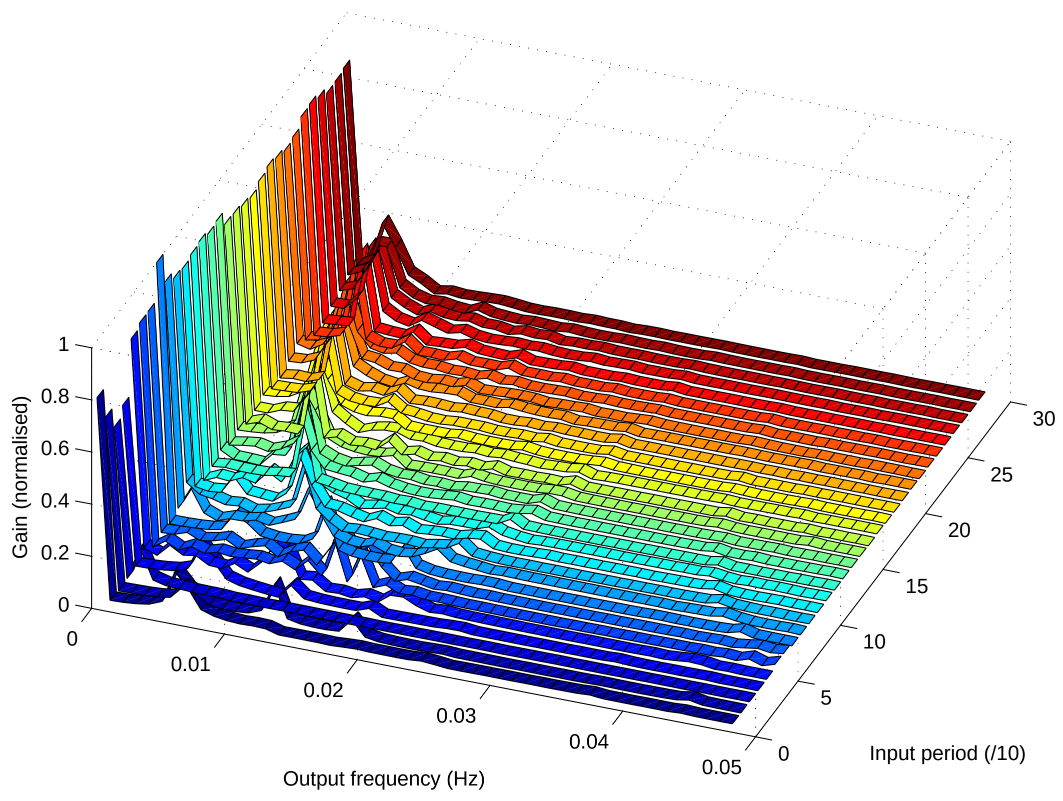


Figure 5.12: Frequency response obtained as a result of the selected hyperpixel size and interpolation scheme (with the replacement mechanism inhibited) with inputs as illustrated Fig. 5.13 and periods represented in units of 10 samples.

2.2. Combine the four background estimates.

2.3. Correct the bias on the mode found.

2.4. Test the previous mode's reliable flag.

2.4.1. True: skip to 3.

2.4.2. False: replace the previous mode depending on the penultimate's and current's flags (p,c).

2.4.2.1. (True, True): Use the average of the penultimate and the current one.

2.4.2.2. (True, False): Propagate the penultimate value.

2.4.2.3. (False, True): Propagate the current value.

2.4.2.4. (False, False): Use the default value.

2.5. Loop to 2.

3. Reset all histograms.

The quadratic fit is carried out in a non-iterative fashion based on the formula proposed in [4]. It corresponds to the first step taken to recover the resolution lost when quantising the sample values and when forming 4-ADU bins. With F_j the electron count in bin j, *index* the index corresponding to the bin of maximum count and *mode* the associated ADU value, the refined mode estimates are obtained as:

$$mode + 2\frac{F_{index-1} - F_{index+1}}{F_{index-1} - 2F_{index} + F_{index+1}} \tag{5.4}$$

The four estimates thus derived can then be averaged to further improve the precision which leads to formula (11.35) presented in chapter 11.

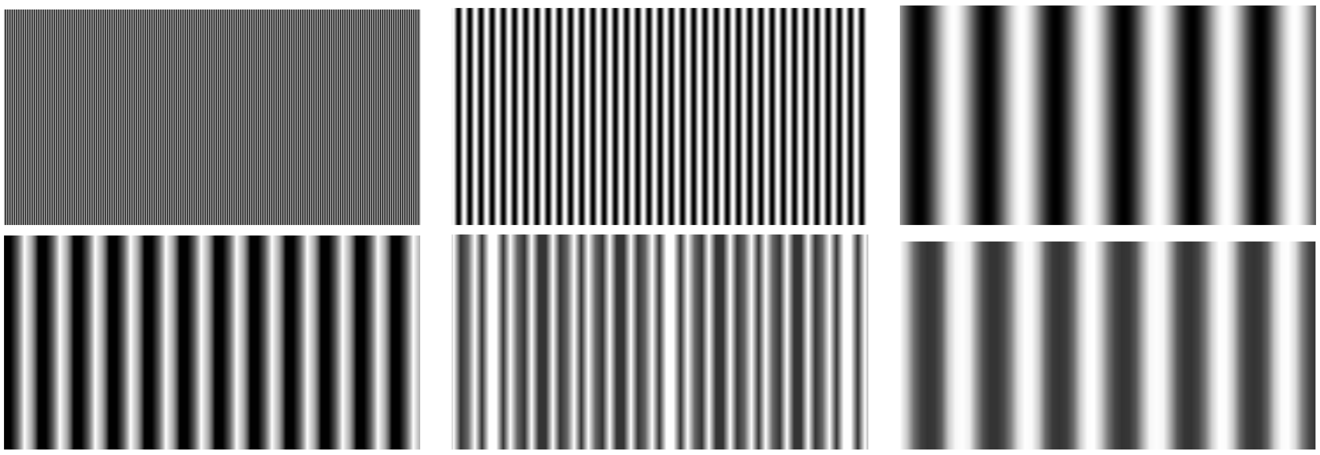


Figure 5.13: Test cases (top) and associated background maps (bottom) for measuring the frequency response: sine signals with 10, 50 and 300-sample periods (from left to right) in AL but constant in AC.

5.4.3 Interpolation

Once the mode values have been determined, they are placed at the centre of each hyperpixel and background estimates can be produced for all lines located between the last two sets of modes (see Fig. 5.8). For obvious reasons of spreading the processing load and of fixed scheduling, the map is not determined for the 32 lines at once but is, instead, produced sample per sample just in time for SNR-thresholding.

With i and j the AL and AC coordinates of the samples within the interpolation cell (W and H the respective sizes of the cells[10]), the two-dimensional bilinear interpolation reads:

$$\begin{aligned} mode_{0,0}(1-\tfrac{i}{W})(1-\tfrac{j}{H}) + mode_{W,0} \times \tfrac{i}{W}(1-\tfrac{j}{H}) \\ + mode_{0,H} \times \tfrac{j}{H}(1-\tfrac{i}{W}) + mode_{W,H} \times \tfrac{ij}{WH} \rightarrow background_{i,j} \end{aligned} \tag{5.5}$$

Since it is performed in the same raster order as that of the samples, two frequencies can be identified respectively for terms changing for every sample or only between TDI lines and this may be exploited to optimise calculations. Since $\frac{i}{W}$ is constant on a line, the interpolation formula may be rewritten as:

$$\begin{aligned} \underbrace{mode_{0,0} + \frac{i}{W}(mode_{W,0} - mode_{0,0})}_{\text{Constant: } A} \\ + \tfrac{j}{H}\underbrace{\left[mode_{0,H} + \frac{i}{W}(mode_{W,H} - mode_{0.H}) - \left[mode_{0,0} + \frac{i}{W}(mode_{W,0} - mode_{0,0})\right]\right]}_{\text{Constant: } B} \rightarrow background_{i,j} \end{aligned} \tag{5.6}$$

to isolate the constant terms for a notable benefit as far as software complexity is concerned since these may then be computed outside of the high frequency loop on the AC positions[11].

```
1. For each cell in the interpolation grid (from 0 to 27).
   1.1. Compute the values of the $A$ and $B$ constants.
   1.2. For each sample in the cell (on the current line).
        1.2.1. Determine the local background estimate: $A + \frac{i}{W}B$.
2. Determine the two constant values to be propagated to the limit of the CCD.
   2.1 $C = mode_{0,0} + \frac{i}{W}(mode_{W,0} - mode_{0,0})$ for the lower AC border.
   2.2 $D = mode_{0,H} + \frac{i}{W}(mode_{W,H} - mode_{0,H})$ for the upper AC border.
3. Assign $C$ to the 15 lower sample locations.
4. Assign $D$ to the 40 upper sample locations.
```

[10]Hence $\{(0,0),(0,H-1),(W-1,0),(W-1,H-1)\}$ correspond to the corners of the square cell (see Fig. 5.8).

[11]This formulation is, however, less hardware friendly since (5.5) lends itself more favourably to resource sharing and will be preferred in section 12.3.6

5.5 Conclusion

The strategy adopted in this chapter for background estimation is fundamentally based on robustness to perturbation while simultaneously providing accurate estimates to fulfil our requirements. As opposed to adaptive or iterative approaches, it has fixed complexity and runs identically for all possible configurations. As opposed to linear methods it achieves a quasi-low-pass behaviour with very limited arithmetics and latency although at the expense of rendering performance evaluation more difficult in the general case. The technical reasons make it perfectly suited to our architecture, while extensive testing has confirmed that it performs well under realistic conditions.

Chapter 6

Simple object model

Contents

6.1 Introduction

Following the strategy defined in section 3.5.2 and after the background estimation described in the previous chapter, the segmentation of the input image may be produced. This is a two-step process: examining the samples individually to select only the relevant ones then collecting them in regions corresponding to a simple object model (SOM). The first, conceptually very simple, essentially requires some care as regards the numerical properties of the fixed-point implementation (section 11.3.4) whose bases are briefly laid out here. For the second, on the opposite, the traditional recursive CC-labelling algorithm is found to be completely inappropriate for the targeted architecture and algorithmic developments are described to derive a method combining fixed complexity with operation compatible with the natural ordering of samples (raster-scan).

6.2 SOM

In line with the reasoning carried out in sections 3.4.3 and 3.5.1, forming objects at this stage has a twofold objective: reducing the data flow by discarding samples associated to the background and structuring the remaining data to transfer it to the software-based tasks. As such, the SOM needs not be extraordinarily complex but should rather rely on few assumptions and be applicable to all objects to set firm grounds for the ensuing object characterisation. Additionally, beside the member samples themselves, it should provide some information on the objects as a prerequisite for Req. 11 and Req. 12 because the limited processing resources on the software side induce the possibility of failing to analyse all the objects produced in all configurations under the fixed time constraint[1].

The incoming optical signal being fundamentally additive, the presence of an object of interest must translate in a contribution at a significant level compared to the local background and to the fluctuations of the total noise (Req. 7). Relying on an SNR-threshold after subtraction of the estimated background is natural for detecting this. Besides, with the essential astrometric and photometric information located in the central lobe of the PSF, observations should

[1]As its operates with fixed scheduling, the hardware is on the opposite capable of handling worst case conditions on a permanent regime by design.

be carried out in this region. This, in turn, implies that detection and the resulting location estimates need only focus on it and emphasises the importance of connectivity. Hence the decision to collect relevant samples in CCs.

From an image point of view, the SOM corresponds essentially to a “blob” of bright samples. Whereas this might seem coarse as compared to the more elaborate models finely mimicking the PSF introduced by matched filtering techniques for instance, it has the considerable advantage of simplicity and of depending on very limited assumptions – something which makes it generic by nature and involves only few parameters. Scientifically speaking, these objects are compound structures composed of a combination of one or many images of stars, of high and textured background, of noise fluctuations and of PPEs. Determining the number of components and separating them are essential prerequisites to interpret the data in meaningful terms and are the purpose of the fine object model introduced in chapter 7, but the SOM answers the need to narrow down investigations to a few areas of interest while encompassing all the necessary information for more subtle characterisation.

6.3 Sample selection

The logic above requires collecting a variety of information concerning the image. For each sample, the pre-calibrated value, the background contribution at the corresponding location and an assessment of the expected fluctuations related to noise are necessary, thereby making sample selection a “rendez-vous” point and a key driver of the overall scheduling. The pre-calibrated value is fetched from the buffer introduced to accommodate the delay related to the background estimation. The background value is produced by interpolation just-in-time for the sample selection. As for noise, the two terms A and B in equation 4.5 are necessarily treated differently. As in section 5.2.3, an ideal CCD is assumed and, hence, noise on B becomes stationary because of the high stability of the FPA. Its variance (σ^2_{RON}) is then dominated by the RON and may be calibrated on ground. Conversely, the uncertainty on A depends mainly on the signal received so that its variance is connected to the measured value. With the assumption above A reduces to a Poisson variable and an estimate of its variance is simply $samp_i$.

In the usual frame of the information theory, the amount of relevant information at location j is then estimated sample-wise with a SNR and compared to the threshold (`params.SNR1_TH`):

$$\frac{samp_j - bkgd_j}{\sqrt{samp_j + \sigma^2_{RON}}} >^? SNR \tag{6.1}$$

which is rather computed as

$$(samp_j - bkgd_j)^2 >^? SNR^2 \times (samp_j + \sigma^2_{RON}) \tag{6.2}$$

for evident reason of simplicity – this last expression being evaluated with fixed-point arithmetics as detailed in section 11.3.4.

To fulfil Req. 8 in all circumstances, a conservative threshold is adopted for this stage with the intent of minimising type II errors (false negative). In spite of this, it achieves a drastic reduction of the volume of data because, in the general case, the majority of the sky appears as almost empty at Gaia’s angular resolution and sensitivity. Tab. 6.1 presents the ratios of samples retained under different density configurations and shows that only a fraction of the samples are retained even in crowded areas. The limit reached at very low star densities corresponds mainly to the PPEs and false detections. They are subsequently handled as part of the characterisation effort (chapter 7) to reduce type I errors. To obtain an optimal balance between type I and type II errors, the corresponding thresholds are determined jointly (section 7.2.1).

Density (stars/deg^2)	Test case	Fraction of samples retained (%)
3.31 million	Baade	7.56
609 000	l54b0	2.37
195 000	l74b0	1.29
22 250	l74b15	0.94
3 500	l74b74	0.94

Table 6.1: Ratio of total number of samples retained as a result of sample-wise thresholding (with the threshold params.SNR1_TH in Tab B.2).

It is worth recalling here that M. Irwin’s original pipeline does not rely directly on $samp_j$ for estimating the signal, but rather on the result of a convolution filter (item 3 in section 3.5) whose kernel mimics the PSF and which is intended to enhance the SNR. It was shown to be essentially useful for detecting objects beyond the magnitude limit

($G > 20$) in our case. In beginning of life conditions, this would quite simply call for extra processing to discard these objects. It might have proved useful later on to recover the sensitivity lost as a result of the ageing of the detectors,but a preferable alternative would consist in releasing the gate permanently active in SM (Tab. 3.1). Hence, although first adopted, it was then suppressed for the benefit of reducing the complexity of the sample-based processing[2].

6.4 Connected-component labelling

6.4.1 Introduction

CC labelling refers to attributing labels to samples in a binary image in such a way that samples of identical label form path-wise connected sets. Intuitive CC-search relies on region-growing mechanisms: once a sample has been identified, the procedure attaches those of its neighbours relevant to the same component according to the selected connectivity relationship and proceeds from there on, until all the samples of the CC have been treated. This translates in recursive algorithms suitable for software implementation but not for hardware because such approaches lead to traversing the data in a manner which is data-dependent and the high addressing costs as well as the unpredictable paths conflict with the gradual availability of the data resulting from the TDI. Even in software, the complex structure of the processing makes it difficult to integrate within a real-time framework and to manage resources in an optimal yet simple way.

Conversely, the high input data rate and the stringent real-time constraint call for low-level implementation with limited addressing overhead and manageable "worst case" complexity. The algorithm described in the next sections has been designed with these flaws in mind. It is a purely TDI-synchronous scheme which processes samples in their raster order with a constant number of operations. It, moreover, relies only on elementary operations and data structures which are both easy and efficient to manage and implement in hardware.

6.4.2 Definitions

We provide in this section a brief reminder of the meaning we associate to CC and of a few terms used in what follows. All apply to binary images for which only those samples valued 1 are considered to exist − *ie.* are part of the *image* set.

Definition 1 *Neighbourhood*
For any sample of a digital image we define a neighbourhood. This formalises the intuitive notion of adjacency of two samples. In particular, for samples in a square mesh, we define a 8-neighbourhood by:

$$\begin{aligned} N_8(p_{x,y}) &= \{p_{x-1,y}, p_{x+1,y}, p_{x-1,y-1}, p_{x+1,y-1}, p_{x-1,y+1}, \\ &\quad p_{x+1,y+1}, p_{x,y-1}, p_{x,y+1}\} \bigcap \{p_{x,y} \in image\} \end{aligned} \tag{6.3}$$

An illustration of two different classical types of neighbourhoods in square meshes is given in Fig. 6.1.

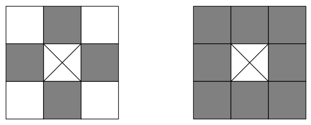


Figure 6.1: Two examples of neighbourhoods in square meshes (left: 4-neighbourhood, right: 8-neighbourhood).

Definition 2 *Connectivity*
We define the connectivity relationship as the equivalence relation[3] *according to which a given sample is connected to any of its own neighbours. 8-connectivity is defined according to the 8-neighbourhood.*

This definition renders the dependency on the definition of the neighbourhood most obvious. Given the adopted definition for the neighbourhood, it is equivalent to considering that the two samples are path-wise-connected inside the *image* set.

[2]It remains as an option in Pyxis which may be enabled during compilation.
[3]An equivalence relation R over a set S is a relation which is:

- reflexive: $xRx \; \forall x \in S$
- symmetric: whenever xRy then yRx
- transitive: if xRy and yRz then xRz.

Definition 3 *CC (Connected Component)*
The equivalence classes of the connectivity relation are the CCs and the quotient set[4] *of the image set is the set of all CCs.*

Despite this formal definition, this a very intuitive concept, as is illustrated Fig 6.2. The labelling procedure consists in assigning a unique index to all samples of the same CC and different indices to samples of different ones. Traversing the map of assigned indices then allows for retrieving the samples of any of the CCs.

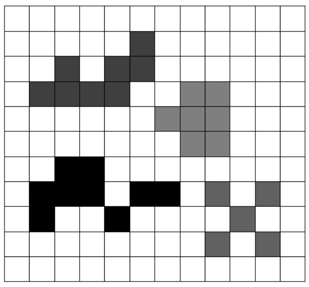


Figure 6.2: Four examples of CCs in 8-connectivity (different colours correspond to different labels).

6.4.3 Principle

Introduction

The principle of the search springs from adapting of the propagation algorithms to the additional constraint of fixed sample access induced by the raster-scan order. Labels still propagate throughout the data matrix, only they do so directionally – as opposed to isotropically: each sample's label is determined by looking at those located within its neighbourhood which have already been labelled. Depending on the configuration found either an existing label is propagated to the current sample or a new one is allocated.

The fact that we want to assign a unique label to each CC combined to the directional nature of the scan requires, however, some sophistication. Fig. 6.3 illustrates this. When sample 2 is processed it is found to be isolated and needs to be assigned a new label because sample 3 has not yet been looked at. It is only when the latter is read-out that the two parts can be united into single CC by relabelling sample 2 or 3. On the contrary, the label of sample 3 is propagated to 4 and 5 straightforwardly.

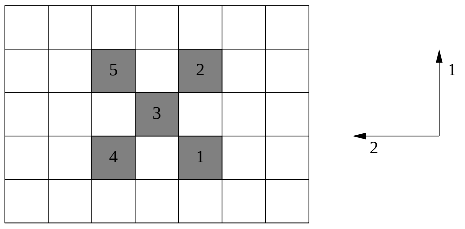


Figure 6.3: Impact of the directional access to samples on the labelling procedure (arrows correspond to scan order and numbers to the order in which the samples are processed). In all figures of this chapter, the addressing of samples is coherent with the raster order resulting from the CCD read-out. It proceeds first in AC since samples come in serial order because of the read-out shift register (arrow 1 from bottom to top) then in AL between one line of samples and the next 2 TDI periods later (arrow 2 from right to left).

This implies that a correspondence table between labels, used to map labels one onto another when existing fragments of CCs are found to be part of a larger one, must be maintained as the samples are being scanned. The equivalence between different labels is stored in this table which, in turn, allows for relabelling the parts of CCs being merged. This is the main difficulty of this approach. There exists a variety of methods to build and process this type of tables in the software case as reviewed in [41]. We here propose one – developed independently but which fits into the general framework established in [41] – adapted to the raster order of samples. As opposed to those optimising the average complexity, particular attention is here devoted to the predictability of operations together with efficient handling of labels and use of elementary logic and data structures in view of the target hardware implementation.

[4]As a set, the construction of a quotient set collapses each of the equivalence classes to a single element.

Architecture

The different labelling operations can be split into two groups, those which are susceptible to occur for each sample, namely at the sample level, and those which can be performed only once every line, the line level. In order to optimise complexity, proper use must be made of each level. Generally speaking, to avoid repetitively performing a given operation and for to reduce complexity, tasks are preferably handled at the line level if possible rather than at the sample one.

Sample level processing

We can summarise the principle of the search in the following way: each sample is processed in raster order and, for each, only a limited number of configurations can present themselves (Fig 6.4). If the sample has been retained by SNR-thresholding, only one of three different actions will need to be taken:

1. if none of the processed neighbours possess a label, then fetch an available label and initiate a new sample packet,
2. if all the processed neighbours possess the same label, then propagate this value and append the sample to the corresponding packet,
3. if the processed neighbours possess different labels, then choose one, append the sample to the corresponding sample packet and update the relabelling table.

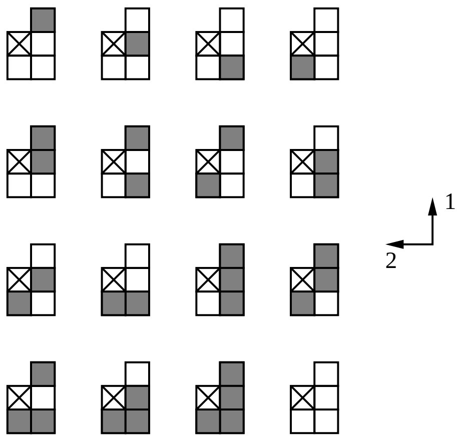


Figure 6.4: Possible configurations while processing a new sample (marked with an "X"). Only those samples already processed are shown (from the previous line (right) and the current one (bottom)) and those having been assigned a label are in grey.

Considering that access to each sample and to its processed neighbours follows a fixed pattern and leads, hence, to sufficiently elementary address computations, there remain two main development drivers: the handling of the relabelling table and of the labels in the image for relabelling the CCs should be very simple while ensuring coherence.

Line level processing

After the completion of the sample level processing for the line, three maintenance tasks need to be performed:

- relabel the on-going CCs and merge associated sample packets,
- extract the completed CCs and update the label statuses,
- update the table of available label.

6.4.4 Relabelling

Considering that there exist equivalence relations between labels and then choosing a unique label value within each class conceptually requires to adopt graph-based approaches to the relabelling problem. The problem can be significantly simplified by doing both at once. Hence, discarding the symmetry of the equivalence relation, the graph structure can be transformed into a set of trees. Each node is then a label value and each directed edge a link to another label value of the same CC to which it is to be mapped.

Labels attached to the same CC are then connected and an appropriate relabelling strategy structures the label space as a set of independent hierarchies – *ie.* a forest of n-ary trees. In particular, one must ensure that circular paths are never introduced but instead that all relabelling paths converge to a single element within each set of "connected" labels – the root of each tree. Merging fragments of CCs and relabelling them can then been seen as a matter of connecting trees together in a way which preserves the hierarchical structure, that is, which ensures than for any given label there exist a path to a unique root element. This representation provides clear insight on how to maintain correctness. As one may imagine, achieving efficient and reliable relabelling in spite of the a priori unlimited depths of the trees and for configurations of arbitrary complexity is a central design driver.

Relabelling is preferably performed at the line level for complexity reasons and to avoid repeatedly changing labels. The delay introduced between the moment when a sample is discovered which connects two parts of the same CC and the effective relabelling calls for some caution however. Whereas one might think that it would suffice to ensure that only root elements are ever mapped to other label values, configurations in which this destination value is itself mapped to a child of the afore-mentioned root element before relabelling is carried out can be imagined and lead to forming circular paths. The previous sufficient condition for correctness only holds if relabelling is not delayed and is performed at the sample level. In the adopted line-level paradigm, this condition must be made somewhat stronger and root elements should only be mapped to other root elements. It is worth noting, complexity-wise, that although this requires an additional search for the second root element, it leads to relabelling trees of reduced depth (Fig. 6.6) and does not incur any extra cost in the end.

6.4.5 Algorithm

This section presents the architecture of the algorithm both at the sample level and at the line level: the labelling and relabelling strategies, then the extraction of completed CCs and the recycling of labels.

Labelling strategy

The labelling and relabelling strategies must be designed jointly. On one hand, the labelling scheme should arrange for minimising the number of updates to the relabelling table. On the other hand, relabelling all already processed samples each time a new correspondence is established must be avoided.

The first item impacts on the strategy used for propagating labels. Assuming the previous line has been properly relabelled, there exist only 3 different configurations leading to a new entry in the relabelling table (Fig 6.5). When any of these configurations occur, the relabelling table must be updated, but the label which is then assigned to the current sample can be chosen to avoid having to repeat this operation for the next sample. Because of the raster order imposed on the samples and the geometry of these configurations, propagating the "upper right" label is optimal. It does not ensure that re-mapping one label to another one will never occur, only that it will not be repeatedly so, thereby making it a rare event.

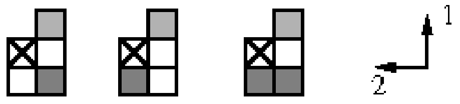


Figure 6.5: Configurations leading to relabelling parts of the same CC ('X' marks the sample being labelled, white ones have been discarded and the different tones of grey indicate different labels).

One work-around the second problem consists in refraining from storing label values at the sample level. Rather than storing these values in an array mimicking the sample matrix, one layer of abstraction is added with an array filled with addresses to label values. This allows for efficiently relabelling all samples which share the same address, independently of their location in the image. The two associated data structures are then: an array which assigns a Label Address (LA) to each retained sample and has a similar structure to the array of sample values[5] and a look-up table providing one Label Value (LV) for each address.

All samples are labelled as `DISCARDED` to start with. Samples are then studied sequentially in raster order. Considering the three different types of operations to be performed (section 6.4.3), the processing at the sample level then consists in either:

[5] *ie.* in raster order.

- fetching an available label address, inserting it in LA then using the numerical value of the address in LV (new CC),
- or assigning the processed neighbours' label address in LA (existing CC),
- or identifying the "upper right" label address, inserting it in LA and modifying the relabelling table (merging CCs).

Relabelling

In order to simplify the handling of the relabelling information and to keep data structures simple, the different trees are not stored as such but relabelling is handled via an operator which maps each label value to another one. Any path to a root element is, hence, the result of repeated compositions of this operator. Relabelling is then a three step process, first the operator is built by defining the target label values (sample level), then the relabelling trees are collapsed so that each leaf is directly mapped to the corresponding root element (sample and line level), finally, the label values are modified (line level).

The operator is efficiently implemented via a look-up table containing one entry for each possible label value: the relabelling map (RM). When a given label address/value pair is first put to use, address and value are identical and the relabelling operator maps the latter to itself – a property referred to as idempotence. Whenever the relabelling information must be updated (part of the sample level processing), two searches for the root elements of the relabelling trees to be merged are performed (for mapping root elements to root elements) by composing the operator repeatedly until idempotence is reached. Finally, the root label address associated to the "upper right" sample is assigned to the current one and RM is updated to connect the root elements.

The process of traversing the relabelling tree may justifiably be regarded as a recursive process of variable complexity. Even taking into account the facts that the previous line has been relabelled and that label propagation scheme is fully coherent with the raster scan order of the samples particular configurations can be imagined to make this problematic, as Fig. 6.6 illustrates. Extending the proposed pattern across the full AC dimension, the complete relabelling path extends over 245 levels since one is added every four samples.

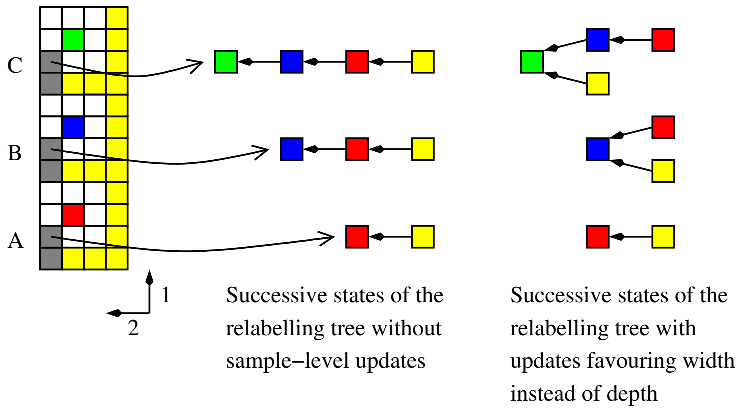

C
B
A
1
2
Successive states of the relabelling tree without sample−level updates
Successive states of the relabelling tree with updates favouring width instead of depth

Figure 6.6: Worst possible configuration as regard the depth of the relabelling tree for the sample of the current line (with samples to be labelled in grey, while yellow, red, blue and green ones are already labelled with different labels and white ones discarded). With the natural approach, when labelling the latest line (to the left), the red, blue and green CCs are successively identified (A, B and C) as being branches of the yellow one. The relabelling tree is then updated, as shown in the centre, which systematically requires traversing the entire tree to find the root element. By gradually collapsing this tree, width is preferred to depth and the relabelling tree is modified to reduce the number of look-up operations, as shown on the right.

This is clearly a concern at the sample level. To reduce the depths of the relabelling trees the structures may be updated whenever they are used to favour width instead of depth. This occurs, at the sample level, after the search for the root elements when building RM but does not completely solve the difficulty since multiple-level trees may still be formed (as shown Fig. 6.6). This calls, at the line level, for entirely collapsing all relabelling trees to keep the situation from further degenerating during the next line: as a consequence, only one-level-deep trees remain in the line just processed. As for the sample level, this operation is not performed element per element but rather path by path to keep the complexity down. Considering things globally, the complexity is only linear with the number of labels since, for each element, the immediate parent is studied only once and may be followed by a single relabelling operation – whereas proceeding label per label would yield a quadratic complexity.

Finally, one pass through RM suffices to update LA and merge the sample packets. Merging the different parts of CCs without moving the data in memory suggests to rely on linked lists of samples and, storing the addresses to the first and last of every list, merging may be efficiently performed. Software-wise it is generally preferable to place data at consecutive addresses in memory for pre-fetch and cache optimisations but as individual accesses are foreseen in hardware this is not a concern.

The situation used in Fig. 6.6 is fortunately only a theoretical case and, in practice, the geometry of the first lobe of the PSF makes such cases improbable. Tab. 6.2 provides representative values depending on density and shows that, in real life, average densities are the most complex to handle in this regard because of a maximum number of samples just above the noise limit and the erratic contours which ensue. Taking some margin above the maxima found a fixed number of compositions of the relabelling operator may be adopted in view of the hardware implantation.

Density	Sample level						Line level
	0	1	2	3	4	Max	Max
3200	70.09	28.50	1.28	0.12	0.01	5	15
22250	70.18	28.48	1.18	0.14	0.01	5	16
195000	68.22	30.40	1.25	0.11	0.008	6	18
609000	64.80	33.81	1.25	0.12	0.01	4	17
$3.31.10^6$	57.30	41.60	1.02	0.08	0.002	4	12

Table 6.2: Frequency of occurrence (%) of N-level (0, 1, 2, ...) deep searches at the sample and line levels for relabelling depending on density.

Extracting CCs

To limit the storage needs and make the best possible use of the available latency, efficient object handling must enable their extraction as soon as the underlying sample information has been fully gathered, as opposed to having to wait for some periodic event which would, additionally, lead to bursts of objects on the software side.

As suggested by [42], this can be simply achieved by:

- flagging the labels (Continued Labels (CL)) when they are assigned to a sample (at the sample level),
- comparing the flags CL of the current line and the previous one to extract all CCs to which no new sample has been added (at the line level).

This procedure, although inefficient in software, is fine for hardware since the underlying complexity is determined by the total number of usable labels and, as CCs can be inserted in the interface with the software with minimum delay after their completion, this number needs not be very large.

Recycling labels

- Values and addresses

  Recycling labels implies recycling both addresses and values and requires some caution because different constraints apply to each. As the algorithm relies on the propagation of label addresses – from one sample to its neighbours, including from one line to the next – each may only be re-used when all samples possessing it have exited the processing window[6]. As a consequence, label addresses and values can only be declared available after a delay corresponding to a full line of samples and are, hence, only scheduled for recycling at first, then recycled. Doing otherwise would result in reallocating a group of samples from its normal CC to the one just begun wherever it may be located.

  This management can be considerably simplified by fully taking into account the constraints which apply. On one hand, maintaining the correspondence between label addresses and values when allocating a new label (the same value is used both as address and value, see section 6.4.5) allows for recycling both at once. Moreover, an available address is then naturally associated to an available value so that only one search is required. While this limits the rate at which label values may be recycled, this is only of secondary importance since they are not the limiting factor: the algorithm uses more addresses than values due to relabelling. On the other hand, taking into account the necessary delay before a label address can be recycled, the recycling operations can be

[6] *ie.* the current line and the previous one.

postponed to the end of the current line. Hence, the management for both may be moved from the sample to the line level.

- Circumstances
  Two circumstances make the recycling of a label possible. The first corresponds to merging parts of CCs. The sample packets are then merged and one of the labels is freed or, rather, scheduled for recycling. The aforementioned delay is enforced thanks to relying on two queue structures which are exchanged once recycling has been carried out after the end of each line. The second case corresponds to extracting completed CCs. This is easily managed since the test for completion implies that these objects have had their last samples on the previous line. The full-line delay has accordingly already elapsed when the object is extracted and the associated label can be immediately recycled.
- Availability
  In order to avoid having to search for available labels, they are inserted in a shift register behaving like a FIFO queue. When each label's delay is spent, recycling simply consists in pushing it on the queue while labels are pulled out on demand.

Data structures

The main data structures required are, accordingly, the following:

- **Label Addresses (LA)** is an array of addresses in raster order allowing for simple correspondence between a sample's position and the label values stored in LV. In order to be able to handle the image boundaries within the exact same framework as the interior samples, LA is slightly over-sized: virtual addresses marked as `DISCARDED` are inserted before and after the samples to be able to process the bordering samples as interior ones. It is used for labelling the current line's samples and for looking up the labels of the previous line and is hence only two lines wide.
- **Label Values (LV)** is a simple LUT providing the label values themselves and allowing for efficient in-place relabelling. It is dimensioned by considering the conditions under which new label addresses are assigned. New entries in LA & LV are created with the same value.
- **Relabeling Map (RM)** is the relabelling LUT. For each possible label value it provides the target one to which it is to be transformed. It is hence an array of the same type and dimension as LV and is initialised to the identical transform – *ie.* maps each value to itself by default.
- **Relabelling Paths Memory (RPM)** is used to store the sequence of intermediate label values on a path to a root element. It is an array sized as LV for worst case conditions and values on the path are placed one after the other with the help of a simple counter tracking the number of elements inserted.
- **Available Addresses (AA)** is a shift register sized as LV and used to implement the FIFO queue of available label values. At initialisation it contains all label values.
- **Continued Labels (CL)** contain the flags indicating whether samples have been aggregated to each CC during the current line and the previous one. It is an array spanning two lines, each sized as LV, which are exchanged after the end of each line.
- **Freed Labels (FL)** are the two queues used to store the labels values scheduled for recycling and enforce the delay. Both are sized as LV and simply exchanged after the end of each line.

Pseudo-code

The identification of CCs proceeds based on four different processes: Seek, Relabel, Extract and Update described successively below.

```
1. Seek (sample level)
    1. Receive a retained sample ($[al][ac]$).
    2. Identify the local configuration by looking at the neighbours already processed ($[al][ac-1]$, $[al-1][ac-1]$, $[al-1][ac]$, $[al-1][ac+1]$): there can be two different labels only if [al-1][ac+1] is labelled, there is only one label if [al-1][ac] is labelled.
      2.1. If none of the neighbours are labelled (new CC or new branch of CC).
```

2.1.1. Pull a label address and value (`label`) from the queue of available ones AA.

2.1.2. Assign the address (`label`) in LA at the $[al][ac]$ position.

2.1.3. Enter the value (`label`) in LV at the `label` position.

2.1.4. Mark the CC as continued in CL.

2.1.5. Add the sample to a new object (initialise the flux, background measurements as well as the maximum value and bounding box with the values corresponding to the sample and increment the count of member samples).

2.2. Otherwise (one of the neighbours is labelled):

2.2.1. Insert the label address corresponding to the maximum AC coordinate among $[al][ac-1]$, $[al-1][ac-1]$, $[al-1][ac]$, $[al-1][ac+1]$ in LA.

2.2.2. Mark the CC as continued in CL.

2.2.3. Add the sample to the corresponding object (increment the flux and background measurements with the values corresponding to the sample, update the count of member samples, the maximum value and the bounding box if required).

2.2.4. If $[al-1][ac+1]$ and one of $[al-1, ac-1]$ and $[al][ac-1]$ have different label values: we need to merge two parts of a same CC.

2.2.4.1. Search for the root element of the tree to which $[al-1][ac+1]$ belongs (`label_1`) and temporarily store all label values on the path in RPM.

2.2.4.2. Insert `label_1` for all labels on RPM in RM.

2.2.4.3. Search for the root element of the tree to which the other labelled neighbour belongs (`label_2`) and temporarily store all label values on the path in RPM.

2.2.4.4. Insert `label_2` for all labels on the path in RM.

2.2.4.5. Schedule `label_2` to be relabelled into `label_1` with RM.

3. Loop to **1.**

2. **Relabel** (line level)

1. Squash all the relabelling trees: update the RM so that all entries map directly to a root element.

1.1. Search for a label value to be mapped to another one: compare RM to the identical transform.

1.2. If none have been found exit to **2.**

1.3. Search for the root element (`label`) of the tree to which the element belongs and temporarily store all label values on the path in RPM.

1.4. Insert `label` for all labels on the path in RM.

1.5. Loop to **1.1.**

2. Relabel.

2.1. Search for a label value (`label`) to be mapped to another one: compare RM to the identical transform (the destination label is now a root element).

2.2. If none have been found exit to **3.**

2.3. Relabel `label` to its tree's root (with RM).

2.5. Merge the objects corresponding to the two labels (connect the linked lists of samples, aggregate the flux and background measurements, combine the counts of member samples and update the bounding box as required).

2.6. Schedule `label` for recycling.

2.7. Reset RM for this label: transfer `label` into `label`.

2.8. Loop to **2.1.**

3. Exit.

3. **Extract** (line level)

1. Search for a label used during the previous line in CL.

2. Compare to the status in the current line in CL.

  2.1. If still used, loop to **1.**

3. Check that the object has not been extracted or merged with another one.

  3.1. Test the number of member samples, if 0 loop to **1.**

4. Publish the object to the software part.

  4.1. Insert the object address in the object queues.

  4.2. Recycle the label value (insert it AA).

  4.3. Loop to **1.**

4. **Update** (line level)

   1. Recycle the labels whose delay has expired from the older line of FL.

   2. Reset the older line of FL and rotate the two lines.

   3. Reset the older line of CL and rotate the two lines.

   4. Reset the older lines of LA and rotate the two lines.

6.5 SOM Descriptors

For the purpose of populating the object queues described in the next section, of meeting Spec. 8 as well as Req. 11, or to avoid having to traverse the entire list of member samples in software, a few descriptors[7] are built as the objects are formed:

- *flux*: sum of member samples' ADU values (with the offset).
- *background_flux*: sum of background estimates at the member samples' locations.
- *nb_pix*: count of member samples (provides a measure of area on the CCD and allows for subtracting the offset from *flux*).
- *saturated*: binary flag marking the presence of saturated member samples.
- *bounding_box*: AL and AC coordinates of the smallest rectangle containing the object (useful for restoring the connectivity of samples for component segmentation (section 7.3)).

6.6 Object queues

With purely synchronous sample-based operations on one side and asynchronous object characterisation on the other, transferring data from the first to the other calls for a flexible interface to avoid propagating unnecessary constraints between the two. With this in mind, static memory (SRAM) may be introduced for this purpose with the benefits of not requiring any direct data communication between software and hardware and acting as an input buffer to the software where objects remain until they can be handled by the system. Finally, the corresponding storage space can be exploited, on the hardware side, to construct the objects in place and minimise data movement.

As a consequence of the limited latency available, this memory space is best structured, from a high level point of view, as a FIFO queue to optimise data accesses and reduce the average time objects remain "in transit". For the sake of satisfying Req. 11, this concept may be extended to several queues, one for each priority level, to additionally make objects accessible on a priority basis as represented in Fig. 6.7. With on-board priorities in the general sense defined as decreasing with magnitude, the *flux* descriptor constructed simultaneously to the CCs can be put to immediate use to form magnitude classes and control object insertion in the queues.

[7] An extension of the previous CC-labelling scheme may be briefly mentioned at this point because reference will be made in section 7.2 to several lists of samples. For a first version of the hardware it represents extra intricacies, however, which given the already sophisticated processing proposed in this part and the numerous complexities of the implementation in the next one, it is probably not desirable to describe in too much detail. It is sufficient to say that:

- samples may be stored in different lists depending on whether they are interior, on the edge of objects (8-connected to samples discarded by the SNR-threshold stage), or saturated and separate *flux* and *nb_pix* descriptors are then built for each list in addition to the global ones;
- this can be carried out when aggregating samples to CCs with two additional tests, one on the sample value for detecting saturation, the other on the retained or discarded status of the four neighbours not tested by CC-labelling;
- the reason for performing this as part of CC-labelling is that it fully exploits the tests required for labelling and offers an opportunity for the efficient creation of more descriptors, either to dismiss irrelevant objects prior even to their transmission to the software or to refine their characterisation afterwards without having to traverse the entire set of member samples.

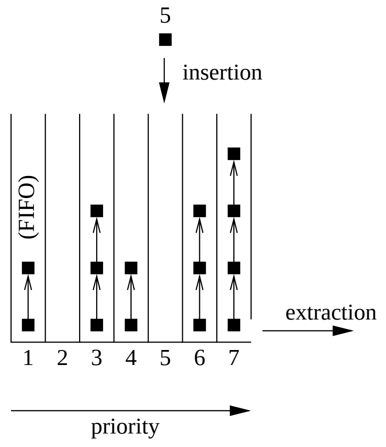


Figure 6.7: Structure of the priority-ordered object queues.

[43] and [5] refer to such an array of priority-ordered FIFO queues as a "hierarchical queue" for their purpose of efficiently implementing the watershed transform (section 7.3) and Klein et al. propose a hardware-friendly implementation for them in [6]. For the purpose of simple address and data management with static resources, both the linked lists of samples and the object queues follow the scheme represented Fig. 6.8. In addition to the links shown on the figure, the unused elements are also collected in a linked list to facilitate fetching them or freeing resources. These two aspects are respectively under the responsibility of the hardware (for forming new CC) and of the software (for recycling object resources), thereby guaranteeing that conflicting accesses can never occur.

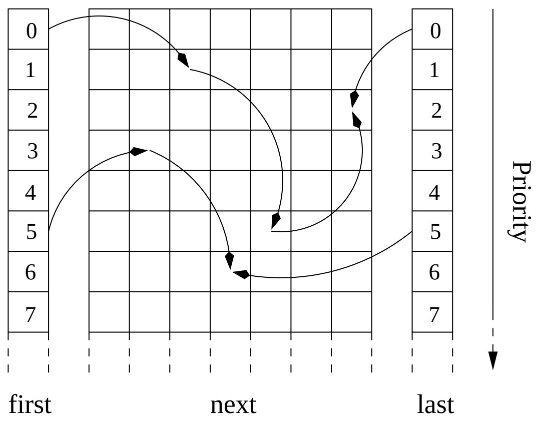


Figure 6.8: Implementation of linked lists based on static resources (courtesy of [6]).

6.7 Conclusion

The CC-labelling algorithm described in this chapter is fit for hardware implementation because it is mostly control logic relying on elementary operations, in predictable numbers, and with simple data management. Technically speaking, it is adapted to the ordering of sample data and can hence be naturally appended to the sample selection pipeline, while as it produces correctly labelled CCs and is of fixed complexity it nicely tops off the hardware implementation.

The resulting architectural organisation, by entrusting the entire chain leading to the formation of SOM-objects to hardware makes the most of this resource. Data-wise, this hardware / software segmentation is optimal as it allows background estimation, SNR-thresholding and CC-labelling to occur just-in-time within the same processing unit, thereby avoiding the need for intermediate storage or complex data exchanges. It is also relevant from a real-time point of view because the data intensive tasks are entirely under the responsibility of the hardware, thereby guaranteeing completion under arbitrary conditions. Finally, in agreement with Req. 11 & 12, it offers a possibility for an object-oriented interface with software, as an elegant transition to the realm of soft real-time for objects arriving at random with varied types.

Chapter 7

Fine object model

Contents

7.1 Introduction

This chapter builds on the objects produced as a result of CC-labelling together with the few assumptions concurring to the SOM. Beside the fact that this first model needs to be completed with measurements to fulfil Spec. 8, it must also be refined because of Spec. 7 and Req. 2 & 11. Indeed, its generic nature adapted to minimising type II errors (false negatives) is inadequate to filter out type I ones (false positives). For this purpose it is desirable to match the individual astrophysical sources (eg. components of DMS), hence a second object model is defined: the fine object model (FOM).

Unlike the sample-based processing, which is mainly concerned with processing a large amount of data in a systematic manner to identify and retain only its relevant fraction, the object-based tasks are driven by relevance. The diversity and the complexity of cases stem from the specialised way in which the objects should now considered. The intent is, naturally, to carry out only the required operations, precisely to make more complex treatments possible in the few cases which need them. With this in mind, the SOM objects are first submitted to a cascade of criteria to confirm their relevance (noise, PPEs) and narrow the field of investigation (single stars, DMS) to ensure the expense of carrying out increasingly complex processing is always justified. The resulting approach is intrinsically adaptive and well suited to software, in coherence with the intended architecture.

7.1.1 Structure

Software implementation, while offering increased flexibility as compared to hardware, faces processing capability limitations which do not, in the general case, permit proving statically that everything will be performed in time for all cases. Real time systems improve this situation by providing a framework in which the response delays are predictable. This allows for calculating the maximum load which can be handled and identifying configurations for which the constraints will not be satisfied – something especially relevant when the input flow is variable as is the case here. The soft real-time option allows the tasks to exceed their deadline, the end of the TDI period in our case, and makes it possible to spread the computations over larger timescales. This trades resources against time and instead of imposing that the processing for every object be completed in one TDI, only requires that it be compatible with the system's capabilities when averaged over the maximum allowed latency.

The available latency corresponds to the interval between the read-out of SM samples for an object and the next hard-real time operation to be carried out in connection to this object. With the flow presented in section 2.2.1 three deadlines may be identified respectively for configuring the charge injection, for enabling the first gate and for read-out in AF1[1] (section 1.3.3) but these delays must necessarily be shared with propagation, selection and management of the CCD interface.

Not being capable of always guaranteeing the completion of the processing within the allocated time slot in all cases (eg. Baade's windows, globular clusters), its structure should both maximise the probability that the most important aims are achieved and that the available information is exploited at each stage to select the next most relevant one and make the best possible use of the resources. The former calls for a priority-driven flow (Req. 11) and the second for decomposing it into successive steps implementing an adaptive behaviour.

Priority-driven processing

The objects queues described in section 6.6 provide flexibility to exploit the available latency because objects will remain in storage until they get a chance of being handled by the software. Implementing not one but several queues offers an opportunity to order objects in the interface and fetch them on a priority basis. Provided the SRAM is sufficiently large, graceful degradation of the performances may be ensured when arbitrary skies and observing conditions exceed the system's capabilities without the need to leave the nominal operating mode. For each object, the delay elapsed since its insertion in the queue may be measured by comparing the coordinates of its bounding box to the current TDI counter value. Those for which the maximum latency has been exceeded may then be signalled before being dismissed in agreement with Req. 12.

Finally, in spite of the argument presented in section 8.2.2 against SMP systems, it is worth mentioning that, since the objects are independent data units as a result of the SOM, it would also be possible to have several software engines characterising them in parallel provided shared access to the SRAM is rendered possible.

Descriptor-driven processing

The characterisation to be performed on board has a double objective: allow for discarding unwanted objects and provide the necessary information for selecting which objects to observe and for managing the data on board. It accordingly corresponds to mapping the detected objects to a number of pre-defined classes based on a set of descriptors. A coarse and incomplete list of such classes is:

- Single star
  - saturated ($G \lesssim 13$)
  - intermediate
  - faint star (notable influence of RON or Poisson noise)
  - fainter than $G = 20$
- False detection
  - related to the background
  - related to RON
  - related to Poisson noise
  - related to PSF rebounds
- PPEs
  - linear-shaped
  - normal incidence
  - low energy
- Others
  - DMS

[1]These correspond respectively to the instants when the object enters the CCD, crosses the first gate and reaches the read-out register.

– resolved objects (eg. planets or minor planets).

– objects on high or textured background

Ideally, to reduce the average complexity, the sequence of tasks should be optimised in respect to a measure of processing cost weighed by a representative distribution of object types and the uncertainty remaining as to their nature after each characterisation stage. Methods based on neural networks might have been used for this purpose to automatically yield an optimal organisation based on the selected learning paradigm and on a database of representative cases. In the absence of such a data set (section 7.4.2), the proposed structure has, however, been crafted manually based on an analysis of situations and simulations. For maximum efficiency, simple individual procedures have been devised based on the overall descriptors systematically computed as part of CC-labelling (except for the component segmentation stage of section 7.3). After almost each of these, a test is carried out to decide on the subsequent course of action. The corresponding cascade of tests and decisions compared to a decision tree rather than to a monolithic engine, as illustrated by Fig. 3.6, with the intent to characterise the objects adaptively.

From a high level point of view, three classes of operations may be identified respectively for filtering undesired detections (type I errors), for refining the object model and for measuring the objects. They are organised in this order for obvious performance reasons: to refrain from segmenting irrelevant objects or from measuring those whose structure will be subsequently altered. We adopt the same structure in the next sections of this chapter.

7.2 Undesirable detections

The first series of descriptors elaborated in this section primarily aims at discarding unwanted objects although this does not preclude using them for commanding observation or for managing the subsequent data for the objects of interest. As opposed to section 6.5, $flux$ designates, in all that follows, the total flux in ADUs but with the offset subtracted according to:

$$flux = total\_flux - nb\_pix \times \texttt{params.OFFSET\_ADU} \tag{7.1}$$

Taking advantage of the possibility of segregating samples based on the topology (section 6.5), we have additionally:

$$flux = flux_{edge} + flux_{interior} + flux_{saturated} \tag{7.2}$$

$$nb\_pix = nb\_pix_{edge} + nb\_pix_{interior} + nb\_pix_{saturated} \tag{7.3}$$

Finally, a number of thresholds and parameters are introduced for tuning the tests to the image properties and to the needs. These are denominated in what follows with the `params` prefix and are recalled in appendix B.

7.2.1 Faint objects

Very faint objects which are not of interest correspond to false detections related to noise or objects fainter than the limiting magnitude ($G = 20$). As these are potentially problematic because coming in large numbers, a series of simple criteria based on integral descriptors form the very first stage of suppressing false positive detections.

Considering that the descriptors are computed in the course of CC-labelling and that the first two tests are simple thresholds on the corresponding values, they could even be moved to the hardware side to save on the utilisation of resources at the interface (SRAM occupation and software time). With the proposed management of the object queues (section 6.6), however, this would require duplicating the logic used for recycling the addresses allocated to the objects. As, conversely, the corresponding objects are of minimum priority and, hence, collected typically in the same queue, they are not in the way of higher priority ones and may be discarded efficiently by the software when time becomes available.

1. Minimum number of member samples.
   After the removal of the filtering step (section 6.3), noise-related false detections are to be expected. Because the realisations of the different noise processes are independent from one sample to another and because their accumulated level is of the order of the minimum signal searched for, the vast majority of these are composed of a single or two samples. As illustrated Fig. 7.1, they are an easily identifiable population which can be efficiently pre-filtered:

$$\text{discard if:} \quad nb\_pix \leq \texttt{params.MIN\_SMP\_SELECTED} \tag{7.4}$$

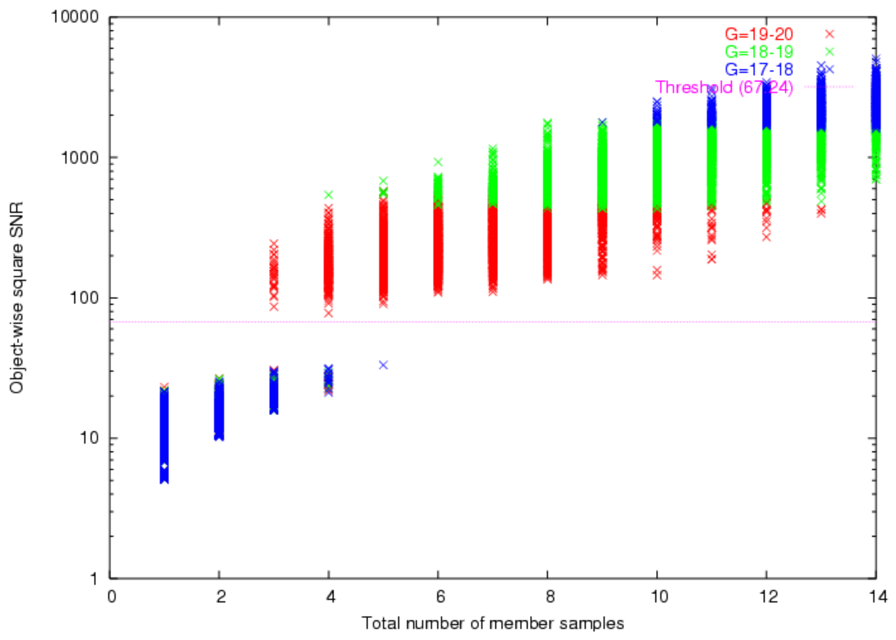


Figure 7.1: Object-wise square SNR as a function of the number of object-member samples in a field populated with magnitude 17 to 20 stars.

2. Magnitude cut-off.
   To further discriminated either very faint sources or false detections caused by noise, a finer test on the minimum admissible total flux value is carried out. The three-sample objects related to noise can be efficiently removed by this means which also allows for rejecting a significant fraction of objects fainter than magnitude 20. The following criterion is used:

$$\text{discard if:} \quad flux < \texttt{params.MIN\_FLUX\_SELECTED} \tag{7.5}$$

3. Object-wise SNR threshold.
   Finally, as in section 6.3, a SNR threshold is introduced object-wise judging the relevance of the objects formed versus the total noise. We use the following formulation with N the number of samples used to estimate the background:

$$\text{discard if:} \quad \frac{(flux - bkgd\_flux)^2}{flux + nb\_pix \times \sigma_{RON}^2(1 + \frac{nb\_pix}{N}) + nb\_pix\frac{bkgd\_flux}{N}} \leq \texttt{params.SNR2\_TH} \tag{7.6}$$

   The numerator corresponds to considering only the object's flux as signal, while the denominator accounts for the various types of noise involved (section 3.2):

   - the Poisson noise associated to the photons' statistics,
   - the Gaussian electronic chain noise,
   - and the measurement noise corresponding to the background (this latter point is only an approximation which would be appropriate were the background measured via a mean or a median).

   The threshold is set jointly with the sample-wise SNR-threshold to achieve a trade-off between type I and II errors based on simulations[2]. Enforcing the condition that at most a given number of false detections per million samples be produced, the node for which the number of detections is highest is retained. The elected SNR pair is: SNR1 = 3.15 (sample-wise) and SNR2 = 7.1 (object-wise) with a completeness at magnitude $G = 20$ fluctuating in the $[98\%, 98.61\%]$ range.

   This computation, being costly due to the floating point numbers and to the complexity of the formula used, is best applied only to objects in the vicinity of the threshold. The two previous tests are introduced precisely in

[2]The corresponding test case should be representative as regards the magnitude distribution of stars and the occurrence of apparent DMS related to crowding in the FOV. A typical density field in the galactic disk, as simulated by Gibis and the Besançon Galaxy model around $l = 74$ and $b = 0$, is found to meet these requirements while collecting sufficient statistics with 195000 stars/deg^2.

this sense to pre-filter objects far below the threshold. Conversely, as the object queues are magnitude-related, the detailed calculation is only required for objects in the class of lowest admissible signal – brighter objects automatically validate this criterion.

7.2.2 PPEs

The second category of objects to be discarded as early as possible are the PPEs, both of solar origin (protons) and from the Galaxy (protons and heavier ions). The intent at detection's level is to suppress PPEs in a manner which strongly favours type I errors (preserving PPEs) against type II ones (rejecting objects of interest). Indeed, this filtering only aims at reducing the number of objects to be observed in AF1 to alleviate the processing load but confirmation (section 2.2.1) remains the preferred means of rejecting difficult PPEs. Accordingly, conservative thresholds are used as a means to achieve very high confidence levels on the tests.

The diffuse contribution of secondary particles (section 2.3.2) being currently neglected by the simulation tools[3], dictionaries are used in Gibis to paste pre-computed patterns in the image. These events are remarkable concerning the amounts of energy deposited which, being largely above the noise level, imply that the profiles are characteristic. The same procedures are then used for discriminating the PPEs from the two origins and derive mainly from geometrical considerations. The possibility of segregating interior, edge or saturated samples as part of CC-labelling (section 6.5) is largely exploited here because the gradients are relevant for discriminating PPEs from optical images.

1. Interior samples.
   The linear geometry of PPEs translates into a limited number of interior samples. For low signals this is not a very discriminatory property, but for larger values of object flux, stars extend over large domains with many interior samples while this number remains limited for PPEs. This is illustrated Fig. 7.2.

$$\begin{array}{rcl} \text{if} & : & flux > \texttt{params.INTERIOR\_FLUX\_TH} \\ \text{discard if} & : & nb\_pix_{interior} < flux/\texttt{params.INTERIOR\_SLOPE\_TH} \end{array} \tag{7.7}$$

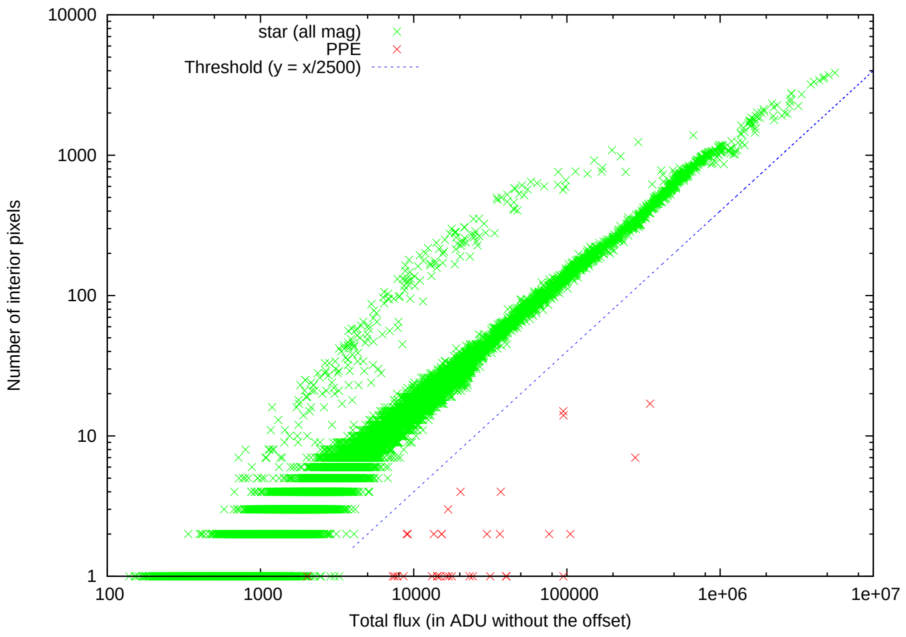


Figure 7.2: Discrimination of PPEs based on the number of interior samples.

2. Mean edge value.
   Objects imaged through the optical system do not have sharp edges because the derivatives of the instrument's PSF are bounded. On the opposite, PPEs feature abrupt edges corresponding to the limited diffusion of the electron-positron pairs generated along the particle's trail. The average directional gradient (towards the exterior) can be simply approximated by considering the difference between the mean edge sample value and the

[3]This statement holds for both Gibis 2 and 3.

background contribution along the object's perimeter. With the low frequency content of the background map, this may be further approximated as the mean sample value on the edge after subtracting the background flux.

$$\text{discard if :} flux_{edge}/nb\_pix_{edge} - bkgd\_flux/nb\_pix > \texttt{params.EDGE\_GRAD\_TH} \tag{7.8}$$

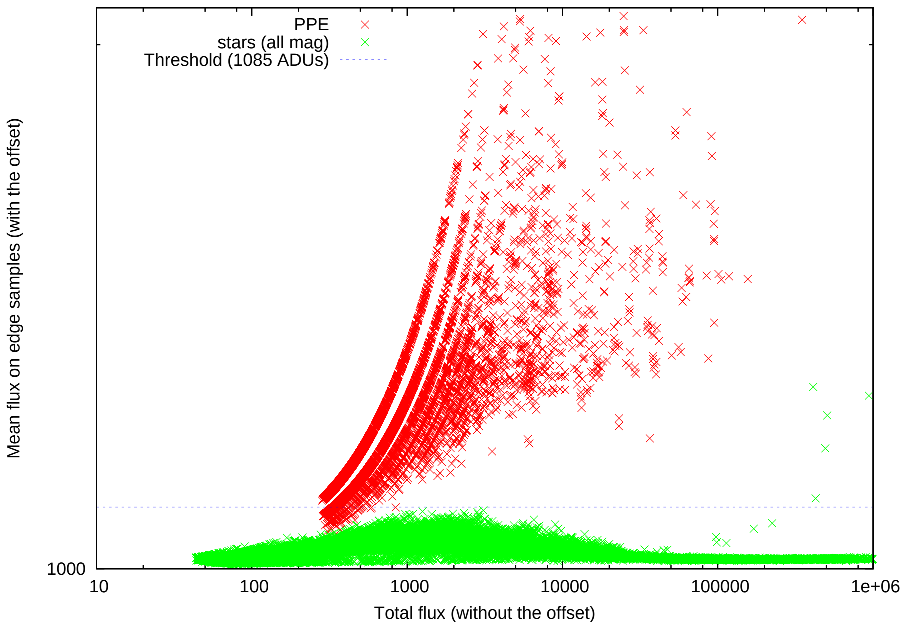


Figure 7.3: Discrimination of PPEs based on the gradient on the edges. The outliers on the bright end correspond to a simulation artifact (the spikes of bright stars are truncated in Gibis due to the insufficient modelling of the PSFs far from the centroid).

3. Flux/surface (mean sample value).
   Because PPEs produce signal concentrated along their trajectory, instead of scattered as a result of diffraction, the more energetic impacts yield images with greater energy densities for a given amount of total signal. This class of events may therefore be discriminated through a test on the average energy density (compared to the total number of samples).

$$\begin{aligned}\text{discard if : } \frac{flux}{nb\_pix} &> \texttt{params.COSMIC\_FLUX\_TEST} \\ &+\texttt{params.COSMIC\_SURFACE\_TEST} \times \sqrt{nb\_pix}\end{aligned} \tag{7.9}$$

4. AL & AC alignment.
   This criterion uses the positions of the samples that are on the perimeter of the object's bounding box to test the AL and AC orientations of the maximum radii. Fig. 7.5 illustrates which samples are retained to be sensitive to deviations from the standard cross-shaped behaviour connected to the PSF. Misalignment is then measured using:

$$\text{misalignment measure : } \left| \frac{ac_2 - ac_1}{ac_3 - ac_4 + 1} \times \frac{al_3 - al_4}{al_2 - al_1 + 1} \right| \tag{7.10}$$

yielding the following piece-wise threshold (Fig. 7.6):

$$\begin{cases} \texttt{params.MISALIGN\_MAX} & \text{if} \quad flux < \texttt{params.MISALIGN\_FLUX1} \\ \texttt{params.MISALIGN\_TH1}/(flux - \texttt{params.MISALIGN\_TH2}) & \text{if} \quad flux < \texttt{params.MISALIGN\_FLUX2} \\ \texttt{params.MISALIGN\_MIN} & \text{otherwise.} \end{cases} \tag{7.11}$$

As illustrated Fig. 7.6, objects exceeding the threshold are then either PPEs or DMS. Working with a population of reduced diversity, these are further discriminated easily by considering a geometrical criterion (Fig. 7.7). The one retained evaluates:

$$\begin{cases} \text{if :} \quad nb\_pix_{edge}/nb\_pix < \texttt{params.PPE\_DMS\_EDGE\_TH} \quad \text{then DMS} \\ \text{otherwise: PPE.} \end{cases} \tag{7.12}$$

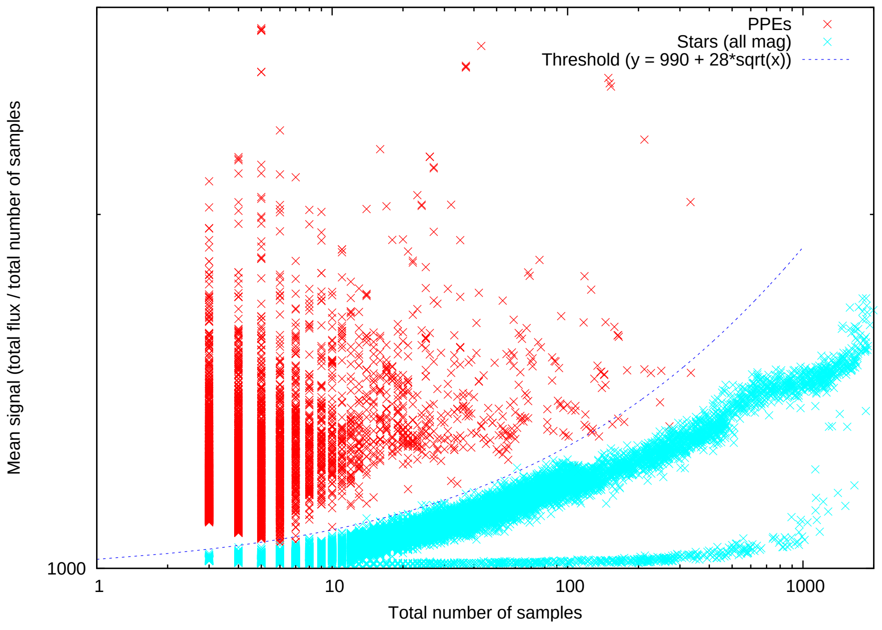


Figure 7.4: Discrimination of PPEs based on the average sample value.

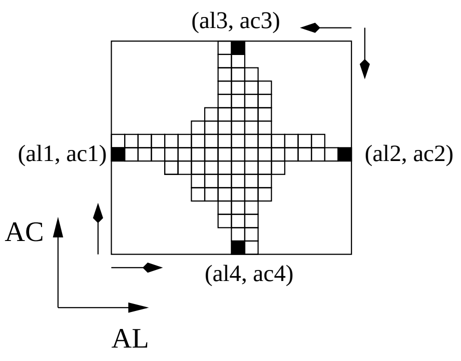


Figure 7.5: Identification of the maximum radii (the arrows in the upper right and lower left corners indicate that the first samples encountered in the corresponding directions are retained for measuring maximum radii).

7.3 Component segmentation

Having discarded a majority of spurious objects, this section describes how components of compound objects may be identified. After discussing the underlying rationale, the selected algorithm is presented with an emphasis on avoiding over-segmentation. This process, although yielding some information on type (DMS), is not, strictly speaking, a descriptor like the previous ones because it alters the object representation and should hence be carried out before the final measurements are made (section 7.4).

7.3.1 Motivation

The SOM, as it does not differentiate between single sources and close-knit multiple ones poses a number of difficulties on board. Although the objects are self-consistent units because all the information associated to the underlying sources is included, the reliability of the characterisation is greatly weakened since one does not know how many sources it describes.

There exists a variety of situations of astronomical and instrumental origins causing sources to blend into a single extended object. The first category is composed mainly of visual binaries or multiple stars and resolved stars in extended sources (galaxies, globular clusters), while the second relates to insufficient angular resolution, high PPE rates or the superposition of the two fields of view (Fig. 7.8).

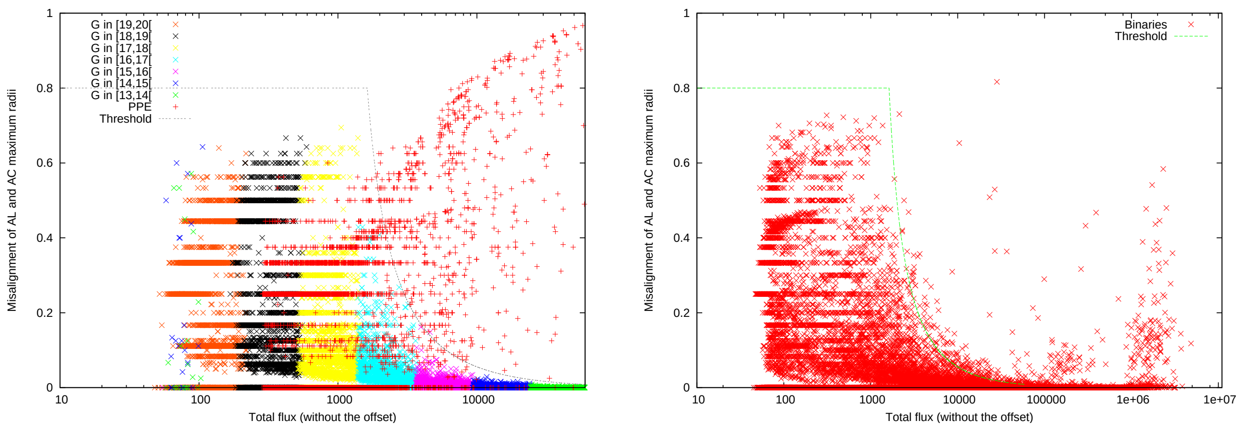


Figure 7.6: Discrimination of PPEs based on the misalignment of AL and AC maximum radii. The plotted threshold is: $y = 0.8$ if $flux < 1626$ ADUs and $y = 500/(flux - 1000)$ otherwise.

Scientifically speaking, the first category is of considerable importance because DMSs are one of the rare cases in which the mass of the stars can be derived, because resolved stars in extended sources provide an extra-galactic distance ladder and because targets in dense regions allow mapping the dynamics of the Galaxy where it is most complex, as near the galactic centre in the vicinity of the black hole. In all these cases, obtaining an individual view of components is mandatory to ensure that a complete set of data is acquired[4] and transmitted to ground in a manner coherent with the needs of the data reduction (joint profile fitting).

Conversely, the second category calls for separating the components since, compound objects being necessarily brighter than each of their components, magnitude would be systematically over-estimated. With Spec. 2 and 3 being relative to individual stars, this would introduce a selection bias or would cause errors in the activation of gates. Finally, the degraded characterisation of objects (type and precise meaning attached to location) would impede confirmation in AF1[5], degrade the reliability of the attitude control loop and make data packets more variable in content and hence the subsequent compression less effective.

Working with an atomic object model is similar, in principle, to introducing basis vectors in a vector space in the sense that it can be used to generate all possible combinations. To form data packets coherent with the scientific exploitation on ground, it is natural to match the model with the underlying physical sources. Objects following this model then come in fewer types which are both easier to describe on board and can be assembled to form complex observation regions with the pre-defined windows.

7.3.2 Segmentation paradigm

Segmentation versus "deblending"

Decomposition of DMSs into components is challenging both as regards the task itself and the constraints which apply. The finite angular size of samples and the real-time deadlines imply, from the start, that not all DMSs can be decomposed into components, irrespective of separation, orientation, colour and magnitude difference. With proper windowing in mind, unresolved DMSs or very close components can remain unnoticed on board since they will nevertheless be observed correctly. Conversely, identification is mandatory when the risk exists that the secondary be only incompletely covered by a window allocated for a compound objects dominated by the primary.

As far as implementation is concerned, object decomposition is clearly a worst case complexity issue because it is when stellar density is high and the objects extend over large regions (bright stars) that DMSs are the more likely. To make things worse, the processing cost is then further increased by the large number of member samples.

These two difficulties combined call for relying on segmentation rather than on "deblending". Segmentation splits a given domain (object) into a partition of sub-domains (components). It is a geometric decomposition obtained by forming sample classes and assigning every sample of the initial domain to a single class. Conversely, "deblending" identifies components by separating their respective contributions. Whatever the method used, it consists in estimating the different sources' contributions to each sample (as a term in the sum of fluxes) and produces objects which overlap in the image. The two concepts merge in the limit case where "deblending" assigns each sample to its dominant contributor: this yields a partition in regions of influence which may also be obtained as a result of segmentation.

[4]That objects are fully covered by the windows of pre-defined size that are commanded for read-out.

[5]Especially since PPEs, because of their linear geometries, tend to connect objects.

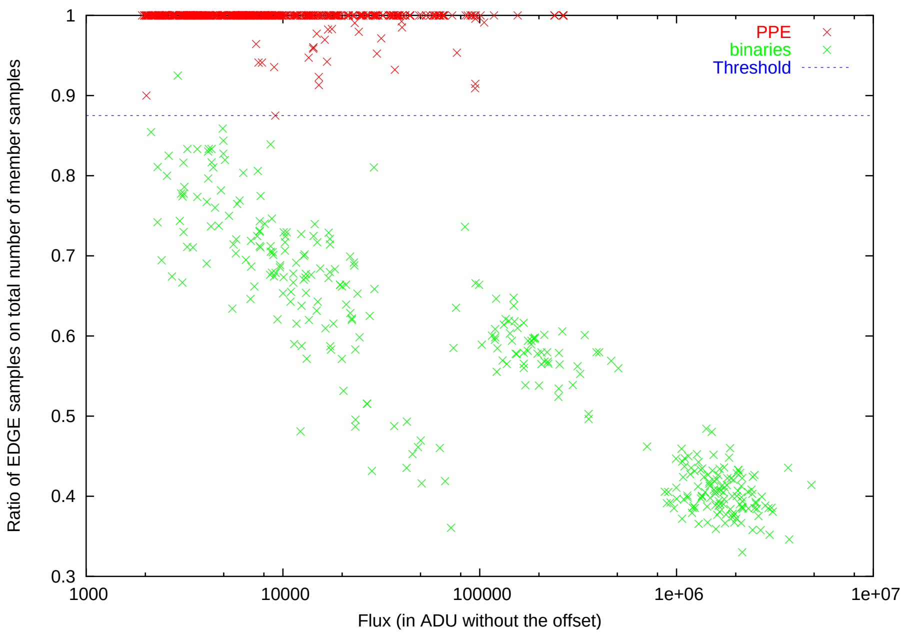


Figure 7.7: Further discrimination of PPEs and DMSs for objects exceeding the threshold defined in Fig. 7.6.

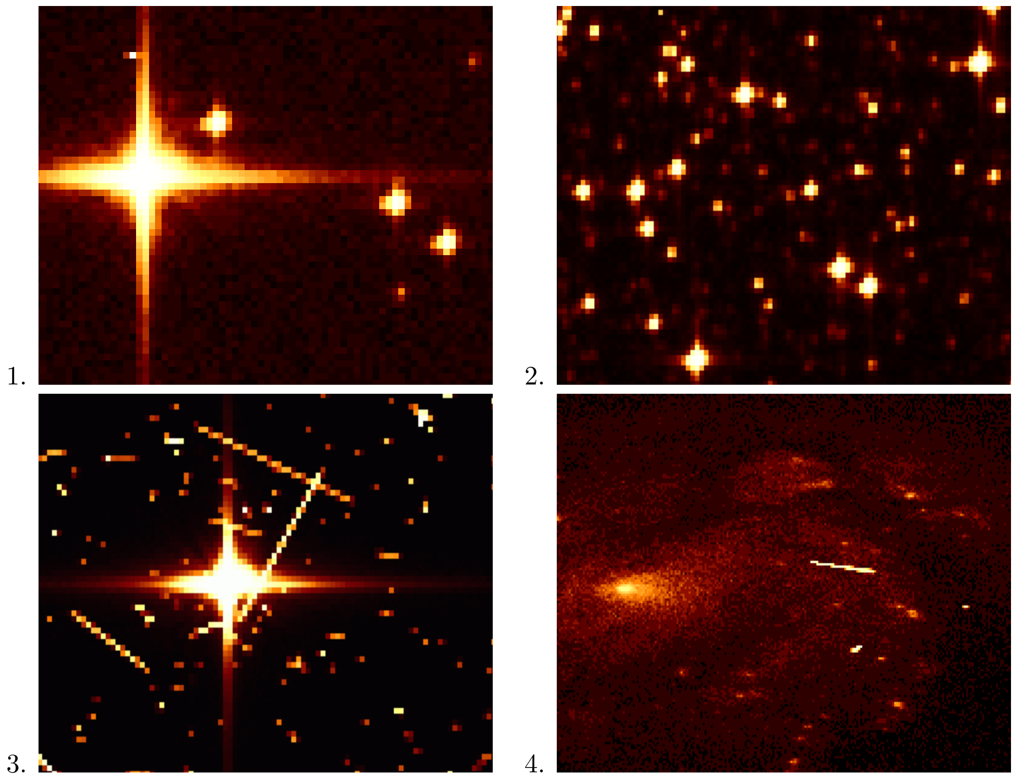


Figure 7.8: Cases of interest: 1. DMS, 2. dense region, 3. PPEs, 4. extended source (M100) as imaged in SM1 or SM2.

Segmentation is simpler because it is a geometrical concept rather than an algebraic one, the counterpart being that the resulting domains constitute only an approximation which neglects the contributions from the other components. Biases are, accordingly, introduced because, on one hand, the source's support is truncated and, on the other hand, because contributions from neighbours contaminate the measurements.

Fine object model (FOM)

In order to refine the SOM, some additional knowledge concerning the objects of interest must be introduced. We here further assume that the PSF features a global maximum at a level significantly higher than the rebounds. Since, as a result of the phase A studies [10] it was shown that obtaining the required astrometric accuracy depended on the ability to concentrate a significant fraction of the energy within the first lobe, this is not an unreasonable hypothesis. It allows for establishing a correspondence between local maxima and components of DMSs, provided inter-maxima relationships are examined to discard rebounds (section 7.3.3). Undetected components then occur when a local maximum is missing, generally because the gradient due to the primary is sufficiently important to absorb the extremum of the secondary.

The maxima are the basis for the construction of the partition through a region-growing scheme starting and expanding from them, as from the sample where the greatest amount of information relative to the underlying source lies. Following this rationale and considering the image as a topographical surface, the limits of our regions of influence are then the “valley lines”. With the presence of rebounds, the situation is not quite this simple but, filtering the corresponding maxima and hence automatically merging the associated domains with those retained, this intuitive representation remains valid.

Structure of the algorithm

The approach selected by both APM and Sextractor, based on iterative thresholding for separating components, is manifestly inadequate on board Gaia for complexity reasons. Instead, the need to identify “valley lines” suggests relying on the watershed transform. As a result, the algorithm proposed has an overall complexity which increases only linearly with the number of member samples and produces an exact partition according to the criteria above while allowing a fine control over over-segmentation.

The corresponding segmentation algorithm is a five step process, involving 4 passes[6] through the list of the object-member samples (Fig. 7.9):

1. Pre-filtering of SOM objects.
   In line with the general philosophy of minimising the average complexity, it is desirable to restrict attempts at segmentation to objects most likely of being DMSs. As a first simplification[7], taking into account the strict windowing objective mentioned above, the SOM may be maintained for objects whose bounding box fits into the faint stars’ standard window[8]. Although the amount of processing for each of these would be low because of the limited number of member samples, they represent the vast majority of cases so that, although the worst case complexity remains unchanged, the average one may be significantly reduced by making segmentation an infrequent event.

2. Identification of local maxima.
   A search for local maxima is performed among the member-samples to generate a set of markers. This search is complex because, due to quantification at the ADC level, sub-sample position of the sources, noise and saturation, extended maxima need to be identified. A label is assigned to each maximum.

3. Filtering of markers.
   The inter-markers relationships are studied to identify and discard the local maxima related to rebounds and noise. This is the key step to adjust the segmentation to the scientific needs and avoid over-segmentation.

4. Modified watershed transform and construction of resulting objects.
   The labels are propagated to neighbouring samples to form the partition according to a modified watershed algorithm. Simultaneously, the FOM objects are formed.

7.3.3 Markers

We restrict the discussion in what follows to our usage of the watershed transform. Concerning the transform itself, the interested reader may turn to [44] for the classical presentation based on the topographic analogy, to [43] and [5] for a refined algorithmic formulation with hierarchical queues or to [6] for a low-level, highly efficient implementation

[6]One pass for identifying candidate maxima, a second for labelling them, a third for discarding plateaus and a fourth for the watershed transform together with the formation of objects.

[7]Other criteria include objects already marked as pertaining to DMS, either because they already correspond to the FOM or because of the descriptors hitherto constructed (eg. criterion 7.12).

[8]This choice is also valid as regards the others reasons for advocating component segmentation since these objects are too faint to be used by the AOCS and do not require the activation of gates.

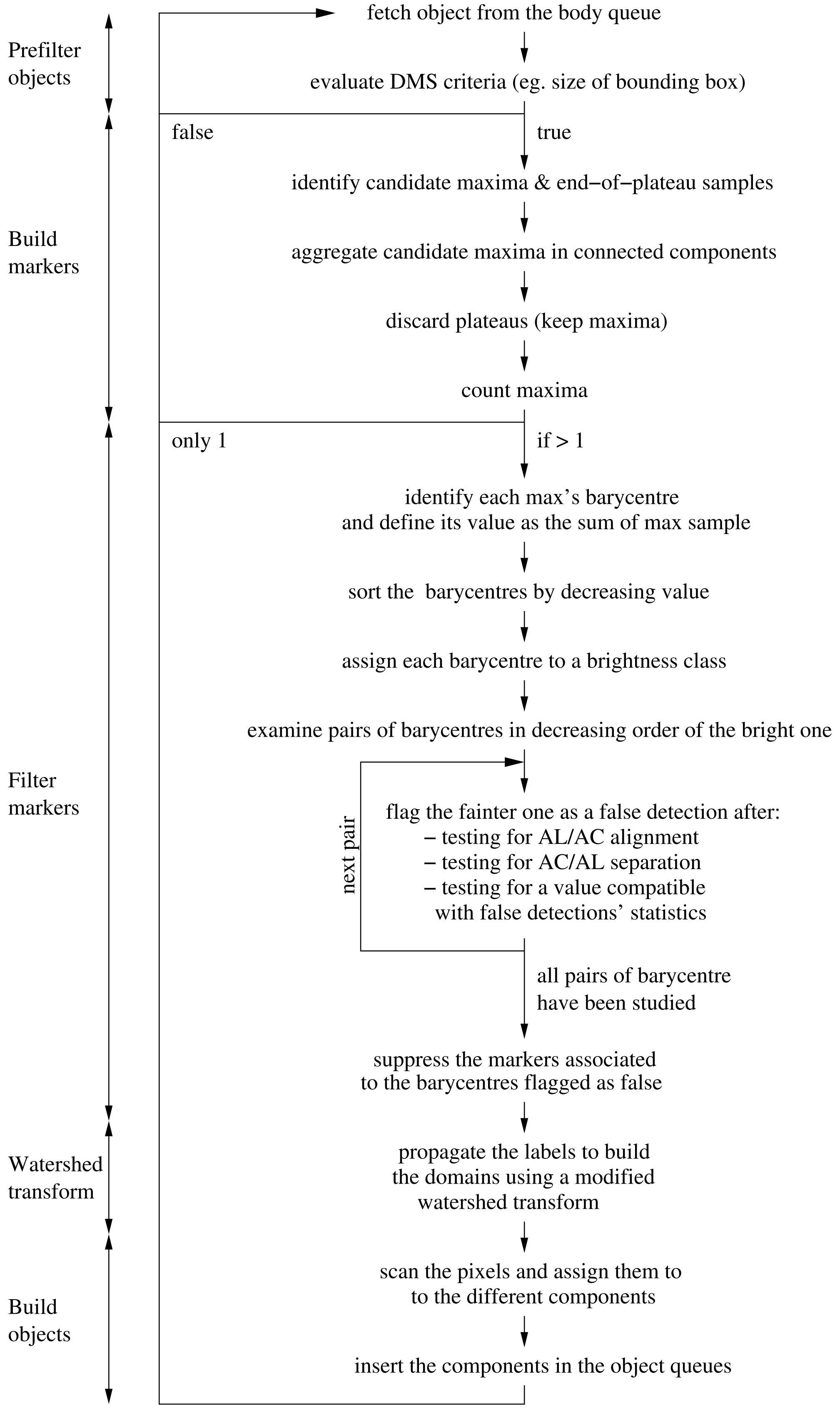

Prefilter
objects
Build
markers
Filter
markers
Watershed
transform
Build
objects
fetch object from the body queue
evaluate DMS criteria (eg. size of bounding box)
false
true
identify candidate maxima & end−of−plateau samples
aggregate candidate maxima in connected components
discard plateaus (keep maxima)
count maxima
only 1
if > 1
identify each max's barycentre
and define its value as the sum of max sample
sort the barycentres by decreasing value
assign each barycentre to a brightness class
examine pairs of barycentres in decreasing order of the bright one
next pair
flag the fainter one as a false detection after:
– testing for AL/AC alignment
– testing for AC/AL separation
– testing for a value compatible
with false detections' statistics
all pairs of barycentre
have been studied
suppress the markers associated
to the barycentres flagged as false
propagate the labels to build
the domains using a modified
watershed transform
scan the pixels and assign them to
to the different components
insert the components in the object queues

Figure 7.9: Structure of the segmentation algorithm. For clarity's sake the watershed transform and the construction of the FOM objects are described as two separate steps although they are carried out in a single pass.

in hardware. It is sufficient to say here that the transform is a label propagation algorithm, from the set of markers (usually local minima) to the neighbouring samples, which functions as sources of water would fill catchment basins and merge along the watershed lines which separate them. As such the algorithm is often used on the gradient image to delineate regions along high gradient samples under the assumption that they correspond to edges in the original image. The formulation used in our case is modified in the sense that it deals with the dual problem, that is, local maxima and valley lines rather than minima and watershed lines, and restricts the growth of regions to the object-member samples[9].

Local extended maxima

The local maxima on which the FOM rests need to be identified as a prerequisite to the transform itself. The main difficulty is that extended maxima must be considered as the rule. These correspond to a set of connected samples above all neighbouring ones and, hence, are not a purely a local property. The classic approach comparing each sample's value to its neighbours' is not applicable because the "$>$" test discards all extended maxima, while the "$\geq$" test also preserves plateaus[10].

The idea underlying this search is straightforward and springs from acknowledging that the local criterion ("$\geq$") retains both plateaus and maxima. A second step is then introduced to discriminate plateaus. This can be performed very simply if the criterion above is split in two ("$=$" and "$>$") to also mark "end-of-plateau" samples. Scanning those samples then suffices to discard the plateau they are part of. This final aspects is based on grouping the samples part of the same plateau through CC labelling. Fig. 7.10 illustrates this process in the frame of 8-connectivity ($i \in \{1, \ldots, 8\}$), and the algorithm then runs:

1. Scan object-member samples to identify those belonging to candidate maxima:

$$\forall i \; value(\text{sample}) \geq value(\text{neighbour}_i)$$

   and the "end-of-plateau" samples:

$$\exists k \; value(\text{sample}) = value(\text{neighbour}_k) \text{ and } \exists j \; value(\text{sample}) < value(\text{neighbour}_j)$$

2. Structure the set of samples belonging to each candidate maxima (maxima and plateaus) by CC labelling.

3. Scan the list of "end-of-plateau" samples to test their respective neighbours' labels:

$$\exists i \; label(\text{neighbour}_i) \neq DISCARDED \text{ (is there a neighbour part of a candidate maximum ?)}$$

   and values:

$$value(\text{sample}) = value(\text{neighbour}_i) \text{ (is it part of the plateau to discard ?)}$$

   to discard the associated CC.

Filtering the markers

Experience shows that over-segmentation is related the irregularities of the PSF, rebounds and photon noise, which generate spurious maxima. Because they relate directly to the PSF, a simple approach for filtering them out is based on the relationships existing between them. The thresholds used are then naturally PSF-dependent and built by examination of the PSFs per magnitude bin. Simulating a large number of single isolated objects for each bin with random colours and sub-sample positions, statistics are built to characterise the spurious maxima.

One difficulty comes from the a priori unknown number of sources underlying the SOM but as, according to the FOM, a correspondence can be established between the maxima and the sources, the maxima may be inspected by pairs. Each secondary maximum is confronted to the potential false maxima associated to a confirmed primary maximum. This allows for rejecting unwanted maxima which do not result from the combination of several PSFs. Other cases, like the intersection of spikes, come in small numbers by construction.

To some extent the sequence of tests carried out on the pairs can be seen as deriving from a subtractive philosophy schematically consisting in iteratively subtracting the PSF of the brightest source then examining residuals. This method is not applicable in our case for three reasons, each sufficient on its own. The first is relevant to selecting an appropriate template for cancelling out the primary because chromatic information will not become available until the

[9] More precisely, propagates the labels geodesically within the CCs.
[10] $N \geq 2$ connected samples of equal value but among which at least one has a neighbour of greater value ("end-of-plateau sample").

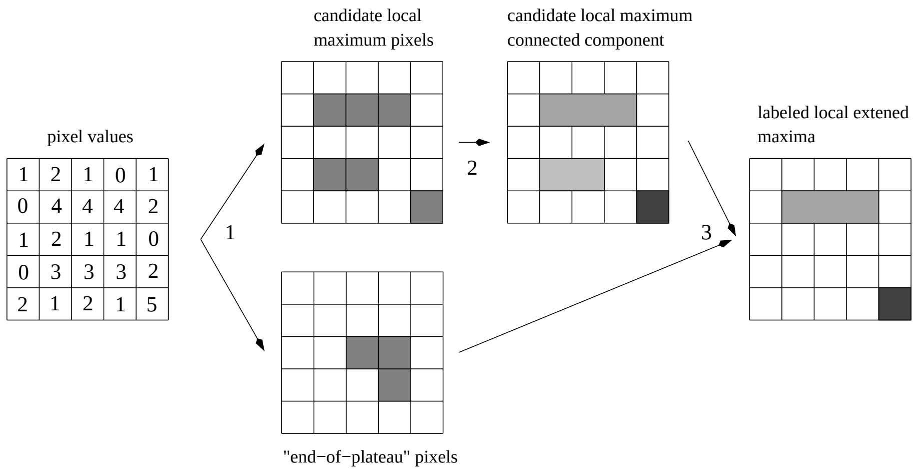

candidate local maximum pixels
candidate local maximum connected component
pixel values
labeled local extened maxima
1
2
3
"end−of−plateau" pixels
1 2 1 0 1
0 4 4 4 2
1 2 1 1 0
0 3 3 3 2
2 1 2 1 5

Figure 7.10: Connected-component-based extended maxima search (8-connectivity). Candidate local maxima are samples whose values are greater than their neighbours' in the sense of "$\geq$" while "end-of-plateau" samples are samples having at least one neighbour of equal value and one of greater value.

object is observed in the spectro-photometers (BP and RP). The second regards the complexity of reliably subtracting this template since the magnitude and the sub-sample location of the primary are difficult to estimate without knowing how many components contribute to the observed signal. Finally, sophisticated schemes are incompatible with our complexity objectives since the entire set of object-member samples must be considered at each iteration.

These considerations nevertheless guide the construction of the statistics relative to the maxima in our approach. As the thresholds need to be robust versus all these effects, they correspond to the bounds of confidence intervals taking into account all these sources of variability. From a computing point of view, considering only the maxima, instead of all member samples, significantly reduces the complexity since the iterations only involve testing them one against another. In an effort to further optimise the method and reduce the number of pairs to be considered, the cross shape of the PSF is taken into account to only compare pairs that are aligned in the AL or AC directions. This simplification only proves invalid for stars brighter than the limit ($G = 6$), which should be discarded earlier since they are not in the range of interest (Fig. 7.11).

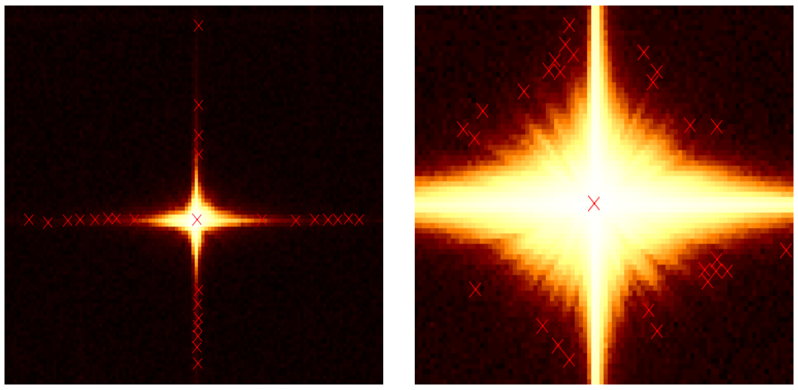


Figure 7.11: Spurious maxima on spikes ($G = 10$) or between spikes ($G = 5.8$).

The calibration of the method, based on a large number of synthetic simulations before launch or based on dedicated data acquired during the mission, consists in recording the relative AL and AC positions and the total flux of spurious maxima for the complete distribution of possible sources. Under the assumption that the primary maximum corresponds to a genuine source, the test then consists in evaluating the secondary maxima versus a threshold which is a function of separation and of the brightness of the primary. To simplify the test and reduce the number of thresholds, brightness classes are formed to model the dependency of the parameters on the source's brightness. As illustrated in Fig. 7.12, the distribution of spurious secondary maxima aligned is studied, then piece-wise linear envelopes are built in both the AC and AL directions to define the functional providing the threshold value for each separation.

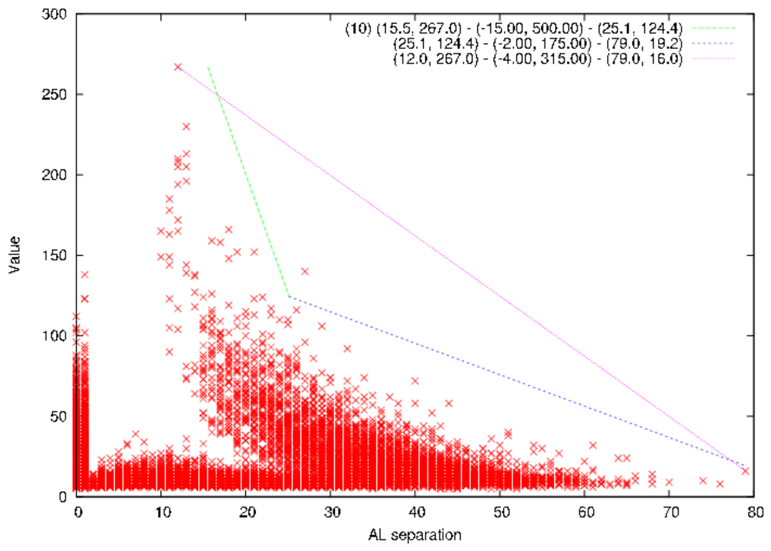


Figure 7.12: Spurious maxima and associated envelopes in the AL direction (in samples) for sources in the range $G \in [10, 11[$. The key provides the coordinates of control points (A, B) and the parameters in the piece-wise linear equation connecting A and B ($y = a.x + b$) in the following format: $(x_A, y_A) - (a, b) - (x_B, y_B)$.

7.3.4 Results

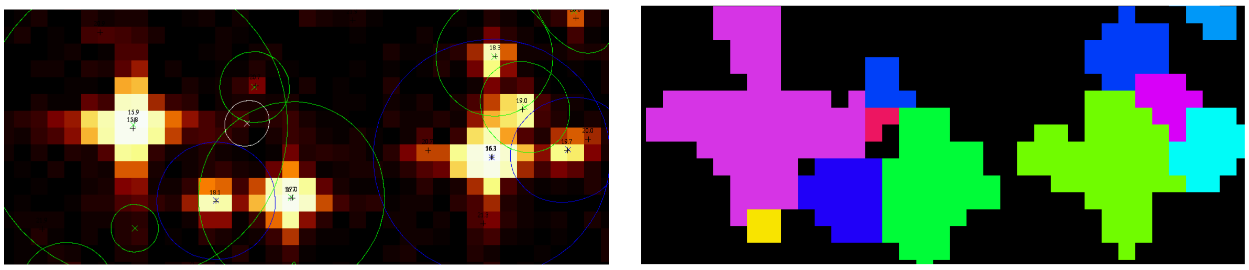


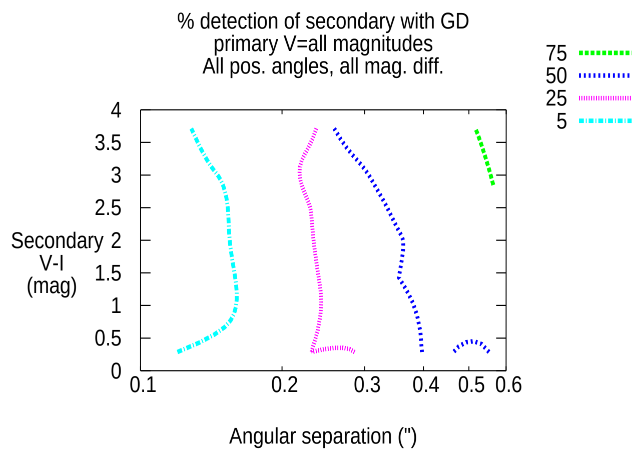
Figure 7.13: Two compound objects and the domains associated to the FOM.

For performance reasons the implementation of the modified watershed transform follows the approach used by Klein et al. [6] in hardware. The main feature of this implementation is to altogether avoid dynamic allocation and, instead perform dynamic linking of statically allocated blocks. While this is mandatory for hardware, it also proves relevant for space-borne software since intervention of the OS should be minimised. Additionally, with simple data structures and predictable accesses, compiler optimisations are also notably more efficient.

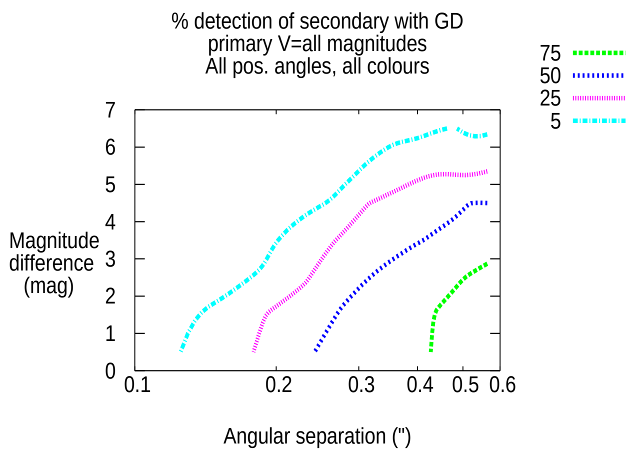
Detection

Assessing the probability of detecting the presence of a companion star with the proposed method has been performed by simulation for a wide range of configurations which jointly sample:

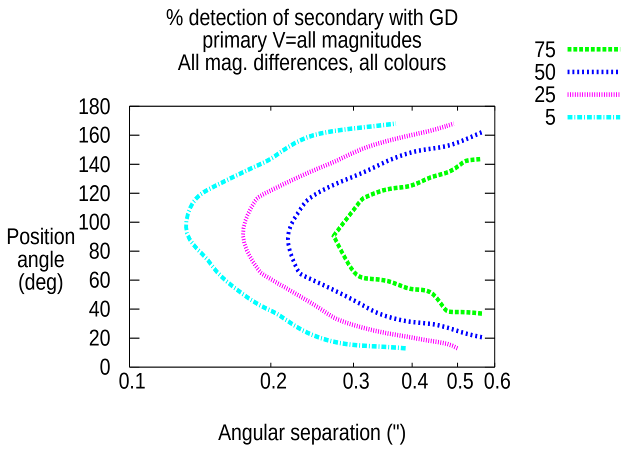
- $V \in [9, 21]$ for the magnitude of the primary,
- $V - I \in [0, 2]$ for the colour of the primary,
- $V - I \in [0, 4]$ for the colour of the secondary,
- separations in the $[1, 18]$ sample range,
- orientations in the $[0, \pi]$ angular range (with the AC axis as reference),
- and magnitude differences in the $[0, 7]$ V range.



1.



2.



3.

Figure 7.14: Detection probabilities for the secondary as a function of colour (1), magnitude difference (2) and orientation (3, with angles measured based on the AC direction).

A total of 15 600 pairs of stars over 25 different simulations have been produced with Gibis under identical conditions for this purpose. As the statistics gathered are hardly sufficient for properly measuring marginal frequency distributions in spite of the large data set, only the overall ones are presented in Fig. 7.14.

The general trends concerning the detection of the secondary are then :

1. It only weakly depends on its $V - I$ colour (7.14 1.).
   This results from using an object model which is not template-based. Instead, the additional assumptions introduced with the FOM are shared by all stars notwithstanding colour differences[11]. Variability with colour, beyond random effects, can be attributed to the fact that the correspondence between V and G magnitudes also depends on the $V - I$ colour index (because of the dependency of the quantum efficiency on wavelength).

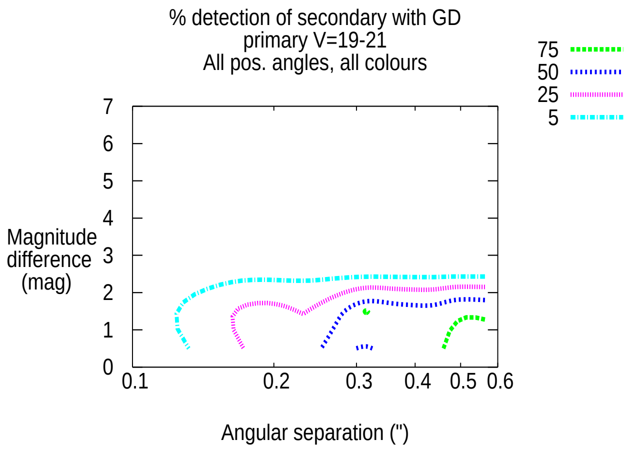
2. It decreases with the magnitude difference (7.14 2.).
   The detection of a companion star crucially depends on being able to identify the associated local maximum. At a given separation, the brighter the secondary, the easier the task. For fainter ones, the additional signal is either absorbed by the strong gradient on the PSF of the primary or drowned in the associated noise, most notably the Poisson noise whose relative importance increases with the magnitude difference[12].

3. The influence of angular separation is not isotropic (7.14 3.).
   The relative orientation of the multiple system versus the AL and AC directions can impacts on the corresponding detection probabilities in two different ways. First and foremost, the different angular resolutions along the two

[11]They are even shared by all sources imaged through the optical system.

[12]Additionally, once detected the secondary must be preserved by the object-wise SNR test (section 7.2.1). This leads to a strong censorship when the secondary approaches the detection limit (as illustrated Fig. 7.15) because of the efforts made to restrict the detections to the magnitude range of sources of interest.

axes favours detection along the AL axis as illustrated in Fig. 7.14 3. One might also think that probabilities would increase between the spikes where the primary's signal is globally lower. If that were indeed the case, restricting the orientation plot to the most sensitive pairs (at high magnitude differences) should show high detection probability regions, around 45° and 135° typically, but this is only marginally the case. Fig. 7.15 demonstrates that the PSF's gradient and the Poisson noise equally hinder the detection of companion stars on the spikes as elsewhere and proves that filtering the markers along the spikes does not significantly depletes detections in these regions.



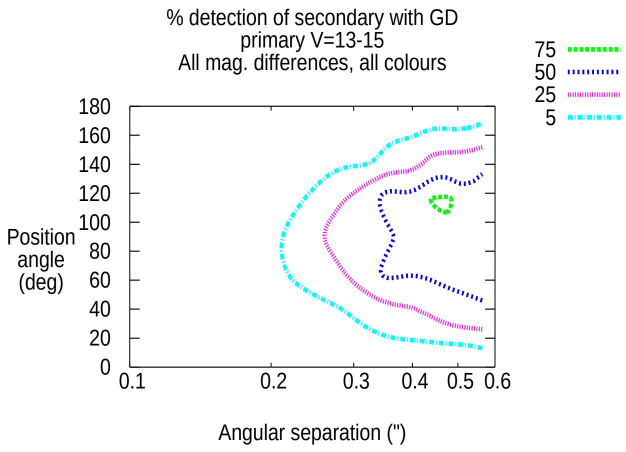


Figure 7.15: Detection probabilities for the secondary at the detection limit and on the spikes at large magnitude differences ($\delta V > 3$).

Measurements

With the FOM, it is not only possible to detect companion stars but also to measure them individually. Although biases are expected due to pollution of the other nearby sources (section 7.3.2), location and magnitude estimates can be produced. Dedicated estimators could be designed to be robust to this pollution, but the standard ones allow for comparing the reliability of the objects produced by segmentation to those associated to isolated stars with Req. 3 in mind. Fig. 7.16 represents the standard deviations of the measurement errors as an indication of the precision achieved[13].

As expected the quality of measurements degrades mostly close to the detection limit where the intermingling of the sources is maximum. This applies to both the primary and the secondary, although naturally, the effect is greater for the latter because it is fainter (this effect is difficult to evaluate precisely because detection probabilities are low for these). Away from the limit, performances improve and compare to those obtained for isolated stars (section 7.4.2).

Windowing

We illustrate in Fig. 7.17 the windowing of DMSs in the AF2 CCD produced by Pyxis as a result of component segmentation. This figure illustrates how the FOM is a key design element for selecting objects and programming their read-out[14] through three different cases of figure:

- cases when undetected companions are close enough to the primary to allow proper imaging with a single normal window,
- cases when detected secondaries lead to the allocation of additional normal windows,
- cases when due to the low separation smaller tiling windows are used to cover the system as a whole.

[13]Only separation / magnitude difference diagrams are presented because, as shown by Fig. 7.14, these are the most discriminating parameters.
[14]"Selection" task in section 2.2.1.

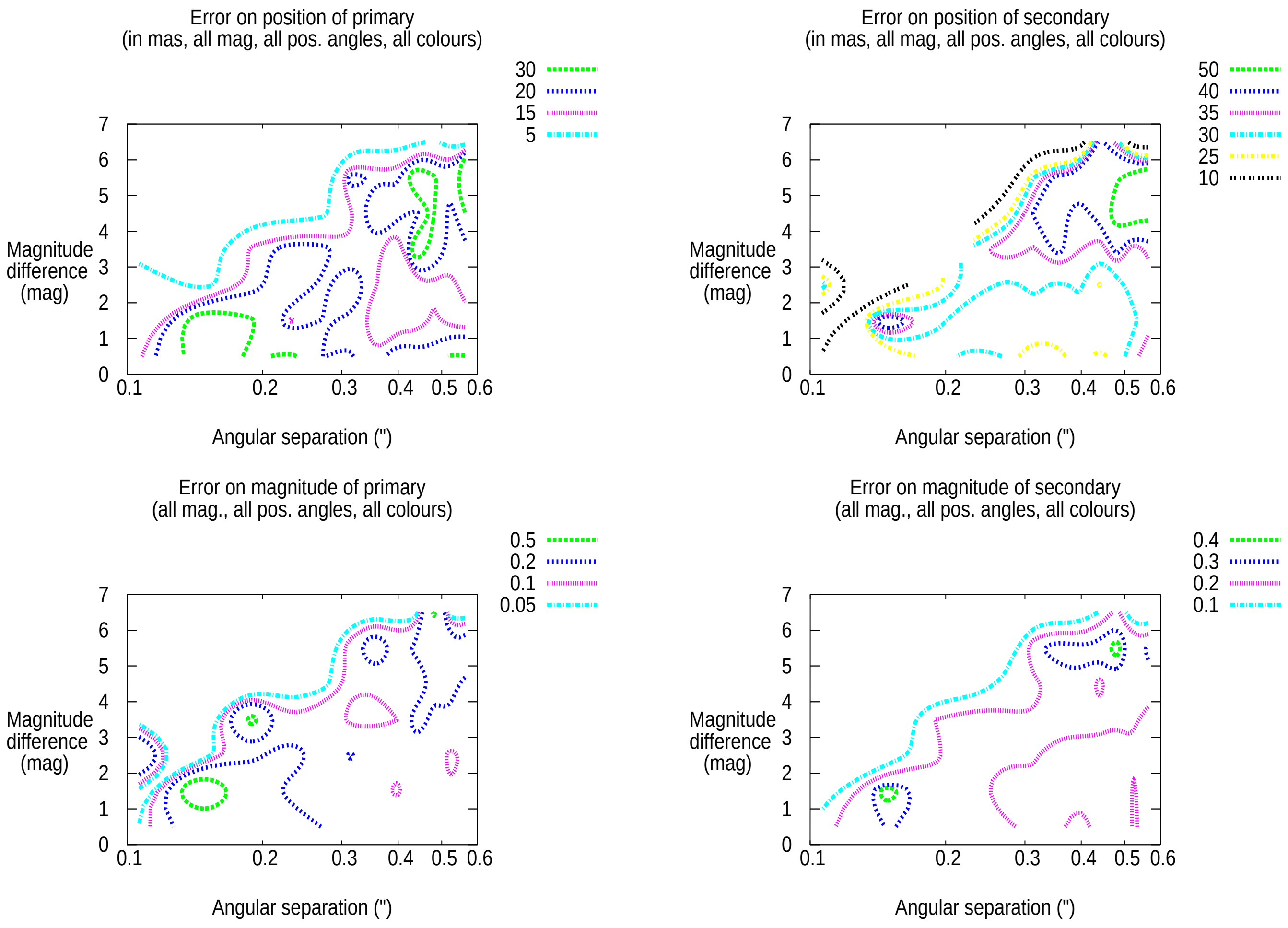


Figure 7.16: Standard deviations of the error on position and magnitude estimates for the primary (left) and secondary (right) components. The lowest contour lines are only relevant to indicate the limits of the domain in which pairs have been successfully detected.

7.4 Measurements

7.4.1 FOM descriptors

As per Spec. 8, object properties need to be measured for the purposes of tracking them down the focal plane, of selection, of attitude control and of data management on board but also of "Initial Data Treatment" on ground (cross-matching, ingestion in the database, calibration etc.). With the decision to let selection and on-board data priority depend only on magnitude[15], only intrinsic measurements are necessary, meaning that the FOM is sufficient as opposed to a combination of the two object models. Nevertheless, the descriptors inherited from CC-labelling (section 6.5) form a basis for the final measurements.

Following the nomenclature of Spec. 8 and with items 1 and 2 under the responsibility of the PDHU when collecting data from the CCDs, detection is only involved in those from 3 to 9. Tab. 7.1 summarises how these needs are met.

3. Positional information
The coarse *bounding_box* information from section 6.5 needs to be refined to the sub-pixel level for tracking objects and the AOCS control loop. Following the logic hitherto adopted, the location is estimated with as limited a priori information as possible and the AL and AC barycentres of the member samples (centroid) are

[15]As opposed to more complex schemes, for example also taking into account the FOV density (to balance FOVs in all cases for attitude reconstruction purposes) or the existence of a companion object (for joint reduction) or the instantaneous PPE fluence (for maximising the usability of downloaded data).

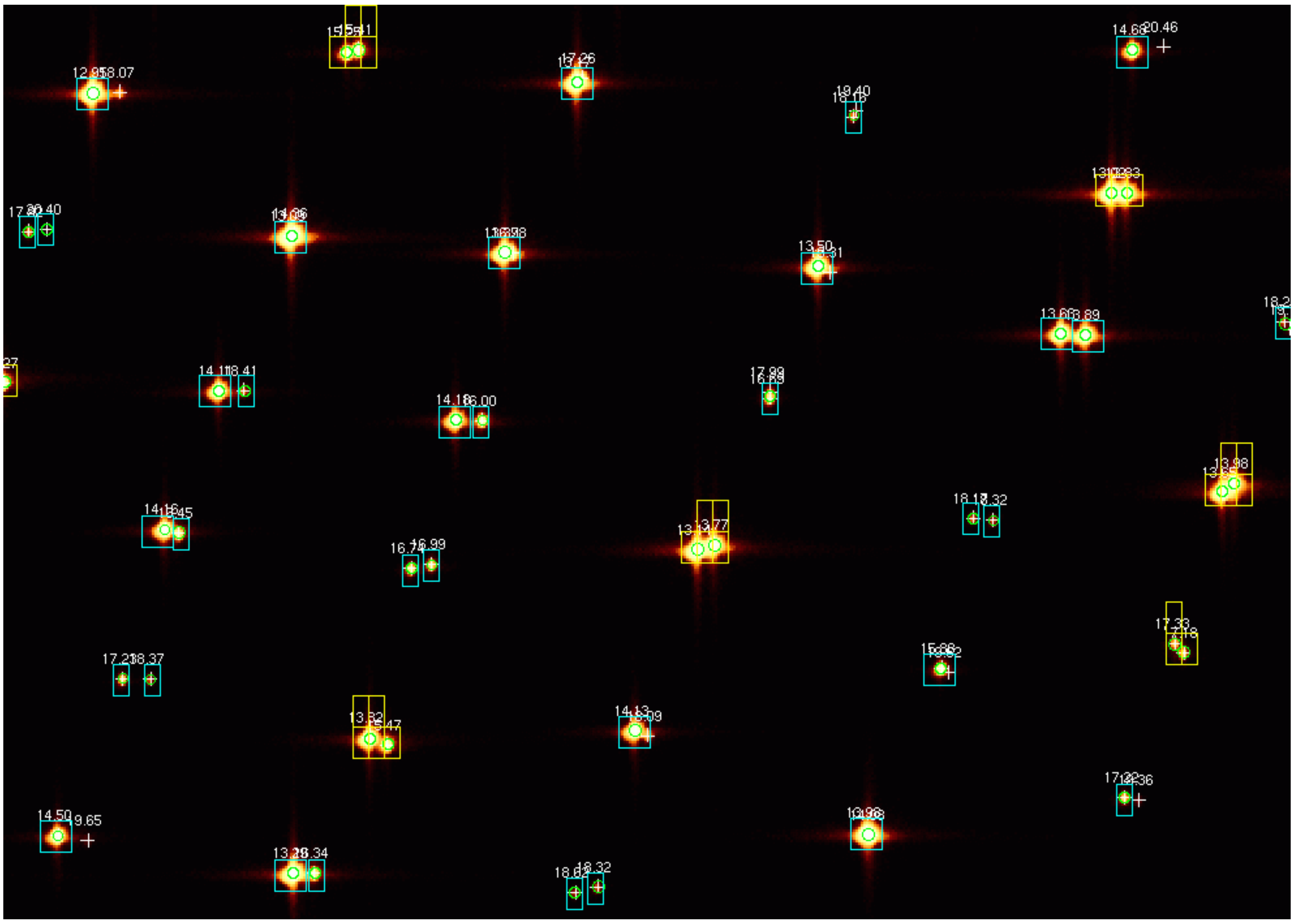


Figure 7.17: Windowing in AF2 for part of one of the test images (white crosses indicate simulated sources and their magnitude, green circles confirmed ones, blue rectangles are normal windows (6 × 12 or 12 × 12 depending on magnitude) and yellow ones indicate tiling windows allocated specifically for covering DMS).

	Category	Measurements
1	object samples	sample data read-out from the AF, BP, RP and RVS CCDs
2	time information	time-tagging of the sample data read-out
3	position information	AL & AC coordinates of the barycentre
4	object type	magnitude classes, DMS and *saturated* flags
5	special processing information	
6	background	sum of background estimates at the member sample locations
7	flux intensity	sum of values of member samples (with offset and background corrected)
8	field information	even identifiers for SM1 objects and odd ones for SM2
9	extended objects	

Table 7.1: Descriptors required to be sent to ground as per Spec. 8 and corresponding on-board measurements.

used[16]:

$$\frac{\sum_{(al,ac)\in object}((samp_{al,ac} - offset) \times (al - min\_al))}{\sum_{(al,ac)\in object}(samp_{al,ac} - offset)} + min\_al \rightarrow pos_{AL}^{obj} \tag{7.13}$$

$$\frac{\sum_{(al,ac)\in object}((samp_{al,ac} - offset) \times (ac - min\_ac))}{\sum_{(al,ac)\in object}(samp_{al,ac} - offset)} + min\_ac \rightarrow pos_{AC}^{obj} \tag{7.14}$$

[16]The formula below proceeds relatively to the object's bounding box to reduce the risk of overflow for very bright objects.

Additional bias correction terms are then introduced before the coordinates, expressed sample-wise up to this point, are converted to follow the general conventions applicable to CCD data [45].

6 & 7. Background & flux intensity
The background and object fluxes are available from the SOM (section 6.5) but need to be updated in the process of component segmentation if the object is found to be a compound one. Besides, with the purpose of using "flux intensity" to select a window geometry from the dictionary of possible ones for the subsequent observation of the object, to configure the gates or to manage the resulting data, this measure should be intrinsic and directly convertible to a magnitude. Accordingly, the background contribution is estimated then subtracted (which also removes the $offset$).

5 & 9. Special processing & extended objects
The requirement to report all special cases is a legitimate one from the point of view of monitoring the status of the instrument. In our case, however, thanks to the decision to rely on limited assumptions and the use generic object models, all objects are submitted to the same processing sequence and it has not proved necessary to introduce any special processing. Nevertheless, data relevant to house-keeping may be transmitted to ground: the existence of undetected objects (Req. 12), the number of objects rejected as PPEs (Req. 2), the number of detected objects (Req. 8) etc.

7.4.2 Performances

Performances at the FOM level correspond to those visible by the on-board processing chain (section 2.2.1) and should hence be compared to URD specifications or to Astrium's "Algorithm Performance Specification" [46] (and subsequent versions). As such, they correspond to the granularity of the testing to be carried out for validation (section 2.4.3). Accordingly, the definition of criteria and the elaboration of a database of representative test cases – necessarily incomplete but devised to be sufficient as regards the expected configurations and the intended use – have been addressed as part of the PDH-S support activity to Astrium. Guidelines for the criteria [SM7] and the necessary simulations [SM10] have been proposed and a test plan [47] issued by Astrium. Input files for use either with Gibis or with Astrium's FPASS have been provided as descriptions of realistic scenes on the sky or of synthetic configurations allowing to test one particular feature of the system with sufficient statistics.

Unfortunately, as of this writing, none of the corresponding simulations have yet been made available to us in return. Accordingly, as in section 5.3, the results presented in this section are based on the simulations produced in the frame of the PDHE TDA (appendix C). These simulations, being targeted at evaluating processing resource usage, are a long way from being a sufficient basis for scientific validation of performances in all cases of interest but nevertheless provide some insight on the behaviour of the system.

Fig. 7.18 collects the performances measured in realistic configurations at different densities. The figures are globally in line with the specifications listed in section 2.2.2 or the needs derived in section 3.3. Completeness is mainly affected by crowding and failure to fully resolve all DMSs in components. This leads to classifying a 14th magnitude star as undetected in the Baade case because it is a very close companion to a correctly detected brighter primary (it would be properly observed with the latter's window).

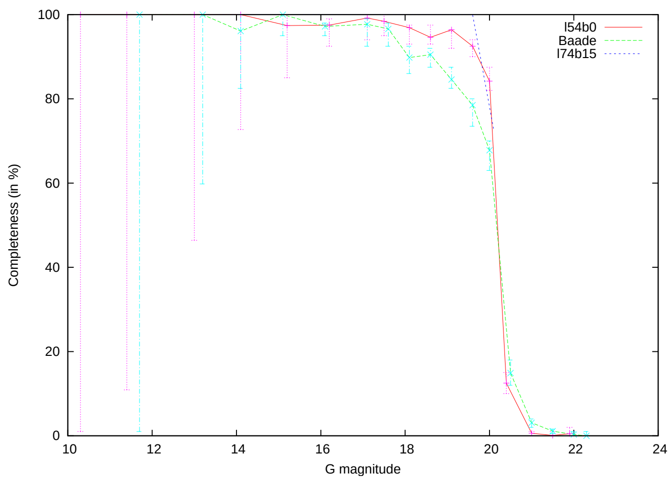
The single star data set (Tab. C.1) provides a synthetic test bench for evaluating performances on isolated stars. Fig. 7.19 presents the performance of location and magnitude estimators in this favourable configuration which corresponds to Req. 8. Ensuring that stars do not overlap makes it problematic to simulate a sufficient number of bright stars to collect good statistics, so only cases with $G \geq 12$ are shown. The completeness plot has been omitted in this figure, because it is above 99.6% in all magnitude classes except for $G \in [20; 21[$ where it drops down to 39.4% (as it should since these stars are outside the range of interest).

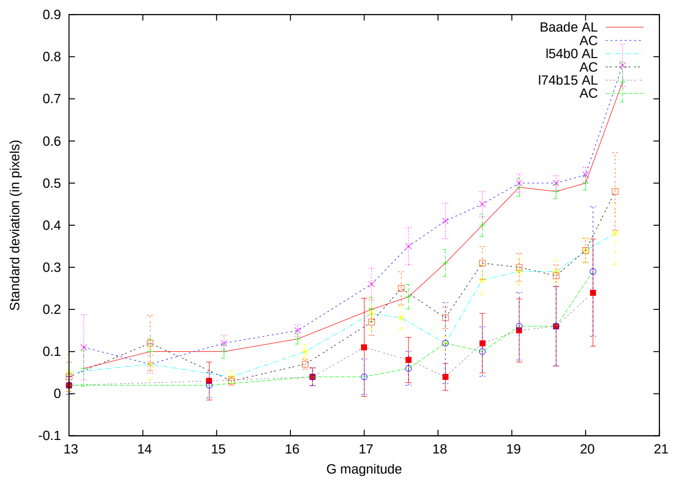
Considering type I errors now, the false detection rate is only meaningful under real sky conditions because they are the result of the combined effects of noise, of the background and of the crowding of objects. Tab. 7.2 presents the rates measured in the realistic simulations. These numbers also show that the most frequent PPE fluences are well handled by the system. To further illustrate the robustness to particles, a case with a fluence 12.5 times greater (100 particles/cm^2/s) is shown on Fig. 7.20, a level expected during approximately 9% of the time. The 14 objects are literally swamped by PPEs, but 100% detection is achieved with only 0.008 false detections per 1000 samples[17].

3.1.1

In terms of computing, although not flight-quality software, the ANSI C code corresponding to the FOM processing has been evaluated as part of Pyxis during the PDHE TDA. Fig. 7.21 presents the corresponding results which are well within the capabilities of a platform like Maxwell's SCS750 (section 2.2.3).

[17]Robustness to PPE fluences allowing uninterrupted operation during 100% of the time would require accommodating as many as 10000 particles/cm^2/s.





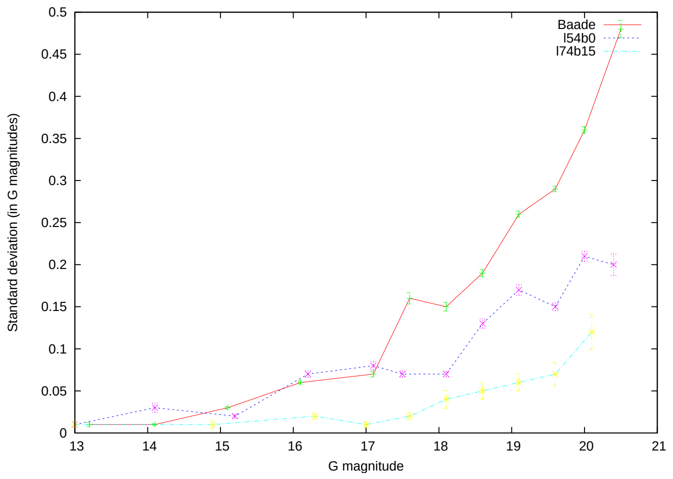
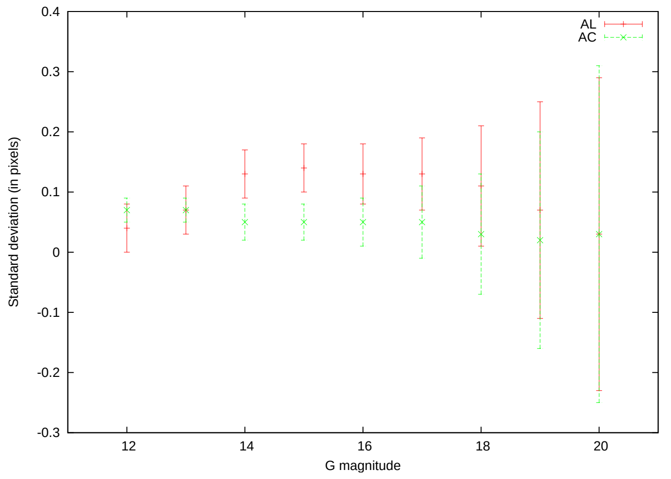


Figure 7.18: Completeness and measurement quality (with dispersions) as measured in representative test cases at different densities (appendix C).



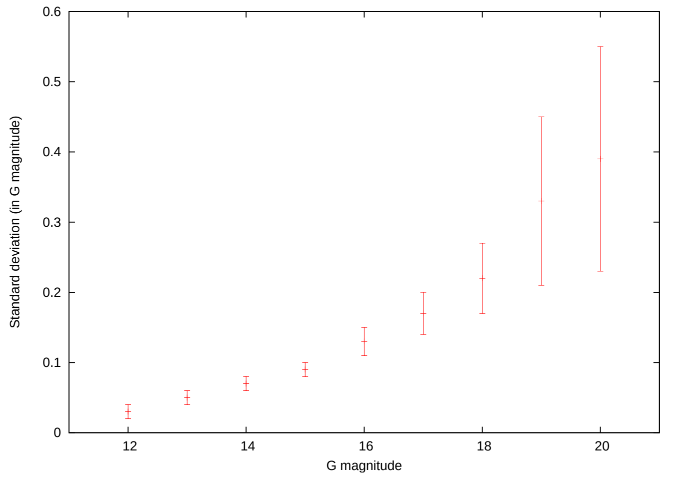


Figure 7.19: Quality of measurements (with dispersions) as measured in synthetic test cases featuring isolated stars aligned on a grid with random sub-pixel locations (appendix C).

Density (stars/deg^2)	Test case	False detection rate
3 310 000	Baade	0.889
609 000	l54b0	0.092
195 000	l74b0	0.019
22 250	l74b15	0.002
3 500	l74b74	0.005

Table 7.2: False detection rate as a function of density (in units of 1 false detection per 10000 samples). The rate increases with density both because of the margin required to obtain a complete survey down to magnitude 20 (because of the Poisson noise which leads to retaining undesirable faint stars) and because of crowding.

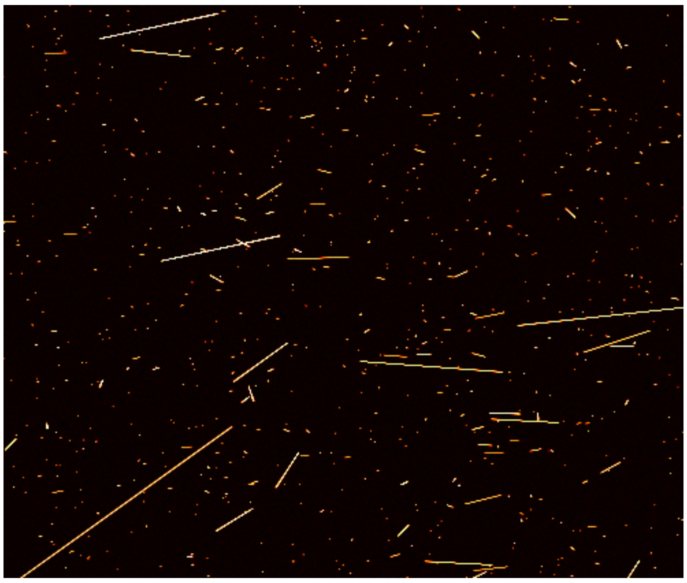


Figure 7.20: An example of FOV dominated by PPEs with a fluence of 100 particles/cm^2/s. As no stars are visible on this snapshot, all objects are discarded as PPEs.

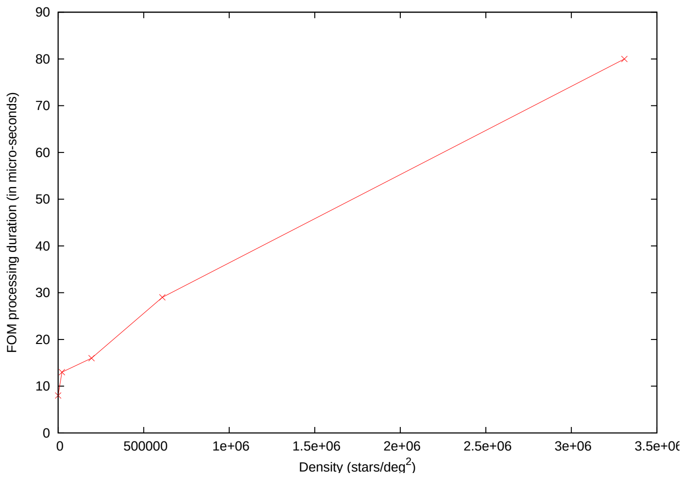


Figure 7.21: Average duration of the processing dedicated to the FOM per TDI as measured on the PDHE real-time platform (courtesy of Astrium GmbH [25]).

7.5 Conclusion

The refinements presented in this chapter build on the SOM to produce the outputs expected by the rest of the on-board processing chain. After targeting minimum false negatives with the SOM, the problem of false positives is addressed in this part of the processing. The solutions retained form a highly reliable mitigation scheme for both noise and PPE-related false detections. By focusing only objects of interest, this in turn offers an opportunity to make the most of the available resources and introduce more complex computations, in selected cases, to identify components and refine the object model (FOM).

Part III

Implementation

Chapter 8

Platform

Contents

8.1 Introduction

This part describes the migration of the algorithms drafted in the previous part to the "hardware world". Following the conclusions of the PDHE TDA, the aim has been to achieve a complete mixed architecture to consolidate the analyses of phase A, with emphasis on the hardware part as something particularly challenging and entirely new at the GEPI. Due to the complexity of such developments, hardware designs are intrinsically hierarchical. For clarity, the structure of this part follows the same logic. After discussing technology and QA criteria for the design of FM and test platforms in this chapter, it moves on to the high level internal architecture and interfaces of the FPGA (chapter 9). It then focuses on two main blocks, first the memory controller which ensures reliable communication with the external asynchronous SRAMs (chapter 10), then the core processing itself (chapter 12), with conversion to fixed-point arithmetics as a prerequisite (chapter 11). Performances can only be assessed globally because synthesis flattens the design to perform cross-component optimisations, hence only overall performances are meaningful. They are presented, as regards the allocated logic resources and timing in a concluding chapter (13) together with a list of shortcomings as compared to an FM.

Although the problem of writing flight-quality software is one obviously worthy of interest[1], it is only alluded to in this part. Several reasons concur to this, one is that we have preferred providing a detailed yet comprehensive account of developments for hardware within the briefness bounds of such a text. Another is that, after the end of the PDHE TDA, we have been fully absorbed in these developments at the expense of improving the software towards a more realistic real time implementation so that software-wise, most of the conclusions are already contained in Astrium GmbH's final report [32]. Finally, with the adopted partition of tasks between hardware and software for object detection, it would have been difficult to run this software without the support of the preceding hardware, which is only being fully integrated and tested at this point in time.

[1]If software offers an opportunity for descriptions at a somewhat higher level than in this part, its inherent flexibility and the increased difficulty in predicting the precise behaviour of the resulting system pose different problems. For instance, [48] emphasises the associated reliability concern which translates into a difficult verification stage, especially since situations cannot be tested exhaustively.

8.2 Candidate platforms

8.2.1 Digital devices

Digital devices exist in a variety of flavors covering a very broad range of use. Starting from the most elementary ones systematically performing a single operation, two main families of programmable devices can be identified depending on whether the behaviour or the structure are modifiable. The first is composed essentially of microprocessors and microcontrollers[2] which, although with an architecture entirely determined at the foundry stage (before the precise use intended for the part is known), are capable of altering their operation based on the interpretation of instructions. Conversely, components featuring a generic structure which can be configured to exactly match the need (hereafter programmable hardware) people the second. While Tab. 8.1 provides examples corresponding to this coarse classification, it is worth stressing that it is not so much structural as based on the use for the part – a microprocessor being an ASIC designed to work as CPU – and indicating the convergence to devices with programmable structure and behaviour with the advent of chips capable of partial dynamic reconfiguration.

Standard logic	Programmable behaviour	Programmable architecture short time-to-market	Programmable architecture long time-to-marker
7400 series TTL	microprocessor	PLD, CPLD	Structured ASIC
4000 series TTL	microcontroller	FPGA	Gate array ASIC
			Full-custom ASIC

Table 8.1: Classification of digital electronic devices (according to [49]). 'Time-to-market" corresponds to the interval separating the emergence of the concept and the capability to commercialise the parts. It provides a measure of the complexity of the design process, not only as concern development but including also extensive validation and production.

8.2.2 Programmable hardware

Families

The central result which ensures that any function can be implemented using programmable hardware is the existence of canonical forms in boolean algebra. With any logical expression convertible to a sum of products (disjunction of minterms) or a product of sums (a conjunction of maxterms), implementing two matrices, one of `AND` gates, the other of `OR` gates, suffices to guarantee the ability to evaluate any expression through proper wiring. Accordingly, the programming of the device consists precisely in the determination of the interconnection matrix. Since the rest of the devices is defined prior to foundry and profits equally from comparable manufacturing processes, the interconnection technology is the limiting factor and determines the potential for performance. Two categories can be distinguished depending on whether the interconnection is physically modified (metallization[3], fuse and anti-fuse[4]) or only reconfigured through intermediate memory cell (flash and SRAM) (Tab. 8.2).

This principle yields combinational cells based on which sequential behaviour can be obtained by connecting outputs to inputs and synchronous operation by inserting flip-flops. Except for the ASICs which correspond to a particular manufacturing process, the different types of programmable hardware then result from organising such cells in different architectures:

PLD. The connections of `AND` gates only are programmable and, depending on devices, different types of output cells (OLMCs) are possible: combinational (wire), sequential (flip-flop) or versatile (combinational or sequential). These devices are only rarely used for themselves nowadays but are the building block of CPLDs.

CPLD. For higher integration purposes, PLDs are integrated to form CPLDs. Their structure is classically based on cells corresponding to 16 PLDs connected through a Programmable Interconnect Array (PIA). The remarkable property of the latter is that a single connecting point exists for all possible connections, so that the propagation delays[5] are all equal. As a consequence, placement and routing simply concentrates on sharing resources.

[2]Microcontroller are similar to microprocessors except that they include elements that are external resources for microprocessors (eg. RAM, ROM, analog-to-digital converter or peripherals) to achieve higher integration for embedded applications.

[3]Connections between transistors by the insertion of 2 to 5 metal layers.

[4]The physical principle for the two is respectively based on destroying a fuse or an insulator by the application of voltage out of the nominal range.

[5]It is worth recalling that these delays are related to the interaction between the resistance and the capacitance at the connecting point, not to the physical distance between the cells.

	Metallization	Fuse	Anti-fuse	Flash EPROM	MOSSRAM
Programmable	1	1	1	10^5 to 10^6	∞
Volatile	no	no	no	no	yes
Density	highest	700 μm^2	1.8 μm^2	25 μm^2	50 μm^2
Power	low	low	low	medium	high
Radiation	rad-hard		rad-hard	no	rad-tolerant [50]
Speed	highest		high	medium	medium
Vendor	eg. IBM (ASIC)	obsolete	Actel	Actel	Xilinx

Table 8.2: Properties of interconnection technologies. The speed line does not relate to the operating frequency of the devices because it is determined by the system's clocks and, hence, does not characterise the part but rather the design running on it. Instead, it is relative to the propagation delays induced by the interconnection technology and sets an upper bound to reachable frequencies. (The radiation, speed and vendor fields are left blank because the information is unavailable due to the obsolescence of the fuse technology).

FPGA. FPGAs are built around simpler elementary cells than CPLDs for the benefit of proposing more together with flexible placement. As routing is not centralised in this case, delays vary depending on the number of connecting points separating the cells. Hence, with FPGAs, the trade-off is between resource utilisation and timing due to more constrained placement and slower signal propagation.

FPGAs

With the intent of complementing the general purpose processor with something offering maximum performance for minimum resources (section 3.5.1), it is only natural to turn to programmable hardware for fine-tuned processing. A solution based on an ASIC would be ideal since these components advertise the highest density at the lowest power consumption and thermal dissipation, however three project-related considerations make this an impracticable approach. First and foremost comes the cost of developing wafers and achieving high reliability foundry for producing only a few units. Were this not sufficient for rejecting this approach, the lack of flexibility in this process is problematic as regards risk management because modifications for debugging or tuning are error-prone and require repeating the entire validation procedure, which is both costly and time-consuming[6]. Of the two remaining possibilities, FPGAs is the preferred one mainly because they outperform CPLDs in density and hence allow for saving mass and volume.

For completeness' sake, it should be mentioned that, as an alternative, a solution relying on a second processor might be envisaged. With a sufficient amount of assembly language pragma directives for optimisation purposes and proper synchronisation between the processors it would likely be possible to fulfil the processing needs by this means, although at the expense of power consumption and thermal dissipation. Yet, notwithstanding the real-time frame imposed by the OS (most probably VxWorks), the time-wise resolution of the scheduling would likely remain insufficient for direct communication between the processors. Accordingly, intervention of the OS in the form of Symmetric MultiProcessing (SMP) would be necessary. Not only is this orthogonal to the policy consisting precisely in refraining from relying on the OS, but discussions with industry suggest than the tools are lacking for the development of such systems at the required level of reliability.

Selection and procurement of FPGAs for space applications are submitted to the procedures forming part of ECSS's standard for electrical, electronic and electromechanical components (ECSS-Q-60B [23] based on ESCC recommendations). The key criteria, beside the project's functional and performance requirements concern reliability, cost and availability. Applied to FPGAs, and with immunity to radiation in mind as a decisive parameter[7], this leads to selecting either an anti-fuse or an SRAM component.

With the amount of processing to be carried out by the FPGA, Actel's RTAX-S and Xilinx's Virtex II are preferred for their high number of system gates. Based on the data in Tab. 8.3, particularly error rates, it is clear that Actel's chips largely outperform those from Xilinx. This results from anti-fuse technology being intermediate between ASICs (one-time programmable, performances) and reprogrammable FPGAs (architecture, programmable). Conversely, Virtex II parts are widely used in consumer electronics and provide the benefits of conventional reprogrammable FPGAs:

[6]Some projects still retain the ASIC alternative in spite of these drawbacks. One example is the Thuraya family of telecommunication satellites equipped with $3.8.10^6$-gate ASICs jointly developed by Boeing and IBM and boasting an approximate twenty-fold increase in performance compared to previous solutions. In this case, the rationale was to multiply the capabilities for simultaneous communications and equip a series of satellites (currently 3) with the same components.

[7]"Radiation tolerant" denotes parts capable of sustaining radiation up to a certain dose (TID in krad) without the damage causing the destruction of the part (SEL) or permanently affecting the functionality. "Radiation hardened" parts, beside being radiation tolerant, additionally feature resistance versus transient effects like SEU and SET thanks to a mitigation strategy (usually some flavour of internal TMR).

	Hardened	Tolerant	Hardened
vendor	Actel	Xilinx	Atmel
parts	RTAX-S	XQR (Virtex II)	ATF280E
technology	anti-fuse	SRAM	SRAM
gates	250k - 2M	400k - 6M	< 280k
pins (total)	< 1272	< 1517	< 472
Radiation			
program	fixed (insensitive)	EEPROM (sensitive)	EEPROM (sensitive)
SEU	transistor-level TMR	TMR tool	self-integrity checks
TID	200 krad (Si)	200 krad (Si)	300 krad (Si)
SEL threshold	> 100 MeV/cm^2/mg	> 125 MeV/cm^2/mg	> 80 MeV/cm^2/mg
SEU threshold	> 50 MeV/cm^2/mg		
flip-flop SEU rate	< 1 per 2500 year	< 10.3 per day	
RAM SEU rate	$< 10^{-10}$ e/b-d	< 2.2 errors per day	

Table 8.3: Comparison of available radiation-hardened and radiation-tolerant parts "out of the box" as per Dec. 2003 (notable additional parts since then are the 4M-gate RTAX-S and the 8M-gate Virtex II). Figures comes from the vendors' data sheets and [50] (rates are given for the interplanetary space with minimum solar activity).

cost[8], availability for developments, shorter time-to-market, development tools. They are not, however, widely used in space applications[9] [50], as opposed to RTAX-S[10]. For designing a representative demonstrator the RTAX-S has had our preference and constitutes our target technology.

8.3 Design methodology

General design flow

The design flow for programmable hardware is somewhat different from its software equivalent due to the need to have finer control over the result (logic blocks, timing, placement and routing) as compared to the assembly-level code produced by a compiler.

1. **VHDL**

   VHDL[11] is central to the design process because of the variety of description levels it offers. With its `variable`, `signal`, `process` and `component` constructs, to name only a few, functional, physical and structural specifications are possible. While the first serves principally for validating the need with a granularity ranging from bit-wise operations to the overall data flow, the second makes it possible to exactly specify synchronous or asynchronous structures (RTL). Finally, with the third, the system can be segmented in units, either at a large scale for board-level engineering or as a network composed of the part's cells ("macros" in Actel's nomenclature) for component design.

   In our case, with the pre-existing software implementation and the preparatory work of P. Laporte [33], it has proven possible to shorten the functional validation and produce a first RTL version rapidly upon becoming familiar with VHDL and its simulation semantics. Such a highly detailed low-level coding has a double advantage: it remains understandable by humans and can hence be more easily reviewed while being sufficiently close to the final netlist for critical examination of the outcome of synthesis.

2. **Synthesis**

   This step is the equivalent of compilation for software. The synthesis tool (Synplicity's Synplify for Actel) interprets the VHDL description together with a set of constraints concerning timing, resource usage, ports and

[8]Although candidate parts for space undergo different manufacturing and verification which translate into increased cost.

[9]Recent evolutions in the design practices for space-borne electronics have seen the growing use of SRAM-based FPGAs to benefit from reprogrammability in space (refer to the MAPLD 2008 conference). Hardening of such designs is, however, challenging for not only the logic but also the configuration must be made robust to SEUs. Accordingly, beside some flavour of TMR these parts must permanently check and refresh their configuration (scrubbing).

[10]Actel boasts that their RT, RTAX and RH parts have been used on board more than 71 satellites, probes or launch vehicles including, for example: Rosetta, Cassini, Hubble etc.

[11]Or Verilog.

technology mapping to translate it into a netlist – an interconnection of cells. While this is straightforward for low-level operators involved in the control logic, conversion of those used in arithmetics is significantly more complex. The amount of resources used for the latter is highly device dependent and while flash-based chips have only basic logic cells, RTAX-S ones feature 1-bit arithmetic primitives[12] and Xilinx's Virtex II parts have more elaborate ones (notably 18×18 multipliers). Fig. 8.1 illustrates the cost of some elementary operators as a result of Synplify runs for completely unconstrained designs targeting the flash-based technology[13].

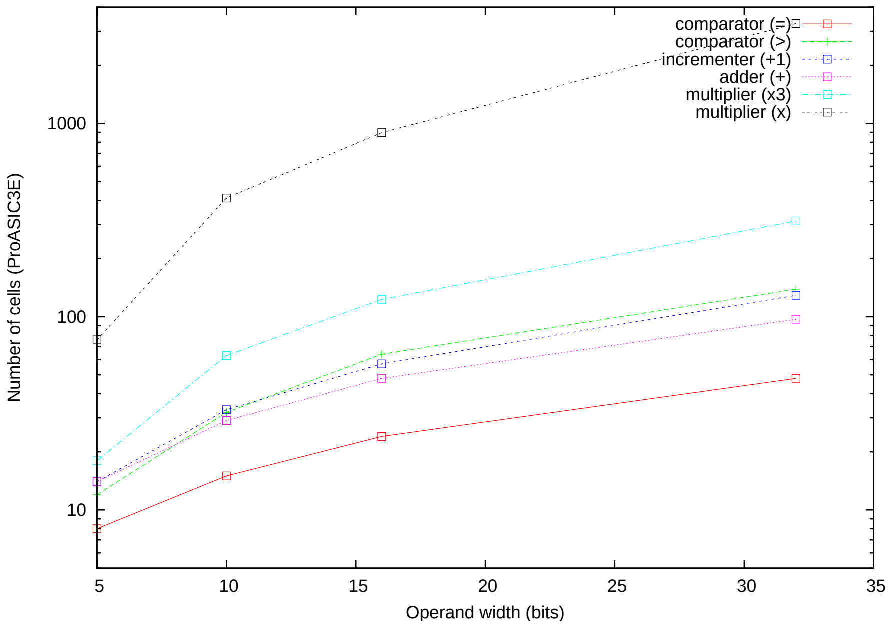


Figure 8.1: Resource allocation for conversion of arithmetic operators to logical cells as a result of synthesis (Synplify) for a flash-based device.

One difficulty, partly related to the flexibility of VHDL, is that not all constructs can be synthesised. This is the case for many of the facilities introduced for functional simulation for instance. The rule for designers is to refrain from thinking functionally but rather interpret the constructs physically – that is as blocks inserted on the data path much as electrical components are in a circuit, serially or in derivation. Furthermore, there exist so-called "semantic mismatches": some constructs are interpreted differently by the simulation tools (Mentor Graphics's ModelSim) and the synthesis engine. A typical example of this concerns the interpretation of sensitivity lists for asynchronous processes. Simulation, being a software task, uses them for optimisation purposes and only evaluates the statements in the process each time a transition occurs on one of the signals in the list. Conversely, in hardware, they are merely optional as a form of declaration of the input signals to the process and do not keep the cells' outputs from evolving between the afore-mentioned events. A cell used for inverting an internal signal, for instance, would repeatedly do so and produce oscillating results at a rate corresponding to the cell's intrinsic timing. Instead, the rule for such configurations is to rely solely on idempotent[14] statements.

3. **Place and route**

The netlist must then be physically placed on the chip. This is usually carried out through the vendor's tool (Designer for Actel) either automatically based on timing and physical constraints (in a first pass) or manually (afterwards). As explained in section 8.2.2, this stage is critical in the sense that the propagation delays between

[12]From the RTAX-S macro library:

- Half-adders: HA1, HA1A, HA1B, HA1C.
- Adders: FA1, FA1A, FA1B, FA2A.

The Axcelerator family from which the RTAX dies are derived and which are designed for ground feature additional adder macros (ADD1 and FA2) as well as a subtractor (SUB1) and a building block for multipliers (MULT1).

[13]Some of the results are highly surprising, but nevertheless confirmed. For example, incrementers are always more expensive than adders for the same word length while they implement reduced functionality. Similarly, > comparators are unexpectedly costly with figures suggesting the implantation of a subtractor together with a comparison to 0.

[14]An idempotent operator is such that repeated application does not change its output.

the cells depend on the data paths and number of interconnection points between them. Only then can the timing model used by synthesis be confirmed or invalidated based on real predictions. Optimisations during place and route determine which parts from a given family of components are compatible with the design as not only should the number of cells available be sufficient but the routing should also remain sufficiently simple to meet the timing constraints. As a consequence, full utilisation of the resources is generally not possible.

4. **Programming and execution**

   The final step consists in programming the device and, for reprogrammable ones, this is generally performed through the JTAG interface. Beside the logic itself, Actel's flash FPGAs offer a mechanism for easily initialising the internal RAM[15] by this mechanism – something of particular relevance for a demonstrator.

5. **Verification**

   With the decision to describe the system, not only at the bit level but also at the nanosecond and micrometre levels due to timing and placement, the need for verification is obvious. It even needs to run parallel to the developments themselves, in good agreement with the QA policy for satellites.

   The multiplicity of descriptions rendered possible by VHDL make it possible to faithfully model the design at every stage in the flow. VHDL accordingly forms the backbone of verification through simulation, first directly with the designer entry, then by converting the netlist to purely structural VHDL, finally by back-annotating it with the timing parameters determined as a result of placement and routing. Beside the benefits of using a single simulator and of full integration to the flow, the designer can then rely on the same test-bench for stimulating his design and validating the conformance of outputs at all stages.

   While simulation allows for inspecting the internals of a yet incomplete design, it only does so through a variety of models. These have different levels of fidelity, yet necessarily remain somewhat idealised. Additional in-place verification is mandatory for all the analog parts of the design, including the FPGA's interfaces. While an external view of events can be acquired with a logic analyser, their internal counterparts are of more difficult access. A number of solutions exist nevertheless, ranging from special operating modes routing signals of interest to unused ports to instantiation of "Identify" components multiplexing them through the JTAG interface. As a consequence, ensuring testability has implications on the architecture and is more extensively discussed in chapter 9.

Space QA

With the high reliability objective set for space applications, QA is an activity in itself and is generally managed by an independent team. It would be much too long and cumbersome to discuss all the aspects involved here, instead only key points are summarised below in something resembling a checklist. The intent is to provide reference criteria to place our demonstrator in perspective regarding feasibility and representativity of FM technology.

- **Project management**

  FPGA or ASIC development activities are covered by a standard from ECSS [51]. It corresponds to a high level management methodology with particular emphasis on phases[16], documentation and reviews carried out by independent teams. Fig. 8.2 illustrates the corresponding recommended QA flow.

- **Design**

  - Failure

    A failure strategy is necessarily at the heart of the design and of the development methodology. It naturally begins with extensive verification through simulation and measurements on the various development and engineering models but also with manual inspection of the netlists, timing analyses (because the EDA tools are known to fail now and then) and of the environment of the FPGA (PCB design[17]). Recognising that in spite of these precautions anomalies can still occur, partly because of radiation, robustness is introduced

[15] This RAM is of static type and should hence be denoted SRAM but to avoid any ambiguity with the external SRAMs, we prefer to refer to the FPGA's internal memory as RAM throughout.

[16] Definition, architectural design, detailed design, layout, prototype implementation and design validation and release.

[17] The SEASAT mission is known to have failed because of a shortcut between the power and ground circuits which were too close on the PCB.

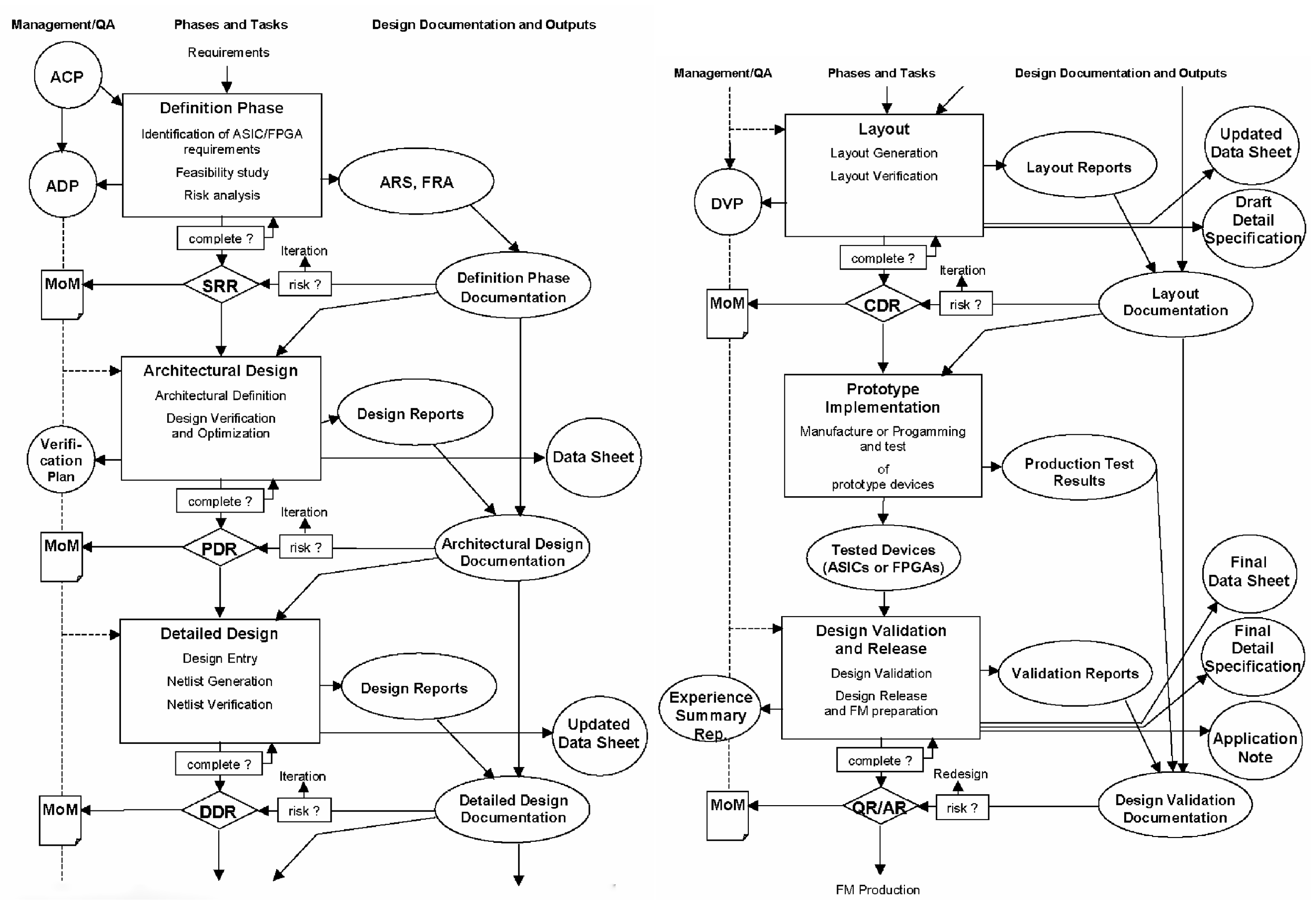


Figure 8.2: Example of development flow for an FPGA or an ASIC solution according to [51] (chronologically from top to bottom and left to right).

at a number of levels: eg. TMR internally or at the component level versus SEUs, recovery capabilities versus "unreasonable" inputs or break of wires and hot, cold or lukewarm redundancy versus part failures – although mainly at the sub-system level, redundancy modes call for support for the switch mechanism at the part level.

– Start up

As all non-permanent regimes, the start-up sequence is a critical phase requiring special attention [48]. Although the specification seems simple enough: power-on the FPGA and place it in a predetermined state from which nominal operation can then proceed, it is also necessary to guarantee that

∗ no transient signals appear on its output ports[18],
∗ cores are powered up before IO banks and PLLs to avoid stressing or damaging the part with inrush currents,
∗ proper delays are allocated for power-on and for reset in spite of varying initialisation times for cores and PLLs depending on temperature, voltage and radiation...

– Timing

Particular attention must be devoted to clock signals because of their central role in synchronous processes. Generally speaking, slower clocks are always advisable – within the limits of the functional needs – because they translate into lower power consumption, reduced sensitivity to SEU, simplified timing and routing as

[18]A problem which is known to have caused mission failure when triggering pyrotechnic elements for the deployment of solar panels or the release of shutters.

well as robustness to variations in duty cycle[19], the flip-flops' setup and hold times[20] and in propagation delays[21] with temperature, voltage and ageing. With this in mind, the introduction of multiple clocks to reduce timing requirements where possible seems desirable but must be traded-off against the additional complexity associated to managing transfers between clock domains.

– Pins

Pin configuration and allocation are also a key element for reliability because they affect signal integrity from and towards the FPGA. Signals are indeed subject to a number of perturbations at their level, be it noise resulting from approximate TTL thresholds, voltage fluctuations on improperly terminated special and unused pins[22] (JTAG, configuration pins etc.) or cross-talk resulting from SSOs. General recommendations concerning these effects consist in spreading out SSOs and interleaving ground pins with critical IOs to reduce the cross-talk.

Pin allocation also influences the internal routing of the FPGA. In a somewhat counter-intuitive way, it is recommended to spread out pins belonging to the same bus to increase the size of the solution space for the routing. This comes from the fact that, beside the timing requirements, routing must ensure that "simultaneous" signals reach their destination within acceptable delays. While this latter constraint could be met be relying on close paths for the signals concerned, more often than not the solution retained by the routing tool consists in selecting very different paths with the benefit of reduced constraints for balancing them out.

With unused pins being just as important as used ones for inspecting the FPGA's internal state, ideal packages include very many pins to properly distribute the IOs while preserving unused pins for test (DFT).

• **VHDL**

ESA requires all programmable hardware entry to be in VHDL or, subject to approval, in schematics or FSM diagrams. The rationale behind such constraints is to guarantee that every part of the description is fully understood by the designer and open to discussion, as opposed to automatically generated code from high level tools for example. VHDL being fundamentally a language for description, additional criteria are defined in [52] to favour readability and open the possibility for independent evaluation during the projects' reviews.

Beyond coding style, reliability objectives also call for the generation of minimal netlists. The reason for this is that circuit area increases with the number of cells used and with it the sensitivity to particles (ie. the interaction cross-section). Besides, the leaner the design, the easier it is to ensure internal consistency and mitigate SEUs – a notable reason for tracking replicated flip-flops.

One particular situation calling for attention in this class of problems relates to FSMs and the risk associated to their evolving towards lock-up states after an SEU. State encoding with a Hamming distance[23] of 2 is generally recommended for detecting abnormal transitions. However, the one-hot encoding which results, and which changes only two bits between every two states in normal operation, implies larger state registers which are themselves more sensitive to SEUs. Accordingly, the two effects require a trade-off for FSMs with a large number of states. Finally, to avoid having to rely on a global reset for returning to normal with an interruption of nominal operation, it is mandatory to insert logic autonomously resetting the FSM locally, especially for all theoretically unreachable states[24].

8.4 Test platform

The development of demonstrators in early phases of large and complex projects answers the need to establish feasibility and obtain order-of-magnitude estimates regarding complexity and resources. As explained in the foreword, these have been the key objectives of our activities in phase A and as part of the PDHE contract. If developments have been continued after the start of phase B, they have rather targeted the consolidation of such assessments and the acquisition of expertise through R&D than migration towards a FM. QA requirements have, hence, not been central in our approach as they ought to be in the industrial phases. Efforts have nonetheless been invested in building a test

[19]Generally expected to be 50%, but often rather in the 40-60% range.
[20]Worst-case values (part speed grade) should be considered.
[21]±10% over mission lifetime.
[22]All special and unused pins should be grounded in the FM according to [48].
[23]In information theory, the Hamming distance between two strings of equal length is the number of positions for which the corresponding symbols are different.
[24]The tools in Actel's IDE provide special options to preserve this part of the specification against the simplifications inspired by its being unreachable.

platform representative of space-qualified equipment and in maintaining a list of deviations from it to support our conclusions. Design choices are the object of this section while shortcomings are analysed in the conclusion to this part (chapter 13).

8.4.1 Platform

Several strategies are possible for prototyping RTAX-S devices. From a general point of view, they propose different trade-offs between flexibility for development purposes and representativity of the final part. Actel's recommended procedure [53] clearly emphasises the latter aspect by simply substituting a commercial Axcelerator[25] part to the RTAX-S through an adapter and only modifying the ultimate programming stage in the design flow[26]. While as representative as possible, this solution is inappropriate for this study because Axcellerator parts, like RTAX-S, are only one-time programmable and hence do not provide much freedom for exploring the solution space.

A second possibility consists in using ALDEC's prototyping environment which replaces the RTAX-S with a ProASIC3 (flash-based) with a soldered adaptor and relies on software to convert the design to this latter part (ActiveHDL). The benefit of reprogrammability is in this case counterbalanced by a degradation of the timing representativity since the two parts have radically different internal architectures.

For cost and flexibility reasons, notably as concerns the simultaneous development of the board supporting the FPGA), this last solution was not retained but pushed on step further instead. If a design running on a ProASIC3 with a number of restrictions can be considered a satisfactory model as ALDEC claims, then enforcing these restrictions manually, it is possible to prototype directly on an equivalent ProASIC3E die before converting the design to the RTAX-S flavour. As previously, developments can then use the same tools (Actel's Libero IDE). Additionally, standard VHDL, use of macros with equivalents for the RTAX-S and relying massively on synchronous operation are some of the precautions which have been taken.

This decision has the advantage of opening the possibility to build a test platform based on one of Actel's Starter Kits. These boards include ProASIC3E chips with the PQFP208 package and standard connectors for virtually plugging anything directly on the 147 available pins[27]. From a project point of view, although not sufficiently representative for high frequency designs, this approach was preferred for its cost effectiveness, its immediate availability and as a further inducement to produce a "slow" design – as power and radiation considerations recommend anyways.

With this philosophy emphasising flexibility, although potentially at the expense of performance, F. Rigaud has developed the test platform presented in Fig. 8.3 and equipped with a ProASIC3E600 die. Its architecture, along with the FPGA's internal organisation, is the object of chapter 9 while the supporting PCBs and the setup of a first test platform is the object of appendix A.

8.4.2 Programming

With the selected ProASIC3E600 chip, the low area constraint general to space applications applies tenfold. Indeed, comparing the 13824 flip-flops available with the cost of arithmetic operators illustrated Fig. 8.1, little more than 4 32×32-bit multipliers can be implanted. Although the algorithms described in part II are more data-management-intensive than computationally so, some adaptations to the processing flow had to be introduced (chapter 9) and particular attention paid to resource allocation (chapter 12).

A few simplifications have been introduced to this end to reduce the amount of control logic. One, already mentioned above, is related to setting the content of the internal RAM through the JTAG port with the `UJTAG` macro. Another corresponds to handling the processing parameters. For the FM, it would be desirable to store their values in some separate memory device, protected from radiation and updated through telecommands, and equip the FPGA with a mechanism for fetching them to update internal registers. Instead, all parameters have here been collected in a package and defined as constants.

With establishing the feasibility of the algorithmic approach as a key objective, the VHDL description has naturally been tailored for synthesis rather than for simulation. Combined with making the best possible use of the limited resources available, this has implied RTL coding and use of standard low-level constructs only – except for `+` or $\times$ deemed sufficiently easily recognisable by synthesis to be optimised respectively into adders and incrementers or multipliers.

For easier inspection, both of operation and structure, all data types are defined based on symbolic constants or `generic` parameters. This allows for reducing the complexity of the design without requiring modifications in the code itself. At the level of the SRAM controllers, for instance, the data and address word sizes or the depth of the FIFO

[25]RTAX-S parts being derived from the Axcellerator family, the two closely resemble each other. The introduction of radiation hardening, however, implies that internal timing differ (notably because of the transistor-level TMR) so one is representative for the other only insofar as purely synchronous designs are concerned.

[26]Designer is capable of converting RTAX-S programs to Axcellerator but not the reverse.

[27]The total of 208 pins includes power, ground and the JTAG pins.

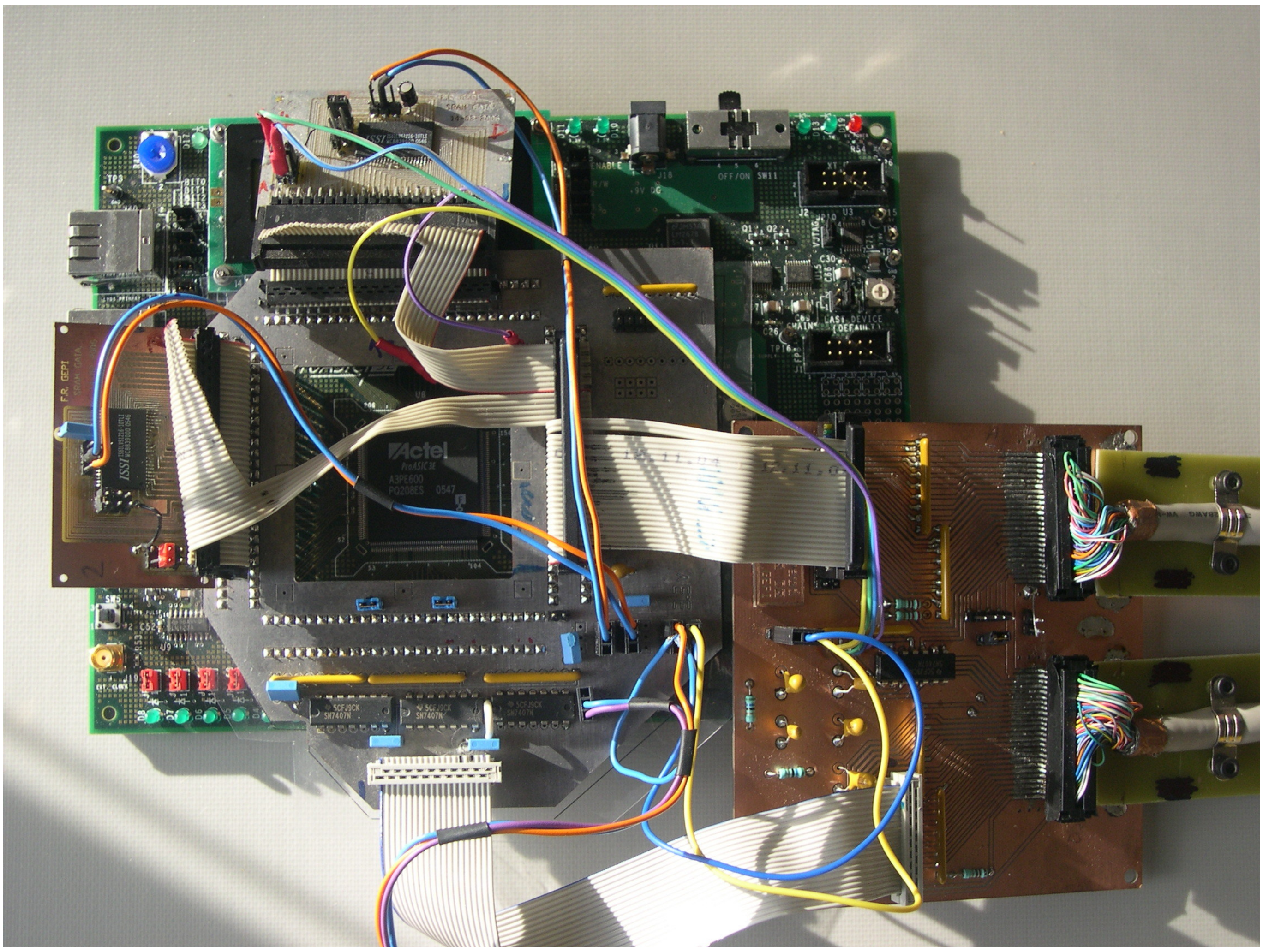


Figure 8.3: The test platform based on Actel's Starter Kit (in the centre with a ProASIC3E600 die with a PQFP208 package) with two external SRAMs mounted on secondary boards for pin assignment flexibility (top and left) and the interface board to the PC (bottom right) stimulating the design and retrieving results.

used to publish read-out data are modifiable by this means. At a higher level, the number of external SRAMs can be configured to simplify the architecture by altogether suppressing one of the controllers and the associated routing resources.

8.5 Conclusion

With the aim of building a complete demonstrator, what follows focuses on developments for the test platform described above. The intent has been to establish feasibility and to this end, the somewhat lower performances of the test hardware and the less stringent quality constraints as compared to space-grade industrial products will provide margin for improvement. Our step-by-step presentation will also render critical design choices or particular difficulties fully apparent, thereby facilitating iterations towards refined solutions as regards quality standards and hardware operation. Until then, let it be borne in mind that scientific performances and exact equivalence with the software model have been placed at the centre of this round, including support for infrequent special cases in spite of the associated costs.

Chapter 9

Internal architecture of the FPGA

Contents

9.1 Introduction

As opposed to modern trends in software targeting genericity and re-usability of code across different applications, FPGA designs offer an opportunity to match the needs as closely as possible, not only for reasons of efficiency, but also to minimise design risk, chip size and power consumption. Demanding embedded applications for the transportation and telecommunication markets for example, although starting with high level models for prototyping, then require the crafting of fine-tuned systems to achieve these objectives. While these considerations are evidently applicable to this study, and even to a higher degree than for ground applications due to the mass, power and reliability constraints in space, a demonstrator's design should remain inherently flexible. This flexibility should not, however, be understood in operational terms, since the aim remains to have the FPGA perform exactly the required operations and no more, but rather in the ability to obtain a number of different operational configurations from a single model or, in our case, VHDL description. Modular architectures, with the encapsulation they provide, are the key to such designs and VHDL offers a powerful mechanism for this through the `if generate` statement which allows for configuring the architecture through a set of global variables.

The first reason for introducing flexibility in a demonstrator such as this one is that it should allow a wide exploration of the solution space. Expanding or reducing it to investigate different resource allocation scenarii is part of the underlying R&D. Besides, modifications of the satellite's concept and of the operating conditions should be easily accommodated to allow the demonstrator to remain in phase with other development activities.

A second key need translating into flexibility is related to the fact that a demonstrator is precisely a prototype. As such it is of primeval importance that every aspect of the design be testable, that is both the hardware and the logic, not only for development purposes (ie. debugging) but also in the light of the validation versus the specifications which must naturally ensue. In our case, this implies for instance, being able to inspect not only the external interfaces but also the internal ones to the storage space available through dedicated SRAM modules. In order to reduce development risks it is also desirable to make the design testable at intermediate stages, which among other things calls for special routing of intermediate data compatible with the nominal interfaces.

Finally, because a number of simplifications are necessarily made, to reduce the cost of the test platform, to facilitate the exploration of various solutions, or to accommodate a test environment itself under development, provision must also be made for migrating the design to the flight equipment. This is of particular importance in our case because the target technology (Actel's anti-fuse RTAX-S chips) being only one-time programmable, developments and tests are carried out on a more flexible one (Actel's flash-based ProASIC3E chips). In spite of having chosen one as a model for the other, which implies some similarity between the two, one of the driving design constraints is to abstract the idiosyncrasies of both. This requires relying on a high level, yet precise, description – for which VHDL is perfectly well suited – and leaving the responsibility of adaptation to the context to the synthesis tool. Non-portable constructs need to be avoided as far as possible and for cases in which they are unavoidable, carefully segregated[1]. Besides, as these differences translate in the use of different macros and, hence, in different timings and resource allocation, even for a design functionally identical, appropriate margins are mandatory. Fortunately, the chosen test platform is favourable in this sense since anti-fuse systems by nature permit higher frequencies and provide higher densities at lower power consumption than the flash-based technology[2].

9.2 Architecture

The FPGA's architecture is shaped by three equally important drivers: fulfilling the functional specification, operating the interfaces for integration in the VPU and providing rich debugging facilities. Although the decision to rely on modularity offers the possibility of gradually building up a single architecture thanks to a careful separation of tasks, it was found more suitable for development purposes (notably as concern the interfaces) and because of limitations in the test platform to introduce an intermediate simplified one. The corresponding somewhat incomplete system, beside the simplifications it allows also results in an additional milestone for validation before it is eventually completed into a full-fledged system.

9.2.1 Simplified architecture

The sample-based processing to be carried out by the FPGA has interfaces (section 9.3), on one hand, with the FPA through a video buffer for the reception of the sample data and, on the other hand, with a processor-based unit for the object-characterisation tasks. Having focused on the hardware part in a first phase, this latter interface, being SRAM-based, poses a testability problem since the soft real-time software-based processing system supposed to fetch data from the SRAM is not yet set up at this point in time. Since, unfortunately, this SRAM cannot be easily simulated using a computer for evident timing reasons, an intermediate architecture has been defined. It implements the processing up to and including the sample selection stage and outputs the selected samples and associated data to the computer already simulating the video buffer interface with the FPA (section 9.3). Fig. 9.1 illustrates this.

Another, by no means less important, reason for deriving this simplified architecture is related to the amount of resources available on our test platform. Two limitations render the implementation of the complete design difficult on the ProASIC3E 600 die. The first comes from the budget in terms of number of pins versus the input and output ports needs (as is shown on Tab. 9.1). Indeed, in spite of the fact that the PQFP208 package features 208 pins, only a tight 147 remain available to the user after subtracting those pre-assigned to input voltages and ground connections. The second is relative to the amount of internal logic accessible. In spite of the advertised 600 000 "system gates" [3], only 13824 logical cells (D-flip-flops) can be used. Besides, for placement and routing purposes, even in a design supposedly "slow" compared to a part's maximum pin-to-pin performance like this one (350 MHz), it is not advisable

[1]This is for example the case with PLLs which are quite simply unavailable in RTAX-S dies (because external ones are usually preferred for FMs) or with internal RAM types.

[2]The frequency gain results from the different interconnect technology, density from the physical size of the individual logic cells and consumption from reduced static dissipation and leakage currents.

[3]The precise meaning of this measure, supposed to characterise the FPGA by an ASIC equivalent, is variable from vendor to vendor so that it is more relevant to marketing than to strict specification of the part.

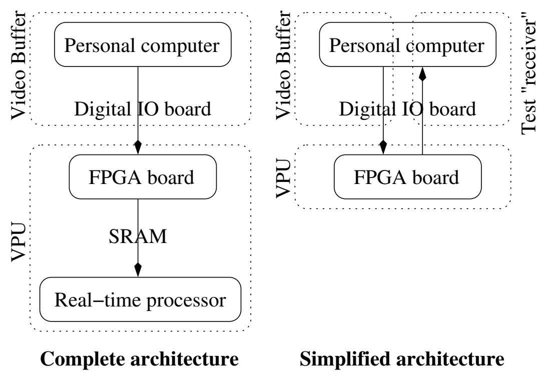


Figure 9.1: Comparison of the complete and simplified architectures.

to target 100% resource utilisation. Hence, the reduction of logic consecutive to the simplified architecture was found to be very welcome.

Fig. 9.2 and 9.3 represent the block diagrams for the intended test platform implementations (simplified and complete). Theses figures illustrate how the required modifications for the simplified configuration are limited to:

- Refraining from instantiating the connected-component labelling engine within the processing core since the processing stops at the sample selection stage. The reason for skipping the connected-component labelling is that, exploiting the storage space provided by SRAM0, it builds objects directly in the interface.
- Removing controller 0 and the associated ports as well as the routing of data through the switch.
- Replicating the handshake interface to export the products of the processing for validation.

Tab. 9.1 summarises the device's ports for the complete and simplified configurations. Active low signals like the SRAMs' write enable are designated either with the $\overline{\text{WE}}$ notation or as nWE when not possible (in the VHDL code and in figures for instance).

9.2.2 External SRAMs

The background estimate presented in chapter 5, although highly optimised from a computational point of view, introduces a delay related to the regional scale at which it applies. As a consequence, samples must wait until a background estimate becomes available at their location before being submitted to the SNR criterion for selection. Another implication is that the histograms are built gradually and must, hence, remain accessible during at least half of the previous delay. Each of these individually represents a significant amount of data, as sections 12.3.3 and 12.3.4 show, implying that the FPGA's storage capabilities are insufficient and that they must be complemented with external memory. Once the decision is taken to render the design more complex with such an interface, one might as well rely on several chips to benefit from a large storage space and more flexibility as regards the scheduling of accesses. This is precisely the option retained and, accordingly, two independent memory chips are used to permit concurrent accesses. It is worth noting that the pin assignment constraint being very tight with PQFP208 packages, careful selection of the SRAM parts is mandatory. Accordingly, section 9.3.3 presents the various possibilities explored as well as the rationale for choosing the components.

9.2.3 Clock generation

For quality reasons, it is generally preferable to rely on external PLLs for the manipulation of clocks based on the signal from the main oscillator. For this reason, and because of the difficulty to protect PLLs against SEUs, Actel's RTAX-S chips do not feature internal PLLs. However, because of the complexities associated with having high frequency signals passing through the FPGA's input ports and the stringent requirements applicable especially to clocks, it is both simpler and more effective for a demonstrator to rely on internal logic. This is all the more valid as the ProASIC3E chip features internal PLLs while the Starter Kit supporting it is not equipped with any. To allow running the algorithms on the available platform while leaving the possibility for simulation on RTAX-S dies open, the PLLs have been isolated in a specific component layer: the `Framework`.

The rationale for clock derivation is discussed in section 9.5.2. Depending on the option retained the strategy implemented on the demonstrator may be considered as part of the simulation environment or of the design itself.

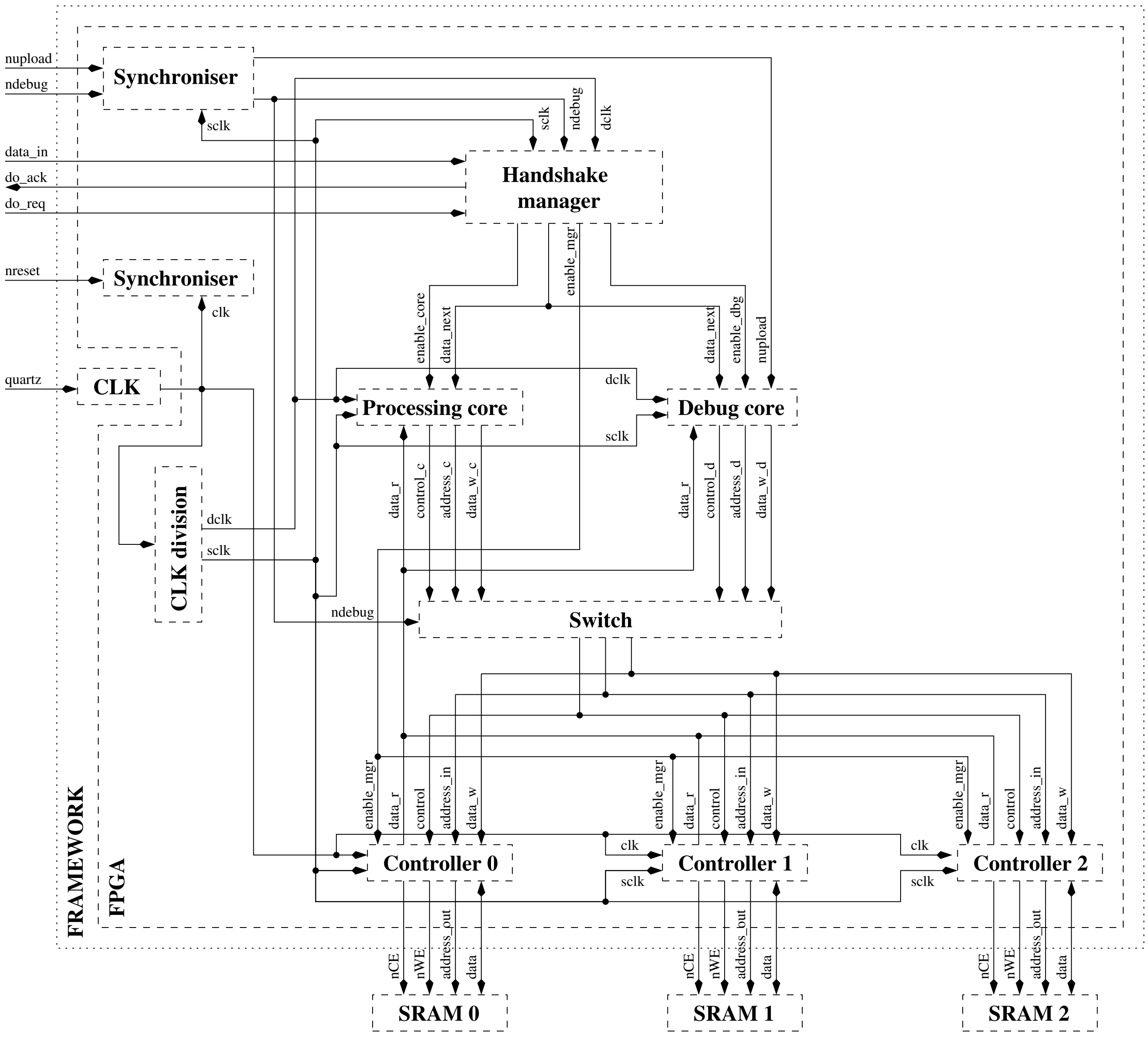


Figure 9.2: Intended complete design including connected-component labelling and forming object entities in the SRAM-based interface with software (SRAM0). Although it is not represented, the $\overline{\text{reset}}$ (ie. the synchronised $\overline{\text{nreset}}$ signal) is distributed to every component except the switch module which is purely combinational based on the $\overline{\text{debug}}$ signal. SRAM0 corresponds to the interface with the processors while SRAM1 and 2 serve for data storage (all three are asynchronous).

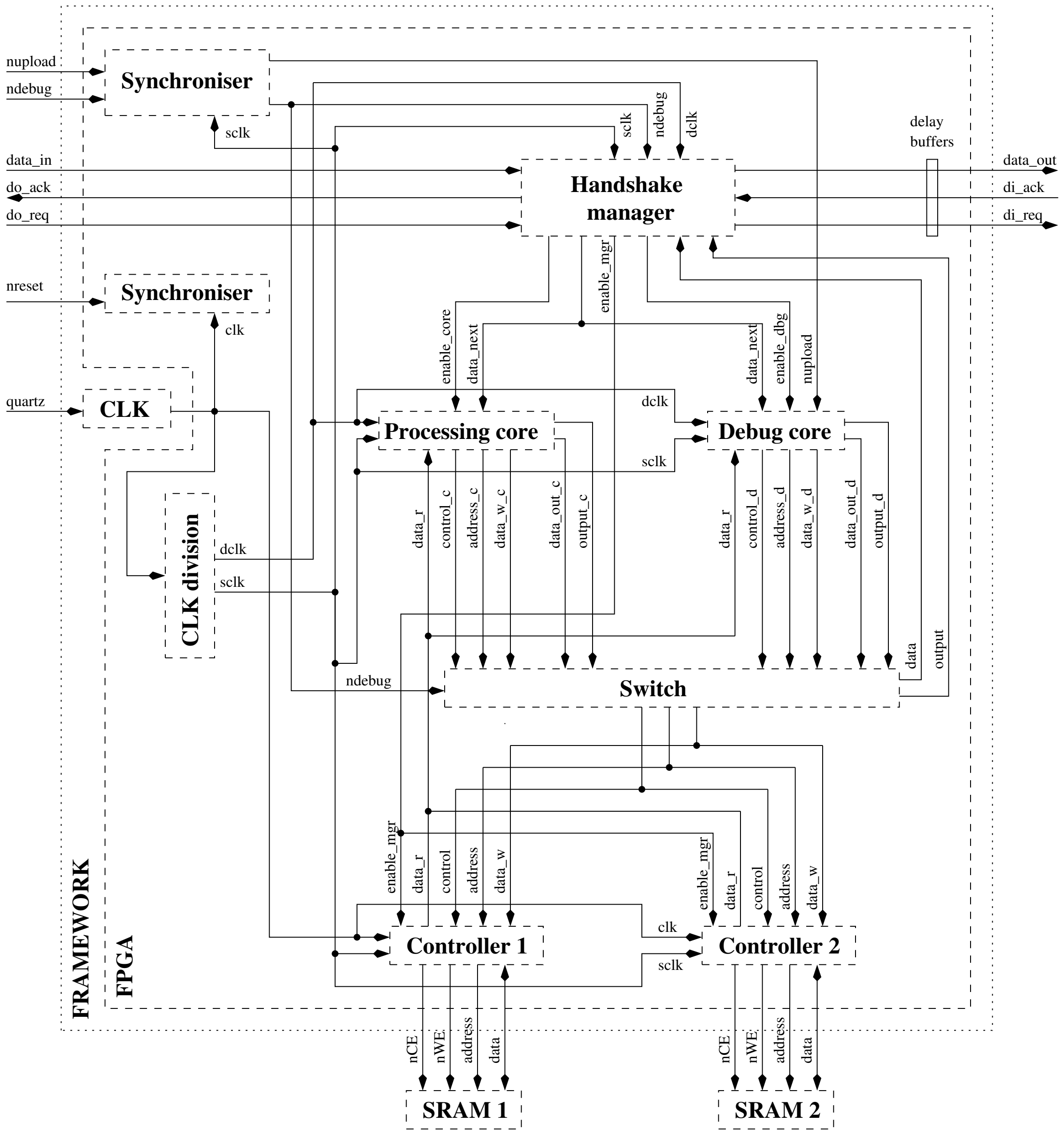


Figure 9.3: Simplified design with processing up to sample selection and outputting selected samples' value, background estimate and AL and AC position through the handshake interface (with configurable delay buffers introduced on the line to reduce the effect of SSOs (cf. appendix A)). $\overline{\text{reset}}$ is distributed to every component except the switch module which is purely combinational based on the $\overline{\text{debug}}$ signal.

Port name	Width	Direction	Function
Control and handshake interfaces			
			Control
`quartz`	1	in	Signal from the quartz oscillator (25 ns)
$\overline{\text{reset}}$	1	in	Asynchronous reset
			Special modes
$\overline{\text{debug}}$	1	in	Debugging switch
$\overline{\text{upload}}$	1	in	Write to SRAMs switch
			Input handshake protocol
`do_ack`	1	out	Input reception acknowledgement
`do_req`	1	in	Input data availability
`data_in`	16	in	Input data word
			Output handshake protocol (simplified architecture)
`di_ack`	1	in	Output reception acknowledgement
`di_req`	1	out	Output data availability
`data_out`	16	out	Output data word
SRAM access			
			Address and data ports
`address`	16	out	Interface addresses (SRAM0)
`data`	16	in & out	Interface content (SRAM0)
`address`	17	out	Buffer addresses (SRAM1)
`data`	16	in & out	Sample values (SRAM1)
`address`	17	out	Histogram addresses (SRAM2)
`data`	10	in & out	Histogram bins (SRAM2)
			Control ports (depending on SRAM types)
`SCLK`	1	out	SRAM clock (32 ns) (SPSS and DPSS)
$\overline{\text{WE}}$	1	out	Write enable (all types)
$\overline{\text{CE}}$	1	out	Chip enable for reduced-power standby (all types)
ZZ	1	out	Snooze for reduced-power standby (SPSS and DPSS)

Totals for the framework depending on the SRAM options (storage & communication)

Configuration	Control	Handshake	SRAM0	SRAM1	SRAM2	Total
2 SPAS (simplified)	4	36		$17+16+2$	$17+10+2$	104
2 SPAS & 2 SPAS	4	18	$2\times(16+16+3)$	$17+16+2$	$17+10+2$	156
2 SPAS & 1 DPAS	4	18	$16+16+2$	$17+16+2$	$17+10+2$	120
2 SPAS & 1 DPSS	4	18	$16+16+4$	$17+16+2$	$17+10+2$	122
2 SPSS & 2 SPAS	4	18	$2\times(16+16+3)$	$17+16+4$	$17+10+4$	160
2 SPSS & 1 DPAS	4	18	$16+16+2$	$17+16+4$	$17+10+4$	124
2 SPSS & 1 DPSS	4	18	$16+16+4$	$17+16+4$	$17+10+4$	126

Table 9.1: Framework ports in the complete and simplified configurations depending on the selected SRAM technology.

The "CLK division" component in Fig. 9.2 and Fig. 9.3 may then be moved either within the `FPGA` layer or the `Framework`. The former is assumed in the figures in spite of the need for improvements concerning the logic used and the quality of the signals generated because, as discussed in section 9.5.2, internal determination of secondary clocks seems preferable to achieve constant phase relationships.

9.2.4 Components

Architecturally speaking, our design philosophy has focused on segregating the problems through modularity. Acknowledging that the image processing, which represents the key functional block, cannot operate without a framework providing input and output interfaces, clocks and the reset, as well as the routing of data depending on the operational mode, a variety of components have been defined. The benefit of such a subdivision is twofold: not only does it permit independent validation of each of these aspects, but it also corresponds to a layer of abstraction which offers services to the different components through easily modifiable mechanisms of simple use.

- Processing core

  All calculations relevant to processing the sample data stream for the detection of objects of interest are gathered in this component. It receives one sample value every DCLK period, processes it using both internal logic, RAM resources and the external SRAM-based data storage. Depending on the configuration used (section 9.2.1), output data is either entirely forwarded to the Handshake manager for transmission to the "test receiver" (Fig. 9.1) or stored in one of the external memories. The algorithmic and implementation details for this component can be found in chapter 12.

Port	Direction	Width (bits)	Comment
`DCLK`	in	1	992 ns clock.
`SCLK`	in	1	32 ns clock.
$\overline{\text{reset}}$	in	1	Command the reset of the component (asynchronous).
`enable_core`	in	1	Enable operations for the next DCLK cycle.
`data_next`	in	16	Input sample data.
`data_r`	in	$N \times 16$	Data read from the SRAMs.
`data_w_c`	out	$N \times 16$	Data to be written to the SRAMs.
`control_c`	out	$N \times 2$	Command read or write operations on the SRAMs.
`address_c`	out	$N \times 19$	Addresses to read or write in the SRAMs.
Simplified architecture			
`data_out_c`	out	$C \times 16$	Output data to the software engine.
`output_c`	out	1	Command handshake transaction cycle on output.

Table 9.2: Processing core ports (N stands for the number of SRAMs connected to the FPGA (section 9.2.1) and C for the number of 16-bits handshake transactions to perform during the next DCLK cycle (section 9.3)).

- Debug core

  A debugging core is introduced as an exact substitute to the Processing core, when the $\overline{\text{debug}}$ signal is asserted[4], for uploading or downloading content for inspection of the SRAMs (depending on the $\overline{\text{upload}}$ signal). This component contributes to making the design easily testable and is further explained as part of the DFT policy presented in section 9.6.

- Handshake manager

  Given that operations during the next DCLK cycle are only possible in the simplified architecture if both the next sample value has been received from the video buffer and the results from the previous cycle successfully sent to the software, management of the handshake interfaces is centralised in the Handshake manager which, in turn, enables the cores on a cycle per cycle basis (section 9.3). Conversely, in the complete architecture, failure to transfer data on output is less critical[5], hence operation during the next cycle can be enabled based solely on the availability of new data.

[4]assert: set a signal to its "active" state / de-assert: set a signal to its "inactive state.
If a signal is active-low, "asserting" that signal means setting it low and de-asserting it means setting it high.

[5]This is because SRAM0 serves also for internal storage and as an intermediate buffer between the FPGA and the software, and because access times being much faster on the SRAM multiple attempts may be carried out during a given DCLK cycle.

Port	Direction	Width (bits)	Comment
`DCLK`	in	1	992 ns clock.
`SCLK`	in	1	32 ns clock.
$\overline{\text{reset}}$	in	1	Command the reset of the component (asynchronous).
`enable_dbg`	in	1	Enable operations for the next DCLK cycle.
`data_next`	in	16	Input sample data.
$\overline{\text{upload}}$	in	1	Whether to upload or download data from the SRAMs.
`data_r`	in	$N \times 16$	Data read from the SRAMs.
`data_w_d`	out	$N \times 16$	Data to be written to the SRAMs.
`control_d`	out	$N \times 2$	Command read or write operations on the SRAMs.
`address_d`	out	$N \times 19$	Addresses to read or write in the SRAMs.
Simplified architecture			
`data_out_d`	out	$C \times 16$	Output data to the software engine.
`output_d`	out	1	Command handshake transaction cycle on output.

Table 9.3: Debug core ports (N stands for the number of SRAMs connected to the FPGA (section 9.2.1) and C for the number of 16-bits handshake transactions to perform during the next DCLK cycle (section 9.3)).

Port	Direction	Width (bits)	Comment
`DCLK`	in	1	992 ns clock.
`SCLK`	in	1	32 ns clock.
$\overline{\text{reset}}$	in	1	Command the reset of the component (asynchronous).
`data_in`	in	16	Input sample data (from the video buffer).
`do_ack`	out	1	Acknowledge reception of data (input handshake protocol).
`do_req`	in	1	Signals availability of data (input handshake protocol).
`data_next`	out	16	Input sample data to the cores.
$\overline{\text{debug}}$	in	1	Switch between processing & debug core (next DCLK cycle).
`enable_core`	out	1	Enable the processing core
`enable_dbg`	out	1	Enable the debug core
Simplified architecture			
`data`	in	$C \times 16$	Output data from the cores.
`output`	in	1	Command handshake transaction cycle on output.
`di_ack`	in	1	Acknowledges reception of data (output handshake protocol).
`di_req`	out	1	Signal availability of data (output handshake protocol).
`data_out`	out	16	Output data to the software engine.

Table 9.4: Handshake manager ports. C stands for the number of 16-bits handshake transactions to perform during the each DCLK cycle (section 9.3).

- Synchroniser

  While the Handshake manager is responsible for asynchronous data transmissions, the Synchroniser ensures that the asynchronous control interfaces do not lead to metastability problems. As discussed in section 9.4.1, assertion of $\overline{\text{reset}}$ remains asynchronous but its removal is synchronised here to ensure the system is left in an entirely predictable state. Conversely, all transitions on $\overline{\text{debug}}$ or $\overline{\text{upload}}$ are synchronised.

- Controllers

  Memory controllers abstract the read and write accesses to the external SRAMs. They are especially useful to hide the timing requirements of the asynchronous parts and allow synchronous requests to be emitted by the cores. They also offer pipeline facilities and isolate the cores' design from the idiosyncrasies of the selected SRAMs. A VHDL model for the asynchronous controller is presented in chapter 10. Depending on the configuration, a significantly simpler one for synchronous accesses might be necessary, but its design is still TBD (section 9.3.3).

- Switch

  For the sake of clarity and to facilitate maintenance, the output data from the two cores and to either the different storage SRAMs or the software is routed through a configurable matrix. The routing essentially consists

Port	Direction	Width (bits)	Comment
			Asynchronous controller
`CLK`	in	1	8 ns clock.
`SCLK`	in	1	32 ns clock.
$\overline{\text{reset}}$	in	1	Command the reset of the component (asynchronous).
`data_r`	out	16	Data read-out from the SRAM.
`control`	in	2	Command read or write operations on the SRAM.
`address_in`	in	19	Addresses to read or write in the SRAM.
`data_w`	out	16	Data to be written to the SRAM.
$\overline{\text{CE}}$	in	1	Enable SRAM.
$\overline{\text{WE}}$	out	1	Enable writing to the SRAM.
`address_out`	out	19	Addresses to read or write in the SRAM.
`data`	in & out	16	Data read or to write to the SRAM.
			Synchronous controller TBD

Table 9.5: SRAM controller ports (address and word sizes correspond to the maximum permitted by the selected technology for the test platform: ISSI ISI61LV51216, see section 9.3.3).

in extracting the Controllers' signals from the cores' output buses while configuration is relevant to switching combinationally between the cores depending on $\overline{\text{debug}}$.

- Clocks

  The oscillator soldered in the Starter kit runs only at 40 MHz to avoid the difficulties related to high frequency input signals. Accordingly, clock generation modules are introduced to generate the main 8 ns clock (CLK) and the slower secondary ones (32 ns SCLK & 992 ns DCLK) (section 9.5).

9.2.5 Initialisation

The FPGA requires both that processing parameters and internal RAM contents be initialised before nominal operation can begin (Tab. B.2 and B.3 in appendix B summarises the processing parameters). While the various thresholds come in small numbers, the pre-calibration tables are more significant. They fit in the general need to initialise the internal RAMs. Indeed, the various pipelines described in chapter 12 introduce latency before the first relevant data is processed. In the meantime, default values should be present in the RAM to ensure the logic can proceed without errors and will not affect the scientific output negatively.

A FM would rely on a dedicated operating mode for this purpose and access an accompanying EPROM used for the storage of processing parameters and default values. This corresponds to a significant increase in complexity as it calls not only for some dedicated logic, but also requires special pins to connect to the EPROM. Assuming this is not critical for the feasibility of the design, the test platform has been designed by taking advantage of the flexibility offered by flash-based FPGAs. While the few parameters have been quite simply hard-coded as constants (declared in VHDL packages), the possibility to initialise the internal RAM through the JTAG port and the `UJTAG` macro has been massively exploited. This is fully supported by simulation yet complicates the initialisation procedure (section 9.6.3) because it translates into a preliminary set up stage is required between each run. Conversely, it allows for discarding both the initialisation logic and the EPROM (which accordingly is not represented in Fig. 9.2 and 9.3).

The memory content is described in two ways depending on whether or not all addresses are filled with the same value. The simple case is managed directly by entering this value in SmartGen and applies to all cases calling only for a default value, ie. all RAM resources except the pre-calibration coefficients. When different values are concerned, files in Motorola's `srecord` format are used. This format encodes the addresses and values and includes a checksum for verification purposes. It is written in ASCII, with one entry types per line. Only the 'S1' type (among 10 different possible ones) is used for `Calib_RAM`, and entries are structured according to Tab. 9.6.

Initialisation of the external SRAMs would also be the responsibility of the dedicated operating mode. For simplicity, the test platform relies on the upload option of the debug mode to write data to the SRAMs – and potentially verify the resulting state. This requires some synchronisation with the data interface to the PC as explained in section 9.6.3.

	Type	Count	Address	Data	Checksum
Number of characters	2	2	4	4	2
Entry format	S1	05	$0 \rightarrow 0F5B$ (3931)	$0 \rightarrow 7FFF$ (32767)	

Table 9.6: Srecord entry format. The checksum is obtained by summing the bytes (two-character hexadecimal values) forming the concatenated count, address and data fields, then taking the 1's complement and finally extracting the least significant byte (rightmost 2 characters).

9.3 Interfaces

9.3.1 Input interface

With pixels binned in both AL and AC to reduce the data flow and increase the SNR, the alternate read-out of the TDI-operated SM1 and SM2 CCDs leads to packets containing 983 unsigned 16-bit sample values sent over the SpaceWire network between the FPA and the VPU every 0.9282 ms. The hard real-time constraint which springs from this interface is managed by a component dedicated to the reception of data from the SpaceWire network on the VPU side and to temporarily storing the decoded packets in a local video buffer.

The introduction of programmable electronics for the sample-based part of detection allows meeting the timing requirements imposed by the reception of data from the FPA. Indeed, the synchronous operation of the FPGA permits fetching this data in hard real time at exactly the rate of reception. The storage need can then be reduced to the latency introduced by the SpaceWire protocol on the receiving side.

There are essentially two ways the FPGA can gain access to this data, either through the afore-mentioned reception component or through direct access to the buffer. This latter approach, although seemingly appealing, is complex as it would require, on one hand, using parts capable of simultaneous read and write or some mechanism for resolving accesses to the video buffer and, on the other hand, call for either some large amount of margin between write and read cycles to the same address or some synchronisation between the two components. Last but not least, in case of failure of the transmission over the SpaceWire network for one or more TDI cycles, some special communication would be needed to suspend operations within the FPGA. Conversely, asynchronous operation for the two components and minimum communications between them suggest to rely on a handshake protocol (section 9.3.4). Assuming the reception component also has capabilities for this interface, access conflicts are naturally solved and publication of data can be very simply suspended in case of transmission problems with the FPA – thus also suspending activities in the FPGA via the Handshake manager (section 9.3.4).

The FPGA's processing pipeline admits a new sample on input every DCLK period (chapter 12), hence a single handshake transaction is required per cycle. For optimal operation, the transmission should be controlled by the FPGA. With the `do_req` signal re-asserted between transactions in less than a DCLK cycle, the factor limiting the transfer rate becomes the FPGA's willingness to admit a new sample and assert the `do_ack` signal. The main benefit is then that publishing the value internally can straightway be DCLK-synchronous.

9.3.2 Output interface

The situation is similar on the output side since the FPGA and the processor do not share a common clock either. With the simplified design affecting mostly the final stages of the processing and the output, two different interfaces are set up:

- Simplified.

  In this case, due to stopping the processing immediately after the sample selection stage, outputs still follow a regular pattern. When samples are discarded, nothing happens. Conversely, when a sample has been retained, the need to output the sample value (16 bits), the sample's AL and AC position (5 and 10 bits), its origin (SM1 or SM2: 1 bit) and the estimated background value at the sample's location (32 bits) calls for transferring 64 bits of data. The modest data rate together with the infrequent need for transfers suggest to also rely on a handshake interface: a sequence of four transactions (during which the 64 bits are multiplexed) is initiated after a sample has been retained and must be completed within one DCLK period.

  The Handshake manager uses a FSM to cycle through the states corresponding to the successive transactions. Given the cost of data storage on the FPGA's side, the transmission should be controlled by the FPGA. This time, it is the `di_ack` signal which should be asserted by the receiving side rapidly after reception of the data (within a fraction of a DCLK period) so that the FPGA should again be the limiting side.

The format used for transmitting AL positions deserves a little explanation. With the CCDs operated in TDI mode a stripmap is acquired which extends over the entire length of the mission. As a consequence, the AL indices of the samples span an interval of considerable length (from 0 to 157.7 billion TDIs in five years) which would require coding over at least 38 bits for absolute identification. Although the on-board time relies on a 64-bit format providing a resolution finer than the TDI and support for the 5 years of the mission (+1 extended time), relative positions are sufficient at this level. Accordingly, to reduce the number of bits only the 5 bits corresponding to the location within the hyperline (in AL) are transmitted instead. With approximate synchronisation, the receiving side can then identify the sample unambiguously.

- Complete.

  In this case, the objective is to transfer entire objects (member samples, object descriptors) once the underlying CCs are made complete by the CC labelling engine. Accordingly, the output functions in bursts and it would be utterly impracticable to rely on the sole handshake channel for transmitting all the data. A faster interface is implemented in the form of an SRAM-based buffer where the bulk of the data is made available to the software. Under the assumption that the FPGA and the software are synchronised by the PDHU (which distributes the clocks within and across the VPUs), a table of pointers providing the addresses to the complete objects can be set up in the SRAM, thereby suppressing the need for any direct communication with the software and for a handshake interface.

  Besides the communication proper, an additional benefit from this architecture is that the SRAMs can also serve as storage space for the FPGA and the software if the data is organised in a way which minimises conflicting accesses. This opens the possibility for performing the various operations on the data in place.

9.3.3 Memory interfaces

Technology

A number of technologies exist to fulfil the need for external memory. The need for random-access read and write rules out all ROM-based solutions, among which flash chips[6], and point to DRAM and SRAM parts. While, DRAM presents higher densities and faster access times due to the intrinsically simpler storage principle[7], refresh cycles are required to maintain the data integrity. These cycles come somewhat in the way of RAM accesses, whether they be managed internally as with Pseudo-Static RAM (PSRAM) or by the Controller, and are hence a source of additional complexity. Hence, considering the rather modest storage space and frequencies needed in this design, SRAMs have been preferred.

Within the SRAM family, main classes of components can be distinguished based on whether they operate synchronously (with read or write cycles triggered by transitions of a clock signal) or asynchronously (with cycles programmed by the transitions on the control signals themselves after precise delays) and based on the number and types of the ports they offer. In the simplest case, the SRAMs offer a single read and write port in the form of one data and one address buses and one set of control signals (eg. $\overline{\mathrm{WE}}$, $\overline{\mathrm{CE}}$ in the asynchronous case), but other flavors exist, principally with separated read and write ports (known as two-port SRAMs) or with two read and write ports (dual-port SRAMs).

Tab. 9.7 collects the properties of some chips of interest in our case (discussed in the next subsections). It first presents some COTS devices identified as possible parts for our test platform, then focuses on radiation-hardened or military grade equivalents to illustrate how representative of a FM each technical solution can be. Browsing through the catalogues of the main electronic manufacturers for space, one finds that SRAMs tend to have their preference over other technologies.

External storage

Noting that the external storage memory is by construction only accessed by the FPGA – although possibly by different components inside of it through the corresponding Controller – use of a single port synchronous SRAM seems recommended for the ease of integration in the overall synchronous processing combined with fast and simple access to the data. Given the SCLK (section 9.5) and the pipeline (chapter 12), response times of order 32 ns, coupled with several MBits of storage space would suffice for our purpose. However, following the argument below concerning the availability of synchronous parts for space, asynchronous SRAMs have been preferred for the test platform as being more representative of space-qualified technologies in use.

[6]The key limitation of flash as regards our intended use is that it does not offer full random-access rewrite capabilities because of the need to erase entire “blocks” at a time before changing bits from ’0’ to ’1’.

[7]One transistor and one capacitor per bit as compared to SRAMs which require six transistors.

Reference	Standard	Speed (ns)	Address (bits)	Data (bits)	Size (bits)	Pins (used, all)	Notes
			COTS				
ISSI ISI61LV51216	3.3V TTL	10	19	16	8 M	37, 44	Single port, async.
ISSI IS61NLF51218A	3.3V TTL	7.5	19	18	9 M	38, 88	Single port, sync.
IDT IDT70V28L	3.3V TTL	15	16	16	1 M	68, 85	Dual port, async.
IDT IDT70V3319	3.3V TTL	7.5	18	18	4.5 M	76, 128	Dual port, sync.
			Radiation hardened or military grade				
ATMEL AT60142FT	3.3V TTL	15	19	8	4 M	29, 37	Single port, async.
Honeywell HLX6228	3.3V TTL	32	17	8	1 M	27, 32	Single port, async.
WED EDI8L32512V	3.3V TTL	12	19	16	8 M	37, 69	Single port, async.
Maxwell 33LV408	3.3V TTL	20	19	8	4 M	29, 34	Single port, async.
WED WED2DL32512V	3.3V TTL	4	19	32	16 M	37, 97	Single port, sync.
ATMEL M67025E	5V TTL	30	13	16	128 k	62, 84	Dual port, async.
Aeroflex ACT-DP16K32A	5V TTL	40	14	32	512 k	48, 130	Dual port, async.
Maxwell 7025E	5V TTL	35	13	16	128 k	62, 77	Dual port, async.

Table 9.7: Comparison of SRAM technologies for the test platform on ground and for on-board operation (sync. stands for synchronous & async. for asynchronous). Speed entries are the values advertised by the manufacturers.

Communications

Conversely, the asynchronous operation of the FPGA and the software suggests decoupling the accesses from the two sides. A number of different solutions can be imagined to permit simultaneous access from the two sides independently. They correspond to placing the resolution logic either within the FPGA, between the FPGA and the SRAMs or within the SRAMs and can be compared based on performance regarding speed, memory size, number of pins and availability of parts for space.

- Single-port asynchronous SRAMs (SPASs) with internal selection

  SPAS components, beside corresponding to the more mature space-qualified technology as can be verified by inspecting the vendors' catalogues, offer fast response times for the creation of an increased throughput interface or for introducing flexibility in the timing on the FPGA or on the software sides. Shared access can be implemented through two independent SRAMs working as a double buffer granting exclusive access to the FPGA for building objects and to the software for retrieving them. Taking advantage of the alternate read-out of the SM CCDs, one SRAM can be devoted to objects from SM1 and the other from SM2, so that access rights can be exchanged at the beginning of each TDI. This would call for precise synchronisation of the FPGA and of the software through the PDHU and for ensuring that neither perform any accesses to the shared memory during the switch uncertainty delay.

  The downside is that connecting two SRAMs to the FPGA requires as many as 74 pins (with ISI61LV51216 parts) although not more than 37 are in use at any given moment. The ProASIC3E 600 soldered on the test platform, with its PQFP 208 package offers an insufficient number of ports (147 in user-space) for that purpose (Tab. 9.1). Although FPGAs with greater number of pins exist, notably for RTAX-S dies, and make this solution viable for the FM, a maximum of 147 usable pins are offered by Actel's existing Starter Kits (PQFP208 packages) and impose a strong constraint for the design of the test platform.

- SPASs with external selection

  An alternative possibility for mounting the two SPAS consists in relying on external part selection. In this scheme, only 37 pins are used for connecting the interface and external electronics is used for routing the signals to either SRAM depending on an extra 1-bit identifier – hence 38 pins in total. Beside the difficulty of designing the external electronics to keep the additional delay incurred within reasonable bounds, this approach suffers from increased cost and reliability problems versus radiation. For these reasons it is not advisable for the FM and has not been retained for the test platform.

- Dual-port asynchronous SRAMs (DPASs)

  Dual port SRAMs fully solve the connectivity problem as they can be treated as SPASs by both sides while featuring conflict resolution logic. With proper organisation of contents, such conflicts can be made infrequent, even in cases of partial desynchronisation, and the address space may be shared between objects originating from the two FOVs thereby offering an opportunity for priority-driven object handling across the FOVs. The cost of implementing the resolution logic within the SRAM, however, translates into limited storage space and slow response for space-qualified parts (Tab. 9.7). The storage space can be extended through address decoding[8], so that speed becomes the decisive parameter in favour of Atmel's M67025E. Conversely, on ground, parts like IDT's IDT70V28L have satisfactory responses and with 1 MBit allow for storing typically the content of 33 full TDI lines for SM1 and SM2 simultaneously ($2 \times 33 \times 983 \times 16 = 1038048$ bits).

- Synchronous SRAMs (SPSS and DPSS)

  Synchronous SRAMs present the benefit of much simpler read and write accesses than their asynchronous counterparts and choosing a dual-port synchronous SRAM (eg. IDT's IDT70V28L for the test platform) equipped with two independent clock ports could permit independent operation on the FPGA and the software sides. Much like the previous discussion concerning internal or external part selection, choosing synchronous SRAMs corresponds to moving the logic required for timing the accesses from the FPGA to the SRAMs themselves, thereby greatly simplifying the underlying controller. Although this has a cost as far as the number of pins is concerned, it would be highly desirable from a complexity point of view.

  Asynchronous devices have however been preferred because we have been quite simply unable to find space-qualified or even military grade synchronous SRAMs[9]. Even White Electronics Designs's WED2DL32512V (Tab. 9.7) sold under the "defence micro-electronics" heading is not of the required grade, although tested in the military temperature range. Accordingly, for the very purpose of establishing the feasibility of the proposed design, the synchronous option should be ruled out. This translates in instantiating three rather complex memory controllers and makes their design critical for the proper operation of the system (chapter 10).

	Function	Technology	Part(s)
Simplified test platform	External storage	SPAS	2× ISSI ISI61LV51216
Complete test platform	External storage	SPAS	2× ISSI ISI61LV51216
	Communication	DPAS	IDT IDT70V28L
Flight model	External storage	SPAS	2×2 ATMEL AT60142FT
	Communication	DPAS	8× ATMEL M67025E

Table 9.8: Proposed memory interfaces for the test platform and the FM. Relying on address decoding for communication in the FM requires extending the Framework's ports with $\overline{\text{OE}}$ signals (this is not necessary for the test platforms since memory devices on ground have sufficiently large address spaces). Conversely, the external storage needs for the FM can be satisfied by placing two SPAS sharing the same address bus side by side on the data bus.

Controllers

Accesses to the external SRAMs, asynchronous or synchronous, should be performed through dedicated controllers introduced to facilitate replacement of the SRAM parts and abstract the details of the timing of requests (especially in the asynchronous case). The design of an asynchronous controller is extensively described in chapter 10. The parts used for the test platform have somewhat higher timing performances than their space-qualified equivalents, however, the controllers operate them at frequencies which are more representative of achievable rates in space. Typically, with a 32 ns SCLK and an 8 ns CLK and the existing controller design, the SRAMs' response delays should not exceed 16 ns.

9.3.4 Handshake protocol

Protocol

The handshake protocol transmits data between the sending and the receiving side asynchronously in a manner which reports the success of the operation. Its principle is simple and involves two additional control bits, emitted respectively

[8]A technique which constructs a larger memory space by using several devices and generating $\overline{\text{CE}}$ and $\overline{\text{OE}}$ signals as part of the addresses: the N most significant bits serve as $\overline{\text{CE}}$ and $\overline{\text{OE}}$ bits to select which device is being addressed (and controls the data bus) while the remaining bits belong to the address bus shared by all devices.

[9]Discussions with memory manufacturers (3D plus, Aeroflex, Atmel, BAE) at MAPLD 2008 have shown that although not posing critical technology challenges, synchronous SRAMs have not been introduced on the market due to a low demand from the designers of space-borne architectures.

by the sending side (do_req) to signal the availability of new data on the data bus and by the receiving side (do_ack) to signal the success of the reception. Each side then listens to the other's control for rising edges (for active high signals) which trigger a new cycle of activity (either publishing new data on the data bus or fetching data from it). As a result, the interface is controlled by both sides, ensuring one never sends or fetches data for nothing, and the data rate is naturally limited by the slower side. The two typical sequence of states for a transaction are then:

- Sending side:
  1. listen on do_ack,
  2. after rising transition on do_ack, publish new data on the data bus (asynchronous),
  3. assert do_req,
  4. return to 1.
- Receiving side:
  A. listen on do_req,
  B. after rising transition on do_req, fetch new data from the data bus (asynchronous),
  C. assert do_ack,
  D. return to A.

Test platform

Our test platform relies on a personal computer for simulating the environment of the FPGA. On input, it directly simulates the VPU's SpaceWire reception component and the content of the video buffer (section 9.3.1): data from the official pixel-level Gaia simulator (Gibis [34]) is stored on the hard drive and passed on to the FPGA. Conversely, two cases need to be considered on output depending on the configuration. In the simplified case, after the calibration of samples and the estimation of the background data, the samples of relevance are identified and retrieved from the FPGA's output port (section 9.3.2) and stored on the computer's disk for further analysis. In the complete case, on the opposite, all the data transit through the communication SRAM to be handled by the software. The requirements for selecting the computer's interface board are then:

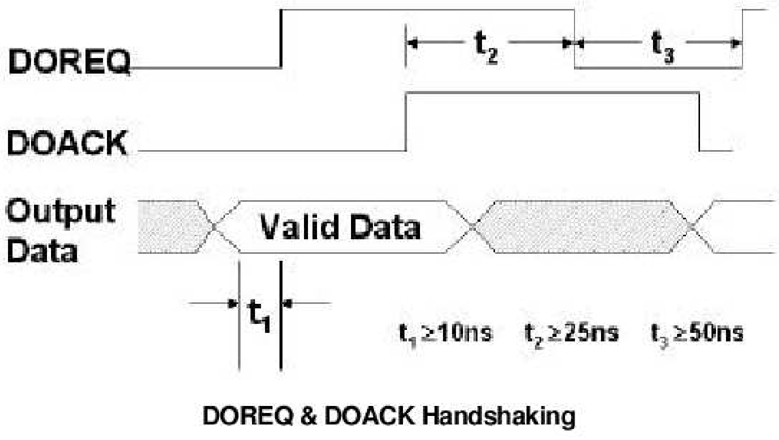
- to have independent digital data input and output capabilities,
- to support handshake transactions,
- to have $\geq$ 16-bit data input and output buses,
- to support at least 15.3 MBit/s transfers on output to the FPGA (one sample value per μs: 16 bits) and 61.1 MBit/s on input from the FPGA (sample value: 16 bits, background estimate: 32 bits, AL position: 5 bits, AC position: 10 bits and SM origin: 1 bit for each retained sample per μs, total: 64 bits).

Adlink's PCI-7300A digital IO board has been selected for this purpose. With 32 channels which can be configured as input or output, 16 bits can be devoted to output and as many to input. For retrieving the output data from the FPGA, four 16-bit transactions per DCLK period are then required so that the last requirement may be reformulated as:

- to assert do_req and di_ack signals respectively in less than 1 and 1/4 DCLK periods,

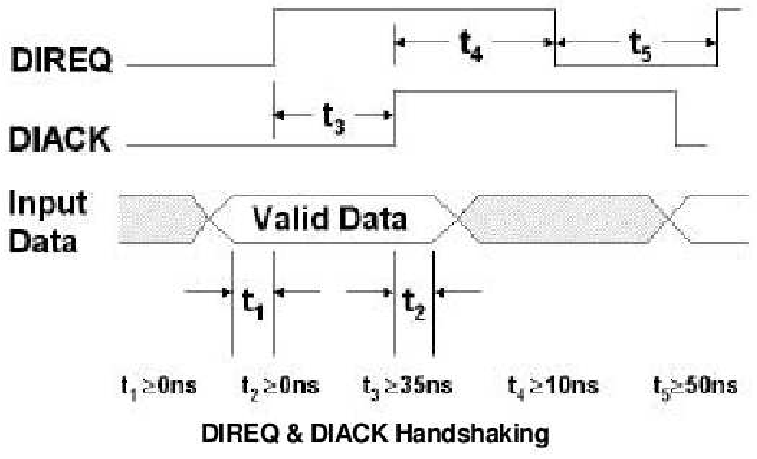
to make the full handshake control of transfers possible in all conditions and ensure the assumption according to which transfers are limited by the FPGA (sections 9.3.1 and 9.3.2).

Fig. 9.4 and 9.5 illustrate the evolution of the do_ack and di_req signals by waveforms. The selected board advertises 80 MBit/s transfers which is confirmed by the the timing indicated on these figures. Besides, the board supports a simplified burst handshake mode transmitting several words at a time should more margin become necessary.


DOREQ
DOACK
Output Data
Valid Data
t_2
t_3
t_1
$t_1 \geq 10$ns $t_2 \geq 25$ns $t_3 \geq 50$ns
DOREQ & DOACK Handshaking

Note: DOACK must be deserted before DOREQ asserts, DOACK can be asserted any time after DOREQ asserts, DOREQ will be reasserted after DOACK is asserted.

Figure 9.4: Data out handshake waveforms from the view point of the computer (from the user manual of Adlink's PCI-7300A board).


DIREQ
DIACK
Input Data
Valid Data
t_4
t_5
t_3
t_1
t_2
$t_1 \geq 0$ns $t_2 \geq 0$ns $t_3 \geq 35$ns $t_4 \geq 10$ns $t_5 \geq 50$ns
DIREQ & DIACK Handshaking

Note: DIREQ must be asserted until DIACK asserts, DIACK will be asserted until DIREQ de-asserted.

Figure 9.5: Data in handshake waveforms from the view point of the computer (from the user manual of Adlink's PCI-7300A board).

Handshake component

The sequences involved in each handshake transaction (Fig. 9.4 and 9.5) rely on two states, respectively for listening and for transmitting. Hence, the handshake functionality may be efficiently implemented as FSMs on either sides, with potential additional intermediate idle states should timing require them.

Sending and receiving are anti-symmetrical which suggests to rely on the same component for both by simply exchanging the request signal (`do_req` or `di_req`) with the acknowledgement one (`do_ack` or `di_ack`). This would reduce the volume of VHDL and facilitate maintenance – a benefit not to be underestimated since, for simulation purposes, four handshake entities are needed, two within the FPGA but also two within the test bench.

Unfortunately, although seductive in principle, this approach is complex to design and calls for highly configurable components to cope with the different control logic and the timing on the two sides. Accordingly, different components are used and while the board's behaviour is best modelled as fully asynchronous, it was found simpler – and easier to integrate within the overall processing – to let the FPGA's function synchronously. Besides, since only one and four transactions should respectively occur on input and output per DCLK period and since the FPGA should be the limiting factor, the need arises for a supervisor (Handshake Manager) from which each component in the FPGA should receive its orders.

From Fig. 9.4 and 9.5, one can extract the "truth" table summarised in Tab. 9.9. The design should be straightforward and necessitate only simple two-state FSMs but intermediate idle states are required to enforce the timing requirements: $t_4\_di = 10ns$ and $t_5\_di = 50ns$ constraints on the test platform's side (see appendix A). A complete 16-bit output transaction then amounts to a minimum of 4 SCLK periods totalling 128 ns and the sequence of four 512 ns. With a DCLK period of 992 ns, 15 extra SCLK periods are available to absorb potential communication delays and represent a comfortable margin[10]. Fig. 9.6 illustrates the resulting design. Should these timing adjustments not be required for the FM, however, they could very easily be suppressed by eliminating the corresponding states.

Consider, as an example, the state of the input interface prior to a transaction. The handshake components on the PC and the FPGA sides are then in the `Wait` state and impose that both `do_req` and `do_ack` are low, as indeed they should be. When data becomes available for transmission, after at least the `t3_do` delay has elapsed, the PC's FSM asserts `do_req` to signal it to the FPGA and remains listening until `do_ack` is asserted by the FPGA (`Send` state). The system remains in this state until the Handshake manager decides it is willing to admit a new value on input, at which stage, `start` is asserted. Upon the next SCLK transition the FPGA's input FSM can then sample `do_req` and, finding it asserted, evolve to the `Rec` state which leads to copying the data to a local signal. During the next SCLK cycle, after this internal copy is well stabilised, it acknowledges reception for the data. This "unblocks" the PC's FSM and lets it return to the `Wait` state as a consequence (after a delay amounting to `t2_do`) so that `do_req` is de-asserted. In parallel, `do_ack` remains asserted for two SCLK periods[11] before the FPGA's FSM also returns to the `Wait` state and, thus, since both FSMs are back to the starting point a new transaction can ensue.

	FPGA			PC		
Input	↗ `do_ack`	when	`do_req` ‾	↗ `do_req`	when	`do_ack` _
	↘ `do_ack`		`do_req` _	↘ `do_req`		`do_ack` ‾
Output	↗ `di_req`	when	`di_ack` _	↗ `di_ack`	when	`di_req` ‾
	↘ `di_req`		`di_ack` ‾	↘ `di_ack`		`di_req` _

Table 9.9: Truth table for transitions of control signals in the handshake protocol.

Handshake manager

The Handshake manager is a supervisor managing and encapsulating all handshake transactions and offering internal data interfaces as if input and output were synchronous with DCLK:

1. Manage input transactions:
   - Allow one transfer per DCLK period.
   - Publish data to the cores DCLK-synchronously.
2. Manage output transactions:
   - Allow four transfers per DCLK period.

[10]Also allow using slow slew and accommodate analogue delays (appendix A).

[11]In practice, experience has shown that this delay is optimal to let the PC sample this signal and acknowledge the end of the transaction (see appendix A).

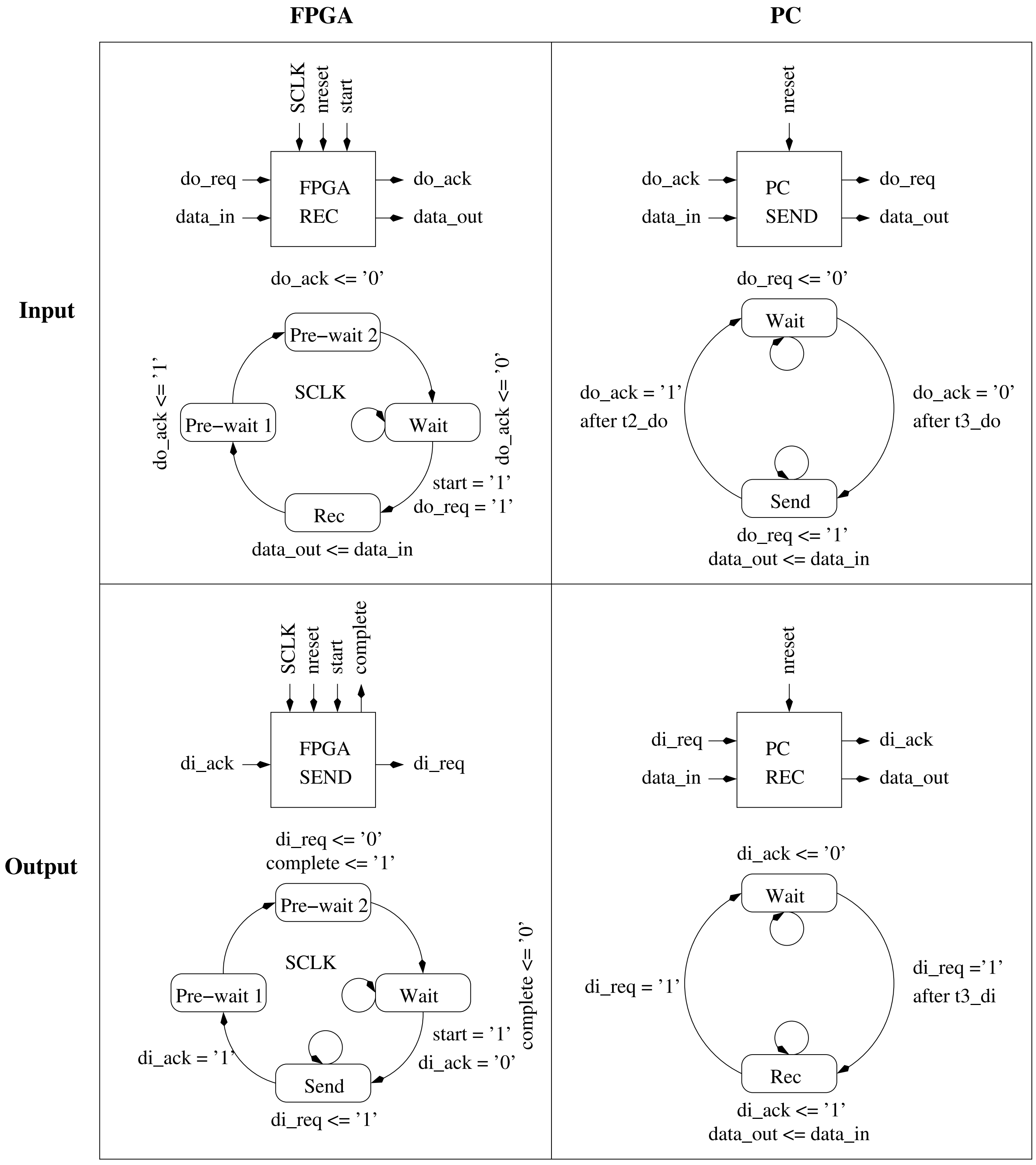


Figure 9.6: Handshake components and FSMs for input and output interfaces (t3_do is 50 ns and t3_di is 35 ns as per Fig. 9.4 and 9.5). The FPGA's output data port is managed by the handshake manager's FSM and is only updated between transactions (Fig. 9.7).

  - Enable transfers conditionally on the active core's `output` signal.
  - Fetch output data from the cores DCLK-synchronously.

3. Control execution:

  - Enable execution during the next DCLK period only if input and output transfers have been successful.
  - Activate processing or debug core depending on the $\overline{\text{debug}}$ signal.

With only one transaction on input, the command of the corresponding component can simply be a DCLK synchronous process while, on output, a FSM (Fig. 9.7) is introduced to force the behaviour into a sequence of four transactions.

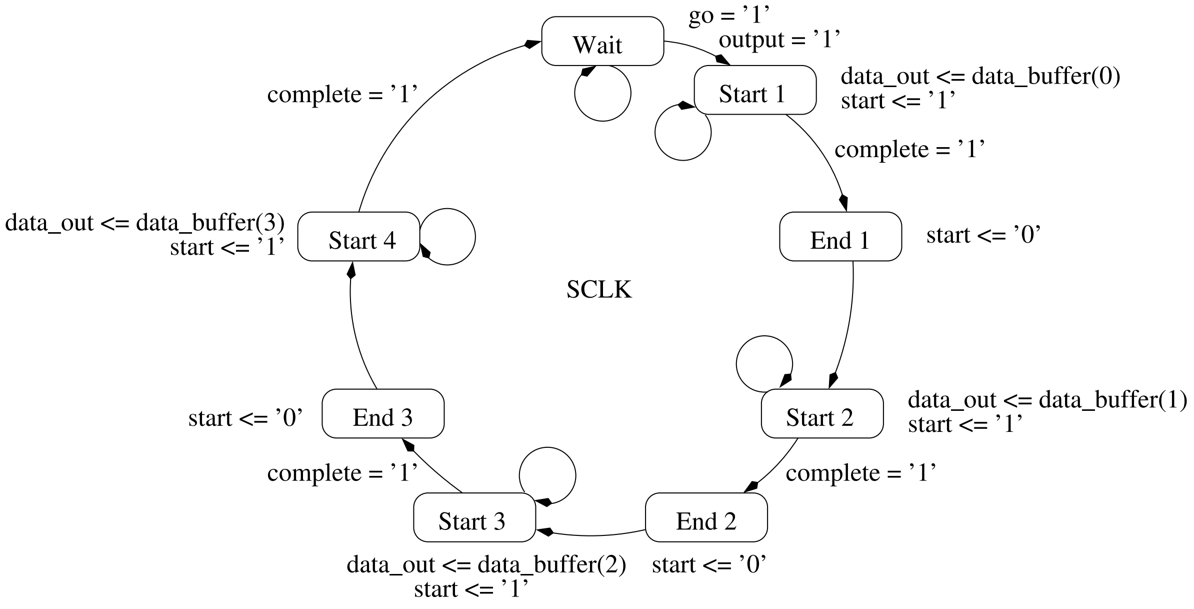


Figure 9.7: FSM for the output sequence of handshake transactions (four full cycles are carried out per DCLK period). The data_buffer signal samples the output from either cores on the rising edge of the DCLK if previous input and output transactions were successful.

Interactions with the handshake components are carried out through the `start` signals, which when low forbids the FSMs from leaving the `Wait` states. On input, upon success of the previous input and output transactions, it is determined by detecting rising edges on DCLK SCLK-synchronously. Conversely, on output, following Moore's style, the lower bit of the FSM's state register serves as the `start` signal.

The mechanism used for managing the data internally and controlling the execution of the cores relies on listening on the `do_ack` signal on the input side (which indicates that data has been successfully received) and on the output FSM (Fig. 9.7) which should have returned to the `Wait` state after the full sequence of output transactions. Fetching and publishing data internally as well as determining the enable signals for the processing and debug cores are then carried out according to the simple decision tree represented in Fig. 9.8.

9.4 Reset policy

9.4.1 Introduction

Use of synchronous or asynchronous reset is a rather controversial design question. Although both approaches fulfil the same functional need of bringing the hardware to a predetermined and hence fully deterministic state, the resulting netlists are so markedly different that the pros and cons of each lead to antagonistic stances. While [54] for instance highly recommends synchronous reset for avoiding metastability issues – something unfortunate indeed when trying to repair to a known state ! – [55] invokes the separate circuitry and the power-on reset in favour of the asynchronous approach.

The essential difference between the two rests in the fact that while synchronous reset happens within normal operation through extra logic inserted in the data path to force flip-flops to the desired state, the asynchronous one, on

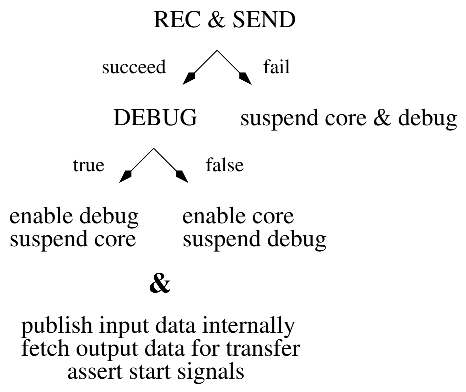


Figure 9.8: Decision tree for enabling processing and transfers.

the opposite, occurs through dedicated circuitry connected to the reset pins of the flip-flops which prevail. Tab. 9.10 summarises some of the consequences beside the intrinsic pros and cons of synchronous and asynchronous operation (first 3 lines).

	Synchronous	Asynchronous
1	100% synchronous design	metastability issues on removal
2	less sensitive to glitches	glitch sensitive
3	need for external synchronous logic	allows reset at power-on
4	needs synchronisation across clock domains	
5	inserted directly in data path	separate circuitry
6	more expensive resource-wise	low-level support: saves resources
7	complicates timing (logic levels)	independent timing analysis
8	allows propagation of reset from inputs	requires resetting all flip-flops
9	complex with clock gating and tristates	complex for internally-commanded reset

Table 9.10: Pros and cons of synchronous and asynchronous reset strategies.

9.4.2 Implementation

We discuss in this section the pros and cons enumerated in Tab. 9.10. The conclusion of this analysis consists in relying on an asynchronous reset with synchronised removal.

3, 9. The pipelined architecture adopted for the processing (chapter 12) together with the Handshake manager's capability to trigger or suspend execution imply that our design's need for reset is exclusively of a global nature. It is hence to be commanded only externally. While this would call for synchronous external logic in the synchronous case, it poses no particular difficulty in the other. Furthermore, adopting an active low reset logic translates into freezing the entire design into the reset state upon power-on. This is not much of a preoccupation for our test platform, but transient states during power-on have been known to cause subsystem failures and can be critical, for instance, for the command of the pyrotechnics used for releasing shutters or solar panels on satellites.

2. Due to the global nature of the reset, the absence of logic on the reset path means it is only a distribution tree and is not subject to glitches.

1. Although purely synchronous designs are often preferred for being simpler to design and more robust to perturbation, the simple and independent asynchronous reset circuitry allows for easily circumscribing the associated metastability problems. As reset prevails, it may be asserted any time but caution is mandatory when de-asserting it. Two classes of difficulties exist: the first related to violation of the reset recovery time (the minimum delay between the removal of reset and the next clock edge) which causes data integrity or metastability problems, the second to different sequential elements being restored to operation during different clock cycles due to the reset's propagation delays.

For both these reasons, Cummings et al. recommend forcing synchronous reset removal by the use of a synchroniser [55]. They propose a simple concept based on a pair of master flip-flops reset by the external asynchronous reset signal and which in turn control the internal reset tree. This approach has been adopted here.

4. With three clocks (CLK, SCLK and DCLK) synchronising the reset across the different domains would require deriving delayed copies and maintaining it asserted long enough to ensure that their intersection have a sufficient duration. This would ensure that all flip-flops are gradually brought to the reset state and that the design if globally reset at one point in time. The design should then be unleashed, so to speak, in a globally synchronous manner. One way of achieving this could consist in having clocks with constant phase relationships and designed so that all three have synchronous rising edges every now and then. The global removal of reset could then take advantage of one of these.

This synchronisation difficulty is automatically solved in the asynchronous case as the asserted reset naturally applies across all domains. Additionally, if its removal is synchronised with the fastest clock and its period exceeds both the reset recovery time and the greatest difference in propagation delays then none of the aforementioned difficulties can occur. This is guaranteed to be valid with CLK running at 125 MHz in a device with 350 MHz maximum pin-to-pin performance and when generating the various clocks by division (section 9.5).

5, 6, 7. With the synchronous strategy, it is necessary to be able to force the inputs of all flip-flops to the reset value. This translates in additional multiplexers or `AND` gates on the path with an associated cost in terms of resources and, because of the increased number of logic levels, in timing. Conversely, the asynchronous reset being connected directly to a dedicated pin of the flip-flop only uses routing resources – delay macros introduced to compensate differential propagation delays to have the reset signal reach all flip-flops "simultaneously" (ie. within a window whose duration results from the timing analysis).

Our design is more constrained in terms of resources than by the target operating frequency, hence it is principally this aspect which guides our choice. Another concern, however, for a demonstrator like ours is the ability to inspect the netlist generated as a result of synthesis for verification or optimisation purposes. As synchronous reset tends to clutter the data paths with extra cells, the asynchronous one is also preferred for readability reasons.

8, 9. The following two criteria have not been considered at this stage in the project because they are considered as design optimisations.

The first is related to the possibility, in the synchronous case of figure, to reset only the inputs directly then rely let the system run for a series of cycles to propagate the values to all internal flip-flops until all evolve to the desired state. This leads to reducing the amount of resources dedicated to the reset since the functional paths are used for its propagation, with a cost consisting essentially in a slower reset mechanism (whose duration then depends on the delay necessary for all flip-flops to reach a steady state).

The second concerns potential conflicts between the synchronous reset and, on one hand, tristate buses and, on the other hand, clock gating. The former, theoretically calling for power-on reset to avoid bus contention at start up, is not relevant in our case since anti-fuse devices do not have internal tristates. For the latter, the problem is that clock-gated parts of the circuit become inaccessible to the reset and call for a two stage reset procedure: re-enabling the gated circuit first before the reset can be addressed to its flip-flops.

9.5 Clocks

9.5.1 Functional

With the decision to implement the vast majority of the processing as a pipeline (chapter 12) and hence to fetch samples from the video buffer one by one, the arrival of a new value represents a natural clock in our problem, with a period of approximately $1 \mu s$ (DCLK). To permit external accesses to the SRAMs within one cycle, chapter 10 shows that two other clocks are necessary, a faster one (CLK) useful for formatting requests meeting the timing criteria and a slower one (SCLK) providing a synchronous interface to the cores.

These three clocks correspond to the main global needs, yet other clocks could be introduced with fine-tuning of the scheduling in mind. Operations like the euclidean division for Mode or accesses to the internal RAM (chapter 12) would benefit from these by reducing the timing constraints to the greatest possible extent. Although only of a local nature, these multiply the number of clock domains, not only making the synchronisation further intricate but also requiring many more cross-domain transfer mechanisms with the associated verification needs. It is possible, as an alternative, to rely on a faster clock and replace synchronous with sequential operation, typically in the form of simple FSMs. The intent is to exploit only a fraction of available periods and keep the logic idle the rest of the time, in an approach akin to clock gating. The downside relies in the timing requirements, then corresponding to the clock's frequency, potentially more constraining than would otherwise be necessary. However, taking into account this design's focus on using "slow" clocks as compared to the target technology's performances, this cost and the overhead associated to the FSMs are expected to remain within acceptable bounds. Using SCLK in replacement of potential slower clocks, this strategy has accordingly had our preference.

The specifications and relationships between clocks may be summarised as follows:

CLK
- use as main system clock
- $\ll$ max device frequency (350 MHz)
- originate from quartz oscillator (derive by multiplication on test platform)

SCLK
- exceed SRAM response delay: $>$ 11 ns + 2 CLK periods (chapter 10)
- allow 12 complete read operations (14 cycles in total) per DCLK/2 period for histogram (chapter 12)
- allow 5 write operations (7 cycles in total) per DCLK/2 period for histogram (chapter 12)
- generate by division from CLK (integer coefficient)
- maintain constant phase relationship with CLK (for reset)

DCLK
- approximate sample arrival times: $\lesssim$ 999.7965 ns (1 TDI period (0.9828 ms) / 983 samples)[12]
- generate by division from CLK (integer coefficient)
- maintain constant phase relationship with CLK (for reset)

9.5.2 Implementation

The clock periods are best determined through a top-down approach starting with the constraint imposed by the input sample interface: selecting the DCLK period as the multiple of the CLK period closest to 999 ns. Assuming it will be no shorter than 990 ns, the SCLK period should then be less than 35 ns, which in turn implies that CLK be less than 12 ns. A number of different solutions are then possible (CLK, SCLK, DCLK):

$$(12, 24, 996), (11, 33, 990), (10, 30, 990), (9, 27, 999), (8, 32, 992), (7, 28, 994), (6, 30, 996).$$

CLK and SCLK pose two different kinds of difficulties. The first as it is only used by the controllers leads to localised timing difficulties and power dissipation but at a frequency closer to the part's maximum performance. The second, on the opposite, is propagated throughout the design and relies on global routing resources for an overall valid timing but farther from the limit frequency. A first filter for selection has consisted in choosing as slow a joint (CLK, SCLK) pair as possible and the (11, 33, 990) and (8, 32, 992) configurations have been preferred in this sense. Although the first is somewhat better as far as power and timing are concerned, the second was nevertheless retained because of the simple relationship between SCLK and CLK (division by a power-of-two factor) – a decision which may be altered later on when optimising the design.

Considering that the Starter Kit which is part of the test platform is equipped with a 40 MHz quartz oscillator, the generation of the FPGA clocks fall into two different classes. A first stage is required for multiplying the 40 MHz to 125 MHz before the secondary clocks may be generated. As this is strongly related to the test platform and given the difficulty of reliably multiplying frequencies by hand, a CCC including a PLL is used for this purpose. It is worth noting that a FM would necessarily proceed otherwise since RTAX-S chips do not feature internal PLLs – which is the reason why this stage has been isolated in the `Framework` rather than being considered as part of the FPGA design (section 9.2.3).

Two possibilities can be imagined for the secondary clocks. A first one would consist in fully utilising the CCC instantiated for CLK and have it also generate SCLK and DCLK. This is unfortunately not possible since Actel's CCC only offers to divide CLK by coefficients limited to 5 bits – which is incompatible with the need to divide CLK (8 ns) by 124 to obtain DCLK (992 ns). A solution would be to cascade the ProASIC3E's second CCC to divide SCLK by 31 but this scheme is undesirable because it introduces delays between the clocks which cannot be jointly optimised because depending on two separate CCCs.

The second possibility consists in producing SCLK and DCLK "manually". It presents the advantage of portability to RTAX-s chips but calls for particular attention to obtain highly synchronised clocks. Considering indeed that any logic inserted between them necessarily introduces a delay, all three clocks should be derived directly from the same

[12]The handshake input interface and enabling mechanism allows for skipping DCLK cycles if data is not yet available in the video buffer.

resource. This would be easy were the dividers all powers-of-two since a single 7 bit counter would suffice; the clocks being typically the least significant, the second and the most significant bits. Due to the need to divide by 124, more elaborate logic is, unfortunately, necessary to implement a modulo 124 counter and masks are used to derive the clocks. For synchronisation purposes, a register barrier is accordingly inserted on output.

Clock	Quartz	CLK	SCLK	DCLK
Period (ns)	25	8	32	992
Frequency (MHz)	40	125	31.25	1.008
Source	-	Quartz	CLK	CLK
Multiplier	-	8	-	-
Divider	-	25	4	124
Generation	oscillator	PLL	counter	counter

Table 9.11: Clocks' characteristics and generation.

9.5.3 Clock domains

The set of cells controlled by transitions on the same clock signal represent what is called a "clock domain". This notion is relevant in FPGA design because the exact phase between any two cells within the same domain is both deterministic and predictable by calculating propagation delays resulting from the paths connecting them. This information is of critical importance to ensure setup or hold times are never violated. On the opposite, phase relationships cannot be assumed between different domains and passing data from one to another is subject to metastability as well as to synchronisation problems. The first is a consequence of varying delays between the edges of both clocks, much as transfers between synchronous and asynchronous circuitry. The second relates to verifying that each side has reached the proper state, ie. that the signals have the expected values, before the other side's sampling occurs and calls for special attention to causal relationships across domains.

Successful transfers, accordingly, necessitate detailed timing analysis or some type of mechanism to ensure that either these effects do not occur or that they can be mitigated. As a preliminary, it is of paramount importance to carefully segregate the use of the different clocks so as to avoid having to deal with highly intertwined domains. This is not only undesirable from a design point of view, as it makes it more complex and less reliable, but also from a performance point of view, as it implies additional control logic and translates into intermediate cycles for synchronisation. Tab. 9.12 summarises the different uses of the three clocks in our design based on the following simple heuristics:

- use of CLK should be kept to a minimum because its high frequency translates into increased power consumption and tighter timing constraints,
- DCLK, being associated to the rate at which a new sample becomes available, should command the various pipelines,
- SCLK, as an intermediate frequency clock, should be used wherever faster operation is desirable (mainly for shortening pipelines' depths).

Clock	Purpose
CLK	Formatting of the requests to the external asynchronous SRAMs.
SCLK	Synchronous interface to the external SRAMs. Sampling of control signals as part of the handshake interface. Introduction of sequential portions in the pipelines: eg. accesses to internal RAM, euclidean division in Mode, intermediate address calculations in Histogram, etc.
DCLK	Scheduling of the components' pipelines and control of components.

Table 9.12: Main uses of the three clocks.

The resulting segmentation in clock domains being then easily traceable, the interfaces can be instrumented to ensure reliability. Only two types of interfaces exist in our design as a result of the use of the three clocks: CLK

$\leftrightarrow$ SCLK and SCLK$\leftrightarrow$DCLK. The first occurs only within the Controllers and is, hence, documented in chapter 10. On the opposite, the second, being relative to the introduction of a sequential portion in a pipeline, can be handled systematically via a precise design methodology for the associated FSM.

The general rule is that the sequence is only carried out once per DCLK period and is hence triggered, not directly by DCLK's edges, but by tracking its edges through SCLK-synchronous sampling. The FSM is then allowed to leave its wait state only when comparison between the most recent sample and the previous one indicates a transition has occurred on DCLK. Thus, even if one of the samples is affected by stability issues, the delay introduced guarantees that DCLK-synchronous signals on input are stabilised and can be sampled by the SCLK domain.

The situation is even simpler on output because SCLK is faster than would strictly be required so the FSM returns to its wait state before the end of the DCLK period. This leaves ample time for the signals to settle before being sampled by the DCLK side.

9.6 Design for test

The need to make the design inspectable is central to the development of a demonstrator. This applies at every level of the design flow, be it as part of simulation, both before and after synthesis or the place and route stage, or as part of the tests carried out during execution. In all cases, the amount of verifications which can be carried out is conditioned primarily by the structural organisation of the design. At the simulation level, RTL descriptions are mandatory, not only because only signals are reported (as opposed to variables) but also because disentangling the results of synthesis, that is identifying key signals for inspection, imposes being able to predict its outcome to some extent. As for execution, because internal signals are then naturally inaccessible, the only information on the state of the FPGA comes from its ports. Due to the limited number of ports, this information is necessarily incomplete, thus calling for incremental inspection and the capability to reconfigure the ports. Actel and Synplicity propose an automatic tool by the name of "Identify", which instantiates a component inside the FPGA during synthesis and multiplexes selected signals to the JTAG ports, precisely in this philosophy[13]. Our design extends this in two respects presented below: a debug core and conditional instantiation of the components.

9.6.1 Debugging

Two separate yet exchangeable cores are placed at the heart of the design for processing and debugging respectively, with a single $\overline{\text{active}}$ one at any given moment in time. The intent is to have the processing core run most of the time, except when $\overline{\text{debug}}$ is asserted, which suspends it and hands control over to the debug core before normal operation is eventually resumed. To make the debug core a substitute to the processing one, it is built around identical input and output ports (Fig. 9.2 and 9.3). Operation of one or the other is commanded through the `enable_core` and `enable_dbg` signals by the Handshake manager while `Switch` grants exclusive access to the external SRAMs to one or the other. It is worth noting that the extra logic corresponding to this instrumentation can be kept to a minimum corresponding to the 19-bit incrementer instantiated within the debug core (approximately 200 cells in total).

To complement the possibilities offered by Identify, emph$\overline{\text{asis is}}$ placed on examining the content of the external SRAMs. Two different modes are possible depending on the $\overline{\text{upload}}$ signal:

- $\overline{\text{upload}}$ low: perform write accesses.
- $\overline{\text{upload}}$ high: perform read accesses.

These modes can be exploited in three main ways:

- to perform an end-of-chain test of the accessibility of all SRAM addresses to validate the test platform's interfaces without the need for re-synthesising the design (by writing data present on the input data port then reading the values to verify coherence),
- to initialise the SRAM's content at the beginning of a run (by writing zeros at all addresses to r reset histograms' bins),
- or to inspect the status of the processing by downloading the content of the SRAMs through the output data port.

Relying on the standard handshake procedure and on the Controllers, the debug core generates addresses with a simple counter and writes the inp$\overline{\text{ut data}}$ to all SRAMs at the rate of one per DCLK period. Integrity of the data can then be verified by de-asserting $\overline{\text{upload}}$, which causes the debug core to generate the same sequence of addresses to output the content of the SRAMs on the output data port.

[13]In the course of the setup of the test platform documented in appendix A, this mechanism has been tested but we have not found it to be reliable for debugging because the additional logic altered the behaviour of the system in unpredictable ways.

9.6.2 Conditional instantiation

Just as the simplified configuration has been defined due to limitations in the test platform, it is relevant for development purposes to generate a series of configurations instantiating only a fraction of components incrementally. This not only increases the simulation and synthesis speeds, thereby allowing more extensive tests to be carried out, but also allows for debugging each component individually or for inspecting their respective performances (in spite of synthesis's flattening of the design for global optimisations purposes). For validating execution, the most relevant data is then output depending on the configuration – another reason is that due the detailed static analysis performed as part of synthesis, it is mandatory to carefully defined the output data or else entire components are suppressed on the basis that their outputs are ignored.

VHDL's `for ... generate` construct is used to this end, in a manner resembling compilation directives in C, so that the design can be automatically configured with a number of boolean constants (Tab. 9.13). Additionally, a special mode is introduced to ensure strict bit-wise correspondence with the software implementation for validation of the outputs (section 12.3.5).

Configuration	Description	Output
	FPGA level	
`CONF_HDSK_ONLY`	input and output handshake	input data (16 bits)
`CONF_MEM_ONLY`	debug core, controllers, handshake	data read from SRAMs (32 bits)
`CONF_SKIP_DBG`	skip debug core and switch	variable
`CONF_SKIP_MGR`	skip controllers	variable
	Core processing level	
`CONF_CALIB_ONLY`	pre-calibration	calibrated sample values (16 bits)
`CONF_SKIP_CALIB`	skip pre-calibration	variable
`CONF_BUFFER_ONLY`	pre-calibration, buffering	calibrated sample values (16 bits)
`CONF_HIST_ONLY`	pre-calibration, buffering, histogram	max data words (36 bits)
`CONF_MODE_ONLY`	pre-calibration, buffering, histogram, mode	mode data words (21 bits)
`CONF_COMPLETE`	complete simplified design	selected samples' data (64 bits).
`CONF_COMPARE_PYXIS`	disable management of the equidistant case in mode for the incomplete hyperpixel (chapter 12)	variable

Table 9.13: Configurations accessible through the conditional synthesis directives (variable means the output depends on the other active configuration words).

9.6.3 Initialisation sequence

This section briefly presents the initialisation sequence required for proper operation.

1. Power on the FPGA ($\sim$ 1 ms).

   Both ProASIC3E and RTAX-S devices are immediately functional when powered up. This is due to the technologies (flash and anti-fuse) which are able to retain their configuration when the device is turned off and do not require to fetch it from some EPROM or EEPROM[14]. As the design has reset on power-on (since $\overline{\text{reset}}$ is active low) the FPGA immediately evolves towards the reset state.

   Power-up amounts to several μs with ramp rates of 0.25V/μs and start-up time is typically less than 1 ms for flash devices.

2. Initialise the internal RAM with Actel's Flash Pro application.

   The internal RAM content is lost in the absence of power and is random as a result of power on. In this phase, the address and data pairs are transmitted to the `UJTAG` macro (part of the unused IO tiles) through the JTAG port serially then written to the various addresses. Due to the serial interface, the transfer is rather slow.

3. Power on the handshake interface on the PC ($\overline{\text{reset}}$, $\overline{\text{debug}}$ and $\overline{\text{upload}}$ are then asserted).

   Asserting these signals on the PC places both sides in a coherent, idle and deterministic state.

[14]Additionally, the simplified configuration relies on hard-coded parameter values which implies that it needs not fetch them from external memory.

4. De-assert $\overline{\text{reset}}$ (after $\sim$ 1 s).

   $\overline{\text{reset}}$ should remain asserted until the PLL which is part of the CCC responsible for the generation of CLK has locked on. De-asserting $\overline{\text{reset}}$ allows the FPGA to evolve to a functional state and as a consequence the handshake interfaces become active. Since the reset only applies to the FPGA, the next stage consists in initialising the external SRAMs. Resetting all addresses to 0 is sufficient to this end and performed through the $\overline{\text{upload}}$ option of the $\overline{\text{debug}}$ mode. As the debug core accesses the two external SRAMs in parallel, the data set of sample values on the PC side must start with 2^{19} zeros (one for each address) and the system should remain in this mode for at least 2^{19} DCLK periods (ie. 520093696 ns $\sim$ 0.52 s)[15].

5. De-assert $\overline{\text{debug}}$ and $\overline{\text{upload}}$ (after $\sim$ 0.52 s).

   From this point on, the FPGA is in nominal operation. The data available on the input handshake interface at this moment in time should correspond to the first sample (AC = 0) of the first line (AL = 0). Initiating a new run requires repeating the steps from 2 onwards.

9.7 Conclusion

The architecture proposed in the chapter is built around DFT principles. These yield flexibility benefits, not only for the development of a simplified architecture but also for the validation of parts of the design. Besides, as the case of the initialisation of the external SRAMs illustrates, considering debugging facilities as part of the design allows for mutualizing resources between verification of the platform, debugging during execution and an initialisation procedure. This latter aspect while yielding significant development simplifications impacts on the test procedure by making it more complex as regards communications with the stimulating PC. As a consequence, a dedicated and automatic sequence should be included in the PC's software managing the interface to mitigate the error-prone nature of these transfers.

[15]More is also acceptable since some addresses would only be written multiple times then, assuming the number of zeros is coherent. Conversely, since all addresses are not used in chapter 12, it would be possible to limit initialisation to those used. Considering the duration of the complete initialisation and that, as the addresses used do not form a contiguous set, a more complex procedure should be devised in this case. This option has not been retained however.

Chapter 10

Asynchronous SRAM controller

Contents

10.1 Introduction

To fulfil the storage needs resulting from the delay introduced by the background estimation (sample values) and the elaboration of background statistics (histograms)[1], external memory modules have been introduced to complement the capabilities of the FPGA (section 9.2.1). To allow a wide exploration of the solution space and for maximum flexibility, the demonstrator was designed based on technology-compatible COTS components, with the intent to narrow down the setup to space-qualified devices and exact needs during a second phase. Hence, high speed 8 Mbit asynchronous SRAMs (10 ns IS61LV51216 chips from ISSI with 16-bit input/output data and 19-bit address ports) have been selected to avoid imposing unnecessary constraints by providing ample resources, flexible timing for R&D purposes and simplified management (no need for refresh cycles). From an architectural point of view, two separate devices have been introduced to permit concurrent accesses and are mounted on the test platform using ribbon cables to facilitate a pin reconfiguration should routing or cross-talk problems occur.

In spite of the above simplifications, the need for a memory controller remains, both to facilitate read and write accesses from the cores and to make the hardware description more generic by abstracting the idiosyncrasies of the selected components. With the intent to propagate the "ease of use" all the way to the processing core where the difficult part really stands, the controller is built for flexibility but, as for the hardware, all features will not necessarily be used in the final system.

This chapter describes the corresponding design, starting from a set of specifications which provide basis for the rationale, all the way to a brief performance assessment. The analysis developed in the next sections leads to write and read accesses composed respectively of two (receive and buffer request, write to SRAM) and three phases (receive and buffer request, read from SRAM, publish data) so that the controller will be subdivided in four parts responsible

[1]Refer to chapter 12 for the exact use made of these storage facilities.

for buffering, writing, reading and publishing data coordinated by a fifth one acting as a scheduler. Finally, timing as well as the asynchronous nature of the SRAMs will be shown to be the key complexity drivers.

10.2 Specifications

The design of the SRAM controller is guided by three key considerations: isolating SRAM-related operations to allow changes of design/technology and to decouple the developments for the cores from the interface with SRAMs, providing additional functionality to implement an "easy-to-use" interface to the SRAMs and integrating the asynchronous devices in the overall purely synchronous flow. The corresponding specifications are presented and briefly discussed below.

1. Decouple behaviour from the selected hardware.
   (a) Allow change of technology.
   Space-qualified SRAMs exist with access times of order 15 ns and timing parameters differing from those of IS61LV51216 chips[2], yet read and write cycles follow similar patterns. The selected read and write modes should hence be compatible with a variety of components and their implementation should depend on a minimum number of parameters.
   (b) Allow introduction of triple redundancy with majority voting.
   Devices designed to be radiation tolerant or rad-hard against a given total dose and latch-up immune below a certain threshold are not necessarily insensitive to SEU caused by high energy particles and which lead to data corruption by altering electrical states. Depending on the frequency of occurrence of such events and the reliability requirements, error protection may have to be introduced. Two types of strategies exist based respectively on EDAC procedures (error detection and correction encodings) and on TMR (replication of data and voting logic). The controller's design should hence be compatible with the insertion of logic levels corresponding to these tasks between the reception of data from the SRAMs and their transmission to the cores.
   (c) Define behaviour between requests and decouple from behaviour of SRAMs .
   Whether or not the decision is taken to "switch" the SRAMs off between accesses for reasons of power consumption, the controller should decouple the cores from the SRAMs between requests to filter transient states on the data bus leading to irrelevant values.
2. Define the interface to SRAMs.
   (a) Encapsulate the details of the read and write cycles.
   The controller should autonomously access the SRAMs through valid signal sequences to allow the cores to simply emit read or write requests with minimum formatting constraints. This additionally allows for altering the read or write cycles used in a manner transparent to the cores.
   (b) Allow requests to be made any time.
   The management of the SRAMs should focus on maximising their availability and minimise the maximum delay between the reception of a request and its completion. With this in mind, it should allow request to be received any time.
   (c) Read-out data should remain accessible to the cores during several consecutive access cycles.
   To avoid propagating SRAM-access constraints to the scheduling of the processing core, the read-out data should remain stable and available to the cores at least until the next one is received from the SRAMs. This imposes requesting data from the SRAM and fetching the value from the controller's port in the same order. Conversely, publishing read-out data for longer durations provides additional flexibility allowing the requests to be made and the data to be fetched independently when convenient as long as causality is respected.
   (d) Access SRAM modules 1 and 2 independently and identically. The interface should allow for concurrent accesses to the two SRAMs in coherence with the architecture's design logic. Data assignments are determined statically to best reduce the risks of access conflicts in the frame of the fixed scheduling of tasks. Accordingly, accesses to the two devices should only differ by a parameter which identifies to which chip they are made.
   (e) Perform validity checks.
   The controller should report invalid data and address content, as well as conflicting requests (simultaneous read/read, write/write and read/write requests). In faulty circumstances, an error message should be output

[2]See ATMEL's AT60142 chip for instance.

in simulation while the conflicting requests are ignored. Such errors should have high severity levels because in the frame of fixed scheduling no flexibility is introduced on the cores' side to suspend operation until the requests are fulfilled. While maintaining the previous valid values should enable the cores to continue, the resulting data will be irrelevant and the error must not remain unnoticed.

3. Allow integration in a purely synchronous frame.
   (a) Operate SRAMs synchronously.
   The cores being fully synchronous, read and write response times should be fully deterministic to allow fixed scheduling. In particular, a mechanism is required to ensure both availability of read-out data and of the SRAMs (for issuing new requests) after predictable delays.
   (b) Handle metastability issues.
   Potential metastability issues exist when retrieving data from the SRAM. An appropriate mitigation scheme should be introduced to achieve reliability performances coherent with the overall admissible failure rate.
   (c) Enforce equal read/write times.
   To simplify the synchronous cores' operations read and write response times should be equal.

10.3 Design

10.3.1 Discussion

Separating the cores from the SRAM interfaces (1a), both structurally and in terms of development, is fulfilled as soon as a controller is introduced as an intermediate layer. This is particularly true in VHDL since connecting SRAMs ports only to the output ports of the controller basically renders all other access methods physically impossible.

We discuss below a number of key design drivers in the light of the specifications given above. Architectural considerations come first and are followed by operational ones in the same order as the logical sequence which leads to each request being fulfilled.

- Dual clocks (2a, 2b, 3a, 3c, 1a)[3].
  The profiles of read (Fig. 10.2) and write (Fig. 10.3) cycles for SRAMs may be decomposed in two phases: one that consists in a valid setup of signals on the chips' ports which corresponds to requesting either a read or a write operation, another which accounts for the chips' response time and consists essentially in enforcing a delay before the next request can be made. Memory access operations are accordingly performed on two time scales: a finer one for formatting requests and a longer one for the maximum admissible request rate.

  While requirement (2a) could be fulfilled by a sequence of operations based on a single clock, the SRAMs' response time would then set the upper bound to accessible frequencies, leading to very slow accesses and idle SRAMs for a significant fraction of the time in violation of item (2b). Instead it is convenient to rely on two clocks (CLK and SCLK) at the cost of some additional complexity. This offers a way of transforming the asynchronous SRAMs into synchronously operated devices (3a). Besides, by defining the SCLK period as encompassing entire read or write cycles, the delays between the emission of requests and the availability of the results may be equated (3c). This delay, although not constant, is then bounded by one and two such clock periods depending on the offset between the reception of the request by the controller and the SCLK cycles. The operation can quite simply be guaranteed to have been performed by the end of the next clock period (3a).

  Clock generation is then submitted to the following three constraints:

  - define the SCLK period as greater than the duration of the longer of the complete read and write cycles,
  - define the SCLK period as a multiple of the CLK period,
  - define the CLK period with a value compatible with the synthesised logic to minimise the total duration of read and write cycles. A trade-off will need to be conducted between power consumption and timing constraints for synthesis and performances. This clock may well be the master clock, in the sense of the finest available clock from which all others are derived by division.

- Parallel controller (2d).
  The SRAMs being independent with accesses driven by the same clock (SCLK), instantiating two separate controller components allows for reducing the complexity of the latter's design, in particular as far as debugging is concerned. The intent is to assign data to each statically to optimise the possibility of concurrent accesses (chapter 12).

[3]The numbering refers to that of the specifications in section 10.2.

- Buffer requests (2b).
  Because we impose that operation be synchronous[4], accesses are only performed between the start and end of the SCLK cycles while request can be made during any CLK cycle by the cores. To allow requests to be formulated any time, they are buffered at the controller's input level, thus alleviating the constraint of maintaining stable signal values on the core's side until the start of the next SCLK cycle (2b). Considering that the core must first signal that a request is about to be made before the data is indeed transferred to the controller, two CLK cycles are necessary at this stage, although, theoretically speaking, each of the signal and the data need only be stable for one CLK cycle with a one-cycle offset.

  Although extension to a request queue allowing to make several requests during the same SCLK cycle is straightforward, only a simple buffering scheme in the sense above has been implemented. This is because this functionality would break the strict predictability of the delay necessary for the request to be processed. Indeed, depending on the number k of pending requests, one would need to wait for the end of the next k^{th} SCLK cycle before finding the read-out value available. This, instead of simplifying the scheduling of the cores would introduce additional complexity at their level so its implementation was not deemed desirable.

  To further simply the interface and considering that the overall scheduling should be determined statically, the cores need not have any knowledge of the status of the controller (ie. of the existence of a pending request). Instead, the controller is left to identify such conflicts, report them through assertions (with high severity levels) and simply ignore the requests in excess. In particular, requests arriving at a rate exceeding the management capabilities of the controller (two or more requests per SCLK cycle leading to a conflict at the buffer level) should trigger such reports.

- Genericity of scheduling (1a, 3a).
  Further analysis of the read and write cycles and associated parameters (Tab. 10.1) reveals that the constraints which apply are essentially of chronological nature. Analysis of the chronograms (fig. 10.2 and 10.3) shows that to be valid a read request must set $\overline{\text{WE}}$ to high before the address and, conversely, that a write request must set $\overline{\text{WE}}$ to high, then the address, then $\overline{\text{WE}}$ to low, then provide the data. Transitions on $\overline{\text{CE}}$ can basically be synchronous with those on the address when reading and with those on $\overline{\text{WE}}$ for writing. What essentially matters is the sequence of operations and not the delays between the steps themselves, insofar as signals are stabilised (1a). Such regularity is natural with synchronous operation of the controller with SCLK, with a cycle then decomposed in a setup phase during which signals are defined one after the other following the faster clock (CLK) and a wait phase until the SRAMs' response becomes available, which marks the end of the cycle and corresponds by construction to a rising edge on SCLK (3a).

Symbol	Parameter	Min	Max
t_{RC}	Read cycle time	10	-
t_{AA}	Address access time	-	10
t_{OHA}	Output hold time	3	-
t_{ACE}	$\overline{\text{CE}}$ Access time	-	10
t_{LZCE}	$\overline{\text{CE}}$ to Low-Z output	3	-
t_{HZCE}	$\overline{\text{CE}}$ to High-Z output	0	4

Symbol	Parameter	Min	Max
t_{WC}	Write cycle time	10	-
t_{SA}	Address setup time	8	-
t_{SCE}	$\overline{\text{CE}}$ to write end	8	-
t_{HA}	Address hold from write end	0	-
t_{AW}	Address setup time to write end	8	-
t_{PWE}	$\overline{\text{WE}}$ pulse width	10	-
t_{HZWE}	$\overline{\text{WE}}$ low to high-Z output	-	5
t_{LZWE}	$\overline{\text{WE}}$ high to low-Z output	2	-
t_{SD}	Data setup to write end	6	-
t_{HD}	Data hold from write end	0	-

Table 10.1: Read and write switching characteristics (from ISSI's IS61LV51216 technical documentation): durations are in nanoseconds and zero delays indicate causality constraints.

- Publish data (3b, 1b, 2e, 2c).
  The introduction of logic on the receiving side (from the SRAMs) is mandatory to control and reduce the risk of transfer errors due to metastability. The dual-clock operation described above allows for flexibly setting the depth of the cascade of registers introduced to this end simply by increasing the duration of the SCLK period (3b). In a similar manner, EDAC or comparison logic may well be introduced at this stage to protect SRAM content (1b)[5].

[4]Although the SRAMs *are* asynchronous.

[5]From a structural point of view in the case of TMR, the address bus may be shared between the SRAMs but the data ports would need to be replicated.

Publishing data in a controlled manner is similar to buffering the requests in the sense that it makes the interface to SRAMs more flexible. Data on the SRAM ports are only guaranteed to be valid at the end of the read and write cycles. Since read-out data can only come at the rate of two every SCLK cycle (one from each chip), the valid values may be held on the controller's output port until the end of the next read cycle. Introducing these registers allows the cores to fetch the data depending on needs, anytime after the end of the SCLK cycle following the request and before the completion of the following one.

While queueing requests was considered undesirable above, further buffering of outputs at the controller's level provides the possibility to access results in an order differing from that of the requests. Depending on whether or not this order is deterministic, selecting which buffer to fetch could be static or determined dynamically. The static approach was preferred resulting in a simple data FIFO queue for the benefit of simplicity and performance, but implementation of the second one, based for instance on tagging values with their address in the SRAMs would also be possible.

- Manage no request states (1c).

For reason of simplicity and because of pin assignment constraints on the FPGA (chapter 9), the retained operational mode is the one implying the minimum number of control signals. With selection between read and write modes performed through the $\overline{\text{WE}}$ port and with the mandatory address and data ports, the $\overline{\text{OE}}$, as well as the optional $\overline{\text{UB}}$ and $\overline{\text{LB}}$ ports can be directly connected to the ground. A first simple approach with constant low $\overline{\text{CE}}$ was devised but two drawbacks have led to revising this scheme. First, the SRAMs are always active which means that power consumption is high while accesses to the SRAMs are rather infrequent. This is something clearly not desirable for space. Second, this implies forcing operations to fall back to a dummy read operation on a pre-determined memory address filled with recognisable content and refraining from publishing this data in the absence of request. This leads to additional logic which may be avoided. Instead, because the use of $\overline{\text{CE}}$ to "turn" down the SRAMs between requests does not incur additional response delays, a second approach using this signal was implemented. Not only does it allow to reduce the power consumption, but since the SRAMs then set their data ports to the high impedance state, the need to filter out undesired transient states disappears, thus also leading to saving resources.

- Verify validity (2e).
  To ensure proper operation and facilitate debugging, the controller is the ideal place to perform a number of sanity checks. Reports can then be emitted during simulation and the component can fall back to valid configurations in undesirable cases. Two categories of events must be monitored: those relative to the requests and to the data.

Assuming the encoding of requests does not allow invalid ones (ie. all are either read or write), monitoring must identify conflicting configurations. In the absence of buffer queues, two particular cases of figure are expected: a request is received while one was already buffered or two requests are made simultaneously. Both are errors, hence to ensure they never happen, proper reporting must be ensured in the development phase.

In the first case, because of the management of incoming requests at the buffer level, the subsequent ones are simply left pending until they can in turn be buffered. This corresponds to increasing the buffer depth by using the port signals themselves to this end. However, requests buffered at this level run the risk of being overridden by other incoming ones without any possibility to automatically signal such events. As noted previously, the delay before completing this request would not meet the specification (3a) and this mechanism should be avoided although it remains possible nevertheless.

The second case corresponds to simultaneous read/read, write/write or read/write requests which must be signalled while continued operation can be ensured by disregarding the conflicting request. Relying on different control ports for read and write, read/write conflicts are easily identified. In this case the write request is ignored to keep the state of the SRAM unmodified. Detecting read/read and write/write conflicts is possible as they translate in ill-defined 'X' `std_logic` signals in simulation as a result of data or address having multiple drivers in the absence of any resolution mechanism.

Address and data verifications are preferably handled at the buffering level, thus allowing to report the error and ignore the corresponding request. A second case of figure concerns the data coming from the SRAM and is necessarily managed after its output signal is considered stabilised, that is after the logic introduced to reduce the metastability risk. The point at which data is being published on the output port towards the processing core is appropriate for this: if the data is valid it is published, otherwise the buffered values are left unaltered.

10.3.2 Interface

Wires to the various controllers are collected in buses to allow for uniformly accessing the different SRAMs through simple indexing by the SRAM ID. Separate buses (`CTRL_BUS`) are used for signalling read (`OP_READ`) or write (`OP_write`) requests as well as for addresses (`ADDRESS`) or data (`DATA`),

Considering the design guidelines of the previous section, overloaded procedures are provided to implement the targeted interface legibly in all contexts of interest. Two families exist managing respectively the scheduling of requests and the controller's ports. The former only synchronously signals the existence of a request for a given SRAM based on the current state of ports.

- Read request: CLK-synchronously signals (`OP_READ`) the existence of a new read request to the designated controller (`INDEX`).

```
procedure READ_SRAM (signal CLK     : in std_logic;
                     signal RESET   : in std_logic;
                     signal OP_READ : inout CTRL_BUS;
                     signal INDEX   : in integer range 0 to DEFAULT_NB_MEM - 1);
```

- Write request: CLK-synchronously signals (`OP_WRITE`) the existence of a new write request to the designated controller (`INDEX`).

```
procedure WRITE_SRAM (signal CLK      : in std_logic;
                      signal RESET    : in std_logic;
                      signal OP_WRITE : inout CTRL_BUS;
                      signal INDEX    : in integer range 0 to DEFAULT_NB_MEM - 1);
```

Conversely, the latter defines the state of ports but relies on the environment in which it is included to define the corresponding scheduling. A variety of prototypes are defined to fit the precise needs of each context (mainly in terms of data types). The two most straightforward examples are given below.

- Read request: sets address and triggers a transition on the control signal (`OP_READ`).

```
procedure READ_SRAM (ADDRESS_I        : in    ADDRESS;
                     signal ADDRESS_O : out   ADDRESS;
                     signal OP_READ   : inout CTRL);
```

- Write request: sets address and data and triggers a transition on the control signal (`OP_WRITE`).

```
procedure WRITE_SRAM (signal ADDRESS_I : in    ADDRESS;
                      signal ADDRESS_O : out   ADDRESS;
                      signal DATA_I    : in    DATA;
                      signal DATA_O    : out   DATA;
                      signal OP_WRITE  : inout CTRL);
```

The read-out data becomes available on the `DATA_OUT` bus after the end of the next SCLK cycle. This bus collects data from the two controllers and is accessible as `DATA_OUT(I)(J)` where `I` identifies the SRAM and `J` the position of the data to retrieve in the shift register serving as a FIFO buffer ("Publish data" in section 10.3.1). In the case of an SCLK-synchronous request, two full SCLK periods are necessary, which corresponds to the maximum latency.

10.3.3 General considerations

The memory controller falls in the control-logic category whose implementation is necessarily complex since it must handle a whole range of configurations while meeting timing and synchronisation constraints. Reliability is a key concern since the functionality it provides is susceptible of being used anywhere in the processing flow, thereby potentially widely distributing causes of malfunction. These reasons combined call for a decomposition in a series of small blocks or processes, in spite of the additional difficulty of managing communications between them, to segment the controller into simpler units and allow unit validation.

With this in mind, particular attention has been paid to segregating operations based on functional objectives, potentially at the expense of making optimisations more difficult at the synthesis level with the risk of increased areas and delays. An example of this rests in the fact that, in a first approach, each process maintained local copies of input port values to reduce the coupling to a few explicitly defined meeting points at a rather significant cost in registers (later reduced through detailed inspection of the timing).

To allow for saving clock cycles, control is achieved through transitions in signals rather that value: the commanded side samples the control signal at its own rate to detect rising and falling transitions through a XOR gate between the previous and the most recent samples[6]. This allows for exploiting both edges instead of, for instance, only rising ones with the need for intermediate steps for lowering the signal.

The functionality of the controller may be described in terms of a combination of request interpretation and of execution of a number of signal sequences. These latter parts, because of they follow fixed patterns determined by the types of the requests are natural candidates for FSMs with each state corresponding to one step in the setup of the signals on the SRAM ports. As a consequence, the controller is structured around a central control logic block which analyses requests and activates different state machines, respectively for the buffering and the formatting of read or write requests. Finally, the mechanism for publishing of read-out data may be more simply described as two processes, one CLK-synchronous for listening on the SRAM port and another, SCLK synchronous, for publishing the data.

This simple concept faces a major difficulty related to the above-mentioned dual-clock operation. Indeed, even with fine synchronisation between the clocks, rising edges cannot be expected to be concurrent. This translates into the material impossibility of designing FSMs with transitions triggered on CLK or SCLK depending on the state[7]. As discussed earlier[8], this leaves no other solution than to make them strictly CLK-synchronous or the access rate to the SRAMs would prove disastrously low. As SCLK was only introduced to enforce the SRAM access delay, the need is to ensure that the FSMs will remain in a stable state for a duration amounting to at least this delay (ie. not necessarily equal). This opens the possibility of asynchronous operation versus the SCLK implemented by detecting the latest high state not when it first occurs (rising edge on SCLK) but instead when sampled at each rising edge of CLK.

Though logical, the representations of read and write cycles in Fig. 10.2 and 10.3 are more complex than necessary as the sequences feature three phases: one for the setup of the signal (driven by CLK), one for awaiting the response of the SRAM (SCLK) and a last one for terminating the cycle (CLK). Instead, it is simpler to consider the following behaviour grouping two of these actions: finish the previous cycle and setup signals (CLK), then wait for the SRAM response (SCLK). The various FSMs follow this modelling in the sense that when signalled by the control logic, their first action deals with terminating the previous access cycle.

To conclude, it is worth noting that even in this simplified frame, the SCLK period does not strictly correspond to the SRAMs' response delay but also encompasses the additional transient steps at the beginning and end of the cycle. The resulting performances will be presented in section 10.5.1. Fig. 10.1 illustrates the overall structure, while Tab. 10.2 provides the truth tables of every FSM.

10.4 Structure

After a first version instrumenting each FSM with input buffers for reducing coupling with other processes and after validation of the proper behaviour, a detailed inspection of transitions on the address and data signals was carried out with optimisation in mind. Defining the FSMs to not only have synchronous transitions from one state to the next but also to sample inputs and publish outputs synchronously, it proved possible to mutualize the buffering between the Scheduler and the read and write FSMs, thereby saving a total of $2 \times (19 + 16) = 70$ flip-flops.

10.4.1 Scheduler

The Scheduler is the central part of the control logic. Being in charge of initiating the read or write request to the SRAM it is naturally an SCLK-synchronous process. Basically, depending on the state of the buffer FSM on rising edges of SCLK, it provokes transitions on one of two signals for either the read or write FSM to move away from the wait state to which both fallback at the end of their cycles.

This activity is complemented by indicating CLK-synchronously to the subsequent processes when new SCLK periods start. To this end, a separate CLK-synchronous process samples SCLK and maintains a `GO` signal by the comparison of the latest and previous samples so that `GO` is high only if a rising edge transition occurred on SCLK during the previous CLK period. This is performed at the controller level to mutualize the logic and to allow all FSMs to watch transitions by simply testing if `G) = '1'`.

The asynchronous nature of the ports implies that the SRAM is sensitive to glitches and hence that output signals must be spotless for reliable operation. To this end, the Scheduler has exclusive access to the SRAM ports. A first approach consisted in letting the read and write FSMs control the SRAM through shared tristate $\overline{\text{CE}}$ and $\overline{\text{WE}}$

[6]Just as the processes cannot be synchronous with two clocks as discussed below, one cannot track edges themselves but rather changes of value in an asynchronous manner.

[7]In fact, the synthesis tools will not allow it.

[8]See section 10.3.1.

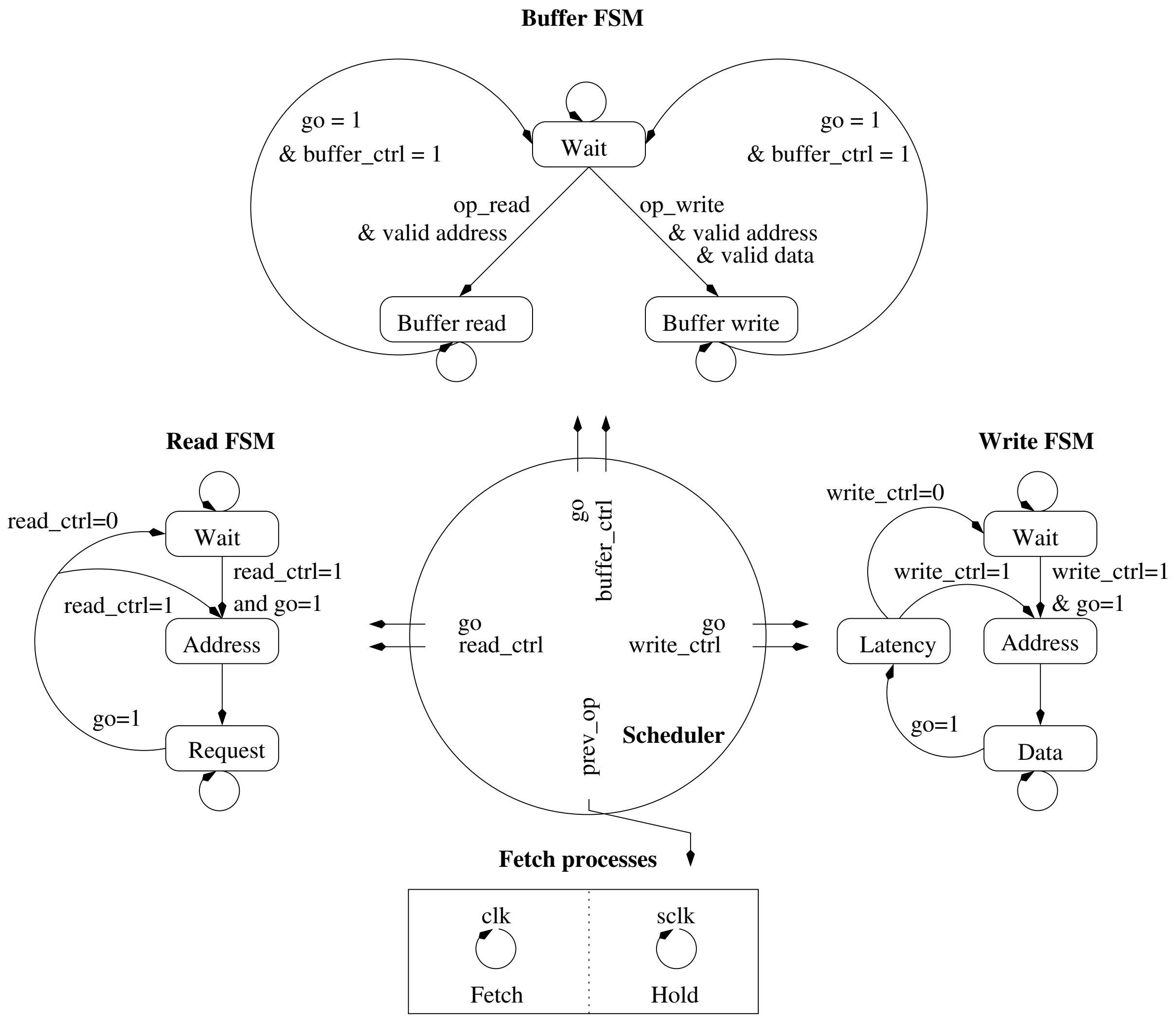


Figure 10.1: Overall controller architecture with buffer, read and write FSM and fetch processes. All FSM transitions are CLK-synchronous so the labels are Boolean conditions on asynchronous changes of values.

signals. In this logic, an idle SRAM translated in having these signals remain in high impedance states and the Scheduler interpreted them to switch the SRAM off by asserting $\overline{\text{CE}}$. The downside of using the signals themselves for communicating between FSMs is that it necessarily leads to transient states on the ports which disrupt SRAMs operations. Accordingly, the two functions were separated in a second approach by introducing intermediate signals manipulated by the FSMs ($\overline{\text{CE}}_r$, $\overline{\text{CE}}_w$ and $\overline{\text{WE}}_r$, $\overline{\text{WE}}_w$) and combined by the Scheduler (to yield $\overline{\text{CE}}$ and $\overline{\text{WE}}$) with XNOR gates to control the SRAM ports. The default status for the SRAM can also be defined at this level by placing the parts in standby mode between requests. Conversely, on the input side, the fetch mechanism allows for filtering out irrelevant signals on the data port (section 10.4.5).

It is worth noting here that the address port has been handled in a similar way – that is through separate `ADDRESS_OUT_R` and `ADDRESS_OUT_W` signals resolved at the Scheduler's level based on which FSM is in its "wait" state – but the reason this time is not related to glitches. It was found that synthesis generates complex logic for the resolution of accesses if a single tristate signal is used for the two FSMs so that, surprisingly, resolving accesses explicitly produces leanier logic. Finally, there is no need for a similar mechanism on the data port as the situation is different since read and write operations on this port are separate.

10.4.2 Buffer FSM

The buffer request FSM implements the interface with the requesting cores. It watches the `op_read` and `op_write` signals for transitions and stores the content of the `address_in` and `data_in` ports upon incoming requests. The state of the FSM indicates the presence of a pending request to the Scheduler. The reception of requests is performed at the highest possible rate, that is CLK-synchronously, while the buffering is constrained by the availability of the SRAM and is controlled by the Scheduler which unlocks the FSM with `buffer_ctrl` and allows it to return to the `Wait` state upon the next `GO`.

A purely synchronous Mealy FSM is used to simplify the logic by differentiating the buffer state depending on whether it is intended for reading or writing. As Fig. 10.1 illustrates, the two states are very similar and differ essentially by the nature of the verifications made on the data and address during the transition from the wait state. These being only meaningful simulation-wise, assert statements are simply inserted to report uninitialised ('U'), forcing unknown ('X'), high impedance ('Z') or weak bits ('W', 'L' or 'H') in either words.

As mentioned earlier, listening for transitions on the `read_op` or `write_op` is only carried out while the FSM is in the `Wait` state. As a consequence, a new request arriving while the FSM is in one of the other two will not be detected before it returns to the `Wait` state. Although not invalid strictly speaking, this behaviour is not desirable since successive requests could override each other during this interval. Such events are accordingly reported with warnings.

10.4.3 Read FSM

With the strategy selected for enforcing the SRAM response delay, the read cycle depicted Fig. 10.2 requires two steps. Leaving the wait state, $\overline{\mathrm{WE}}_r$ is first set to high, which terminates the previous cycle if it corresponded to writing, then the address of the data to be read-out is set, which triggers a new cycle if the previous one consisted in reading. Values of signals are then maintained stable until the next `GO` indicates that the SRAM's response delay is elapsed. If the subsequent request is also of type "read" and is already buffered at this stage, the FSM cycles to "Address" and "Request" without returning to "Wait" for maximum performance. Conversely, the absence of request or writing to the SRAM cause the FSM to sleep in the "Wait" state.

$\overline{\mathrm{CE}}_r$ control is easily achieved by setting it to low simultaneously to $\overline{\mathrm{WE}}_r$. It remains so until the return to the "Wait" state, at which stage it is set to high.

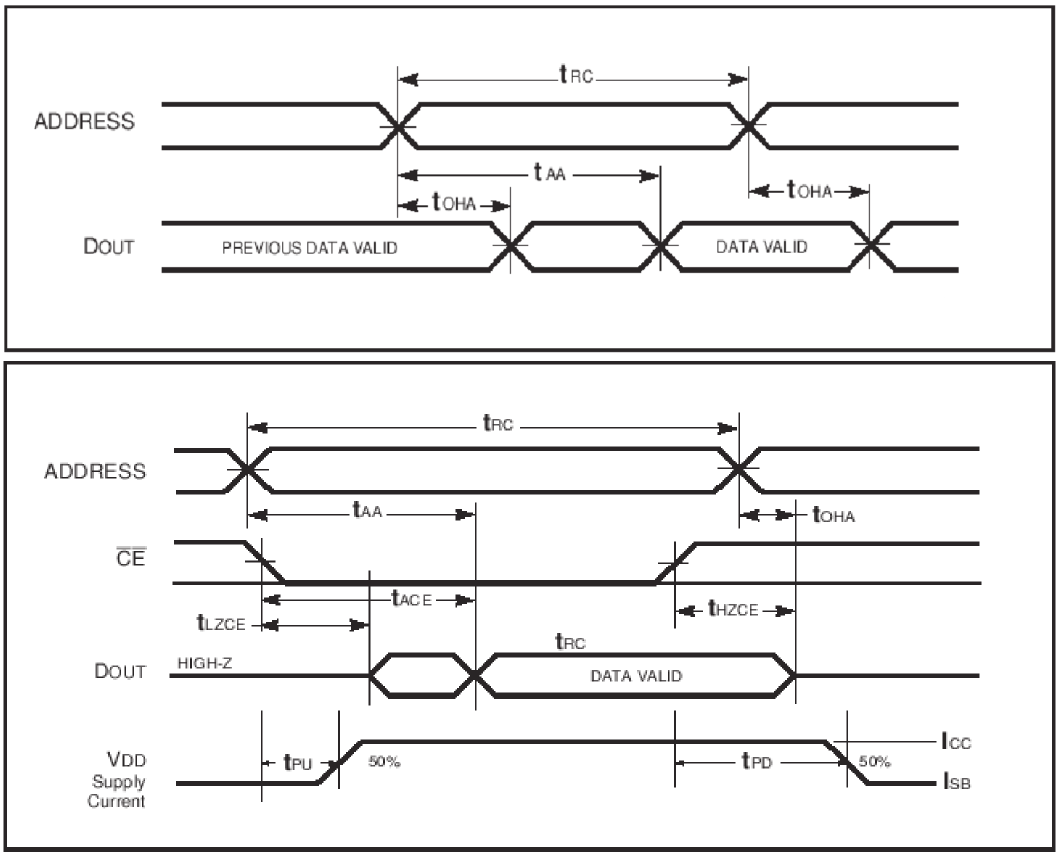


Figure 10.2: Read cycles with and without $\overline{\mathrm{CE}}$ control respectively (from ISSI's IS61LV51216 technical documentation). The waveforms assume $\overline{\mathrm{WE}}$ is high while $\overline{\mathrm{OE}}$, $\overline{\mathrm{UB}}$ or $\overline{\mathrm{LB}}$ have low input voltages.

10.4.4 Write FSM

The situation for write cycles is similar to that for read ones except that the sequence (Fig. 10.3) corresponding to the setup of signals requires three steps. While successive read cycles are triggered by transitions on the address, successive write ones require impulses on $\overline{\text{WE}}$. Hence the first step sets the address to which data is to be written and $\overline{\text{WE}}$ to high (which terminates the previous write cycle if any), the next one re-enables writing to the SRAM via a low $\overline{\text{WE}}$ before the value is published on the data port. As for reading, the FSM can cycle directly through "Address", "Data" and "Latency" if a sequence of consecutive write operations is requested.

As a consequence of the transition on the address bus and before one on the data bus, the SRAM would engage in a read cycle as per Fig. 10.2 because of its asynchronous nature. If the CLK period is slow enough compared to the SRAM's timing, this leads to undefined or read-out values being written on the data port by the SRAM. Otherwise, the transition on $\overline{\text{WE}}$ interrupts the read cycle during the previous cycle's data hold time and the state of the data port remains unchanged. In any case, the "fetch data" part of the controller ensures such transient states do not propagate to the cores via the data publishing policy.

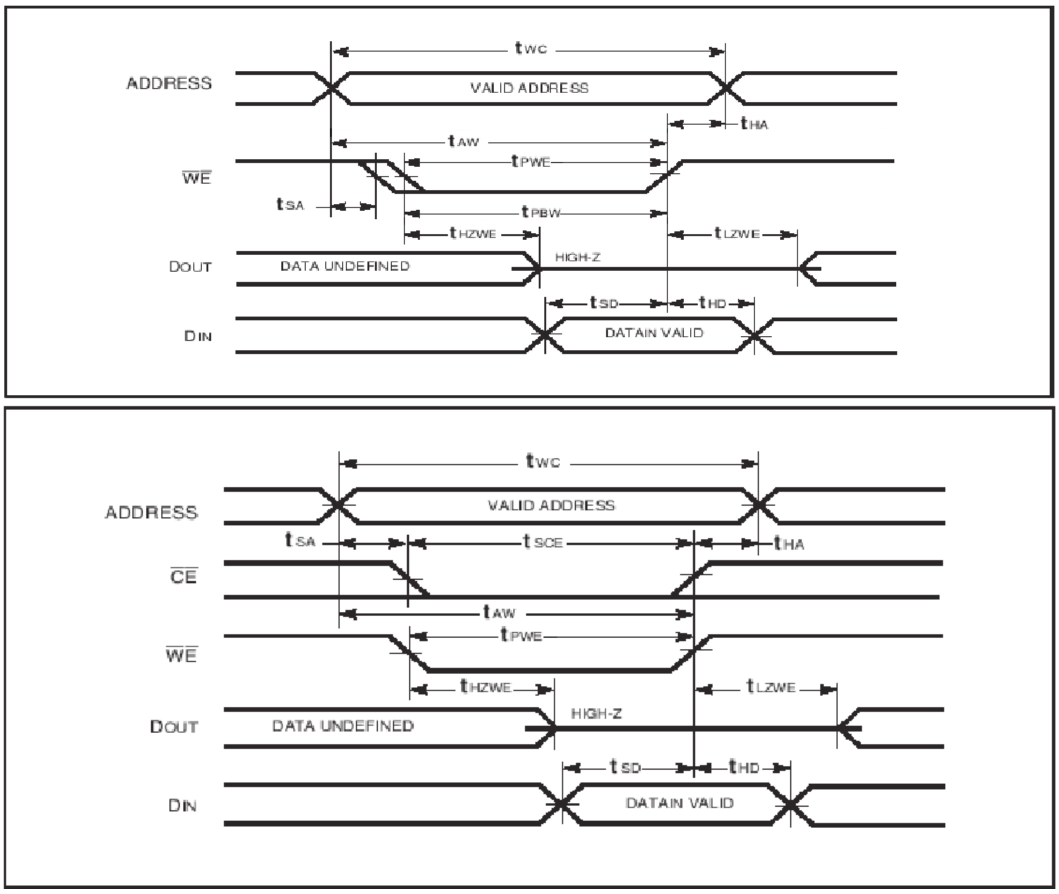


Figure 10.3: Write cycles with and without $\overline{\text{CE}}$ control (from ISSI's IS61LV51216 technical documentation): $\overline{\text{OE}}$, $\overline{\text{UB}}$ or $\overline{\text{LB}}$ have low input voltages.

10.4.5 Fetch processes

Fetching the data from the SRAM data port could well be performed through a FSM, however the combined need to handle metastability issues and to enforce equal read and write times (items 3b and 3c in section 10.2) and the fact that the write FSM has structurally more stages than the read one suggest to proceed differently. Classical ways of managing metastability at interfaces consist in inserting a cascade of registers, each of which imposes synchronisation constraints and reduces the probability of sampling a transient signal. When fetching data from the SRAM, inserting such registers in numbers equal to the difference of stages between the read and write FSMs solves potential metastability issues and leads to equal durations between the two types of accesses. As a consequence two processes are implemented, one which samples the SRAM data port CLK-synchronously and propagates values through the registers and one which samples the last register SCLK-synchronously to make the value accessible to the cores.

Buffer FSM	
Wait	
Buffer read	
prev_read	op_read
address_buffer_b	address_in
Buffer write	
prev_write	prev_write
data_buffer_b	data_in
address_buffer_b	address_in

Read FSM	
Wait	
$\overline{\text{CE}}_r$	1
$\overline{\text{WE}}_r$	1
address_out	Z
Address	
$\overline{\text{CE}}_r$	0
$\overline{\text{WE}}_r$	1
address_out	address_buffer_s
Request	
$\overline{\text{CE}}_r$	0
$\overline{\text{WE}}_r$	1

Write FSM			
Wait			
$\overline{\text{CE}}$	1	Data	
$\overline{\text{WE}}$	1	$\overline{\text{CE}}_w$	0
address_out	Z	$\overline{\text{WE}}_w$	0
data_io	Z	data_io	data_buffer_s
Address		Latency	
$\overline{\text{CE}}_w$	0	$\overline{\text{CE}}_w$	0
$\overline{\text{WE}}_w$	1	$\overline{\text{WE}}_w$	1
address_out	address_buffer_s		

Table 10.2: Buffer, read and write FSMs' signal and values. A unique value is provided for vectors with identical bits.

The need to publish only relevant values calls for inserting some logic before publishing the values. It suffices to impose the following two conditions:

- the previous request should be "read" (memorised in `PREV_OP`),
- the value should be valid.

The first condition is rather obvious after noticing that the data does not correspond to the latest request (in preparation) but to the previous one sent to the SRAM and answered after the response delay. The nature of the previous operation is easily memorised at the Scheduler's level. The second corresponds to validity checking (in the sense of `std_logic` simulation semantics) and refraining from propagating incorrect values to the cores.

Finally, introducing some flexibility in the way the cores access the read-out data (as of item 2c in section 10.2), leads to publishing data via a set of registers as discussed in section 10.3.1. As a matter of fact this facility is best organised as a FIFO queue, itself best implemented as a shift register. To fulfil the specification it is then necessary to refrain from replicating identical values in the shift register – which would push older ones out – in the absence of new read request. This is naturally guaranteed by the first condition above.

10.5 Implementation

This section dwells on the implementation itself. For reasons of simplification, the data and the address word sizes have been implemented as generic parameters (underlying the declaration of specific data subtypes built on the `std_logic_vector` type). This is convenient not only for debugging but, beyond, in making the output of synthesis understandable as compared to the full-fledged model. The output data buffer depth is also defined as generic to allow to scale this facility according to needs. Finally, all timing constraints are imposed by means of the CLK and SCLK signals so that the overall timing results from the CLK period and the divisor used.

10.5.1 Performances

This section derives the expected performances, in terms of number of SCLK and CLK cycles, both before a new request can be managed by the controller and before a request is considered fulfilled. The first impacts on the maximum achievable bandwidth obtained by pipelining requests at this rate, the second on the latency management to be carried out at the cores' level in the form of either wait states or data pre-fetching.

Although advertised with 10 ns access time, the IS61LV51216 chip cannot reach the theoretical limit of 1.6×10^9 bit/s for both read and write operations. While possible for successive "reads" commanded by changes of address every 10 ns, $\overline{\text{CE}}$ and $\overline{\text{WE}}$-commanded write cycles require at least $t_{SA} + t_{HZWE} + t_{SD} + t_{HA}$ which can be decomposed into an 11 ns write cycle and an additional $\overline{\text{WE}}$ pulse (which may be simultaneous with one on $\overline{\text{CE}}$) during which the address may be changed (at least 2 CLK periods).

Tab. 10.3 summarises the sequence of states and the number of clock cycles attached to read and write cycles. It renders explicit that SCLK must span the entire cycles and hence comprise for reading: the transition on address and the SRAM response delay, for writing: the $\overline{\text{WE}}$ pulse and address change and the delay. This sets the lower bound of accessible values for SCLK: the maximum between 10 ns (reading) and 11 ns + 2 CLK periods (writing). It is worth noting that this applies whether or not the system passes through the `Wait` states – the special transitions between "Request" and "Address" for reading or "Latency" and "Address" for writing have been introduced as shortcuts especially to avoid unnecessary intermediate delays.

Cycle	Sequence		
Read	Address $\xrightarrow{CLK}$	Request $\xrightarrow{SCLK}$	
Write	Address $\xrightarrow{CLK}$	Data $\xrightarrow{SCLK}$	Latency $\xrightarrow{CLK}$

Table 10.3: Timing constraints for read and write cycles (conditions are transitions on CLK and SCLK clock, not durations).

These constraints are taken into account for the general determination of clocks in chapter 9. The maximum data rate then follows from specification (3c) and amounts to $\frac{16}{P_{SCLK}}$ bit/s (59.6 MBytes/s with $P_{SCLK} = 32$ ns). The buffer FSM allows for chaining requests and reaching this limit.

Conversely, the presence of this buffer makes it more complex to predict when a given request will be fulfilled depending on the phase between its reception by the controller and SCLK. In normal operation, a request is buffered until the next SCLK cycle (the SRAM may be busy with another one at this stage), then a read or write cycle is initiated. Read and write operations can hence be considered completed by the end of the next SCLK period – for reading: the transition on the controller's data port coincides with the rising edge on SCLK, for writing: a new read or write cycle may begin. To avoid stability problems related to imperfect synchronisation between CLK and SCLK, requests received simultaneously with a rising edge of SCLK are nevertheless buffered. This case of figure corresponds to the one with maximum latency, which then amounts to two full SCLK periods. As mentioned earlier, a configuration is possible with further degraded latency which corresponds to emitting a request while one is already buffered. In this particular case, to be avoided, the delay before the fulfilment of requests would be variable but bounded by three SCLK periods.

10.6 SRAM VHDL model

For the development of the controller and to allow extensive debugging a VHDL model with detailed timing parameters was developed for the selected SRAMs[9]. A model from the Hamburg VHDL archive served as a basis but underwent in depth reconfiguration because of its lacking support for read and write cycles with $\overline{\text{OE}}$ permanently low as is the case in our setup. This model introduces two Boolean variables which control whether or not the chip is writing on the data ports (`read_active`) and if the data is valid or not (`read_valid`). The corresponding truth table has been established based on a broad range of timing considerations relative to $\overline{\text{WE}}$ and the address (see Tab. 10.5). Three different configurations are identified (I, II and III) and are defined in Tab. 10.4. After simplification, they lead to the Boolean equations (10.1), (10.2) and (10.3) below where the numbering refers to entries in the truth table. A simplified model of the influence of $\overline{\text{CE}}$ was added on top of it to make the complexity manageable according to the following three rules:

- impose t_{HZCE} and t_{LZCE} transitions for both low and high $\overline{\text{CE}}$ at all times,
- rising edges of $\overline{\text{CE}}$ terminate read and write cycles,
- falling edges of $\overline{\text{CE}}$ initiate read or write cycles depending on $\overline{\text{WE}}$.

State	`read_active`	`read_valid`
I	1	0
II	1	1
III	0	-

Table 10.4: Configurations and correspondence with read_active and read_valid variables.

To make expressions more readable, two notations are introduced:

- Subscripts designate stability durations: eg. $address_{OHA}$ is equivalent to `address'stable(tOHA)` in VHDL. The corresponding variable is a Boolean one valued `True` if the address signal has been stable for the t_{OHA} duration when it is evaluated, and `False` otherwise.
- Superscripts designate the result of tests: eg. $address^{1}_{OHA}$ is equivalent to the `address'stable(tOHA) = 1` comparison in VHDL and is `True` or `False` as the latter.

I if:

$$
\begin{aligned}
\overline{\mathrm{WE}}^{1}_{LZWE}.(\quad & \overline{\mathrm{WE}}^{0}_{OHA}.(\overbrace{A^{0}_{OHA}.\overline{\mathrm{WE}}^{0}}^{4} + \overbrace{A^{1}_{OHA}.\overline{\mathrm{WE}}^{1}.A^{0}_{AA}}^{20}) \\
& +\overline{\mathrm{WE}}^{1}_{OHA}.(A^{1}_{AA}.\overbrace{(\overline{\mathrm{WE}}^{0}_{HZWE}.\overline{\mathrm{WE}}^{0}}^{9} + \overbrace{\overline{\mathrm{WE}}^{0}_{RC}.\overline{\mathrm{WE}}^{1}}^{24\ 27}) \\
& \quad + A^{0}_{AA}.(\overline{\mathrm{WE}}^{0}.\overbrace{(A^{0}_{OHA}}^{7\ 10\ 13} + A^{1}_{OHA}.\overbrace{(\overline{\mathrm{WE}}^{0}_{HZWE}}^{8} + \overbrace{\overline{\mathrm{WE}}^{1}_{RC}}^{14}) + \overbrace{\overline{\mathrm{WE}}^{1}.A^{1}_{OHA}}^{23\ 26\ 29}))))
\end{aligned}
\tag{10.1}
$$

II if:

$$
\overline{\mathrm{WE}}^{0}.\overbrace{(\overline{\mathrm{WE}}^{0}_{LZWE}.A^{0}_{OHA}}^{1} + \overbrace{\overline{\mathrm{WE}}^{0}_{OHA}.A^{1}_{OHA}}^{2\ 3\ 5\ 6}) + \overline{\mathrm{WE}}^{1}.\overline{\mathrm{WE}}^{1}_{RC}.\overbrace{(A^{0}_{OHA}}^{28} + \overbrace{A^{1}_{AA}}^{30}) \tag{10.2}
$$

III if:

$$
\begin{aligned}
& \overbrace{\overline{\mathrm{WE}}^{1}_{RC}.\overline{\mathrm{WE}}^{0}.A^{1}_{AA}}^{15} \\
& + \overline{\mathrm{WE}}^{0}_{RC}.(\quad \overline{\mathrm{WE}}^{1}_{HZWE}.(\overbrace{A^{1}_{AA}.\overline{\mathrm{WE}}^{0}}^{12} + \overbrace{A^{0}_{OHA}.\overline{\mathrm{WE}}^{1}}^{25} + \overbrace{A^{1}_{OHA}.\overline{\mathrm{WE}}^{0}}^{11}) \\
& \qquad +\overline{\mathrm{WE}}^{0}_{HZWE}.(\overbrace{\overline{\mathrm{WE}}^{1}_{OHA}.\overline{\mathrm{WE}}^{1}.A^{0}_{OHA}}^{22} \\
& \qquad\quad + \overline{\mathrm{WE}}^{0}_{OHA}.\overline{\mathrm{WE}}^{1}.\overbrace{(\overline{\mathrm{WE}}^{0}_{LZWE}}^{16\ 17\ 18} + \overline{\mathrm{WE}}^{1}_{LZWE}.\overbrace{(A^{0}_{OHA}}^{19} + \overbrace{A^{1}_{AA}}^{21}))))
\end{aligned}
\tag{10.3}
$$

[9]The resulting detailed simulation framework is also of value for testing the design directly on RTAX devices for further validation of its feasibility.

#	$\overline{\text{WE}}$	$\overline{\text{WE}}_{LZWE}$	$\overline{\text{WE}}_{OHA}$	$\overline{\text{WE}}_{HZWE}$	$\overline{\text{WE}}_{RC}$	A_{OHA}	A_{AA}	read_active	read_valid	state	data	Note
1	0	0	0	0	0	0	0	1	1	II	data	Valid if A changed then $\overline{\text{WE}}$. Previous cycle was read: hold value.
2	0	0	0	0	0	1	0	1	1	II	data	A changed then $\overline{\text{WE}}$. Previous cycle was read: hold value for tOHA.
3	0	0	0	0	0	1	1	1	1	II	data	A changed then $\overline{\text{WE}}$. Previous cycle was read : hold value for tOHA.
4	0	1	0	0	0	0	0	1	0	I	X	A changed after $\overline{\text{WE}}$. Invalid write cycle.
5	0	1	0	0	0	1	0	1	1	II	data	A changed then $\overline{\text{WE}}$. Previous cycle was read: hold value for tOHA.
6	0	1	0	0	0	1	1	1	1	II	data	A changed then $\overline{\text{WE}}$. Previous cycle was read: hold value for tOHA.
7	0	1	1	0	0	0	0	1	0	I	X	A changed after $\overline{\text{WE}}$. Invalid write cycle.
8	0	1	1	0	0	1	0	1	0	I	X	Write cycle is past the hold time but not tHZWE, data is undefined.
9	0	1	1	0	0	1	1	1	0	I	X	Write cycle is past the hold time but not tHZWE, data is undefined.
10	0	1	1	1	0	0	0	1	0	I	X	A changed after $\overline{\text{WE}}$. Invalid write cycle.
11	0	1	1	1	0	1	0	0	-	III	Z	To be valid, A must have changed before $\overline{\text{WE}}$: listen on data port.
12	0	1	1	1	0	1	1	0	-	III	Z	Write cycle listens for data on data port (valid write sequence).
13	0	1	1	1	1	0	0	1	0	I	X	A changed after $\overline{\text{WE}}$. Invalid write cycle.
14	0	1	1	1	1	1	0	1	0	I	X	A changed after $\overline{\text{WE}}$. Invalid write cycle.
15	0	1	1	1	1	1	1	0	-	III	Z	Write cycle listens for data on data port (valid write sequence).
16	1	0	0	0	0	0	0	0	-	III	Z	Previous cycle was write, remains high impedance during tLZWE
17	1	0	0	0	0	1	0	0	-	III	Z	Previous cycle was write, remains high impedance during tLZWE
18	1	0	0	0	0	1	1	0	-	III	Z	Previous cycle was write, remains high impedance during tLZWE
19	1	1	0	0	0	0	0	0	-	III	Z	Previous cycle was write, beginning of a read cycle (A-controled).
20	1	1	0	0	0	1	0	1	0	I	X	A changed during previous write cycle. Invalid read cycle.
21	1	1	0	0	0	1	1	0	-	III	Z	Falls back to read after write cycle: hold period.
22	1	1	1	0	0	0	0	0	-	III	Z	Previous cycle: incomplete read (no hold), new read cycle.
23	1	1	1	0	0	1	0	1	0	I	X	$\overline{\text{WE}}$ and A changed simultaneously. If valid: intermediate stage.
24	1	1	1	0	0	1	1	1	0	I	X	Falls back to read after write cycle: intermediate stage (undefined).
25	1	1	1	1	0	0	0	0	-	III	Z	Previous cycle was write (nothing to hold), new read cycle.
26	1	1	1	1	0	1	0	1	0	I	X	Intermediate read stage.
27	1	1	1	1	0	1	1	1	0	I	X	Falls back to read after write cycle: intermediate stage.
28	1	1	1	1	1	0	0	1	1	II	data	Previous cycle was read, hold value (successful or not)
29	1	1	1	1	1	1	0	1	0	I	X	Intermediate read stage.
30	1	1	1	1	1	1	1	1	1	II	data	End of read cycle.

Table 10.5: Truth table of the SRAM model (A stands for address). Signal values are of the std_logic type ("Z" for high impedance, "X" for undefined, "-" for indifferent).

10.7 Conclusion

The design presented in this chapter for the memory controller is reliably instantiated multiple times as part of the FPGA's interfaces to the external SRAMs. With moderate timing constraints, resulting from the decision to rely massively on pipelined processing (chapter 12), the model is not, however, used in maximum performance conditions.

- To limit power consumption, the CLK period (8 ns) has been chosen comparable to the SRAM's response delay (11 ns), resulting in an overall duration of 32 ns (SCLK) (chapter 9). Although a faster clock would significantly improve throughput, another reason for selecting such figures is that they are in line with the performances of radiation-hardened space qualified SRAM devices.
- The processing pipeline (chapter 12) handles the SRAM accesses through FSMs which systematically retrieve the read-out values after the request. Buffering of outputs through a queue as described in section 10.3.1 is not necessary as a consequence. The corresponding resources are freed by setting the queue's depth to 1, thereby transforming the queue in a simple register and avoiding any over-design cost.
- Requests from the FPGA's processing cores are scheduled SCLK-synchronously, which while being simple from a control point of view, corresponds to the case of maximum latency (2 SCLK periods).

As an illustration, Fig. 10.4 shows the behaviour of the controller coupled with the SRAM model. This example is based on pre-synthesis simulation because post-synthesis behaviour is more difficult to interpret due to the shared resources and the removal of logic which result from synthesis's optimisations.

The SRAM-controller pair's behaviour compares to that of a synchronous flow-through "no-wait" SRAM but with reduced address to data delay since the controller is capable of exploiting the SRAM's output hold time (t_{OHA}) to retrieve the result of the previous access yet place the new data for the current one during the same SCLK period.

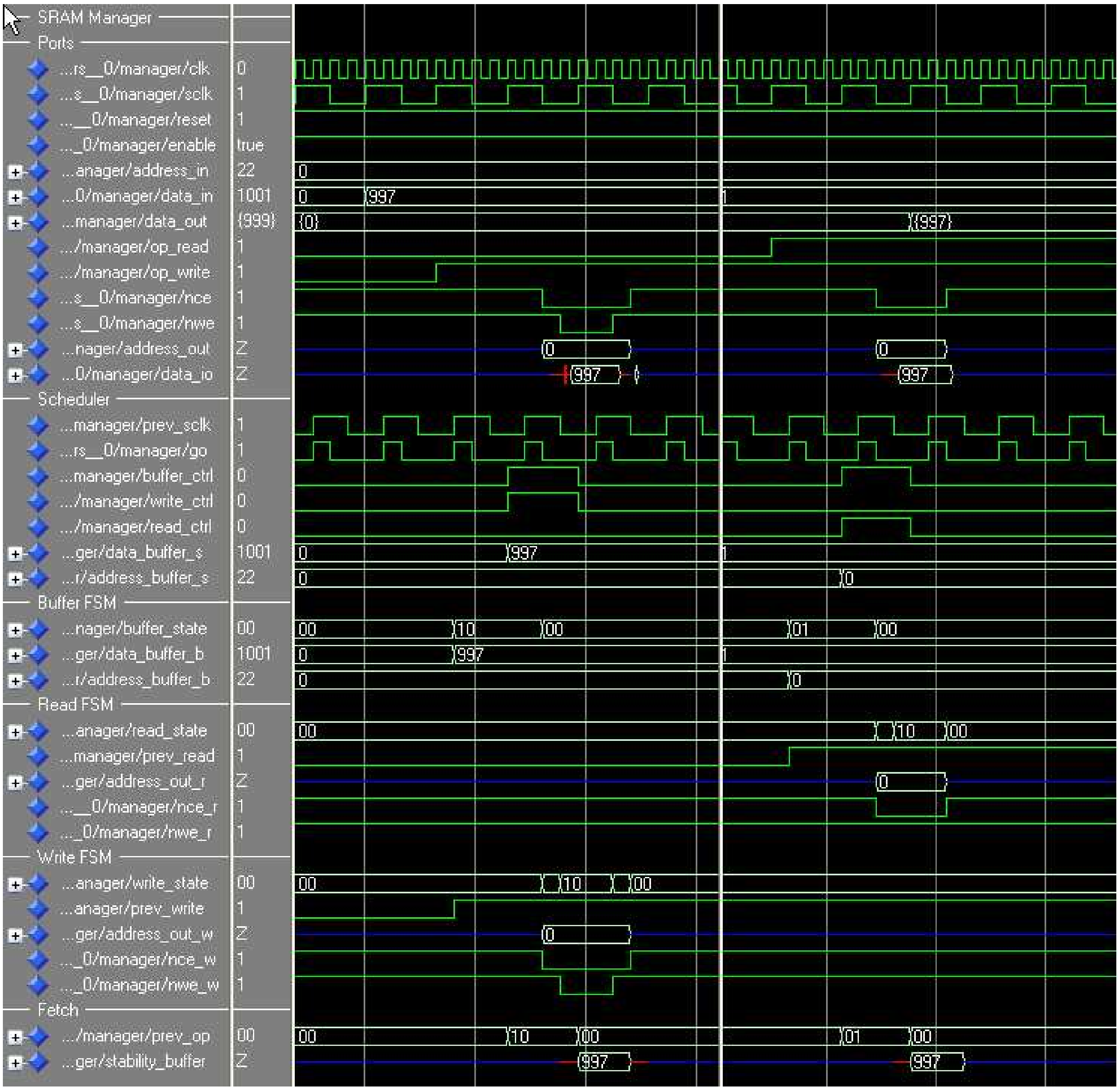


Figure 10.4: Example of write (left) and read (right) cycles at address 0 from the controller's point of view (pre-synthesis simulation with ModelSim). Signal names are legible on the left with truncated paths (they are all defined within the frame of the controller).

Chapter 11

Fixed-point formulae

Contents

11.1 Introduction

Motivation

Fixed-point arithmetics have, historically, been introduced in Pyxis with a threefold objective. In the context of the PDHE study, with the associated need to validate the results produced by the breadboard, emphasis was placed on achieving exact reproducibility across platforms to allow for bit-wise comparisons with a set of reference results. In this frame, floating-point arithmetics are the source of slightly varying results not only because of different machine epsilon values, but also because of the loss of associativity[1] and hence, of the compiler-dependent interpretation of statements[2]. Conversely, fixed-point arithmetics, being performed exactly, ensure the uniqueness of results on all machines.

Additionally, the structure of the on-board processing is such that the propagation of errors can lead from apparently minor changes in intermediate data to detectable effects when looking at final results. For instance, threshold effects for stars at the faint limit might artificially induce variable detection rates affecting both the processing load and the scientific performances, thereby making comparison of different platforms even more complex than it already is. On the opposite, arithmetics with fixed-point operators render the sources of errors explicit and considerably simplify the prediction of numerical errors and of their propagation.

Last but not least, the migration to fixed-point arithmetics is a prerequisite for porting the algorithms described in part II to a mixed hardware/software architecture. Although recent progress in technology has seen the introduction of processor cores within some FPGAs, thus greatly extending the range of designs susceptible of system-on-chip implementation, flight-qualified dedicated hardware (ASIC or FPGA) still lack FPUs. Hence, the arithmetics intended to run on them must be formulated with integers.

[1]The approximate nature of the results of floating-point operations makes the accuracy dependent on the order of operations.

[2]Not to mention the fact that, for optimisation purposes, numerous compilers do not enforce the ANSI C standard for type casts between integers and floating-point values (i.e. rounds-off instead of truncating) in type-heterogeneous statements.

Outline

After a general introduction to fixed-point arithmetics, the formalism used for deriving the errors is presented with a few prerequisites useful throughout the chapter. Then the different algorithms are examined one after the other in an order corresponding to the general sample flow. Particular emphasis has been placed on exact control over the numerical errors with the intent to output results with only valid bits. Accordingly, more than just formulae, entire procedures to be followed for the calculations are rather presented in what follows, each time accompanied by detailed numerical analysis.

11.2 Fixed-point arithmetics

Fixed-point arithmetics allow for approximating the results of real-numbered calculations with decimal numbers – or rather their equivalent in base 2. As opposed to floating-point numbers which use a flexible (sign, exponent, significand) decomposition, fixed-point ones rely on an adopted convention for the representation format which defines how many bits are used before the radix point and how many after, for instance:

$$\text{In 0.11.21 format: } 1035.031250486589 \rightarrow \begin{cases} \underbrace{}_{\text{no sign bit}} \; \underbrace{10000001011}_{\text{11 integer bits (1035)}} \quad \underbrace{000010000000000000001}_{\text{21 decimal bits (.031250476837)}} \\ \underbrace{-9.75184 \times 10^{-9}}_{\text{representation error } (<2^{-21})} \end{cases}$$

The format is implicit so that computations can be performed exactly and entirely within the integer unit with usual arithmetics.

Three different signed integer representations exist. The rather naive one given above, called "signed-magnitude", is in practice seldom used because it is essentially convenient for performing the $x \rightarrow -x$ unary operation, which is not of very frequent use, and leads to complex bit manipulations for other arithmetic operators. A second, called "1's complement" consists in representing negative integers by complementing each bit. It suffers from having two representations of 0, which requires two tests, respectively for +0 and -0. The "2's complement" representation solves this difficulty by adding 1 to the "1's complement". It is overwhelmingly used in both modern computer systems and dedicated hardware. For a negative value λ encoded as a signed integer on N bits ($\tilde{\lambda}$) the conversion to this format is given numerically by $\tilde{\lambda} = 2^N + \lambda$ – which allows for computing $x + y$ notwithstanding the signedness of x and y so long as operations are carried out modulo 2^N – or, bit-wise, by negating the bits of $|\lambda|$ then adding 1. Tab. 11.1 illustrates the encoding of signed integers on 4 bits.

0	0000	4	0100	-8	1000	-4	1100
1	0001	5	0101	-7	1001	-3	1101
2	0010	6	0110	-6	1010	-2	1110
3	0011	7	0111	-5	1011	-1	1111

Table 11.1: 2's complement representation on 4 bits.

11.2.1 Methods

As mentioned in the introduction, the fixed-point formulae can apply to both the hardware and the software. The methodologies available for performing the conversion differ slightly in the two cases since the software world offers additional possibilities as compared to programmable hardware. A first manual one, so-to-speak, corresponds to translating the corresponding expressions to a sequence of fixed-point ones, generally more numerous but based on a reduced number of arithmetic operators. The second one proceeds by emulating the behaviour of a FPU with a suitable library. The portions intended to run on the FPGA must therefore benefit from complete manual conversion to be compatible with synthesis. Conversely, for those intended as part of the real-time software, this quality constraint can be somewhat relaxed for simplicity insofar as the computational cost remains acceptable.

For the purpose of developing a full-fledged bit-comparable software model for validation of the hardware, all fixed-point formulae need nevertheless to be implemented in the software. Accordingly, a mix of both approaches is used in software depending on whether it is intended for simulation only or also to run in real life. Hardware and software should in theory be identical since fixed-point computations are exact and the same 2's complement representation is adopted by both. A subtle difference is, however, introduced by a flaw in the ANSI C norm. Explicit mention of the required adaptations will be made where necessary but, as this part of the software serves only as a reference for debugging, they have limited impact.

Fixed-point translation

Beside the usual operations (addition, subtraction, multiplication and Euclidean division[3]), fixed-point arithmetics extensively rely on bit shifts which allow for conversions between formats at a negligible cost. All conversions, represented as fractions or products in what follows, are in fact performed in this way. Similarly, a number of modulo operations are efficiently handled via bit-wise AND with the proper masks so that the most intensive constructs used are those induced by the original formula itself.

For the software implementation, the main difficulty in this exercise lies in the fixed word length which is either resolved by a trade-off between integer and decimal bits or by splitting the variable over several words, most often with a complexity and performance penalty. Conversely, in hardware, no such limit exists but the number of bits used impacts on the resources used or the number of cycles required to perform any given operation on the resulting signal. Hence, in both cases, the design is driven by the final precision requirements back-propagated through the calculation.

The following tasks have been translated to fixed-point:

- pre-calibration of sample values.
- computation of background mode estimates.
- interpolation of the background map.
- filtering of sample values.
- evaluation of the sample-wise SNR.
- estimation of the background flux.

Floating-point emulation

Reliable and simple emulation of an FPU can be obtained by using a subset of the SoftFloat IEC/IEEE Floating-point Arithmetic Package[4]. Only one key routine is mentioned in what follows, `mul32AndDivTo32`, which allows for multiplying 2 arbitrary 32-bit integers (A, B), then divide the result with a third (C) (to yield $\frac{A \times B}{C}$). This routine was developed for Pyxis via calls to the SoftFloat library and is capable of emulating an extension of the word length from 32 to 64 bits using two 32-bit words. With `U32` a 32-bit long unsigned int type, it reads:

```
/* Multiplies a by b then div by c to return it in 32 bits
   if result larger than 32 bits, returns 0xFFFFFFFF. */

U32 mul32AndDivTo32( U32 a, U32 b, U32 c )
{
    U32 z0, z1; int i=0;

    mul32To64( a, b, &z0, &z1 ) ;
    while ( (c<<i) < (1L<<31) ) i++;
    shift64Left( z0, z1, i, &z0, &z1 ) ; c <<= i;
    z1 = estimateDiv64To32( z0, z1, c) ;
    return z1;
}
```

Conversion of the formulae is then rather straightforward but does not provide any theoretical insight on the quality of the computation. Instead experimental measures have been carried out and show that the two approaches behave similarly in terms of the nature of the errors introduced (see section 11.2.2).

This method has been used for:

- the computation of the average background level.
- the calculation of barycentres,

Floating-point

For lack of time, some items intended to run in software (hence most likely on a CPU equipped with a FPU) and connected to the calculation of the object descriptors are still computed using floating-point arithmetics (the object-wise SNR, the misalignment criterion, the perimeter over area ratio).

[3]All divisions in the forthcoming formulae should be understood as Euclidean divisions with discarded remainders.

[4]Release 2b: http://www.jhauser.us/arithmetic/SoftFloat.html.

11.2.2 Calculation error

Fixed-point translation

FPUs have their own internal number representations and produce externally representable results at the end of calculations via round-off (using hidden bits for precision)[5] as of the IEEE754 standard [57]. Fixed-point arithmetics on the opposite proceed by truncation. The typical example of this is integer division which discards the remainder according to the ANSI C norm. The signedness of the corresponding error then depends on those of the operands.

In order to handle both paradigms on an equal basis, we choose to treat every numerical result as an estimate of the inaccessible real number – as if it were an estimator for a random variable. If the sequence of operations in the computation does not depend on the value of x, then this formulation has the advantage of allowing for easily deriving the properties of the resulting estimator. It may then be included in a more complete error budget, modelling the input distribution of x for instance, to produce overall quality quantifiers through a conditional decomposition of the sample space. With this in mind, the error analyses presented here only address the computational effects, not the statistical quality of the estimator being approximated.

Let Δ_b be the smallest representable increment and x be a uniformly distributed random variable, then we have in one and the other cases[6]:

- round-off (floating-point)

$$\text{Bias:} \quad E(\hat{x} - x) = 0 \tag{11.1}$$

$$\text{Variance:} \quad \sigma^2_{\hat{x}-x} = \frac{\Delta_b^2}{12} \tag{11.2}$$

- truncation (fixed-point)

$$\text{Bias:} \quad E(\hat{x} - x) = -\frac{\Delta_b}{2} \quad \text{for positive } x \tag{11.3}$$

$$= +\frac{\Delta_b}{2} \quad \text{for negative } x \tag{11.4}$$

$$\text{Variance:} \quad \sigma^2_{\hat{x}-x} = \frac{\Delta_b^2}{12} \tag{11.5}$$

Beside the fact that Δ_b equals the machine epsilon value for floating-point computations[7] but only the available precision for fixed-point ones [8], the most notable difference is that truncation leads to the introduction of a bias which depends on the sign of x.

The truncation bias can be corrected if the last precision bit is used for rounding-off:

$$\text{Positive } x: \quad \tfrac{\hat{x}+1}{2} \tag{11.6}$$

$$\text{Negative } x: \quad \tfrac{\hat{x}-1}{2} \tag{11.7}$$

$$0: \quad 0 \tag{11.8}$$

This corresponds exactly to trading accuracy (reduction of the bias) for precision (increase of the estimator's variance) whenever the achieved precision is limited by the word length. Cancelling the bias, however small, has the benefit of significantly simplifying the control of the propagation of errors and is, in most cases, a highly desirable operation in spite of the expense in terms of precision. This is particularly the case concerning the interpolated background whose contribution to the objects' flux results from the sum of potentially very many terms.

- truncation followed by round-off

$$\text{Bias:} \quad E(\hat{x} - x) = 0 \tag{11.9}$$

$$\text{Variance:} \quad \sigma^2_{\hat{x}-x} = \frac{\Delta_b^2}{3} \tag{11.10}$$

[5]Depending on the mode (extended precision or double precision mode) results can vary [56].

[6]Where $E()$ denotes the function extracting the integer part of its argument.

[7]$\sim 1.2.10^{-7}$ for floats and $\sim 2.2.10^{-16}$ for doubles on i386 processors.

[8]2^{-n} with n the number of allocated decimal bits.

The subtle difference between software and hardware previously mentioned stems from the fact that the behaviour when shifting bits right is not fully constrained in the ANSI C norm. Indeed, the bits introduced at the left end may either be 0s or copies of the previous most significant bit depending on the compiler and architecture. The latter option is easily understood after noticing that, even on the 2's complement representation, the sign of the integer can be determined by the most significant bit, so that replicating it intends to preserve the signedness. It is because this behaviour is not reliably portable, however, that when manipulating signed integers, division by a power of two has been preferred[9] – for which the ANSI norm specifies that this division must discard the remainder and so falls into the truncation case above. Conversely, for positive integers or concerning the hardware implementation, bit shifts are used for evident reason of simplicity and efficiency. It is equivalent to truncation except that the bias introduced does not depend on the sign of x:

- shift right (fixed-point)

$$\text{Bias:} \quad E(\hat{x} - x) \quad = -\frac{\Delta_b}{2} \tag{11.11}$$

$$\text{Variance:} \quad \sigma^2_{\hat{x}-x} \quad = \frac{\Delta_b^2}{12} \tag{11.12}$$

Distribution of errors

Of all arithmetical operations, only the division introduces errors because it discards the remainder of the euclidean division. Hence, in an effort to derive the final distribution of errors, any computation we develop will need to be decomposed in as many steps as there are divisions. The error introduced by each is of the form:

$$Y = a \times X[M] \tag{11.13}$$

where a will generally denotes the multiplicative coefficient introduced to achieve the desired precision on the quotient, M the divisor and X the dividend. Considering X and M as independent random variables but a fixed, since it relates to the selected fixed-point format, we discuss in this section the distribution of Y.

The case when $M = m$ is fixed and $X_i = Y_{i-1}$ is of particular interest here because it is the basis of most uniform random number generators, classically referred to as the multiplicative congruential method (see [58]).

Let f, g and h be the PDFs of X, M and Y respectively. Since the inverse image of a given value y by the mapping (11.13) is:

$$\{\frac{y + k \times m}{a} \quad / \quad \forall k \in \mathbb{Z}\} \tag{11.14}$$

then $h_{Y|M=m}$ reads:

$$h_{Y|M=m}(y) = \frac{1}{a}\sum_{k\in\mathbb{Z}} f(\frac{y + k \times m}{a}) \tag{11.15}$$

and h is:

$$h(y) = \int_{\mathbb{R}} h_{Y|M=m}(y)g(m)dm = \frac{1}{a}\int_{\mathbb{R}}\sum_{k\in\mathbb{Z}} f(\frac{y + k \times m}{a})g(m)dm \tag{11.16}$$

The mechanism leading to Y being generally approximately uniformly distributed is fairly intuitive and apparent in formula (11.15) first, when $M = m$ is fixed for simplicity, then in (11.16). Indeed, formula (11.15) expresses the fact that h is built by summing regularly distributed values of f, one in each each interval of length m. Geometrically speaking, this corresponds to slicing the graph of f along the limits of these intervals then summing them together to yield the graph of h on the $[0; m]$ interval. This is clearly an averaging process since normalisation is imposed by the fact that $\int_{\mathbb{R}} f = \int_{[0;m]} h = 1$. In our frame, the coefficient a introduced to increase the precision further strengthens this effect since it dilates the function f in the direction of the axis along which the slices are then taken. The larger the value of a, the slower the variations of $f(\frac{y}{a})$ on each of the intervals of length m, hence the effectiveness of the averaging is intuitively connected to the balance between m, a and k when f is k-lipschitz.

The effect of the distribution of M is also clear from (11.16). Assuming, as the limit and most favourable case, that (11.15) yields distributions $h_{Y|M=m}(y) = \frac{\mathbb{I}_{[0;m]}(y)}{m}$, the PDF h reduces to an average of indicator functions weighed by the distribution of M. Deviation from uniformity results from the propagation of boundary effects related to the variability of the length of the $[0; m]$ interval and, accordingly, relate directly to the dispersion of the distribution of M. The smaller the dispersion, the larger the value of $\tilde{m}$ for which Y can be regarded as uniform on the $[0, \tilde{m}]$ interval.

[9] This only applies to pre-calibration (section 11.3.1) in what follows.

Floating-point emulation

Using the previous lemma requires having exact knowledge of all the details of the calculations to allow for deriving the properties of the final estimator. This is clearly not applicable when relying on the SoftFloat library which has some built-in intelligence and whose calculations are hence not fully systematic. In its stead, one must determine these properties experimentally. This is simplified by the fact that in Pyxis 2.4.1, this library is exclusively used via calls to `mul32AndDivTo32`.

Let A, B and C be uniformly distributed random variables over the $[A_m, A_M]$, $[B_m, B_M]$ and $[C_m, C_M]$ domains. Noting that we have:

$$\begin{aligned} P(\frac{A \times B}{C} = \lambda) &= \int_{\mathbf{R}} \int_{\mathbf{R}} f_A(v).f_B(\frac{\lambda.u}{v}).f_C(u).du.dv \\ &= \frac{1}{A_M - A_m}.\frac{1}{B_M - B_m}.\frac{1}{C_M - C_m}.\int_{C_m}^{C_M} \int_{A_m}^{A_M} \mathbf{1}_{[B_m,B_M]}(\frac{\lambda.u}{v}).dv.du \end{aligned} \tag{11.17}$$

one can generate uniformly distributed $\lambda = \frac{A \times B}{C}$ ratios in $[\Lambda_m, \Lambda_M]$ as soon as $B_m \leq \frac{\Lambda_m \times C_m}{A_M}$ and $B_M \geq \frac{\Lambda_M \times C_M}{A_m}$.

Using this result, two different setup have been used to test the `mul32AndDivTo32` routine depending on whether $A \times B$ can be computed on a single 32-bit word or not. For each the differences between the floating-point and the fixed-point values have been studied as an approximation of the calculation error. Details and results are contained in Tab. 11.2

	No overflow	Overflow
Number of samples	500000	
$[\Lambda_m; \Lambda_M]$	$[1; 10^3]$	$[10^6; 10^9]$
$[A_m; A_M]$	$[10; 50]$	$[500; 10^4]$
$[B_m; B_M]$	$[1; 2.10^6]$	$[100; 10^9]$
$[C_m; C_M]$	$[100; 10^4]$	$[10; 100]$
minimum	0.0	0.0
maximum	0.99990	0.98990
mean	0.49941	0.46906
median	0.49898	0.46988
standard deviation	0.28901	0.29408
skewness	0.00145	0.01294
kurtosis	-1.20213	-1.19942
χ^2_{19}	16.966	2.2×10^4

Table 11.2: Statistics of the mul32AndDivTo32 error

Since a reference case can be obtained via a fixed-point translation which results in a uniform error distribution, comparison of the floating-point values (λ) to a uniform model has been performed via a simple χ^2_{19} test. Valid when no overflow occurs, which indicates that `mul32AndDivTo32` is then equivalent to simply performing truncation, it reveals the effect of the more sophisticated procedure put in practice otherwise. The histogram in Fig. 11.1 does plainly show that the algorithm used has a number of accumulation points: 0, 0.2, 1/4, 1/3, 0.4, 1/2, 0.6, 2/3, 3/4, 0.8.

11.3 Modules

We now present the arithmetics introduced module per module. For each we first explain the rationale behind the calculation (where the formulae have the usual real-numbered mathematical meaning). We then list the variables with associated ranges as they are represented[10] internally (eg. in ADU for sample values). Before finally introducing the fixed-point arithmetics – where notably the ratios should be interpreted as bit shifts in general or as Euclidean divisions otherwise – the format determination when necessary and the corresponding error analysis. The results of the computations, being considered as estimators, are denoted as $\widehat{result}$.

It is worth emphasising that the format indications, derived from the ranges in which the variables live, serve two key purposes. First and foremost, they are of use for the interpretation of the data and tracking the precision achieved at each stage in the computation. Besides, the integer bits, only indicative of the risk of overflow in software where

[10]We abbreviate integer with "i" and float with "f" as far as the intrinsic variable types are concerned – i.e. not their representations which are all either int or unsigned int depending on the need for a sign bit.

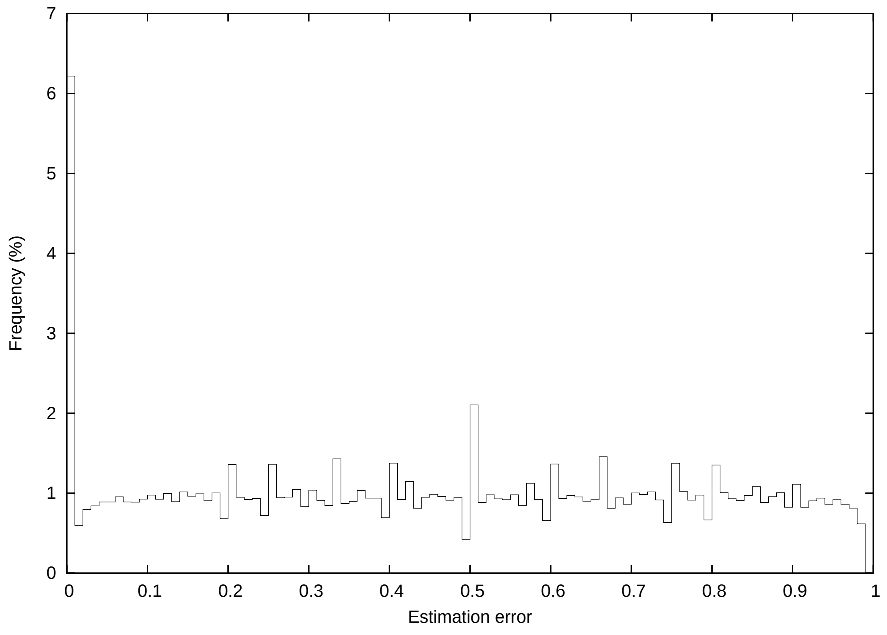


Figure 11.1: Error distribution of mul32AndDivTo32 when overflow occurs.

only 8, 16 or 32-bit variables can be declared natively, participate to the allocation of proper amount of resources in hardware and are hence, adjusted tightly as possible.

We restrict our discussion to the aspects calling for fixed-point arithmetics here. Please refer to chapters 4, 5 and 6 for an overall description of the underlying needs and designs. Real-numbered computations are required for four distinct objectives:

1. the calibration of the CCD data,
2. the computation of the sample-wise SNR,
3. the computation of the object-wise SNR,
4. the computation of the coordinates of the member samples' barycentre.

Hence, the precision constraints to take into account when devising a fixed-point analogue are those which apply at these levels.

11.3.1 Pre-calibration

- Rationale

  The CCD pixels gather photo-electrons in the $[0; 190000e-]$ range before handling them over to the electronic chain which converts them to an unsigned 16-bit integer value representing the sample count (in ADU). To allow for negative electron counts due to noise, conversion relies on:

$$samp_{e^-} = samp_{ADU} \times gain + zero \quad \text{(with } gain = 4.07 \text{ and } zero = -4070\text{)} \tag{11.18}$$

  Pre-calibration aims at correcting cosmetic defects (hot and dead pixels) and at ensuring uniform sensitivity across the samples and the detectors. The former relies on an average of neighbouring valid samples. The latter is the object of this section and is achieved by integrating the customary flat-field and dark current corrections in a linear model. These corrections are applied to each and every read-out value used by the on-board processing to determine the integer values which would be produced by an ideal CCD of identical characteristics. The resulting corrected sample value is only used on-board (only raw values should be downloaded). Current simulations only feature ideal CCDs as regards the PRNU and DSNU, hence the model developed below is currently only used to perform the identical transform !

The linear model reads:

$$a^j \times samp^j_{raw} + b^j \rightarrow samp^j_{cor} \tag{11.19}$$

and, with the set of assumptions collected in Tab. 11.3, generalises equation (4.1).

Variable	Meaning	Type	Domain
j	Sample identifier (depends on CCD, position and size)	i	
a^j	$\frac{\overline{flat\_field}}{flat\_field^j}$	f	$]0.9374; 1.0625[$
b^j	$dark^j \times a^j$	f	$]-4096; -4096[$
$samp^j_{raw}$	raw ADU sample value	i	$[0; 65535]$
$samp^j_{cor}$	corrected sample value	i	$[0; 65535]$

Table 11.3: Variables and assumptions for pre-calibration.

Lacking information on the expected ranges for the flat-field and dark effects over the course of the mission, and in particular in end-of-life conditions, the following analysis has been conducted based on those specified in [19] which correspond to procurement specifications. This document defines two categories of defective pixels, "white pixels" and "dark pixels" respectively characterised by the fact that they generate spurious charges at a rate greater than 20 electrons per second (at $-80\ ^oC$) or have a response lower than 80% of the local average. Additionally, columns featuring more than 100 such pixels (either white or dark) are considered defective. This specification is imprecise in the sense that it rather defines the behaviour of nominal pixels than that of defective ones. It must be interpreted at a number of levels to derive the domains in which the correction coefficients should be defined:

– It is assumed that the CCDs do not feature regional-scale pixel response biases – in the sense that the afore-mentioned local means are all equal to the sensitivity reported by the detector's quantum efficiency. Being mean values, however, they imply the existence of "bright pixels" whose response is above this value. The corresponding upper bound is unknown but it may be supposed that their impact on the sensitivity is symmetrical to that of dark pixels.

– Assuming the anti-blooming facility to work correctly, charge generation can at most lead to saturation. This kind of white pixels are not of interest here, however, since they would swamp the signal beyond any possibility of recovery so that columns with a single such pixel should be declared defective. A reference value to which performances will need to be compared can be inferred via the limit case, for which 100 white pixels lead to saturation. It sets the upper bound of charge generation at $1.93.10^6$ electrons per pixel and per second (1900 electrons per white pixel per TDI).

– Pixels covered with an aluminium mask (lines 1, 2, 5, 6, 9, and 10) are assumed to perfectly transmit the charge inherited from the previous pixel with the addition of those generated by the pixel itself. This translates into ignoring the pixel's response in the computation of the resulting overall sensitivity yet account for its potentially being a white pixel.

– The spatial distribution of the defective pixels is of importance as soon as gates are in use but in SM, the permanent activation of the first gate leads to a single case. Additionally, the threshold for considering columns defective should depend on the number of TDI stages effectively used for integration since the relevant parameter is the fraction of defective pixels as illustrated in Tab 11.4.

Tab. 11.4 illustrates the conditions which can be obtained under these assumptions with the limiting number of dark or bright pixels and no white ones. The bounds determined correspond to assuming that nominal pixels have sensitivities which average to the specification – and hence tend to outweigh the impact of the others for this aspect – yet generate charges at the rate of 20.5 electrons per second (white pixel threshold plus dark signal). Hence, these values should not be taken at face value but rather as being indicative of the ranges in which the coefficients vary, in particular insofar as they depend on gating. The conclusion is that the $]0.96; 1.04[$ range should at least be accessible for a^j in SM where the first gate is permanently active. Concerning b^j, the calibration of nominal pixel clearly does not impose strong design constraints since values remain small with limited needs in term of precision. Conversely, correcting the error introduced by white pixels calls for dynamics extending several times beyond $\frac{1900}{4.07} = 466.83$. The selected format (values in brackets in Tab. 11.3) is signed for technical

Gate	Pixels	Domain for a^j	Pixels	Spurious electrons	Min b^j
-	4494	$]0.97, 1.03[$	4500	90.66	-22.78
1	2900	$]0.96, 1.04[$	2906	58.55	-14.89
2	2048	$]0.95, 1.05[$	2054	41.38	-10.67
3	1024	$]0.90, 1.10[$	1030	20.75	-5.60
4	512	$]0.80, 1.20[$	518	10.44	-3.07
5	256	$]0.60, 1.40[$	262	5.28	-1.81
6	128	$]0.21, 1.79[$	134	2.70	-1.19
7	64	$]0, 2[$	70	1.41	-1.19
8	32	$]0, 2[$	38	0.77	-0.71
9	16	$]0, 2[$	22	0.44	-0.44
10	8	$]0, 2[$	14	0.28	-0.28
11	4	$]0, 2[$	9	0.18	-0.18
12	2	$]0, 2[$	5	0.10	-0.10

Table 11.4: Worst case sensitivity (100 dark pixels (with zero sensitivity) or bright pixels (with symmetrical effects) are assumed) and charge generation (dark current in the absence of white pixel) budgets as a function of gates.

reasons although only charge generation is mentioned in [19]. This would allow for handling systematic charge loss if it were to occur. Besides, the accessible range of values makes it possible to compensate the contribution of 8 reference white pixels with margin for others.

- Fixed-point arithmetics

The key element for the design of this fixed-point computation is to recognise that the maximum error has a lower bound which is imposed by the decision to produce integer ADU values in the end. This lower bound is asymptotically reached when the precision increases, that is as the decimal representation approaches the corresponding real number. With finite precision, the design can only attempt to minimise the additional term resulting from the computational errors within the constraints imposed. In the frame of an implementation on dedicated electronics (FPGA or ASIC), the number of bits available for the computation itself is only limited by the number of gates of the device. Conversely, the large parameter sets (one a^j and one b^j coefficients per CCD and sample position in SM) impose storage requirements. To either save the FPGA's internal memory or to allow using the external SRAM devices for this purpose, we impose that each of the a^j and b^j coefficients be encoded with a maximum of 16 bits. Hence, a trade-off will need to be made for the b^j terms between the contribution to the error and the range of values available. The preferred encoding (in 1.12.2 format) will be justified below by the error analysis and leads to the range indicated in brackets in Tab. 11.3.

Dropping the j superscript (which serves to indicate that coefficient values depend on the sample considered), it is preferable to rewrite (11.19) in the following way to reduce the number of bits involved in the computation:

$$samp_{raw} + ((a - 1) \times samp_{raw} + b) \rightarrow samp_{cor} \tag{11.20}$$

Considering that $a \in]0.9375; 1.0625[$, $(a - 1) \in] - 0.0625; 0.0625[$ can be encoded in 1.0.18 format in spite of the 16-bit limitation. This format is particularly suitable to ensure that $(a \hat{-} 1) \times samp_{raw}$ will always be correct at the ADU level and reserves two decimal bits for the subsequent summation with $\hat{b}$ and for round-off. The calculation is then [11]:

$$\underbrace{co\hat{r}r_1}_{1.12.18} = \underbrace{(a \hat{-} 1)}_{1.0.18} \times \underbrace{samp_{raw}}_{0.16.0} \tag{11.21}$$

since the integer part of $co\hat{r}r_1$ is bounded by $0.0625 \times 2^{16} = 2^{12}$. The charge generation specification does not require introducing a sign bit for $\hat{b}$, however, the operation carried out by formula (11.22), being a signed addition, calls for signed operands. Then, let N be the number of decimal bits used to encode $\hat{b}$ and $M = max(12, 14-N)$,

[11]The warning issued previously concerning the use of bit shift operators with negative integer applies here particularly.

the charge generation correction writes[12]:

$$\underbrace{\hat{corr_2}}_{1.M+1.18} = \underbrace{\hat{corr_1}}_{1.12.18} + \overbrace{\underbrace{\hat{b}}_{1.14-N.N} \times 2^{18-N}}^{1.14-N.18} \tag{11.22}$$

Since, $\hat{corr_2}$ is signed, round-off to the nearest ADU takes place differently in software and hardware as explained in section 11.2.2. In hardware, the bias is constant so let $s = 1$. Conversely, in software, the bias introduced by division depends on sign, so we use instead $s = sign(\hat{corr_2}) = \begin{cases} -1 & & < 0 \\ 0 & \text{if} & \hat{corr_2} = 0 \\ 1 & & > 0 \end{cases}$.

– if $|\hat{corr_2}| \equiv 2^{17}[2^{18}]$, let k be a uniform binary random variable, then

$$\underbrace{\hat{corr}}_{1.M+1.0} = \frac{\overbrace{\hat{corr_2} + s \times 2^{17}}^{1.M+1.18}}{2^{18}} - s \times k \tag{11.23}$$

– otherwise:

$$\underbrace{\hat{corr}}_{1.M+1.0} = \frac{\overbrace{\hat{corr_2} + s \times 2^{17}}^{1.M+1.18}}{2^{18}} \tag{11.24}$$

The final result is obtained with special attention to underflow[13] or values exceeding the saturation value according to:

$$\underbrace{\hat{samp}_{cor}}_{0.16.0} = \begin{cases} 0 & \text{if negative,} \\ sat & \text{if } \geq sat, \\ \underbrace{samp_{raw}}_{0.16.0} + \underbrace{\hat{corr}}_{1.M+1.0} & \text{otherwise.} \end{cases} \tag{11.25}$$

- Error analysis

This error analysis bears much resemblance with the one carried out in section 11.3.2 below because the computation features a sum of variables and a truncation step[14]. Similarly, it leads to setting the value of the parameter N.

As elsewhere in this document, we consider the input value $samp_{raw}$ as exact. This is of particular importance here since the performance estimates derived in this section will depend explicitly on the value of $samp_{raw}$[15]. Conversely, the coefficient values $\hat{a}$ and $\hat{b}$ represent estimates of the inaccessible real-numbered parameters of the computation. We assume a and b to be distributed uniformly and $\hat{a}$ and $\hat{b}$ rounded-off to the nearest 18 and N bits. Hence errors are themselves distributed uniformly on $]-2^{-19}; 2^{-19}[$ and $]-R; R[=]-2^{-N-1}; 2^{-N-1}[$ respectively. Accordingly, $\hat{corr_1} - corr_1$ is bias-free and distributed uniformly on $]-S; S[=]-2^{-19}samp_{raw}; 2^{-19}samp_{raw}[$ so f_N, the PDF for $\hat{corr_2} - corr_2$, is:

$$\begin{aligned} f_N(x, samp_{raw}) &= \frac{\mathbb{I}_{]-S;S[} * \mathbb{I}_{]-R;R[}(x)}{4SR} && (11.26) \\ &= \begin{cases} 0 & \text{if} \quad x \in]-\infty; -R-S] \cup]R+S; +\infty[\\ \frac{min(R,x+S)-max(-R,x-S)}{4SR} & \text{otherwise} \end{cases} && (11.27) \end{aligned}$$

[12]It should be noted that an implicit constraint on the selection of the formats for a and b is imposed by this equation. Because integers are encoded using 32 bits in both software and VHDL, care has been taken to ensure that $1 + M + 1 + 18 \leq 32$ to avoid overflow on $\hat{corr_2}$. The bit reserved for performing the sum implies in turn that the dynamics available for encoding a and b on the RAM is not fully used (15 bits would suffice for storing $\widehat{a-1}$ and $\hat{b}$ in the selected formats) or conversely, that the ranges of accessible values of a and b could be extended were $\hat{corr_2}$ encoded with 33 bits.

[13]Underflow should never occur in normal operation as it would imply a negative electron count beyond the margin taken for representing noise-induced negative values (see 11.18). Such a case of figure would rather signify that correction coefficients (notably the corresponding b value) need to be updated.

[14]The reader is advised to turn to section 11.3.2 where calculations are explained in greater details before examining this one more closely.

[15]As stated in section 11.2.2 a more complete error model can subsequently be built by considering the results as obtained conditionally under the hypothesis that $\hat{samp}^{raw} = samp_{raw}$.

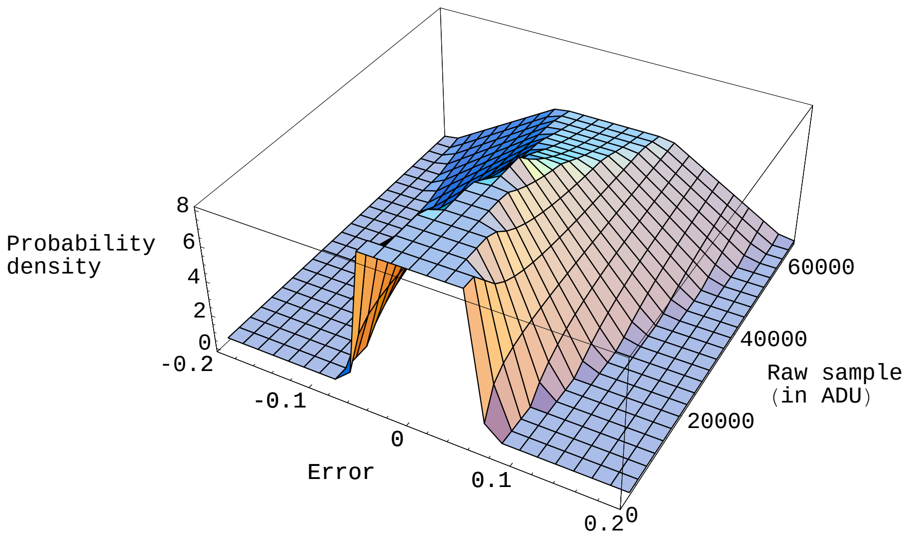


Figure 11.2: Distribution of numerical errors (f_N) with $N = 3$.

The subsequent truncation step must convert the intermediate result from the $1.M + 1.18$ format to the nearest integer. Since the $\hat{corr}_2$ estimate is bias-free, the underlying analysis is similar to the one conducted in section 11.3.2. As mentioned in section 11.2.2, the bias introduced by truncation depends on the sign as can be seen in (11.29), yet this does not modify the computation since it only depends on the square of the offset in (11.31). Besides, the special treatment of the unbalanced round-off configuration ensures that no bias is introduced. This time the offset between the two grids is $offset = \hat{corr}_2[2^{18}]$, with $K = \{0, \dots, 2^{18} - 1\}$, so that the PDF for $\hat{corr} - corr$ reads:

$$g_N(x, samp_{raw}) = \frac{1}{2^{18}} \sum_{i \in K} g_N(x, samp_{raw} | w \in \Omega_i) \quad \text{since offsets are equiprobable.} \tag{11.28}$$

In each class Ω_i, round-off corresponds to[16]:

$$\hat{corr} = \hat{corr}_2 + sign(\hat{corr}_2) \times \underbrace{(i - 2^{17}) \times 2^{-18}}_{k_i} \quad \forall i \in K \tag{11.29}$$

hence the conditional PDFs:

$$g_N(x, samp_{raw} | w \in \Omega_i) = f_N(x - sign(x) \times k_i, samp_{raw}) \quad \forall i \in K \tag{11.30}$$

and:

$$\sigma^2_{\hat{corr}-corr}(N, samp_{raw}) = \int_{\mathbb{R}} u^2 f_N(u, samp_{raw}) du + \frac{\sum_{i \in K} k_i^2}{2^{18}} \tag{11.31}$$

Although the integral above could very well be evaluated using (11.27) above, it is more elegant to notice that it is precisely the variance of the $\hat{corr}_1 + \hat{b} \times 2^{18-N}$ sum. Since the two terms are independent and uniformly distributed, formula (11.2) applies directly to yield[17]

$$\sigma^2_{\hat{corr}-corr}(N, samp_{raw}) = \frac{2^{-2N} + 2^{-36} samp^2_{raw}}{12} + \frac{11453246123}{137438953472} \tag{11.32}$$

This result can be interpreted like (11.53) as decoupling the errors related to the computation and to the round-off step. It is worth pointing out that, thanks to the high precision representation of $\hat{corr}_2$, the second term only amounts to 0.0833 ADU^2, a value remarkably close to the theoretical bound (11.2).

[16]These formula correspond to the software case. Although formulae are different in hardware, the conclusion still holds since the two produce identical results.

[17] $\frac{\sum_{i \in K} k_i^2}{2^{18}} = \frac{1}{2^{54}} \sum_{i=-2^{17}} 2^{17} - 1 i^2 = \frac{1}{2^{54}} (\sum_{i=0}^{2^{17}} i^2 + \sum_{i=0}^{2^{17}-1} i^2$ with $\sum_{i=0}^{n} i^2 = \frac{n(n+1)(n+2)}{6}$.

Formula (11.32) shows that the worst precision is achieved close to saturation. Below this value, however, the contribution from $\hat{b}$ is bound to rapidly dominate that of $\hat{a} \times samp_{raw}$. If we choose to make contributions equal at the saturation level (with $N = 2$), then precision is only weakly dependent on $samp_{raw}$ and, at the $0\ e^-$ level (corresponding to 4070 ADU), the standard error amounts to $\frac{\sqrt{\sigma^2_{\hat{corr}-corr}(2,4070)}}{4070} = 0.00731\ \%$. With this value, the worst case standard error remains within reasonable bounds with a value of 0.306186 ADUs and figure (11.3) illustrates how the variance of errors depend on the sample value. Besides, the parity of g_N implies that the probability of producing an erroneous ADU value is given by $2\int_{0.5}^{+\infty} g_2(u, samp_{raw})du \simeq 3.3617 \times 10^{-7} \times samp_{raw} + 0.0591$. [18]

$$E(\hat{corr} - corr) = 0 \tag{11.33}$$

$$\sigma_{\hat{corr}-corr}(2, samp_{raw}) = \sqrt{\frac{2^{-4} + 2^{-36}samp^2_{raw} + 1}{12} + 2.4253 \times 10^{-12}} \tag{11.34}$$

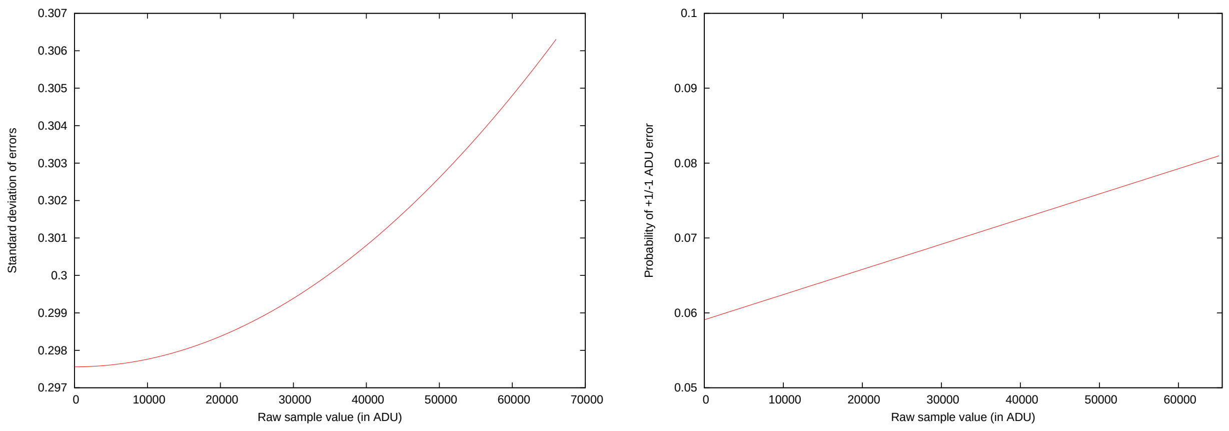


Figure 11.3: Estimator's standard deviation (left) and ±1 ADU error probability (right) with $N = 2$ as a function of the raw sample value.

11.3.2 Mode determination

- Rationale
  The regional background level is determined as the mean mode value of all four histograms[19] of sample values on each rectangular 32 × 32-sample region (hyperpixel). Each histogram is built then the index of the most populated bin is determined. Following [4], a parabola is then adjusted to refine the estimate to the sub-ADU level before all four estimates are averaged. The underlying analysis is presented in chapter 5 and leads to the following formula:

$$\frac{1}{4}\sum_{i\in\{0\ldots3\}}\left(mode^i + 2\frac{F^i_{index^i-1} - F^i_{index^i+1}}{F^i_{index^i-1} - 2F^i_{index^i} + F^i_{index^i+1}}\right) \rightarrow mode \tag{11.35}$$

  with the set of assumptions collected in Tab. 11.5.

  The resulting values (one for each hyperpixel) are then interpolated (sub-section 11.3.3) to yield a background map. In case there exist several most populated bins, the one retained has the minimum index value ($F^i_{index^i} > F^i_{index^i-1}$) – hence the "divide-by-zero" case is avoided. Additionally, the selected range renders finding $index^i = 0$ very unlikely so boundary effects can be neglected.

- Fixed-point arithmetics
  Let us first summarise the available resources to define to which maximum error formula (11.35) needs to be evaluated:

[18]This is the probability of obtaining a value other than what would result from rounding the real number to the nearest integer. However, it also corresponds to obtaining a value ±1 ADU away from it since the distribution falls off to 0 rapidly.

[19]As the expected sky background is low, the histogram is truncated to the lower sample values only.

Variable	Meaning	Type	Domain	Property
i	index of the re-binned histogram (4 ADU per bin)	i	$\{0,1,2,3\}$	
$mode^i$	value of the mode (corresponds to $index^i$)	i	$\{936\ldots1960\}$	
$index^i$	index of the bin of maximum count	i	$\{0\ldots255\}$	$F^i_{index^i} \geq F^i_j \quad \forall j$
F^i_j	count of the j^{th} bin	j	$\{0\ldots1024\}$	$\sum_j F^i_j = 1024$
$mode$	mode estimate (ADU)	f	$]935;1961[$	

Table 11.5: Mode calculation variables and assumptions.

– As $mode \in]935;1961[$, the sign and integer parts of the targeted fixed-point format can be 0.11.

– Increments to mode are bounded under the conditions imposed above. The lower bound being greater than $\frac{1}{4096}$, all possible values of increments can be differentiated if a 0.11.12 format is used.

– The resulting *mode* values will be bi-linearly interpolated in section 11.3.3. This operation, which is delicate otherwise, can be carried out very simply and exactly if 10 bits are reserved at this stage for the subsequent multiplication.

For the sake of both simplicity and tight control of the propagation of errors, the last item is decisive and implies that *mode* must be eventually represented in 0.11.11 format. As below, additional bits can be used for intermediate results to achieve maximum numerical performances within the bounds of the eleven decimal bits available.

Written as in (11.35), the computation faces four contributors – the $mode^i$ terms being exact – to the final error and bias:

1. The $2\frac{F^i_{index^i-1}-F^i_{index^i+1}}{F^i_{index^i-1}-2F^i_{index^i}+F^i_{index^i+1}}$ terms may be computed with a maximum error bounded by 2^{-21} and a bias of 2^{-22} since the numerator cannot exceed 2^{11}. If round-off is performed using the last bit, the maximum error increases to 2^{-20}.
2. The maximum error then increases proportionally with the number of terms in the sum. With four terms, although the format remains unchanged, the result's last 2 bits are not significant. Accordingly, some margin for precision must be taken and the result rounded-off to the significant bits. Conversely, the size of the integer part may require up to 13 bits. Finally, it is worth noting that, as the sum of independent uniform variables (see 11.2.2), the distribution of the resulting error is not uniform.
3. The final normalisation further complicates the picture, mainly because of the non-uniform distribution of errors. Indeed, it introduces a bias which is smaller than the representable precision at this stage – as will be seen below – and hence cannot be corrected with a procedure analog to (11.6), (11.7) and (11.8).
4. The result produced has a lower precision than the number of bits used for its encoding because of the summation. The additional bits used for improved precision must be rounded-off for the result to fit into the 11 allocated decimal bits. This must, hence, be carried out in a finely controlled way to achieve a precision close to the level imposed by (11.10), and to avoid any additional bias.

Because of the fact that the divisor is a power of two in item 3, items 3 and 4 could be performed jointly and any bias avoided, however, we prefer to rely on the distributivity of division to exchange items 3 with 2. The normalisation can then be carried out jointly with item 1 according to the following re-formulation:

$$\frac{\sum_{i\in\{0\ldots3\}} mode^i}{4} + \sum_{i\in\{0\ldots3\}} \frac{F^i_{index^i-1} - F^i_{index^i+1}}{2\times(F^i_{index^i-1} - 2F^i_{index^i} + F^i_{index^i+1})} \quad \rightarrow \quad mode \tag{11.36}$$

The first term can be computed without errors since 2 decimal bits suffice to exactly represent the resulting decimal. Similarly, each of the terms in the sum can be produced with a predefined number of exact bits after bias correction. The resulting distribution of errors can be fully characterised based on the argument in section 11.2.2 to fine-tune the round-off step and obtain the final fixed-point format without any bias. This error is only a function of the number of decimal bits (N) used for the computation of the ratios in (11.36) and of the format retained at the round-off level – the preferred value $N = 16$ is justified in the error analysis below. The calculation is then performed as follows:

1. Individual ratios for the sub-ADU correction.

For each $i \in \{0 \dots 3\}$ we compute:

$$\underbrace{sub\hat{\_}adu^i}_{1.0.N} = \frac{\overbrace{(F^i_{index^i+1} - F^i_{index^i-1}) \times 2^N}^{1.9.N}}{2 \times \underbrace{(2F^i_{index^i} - F^i_{index^i-1} - F^i_{index^i+1})}_{0.12.0}} \tag{11.37}$$

Then correct the bias depending on the sign of $sub\hat{\_}adu^i$ using either (11.6), (11.7) or (11.8). This leads to $sub\hat{\_}adu^i_2$, which is in $1.0.N-1$ format.

2. ADU contribution.

This term is computed exactly then transposed to a format compatible with the terms above according to:

$$\underbrace{\hat{adu}}_{0.11.N-1} = \frac{\overbrace{\Big(\sum_{i\in\{0\dots3\}} mode^i\Big) \times 2^2}^{0.13.2}}{4} \times 2^{N-3} \tag{11.38}$$

A formula which is rather effectively computed through a single left shift operation (by $N-3$ bits).

3. Sum.

All contributions are collected:

$$\underbrace{s\hat{u}m}_{0.11.N-1} = \underbrace{\hat{adu}}_{0.11.N-1} + \sum_{i\in\{0\dots3\}} \underbrace{sub\hat{\_}adu^i_2}_{1.0.N-1} \tag{11.39}$$

4. Round-off

Finally, round-off is performed to reduce the number of decimal bits to 11. Two cases of figure are considered here, as will be justified by the error analysis below:

– If $s\hat{u}m \equiv 2^{N-13}[2^{N-12}]$, let k be a uniform binary random variable, then

$$\underbrace{m\hat{o}de}_{0.11.11} = \frac{\overbrace{s\hat{u}m + 2^{N-13}}^{0.11.N-1}}{2^{N-12}} - k \tag{11.40}$$

– otherwise:

$$\underbrace{m\hat{o}de}_{0.11.11} = \frac{\overbrace{s\hat{u}m + 2^{N-13}}^{0.11.N-1}}{2^{N-12}} \tag{11.41}$$

- Formats

– $F^i_{index^i}$ (0.10.0), $F^i_{index^i+1}$ (0.9.0), $F^i_{index^i-1}$ (0.9.0).

The hyperpixels built in GD feature a total of 1024 samples being 32×32 in the AL and AC directions. Coding $F^i_{index^i}$ using 10 bits allows for handling all cases of figure except the improbable one in which all samples share the same value. This may be the case if all are saturated. This value being outside the background range the histogram would remain empty and lead to propagating the value of a neighbouring hyperpixel or to the use of the default background value as described in chapter 5. Conversely, in the highly unlikely event that all would be equal with a value in the background range, $F^i_{index^i}$ would overflow to 0 and hence, as before, trigger the background value replacement mechanism. Similarly, because $F^i_{index^i} + F^i_{index^i+1} \leq 2^{10}$ and $F^i_{index^i} \geq F^i_{index^i+1}$, $F^i_{index^i+1} \leq 2^9$. Similarly, $F^i_{index^i-1} < 2^9$, so both may be coded on 9 bits only, if we deem acceptable to then approximate the background mode as $index^i$ (since $F^i_{index^i+1}$ would overflow to 0).

– $\widehat{sub\_adu}^i$ (1.0.N).

N decimal bits are introduced through the product with 2^N and a sign bit is required to account for the different possible orderings of $F^i_{index^i+1}$ and $F^i_{index^i-1}$. The determination of the number of integer bits necessary results from the study of the function g defined as:

$$g : \begin{cases} [0, 2^9 - 1] \times [0, 2^{10}] \times [0, 2^9] & \rightarrow \quad \mathbb{R} \\ \qquad (f_1, f_2, f_3) & \rightarrow \quad 2^{N-1} \frac{f_3 - f_1}{2f_2 - f_1 - f_3} \end{cases} \tag{11.42}$$

Strictly speaking the values of this function are only of interest on the point of integer coordinates, but since we are interested in minimum and maximum values, we may look at it as a mapping of $\mathbb{R}^3$ onto $\mathbb{R}$.

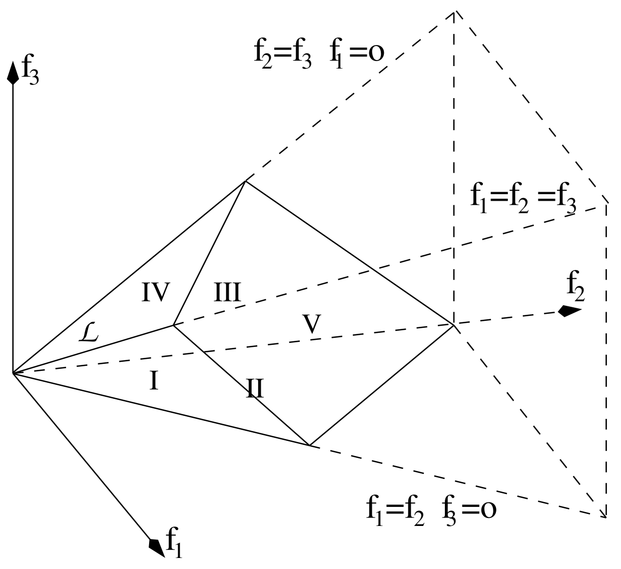


Figure 11.4: Geometry of the domain $\mathcal{D}$

The domain of interest $\mathcal{D}$ results from the intersection of the quadrant of positive f_1, f_2 and $f3$ with the half-spaces $f_2 > f_1$ and $f_2 \geq f_3$ and the octahedron corresponding to the "sphere" $|| \quad ||_1 \leq 1024$. Since g has no critical points except on the $f_1 = f_2 = f_3$ line ($\mathcal{L}$), where it is not defined as its denominator vanishes, its upper and lower bounds are reached on $\mathcal{D}$'s boundaries, that is surfaces I, II, III, IV and V on Fig. 11.4:

I. $g(f_1, f_2, f_3) = -2^{N-1}$ when $f_1 = f_2$.
II. $g(f_1, f_2, f_3) = -2^{N-1} f_1/(2f_2 - f_1)$ when $f_3 = 0$, which is minimum when $f_1 = 2^9 - 1$ and $f2 = f_1 + 1$: $-2^{N-1}.(2^9 - 1)/(2^9 + 2)$ and maximum when $f_1 = 0$: 0.
III. $g(f_1, f_2, f_3) = 2^{N-1} f_3/(2f_2 - f_3)$ when $f_1 = 0$, which is maximum when $f_3 = 2^9$ and $f2 = f_3$: 2^{N-1} and minimum when $f_3 = 0$: 0.
IV. $g(f_1, f_2, f_3) = 2^{N-1}$ when $f_3 = f_2$.
V. $g(f_1, f_2, f_3) = 2^{N-1}(f_3 - f_1)/(2^{11} - 3(f_1 + f_3))$ on the plane $f_1 + f_2 + f_3 - 2^{10} = 0$.
The function g being continuous on all transitions between the enclosing planes (I, II, III, IV, V) except along $\mathcal{L}$ where it may be prolonged by continuity to -2^{N-1} and 2^{N-1} on I's and IV's sides respectively, no additional extrema are possible.

The conclusion is that on $\mathcal{D}$, g has -2^{N-1} and 2^{N-1} for lower and upper bounds respectively. This translates in the absence of integer bits, hence the 1.0.N format.

- Error analysis

The following error analysis leads to setting the value of the free parameter N and justifies the special treatment of the bias above (11.40) and (11.41).

Let us first derive the properties of the intermediate variable $s\hat{u}m$ in equation (11.39). The properties of each of the $\widehat{sub\_adu}^i_2$ terms follow directly from (11.9) and the argument in section 11.2.2. For each i, the distribution of $\widehat{sub\_adu}_2^i - sub\_adu_2^i$ is hence uniform over the $[-2^{-N+1}; 2^{-N+1}]$ range. The $\hat{adu}$ term, being computed exactly, does not contribute to the error budget, so that its distribution simply results from the sum of the four $\widehat{sub\_adu}_2^i - sub\_adu_2^i$ terms. These numerical errors follow from independent processes, for $N = 2$ the PDF f_2 for $\hat{sum} - sum$ then reads:

$$f_2(x) = \mathbb{I}_{[-\frac{1}{2};\frac{1}{2}]} * \mathbb{I}_{[-\frac{1}{2};\frac{1}{2}]} * \mathbb{I}_{[-\frac{1}{2};\frac{1}{2}]} * \mathbb{I}_{[-\frac{1}{2};\frac{1}{2}]}(x) \tag{11.43}$$

$$= \begin{cases} 0 & & x \in [-\infty; -2[\\ \frac{x^3}{6} + x^2 + 2x + \frac{4}{3} & & x \in [-2; -1[\\ \frac{-x^3}{2} - x^2 + \frac{2}{3} & \text{if} & x \in [-1; 0[\\ \frac{x^3}{2} - x^2 + \frac{2}{3} & & x \in [0; 1[\\ \frac{-x^3}{6} + x^2 - 2x + \frac{4}{3} & & x \in [1; 2[\\ 0 & & x \in [2; +\infty] \end{cases} \tag{11.44}$$

Using which the general distribution can be expressed for all N as:

$$f_N(x) = 2^{N-2} f_2(2^{N-2}x) \tag{11.45}$$

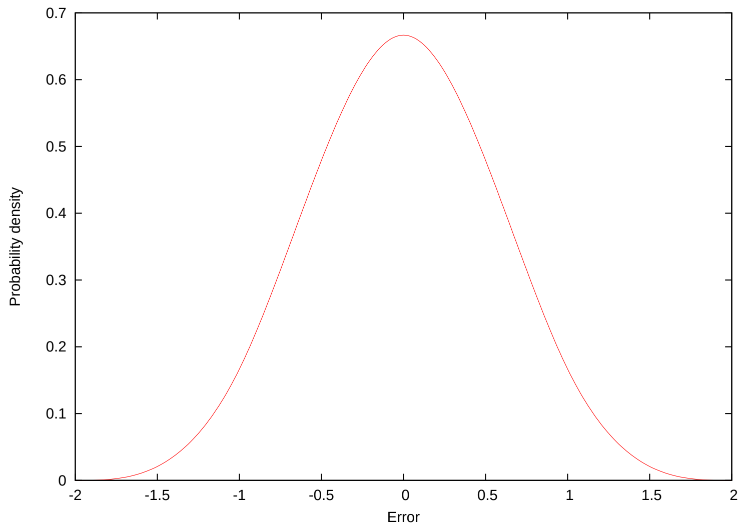


Figure 11.5: Distribution of numerical errors ($\hat{sum} - sum$) for $N = 2$

The $\hat{sum}$ estimate, as the sum of four unbiased estimates, is bias-free. However, although the smallest increment representable amounts to 2^{-N+1}, maximum and minimum errors are separated by 2^{-N+3}. This implies that the last two bits of $\hat{sum}$ hold a limited amount of information and suggests to exploit them for round-off purposes.

Round-off consists in transforming a result located on a grid of a given precision to another of lower one. The relative positions of the two grids condition the nature of the error introduced as is illustrated Fig. 11.6.

Offsets between the grids can be expressed as: $offset = \hat{sum}[2^{N-12}]$. This expression has the same form as the linear congruential generators discussed in section 11.2.2, so that the four possible offset values can be considered equiprobable. Combined with the symmetry of the distribution (11.44), this favourable circumstance simplifies the problem in our case since cases 2 and 4 in Fig. 11.6 then balance each other.

Whatever the chosen format, the remaining unbalanced case (case 1 in Fig 11.6) leads to the introduction of a bias smaller than the smallest representable increment, even in high precision. However, it is worth noticing that it corresponds to the situation in which two low precision nodes are equidistant from the high precision result. Assignment to either ones is accordingly somewhat arbitrary, so one simple way of solving this difficulty without affecting the estimator's variance consists in making this case symmetrical by assigning the result to one or the other alternatively (dotted arrow of Fig. 11.6). Hence, concerning bias, we obtain:

$$E(\widehat{mode} - mode) \quad = 0 \quad \text{ADU} \tag{11.46}$$

To compute the variance, the error distributions after round-off need to be derived starting from (11.44). Let

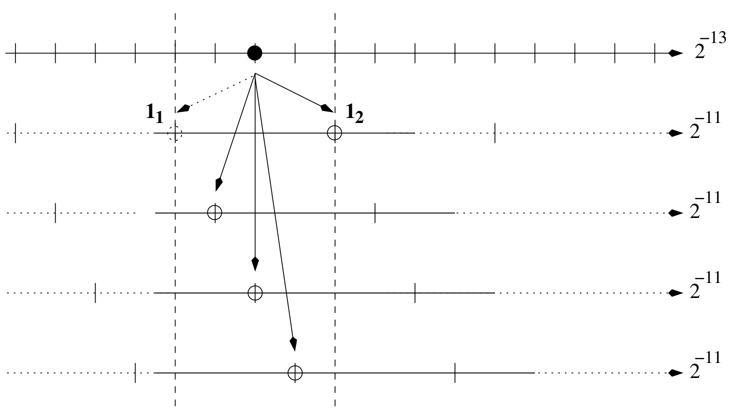


Figure 11.6: Relative positions of the high (2^{-13}) and low (2^{-11}) precision grids for performing round-off from one to the other (for $N = 14$).

$K = \{0, \ldots, 2^{N-12} - 1\}$ be the set of indices for the different offset cases (Ω_i), then since:

$$\Omega = \bigcup_{i \in K} \Omega_i \text{ with } \Omega_i \neq \emptyset \quad \forall i \in K \text{ and } \Omega_i \cap \Omega_j = \emptyset \quad \forall (i,j) \in K^2 \text{ when } i \neq j \tag{11.47}$$

$\{\Omega_i | i \in K\}$ constitutes a partition of the sample space Ω. So, that for the $\hat{mode} - mode$ variable, the PDF can be decomposed as:

$$\begin{aligned} g_N(x) &= \sum_{i \in K} g_N(x|w \in \Omega_i) \times P(w \in \Omega_i) \\ &= \frac{1}{2^{N-12}} \sum_{i \in K} g_N(x|w \in \Omega_i) \quad \text{since offsets are equiprobable.} \end{aligned} \tag{11.48}$$

In each class, round-off corresponds to the following change of variable:

$$\hat{mode} = s\hat{u}m + \underbrace{(i - 2^{N-13}) \times 2^{-N+1}}_{k_i} \quad \forall i \in K \tag{11.49}$$

The conditional PDFs can thus be expressed as:

$$g_N(x|w \in \Omega_i) = f_N(x - k_i) \quad \forall i \in K \tag{11.50}$$

To account for the special treatment of case 0 (see above for the correction of bias), we could further decompose Ω_0 into Ω_{0_0} and Ω_{0_1} with:

$$\hat{mode} = \begin{cases} s\hat{u}m - 2^{-12} & \text{for} \quad \omega \in \Omega_{0_0} \\ s\hat{u}m + 2^{-12} & \text{for} \quad \omega \in \Omega_{0_1} \end{cases} \quad \text{and} \quad \begin{cases} g_N(x|w \in \Omega_{0_0}) = f_N(x + 2^{-12}) \\ g_N(x|w \in \Omega_{0_1}) = f_N(x - 2^{-12}) \end{cases} \tag{11.51}$$

but this will not prove necessary for our calculation (as (11.53) only depends on k_i^2).
Finally, the variance can be computed through:

$$\begin{aligned} \sigma^2_{\hat{mode}-mode}(N) &= \int_{\mathbb{R}} x^2 g_N(x) dx \\ &= \frac{1}{2^{N-12}} \sum_{i \in K} \int_{\mathbb{R}} x^2 f_N(x - k_i) dx \\ &= \frac{1}{2^{N-12}} \sum_{i \in K} \int_{\mathbb{R}} (u + k_i)^2 f_N(u) du \end{aligned} \tag{11.52}$$

We find:

$$\sigma^2_{\hat{mode}-mode}(N) = \underbrace{\int_{\mathbb{R}} u^2 f_N(u) du}_{\frac{1}{2^{2(N-2)}} \cdot \int_{\mathbb{R}} x^2 f_2(x) dx = \frac{1}{3 \times 2^{2(N-2)}}} + \frac{\sum_{i \in K} k_i}{2^{N-13}} \underbrace{\int_{\mathbb{R}} u f_N(u) du}_{0} + \underbrace{\frac{\sum_{i \in K} k_i^2}{2^{N-12}}}_{\frac{8400899}{12582912.2^{2N}} + \frac{1}{50331648}} \tag{11.53}$$

This formula shows that two decoupled effects impact on the estimator's variance: the combined numerical errors from the individual high precision estimates (first term above) and the truncation error (third term). Not surprisingly, the latter dominates rapidly when N increases and sets the variance's limit as is illustrated Fig. 11.7. Interestingly, this value $\simeq 1.987.10^{-8}$ ADU^2 is smaller than what would be achieved had we directly derived the estimate at the 2^{-11} precision level (11.10): $(2^{-11})^2/3 \simeq 7.947.10^{-8}$ ADU^2. This is a consequence of the fact that the distribution of errors (11.44) is sharper than in the uniform case (Fig. 11.5).

We can now set the value of N, taking into account the word-length constraint which imposes: $N \leq 18$ because of (11.40) and (11.41). Besides, we want to have $N \geq 12$ to avoid degrading the precision of the $\widehat{sub\_adu^i}$ terms below the selected fixed-point format. Fig. 11.7 shows that the performance improvement becomes almost invisible for $N > 16$. One solution to save on resources in the frame of a dedicated hardware implementation, is to select this value. We choose $N = 16$ and have:

$$\sigma^2_{\hat{mode}-mode}(16) \quad = \quad 2.1265.10^{-8} \text{ ADU}^2 \tag{11.54}$$

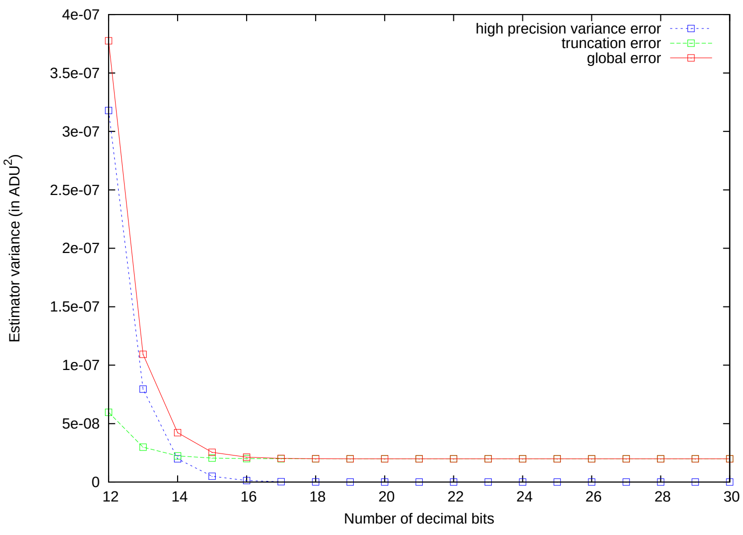


Figure 11.7: Contributions to the estimator's variance as a function of the number of decimal bits.

11.3.3 Background interpolation

- Rationale
  The regional background values are linearly interpolated in both the AL and AC directions to produce a two-dimensional background map. For each sample, the modes derived for the neighbouring four hyperpixels are used according to:

$$\begin{aligned} &mode_{0,0}(1-\tfrac{i}{W})(1-\tfrac{j}{H}) + mode_{W,0} \times \tfrac{i}{W}(1-\tfrac{j}{H}) \\ &+mode_{0,H} \times \tfrac{j}{H}(1-\tfrac{i}{W}) + mode_{W,H} \times \tfrac{ij}{WH} \rightarrow samp^{bk}_{i,j} \end{aligned} \tag{11.55}$$

with the set of assumptions recalled in Tab. 11.6.

Variable	Meaning	Type	Domain
$mode_{W,H}$	mode estimate at position (W, H)	f	$]935; 1961[$
i, j	AL & AC coordinates	i	$[0; 31]$
W, H	AL & AC dimensions of the hyperpixel	i	32
$samp^{bk}_{i,j}$	background value at position (i, j)	f	$]935; 1961[$

Table 11.6: Variables and assumptions for interpolation.

The background map is then used to estimate the contribution of the sky background to the measured flux for the SNR computations on-board (section 11.3.4 and 11.3.7).

A sample of residuals obtained between background calculations performed entirely with fixed-point vs. floating-point arithmetics is shown in Fig. 11.8. Values have been re-normalised and false colours are used to increase the contrast and allow identification of the structures. The dimension of the hyperpixels is clearly visible and the circular structure of similar size may be interpreted as sections in the interpolation errors (Fig. 11.9). There exist both larger and smaller scale structures difficult to explain, as the presence of "flat" regions with comparatively uniform residuals interpreted as resulting from compiler optimisations or from the internal mechanisms of the floating-point computations.

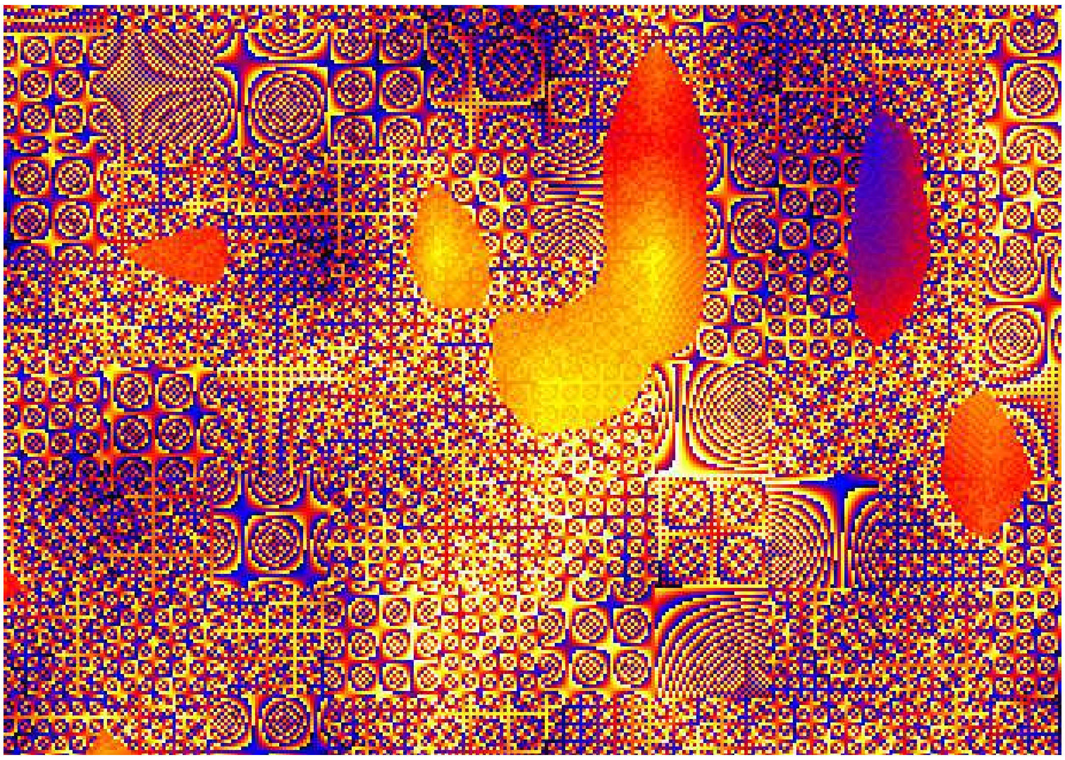


Figure 11.8: Residuals between purely fixed-point and purely floating-point background maps (in false colours).

- Fixed-point arithmetics
  Adopting $W = H = 32$ (i.e. a power of two) is a considerable simplification in the case of fixed-point arithmetics. Indeed, not only may the interpolation itself be computed exactly but it then simply corresponds to a multiplication with an implicit change of the fixed-point format to 0.11.21.

$$\underbrace{sa\hat{m}p^{bk}_{i,j}}_{0.11.21} = \underbrace{mode_{0,0}}_{0.11.11} \times 2^{10} - mode_{0,0} \underbrace{(i+j)}_{0.0.6} \times 2^{5} + mode_{0,0} \times \underbrace{i \times j}_{0.0.10} + \ldots \quad (11.56)$$

  A transient overflow may occur when computing the $mode_{0,0}(i+j) \times 2^5$ term because we can have $i + j \geq 32$, however the sum $-mode_{0,0}(i+j) \times 2^5 + mode_{0,0} \times i \times j$ is guaranteed never to overflow (as is visible in formula (11.55)).

- Error analysis
  As the interpolation is computed with exact coefficients and the mode values used at the four nodes are bias free, that of $sa\hat{m}p^{bk}_{i,j}$ reduces to:

$$E(sa\hat{m}p^{bk}_{i,j} - samp^{bk}_{i,j}) \quad = \quad E(\hat{mode} - mode) = 0 \quad \text{ADU} \quad \forall (i,j) \in [0;31]^2 \quad (11.57)$$

  In terms of variance, we similarly have equal variances at the four nodes, with null covariances since the individual $\hat{mode}$ computations are entirely independent. Taking into account the change of fixed-point format to 0.11.21, it reads:

$$\sigma^2_{sa\hat{m}p_{i,j}{}^{bk} - samp^{bk}_{i,j}} \quad = \quad (2(\frac{i}{2^5})^2 - 2\frac{i}{2^5} + 1)(2(\frac{j}{2^5})^2 - 2\frac{j}{2^5} + 1) \times \sigma^2_{\hat{mode} - mode} \quad (11.58)$$

As illustrated by Fig. 11.9, the variance is maximum at each of the node (equal to $\sigma^2_{\hat{mode}-mode} \sim 2.1265.10^{-8}$ ADU^2) and minimum at the centre where it reaches $\sigma^2_{\hat{mode}-mode}/4 \sim 5.3164.10^{-9}$ ADU^2. Its variations have the symmetries of the interpolation itself.

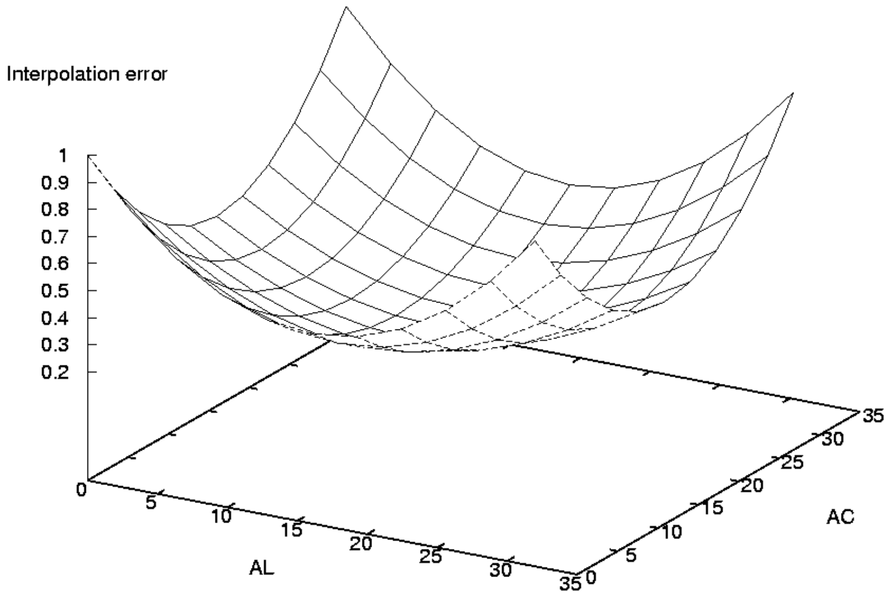


Figure 11.9: Relative interpolation error $(\frac{\sigma^2_{\hat{samp}_{i,j}{}^{bk} - samp^{bk}_{i,j}}}{\sigma^2_{\hat{mode}-mode}})$ over the hyperpixel

11.3.4 Sample-wise SNR

- Rationale

  Relevant samples are identified by thresholding them on a SNR basis. The contributions of the background and of the noise are taken into account in the following formula:

$$\frac{samp_j^{cor} - samp_j^{bk}}{\sqrt{samp_j^{cor} + RON^2}} >^? SNR \tag{11.59}$$

  Yet, this holds insofar as the *samp* variables are in electrons while they are naturally expressed in ADU at this stage. In ADU, after squaring the expression to remove the need to compute a square root and removing all divisions for maximum performance, the test condition (11.59) transforms into:

$$G^2 \times \sup(samp_j^{cor} - samp_j^{bk}, 0)^2 >^? SNR^2 \times |G \times samp_j^{cor} + RON^2 - offset| \tag{11.60}$$

  with the assumptions collected in Tab. 11.7.

- Fixed-point arithmetics

  The essential need here is to be able to reliably evaluate the test expression to retain the proper samples at all times, not to compute the sample-wise SNR in all cases. Hence, two different cases of figure may be envisaged: when the condition is trivially fulfilled and when it requires a precise calculation. Separating the two, as will promptly become apparent, allows for reducing the domains in which the variables reside and hence for increasing the number of free bits to achieve maximum precision.

  Decomposing the signal into k_1 and k_2, respectively the fraction above the local background level and its complement above the 0 ADU value,

$$samp_j^{cor} = samp_j^{bk} + k_1 = \frac{offset}{G} + k_2 + k_1 \tag{11.61}$$

Variable	Meaning	Type	Domain
j	AC coordinate of the sample	i	$[0; 983[$
$samp_j^{fil}$	filtered sample value	f	$[0; 65535]$
$samp_j^{cor}$	corrected sample value	i	$[0; 65535]$
$samp^{bk}$	background value	f	$]935; 1961[$
RON	standard deviation of the read-out noise	f	$[0; 20[$
G	sample encoding gain	f	$[3; 10]$
$offset$	sample encoding offset	i	$\sim G \times 1000$
SNR	sample-wise signal-to-noise ratio	f	$[0 : 6[$

Table 11.7: Variables and assumptions for the SNR calculation.

condition (11.60) becomes:

$$G^2 \times k_1^2 - G \times k_1 \times SNR^2 - SNR^2(G \times k_2 + RON^2) >^? 0 \tag{11.62}$$

This is automatically fulfilled as soon as

$$k_1 \geq \frac{SNR}{2 \times G}(SNR + \sqrt{4 \times G \times k_2 + 4 \times RON^2 + SNR^2}) \tag{11.63}$$

With the assumptions above and knowing that $k_2 \leq 1024$, this condition is verified whenever $k_1 \geq 128$ in which case the sample is retained without the need to compute the exact value of the SNR. Otherwise, k_1 can be encoded in 0.7.9 format then squared to 0.14.18 and tested versus the threshold.

The resulting 5-step procedure is then:

1. Verify that $\hat{samp}_j^{cor} \geq \frac{offset}{G}$, discard the sample if false.
2. Compute $\hat{samp}_j^{cor} - \hat{samp}_j^{bk}$ in 1.22.9 format.
3. Test the sign bit: discard if negative,
4. else test the integer part: if ≥ 128 keep the sample,
5. otherwise go through the detailed computation to test versus the threshold.

Having verified that $\hat{samp}_j^{cor} - \frac{offset}{G} \geq 0$, this last step may be written without risk of overflow as:

$$\underbrace{(\hat{samp}_j^{cor} - \hat{samp}_j^{bk})^2}_{0.14.18} >^? \underbrace{\underbrace{\frac{SNR^2}{G}}_{0.4.18} \times \underbrace{(\hat{samp}_j^{cor} - \frac{offset}{G})}_{0.11.0}}_{0.14.18} + \underbrace{\frac{SNR^2}{G^2} \times RON^2}_{0.11.18} \tag{11.64}$$

or alternatively, tolerating a transient overflow, as:

$$\underbrace{(\hat{samp}_j^{cor} - \hat{samp}_j^{bk})^2}_{0.14.18} >^? \underbrace{\frac{SNR^2}{G}}_{0.4.18} \times \underbrace{\hat{samp}_j^{cor}}_{0.12.0} - \underbrace{\frac{SNR^2}{G^2} \times (offset - RON^2)}_{0.14.18} \tag{11.65}$$

In terms of computing resources, the first form (11.64) has the advantage of slightly reduced dynamics, but requires one adder, one subtractor and one multiplier while the second (11.65) has operands of larger size but calls only for one adder and a multiplier. Besides, in the case of (11.64), the computation of $\hat{samp}_j^{cor} - \frac{offset}{G}$ may be efficiently mutualized between testing the sign (ie. the value of the most significant bit) and performing the detailed computation.

In software, since data words are necessarily 32-bit long, for the benefit of pre-computing a maximum number of terms, expression (11.65) has been preferred, while in hardware, the first form (11.64) was rather selected – a choice to be confirmed by the amount of resources allocated during synthesis.

- Error analysis
  In terms of error analysis, there are three contributors in (11.65):

  1. $|\epsilon_1| < \frac{1}{2048}$ for the error on $\hat{samp}_j^{bk}$ from the previous steps.

2. $|\epsilon_2| < \frac{1}{2^{18}}$ for the error when encoding $\frac{SNR^2}{G}$ in 0.14.18 format.

3. $|\epsilon_3| < \frac{1}{2^{18}}$ for the error when encoding $\frac{SNR^2}{G^2}(RON^2 - offset)$ in 0.14.18 format.

The corresponding contributions, however, differ considerably in magnitude, their respective upper bounds being:

1. $|2 \times \epsilon_1 \times (samp_j^{cor} - samp_j^{bk}) + \epsilon_1^2| < |256 \times \epsilon_1 + \epsilon_1^2|$ of order $\sim 0.125 \quad \text{ADU}^2$,

2. $|\epsilon_2 \times samp_j^{cor}| = |\epsilon_2 \times (samp_j^{bk} + 128)| < |2089 \times \epsilon_2|$ of order $\sim 8 \times 10^{-3} \quad \text{ADU}^2$,

3. $|\epsilon_3| < \frac{1}{2^{18}}$ of order $3.8 \times 10^{-6} \quad \text{ADU}^2$.

So that the total error is bounded by 0.133 ADUs. Although this value may seem high, it is worth mentioning that it greatly benefits from the reduction of the $samp_j^{cor}$ domain for which (11.65) is computed and from isolating the terms which can be pre-computed. The remaining error is largely related to the computation of the square of the $\hat{samp}_j^{cor} - \hat{samp}_j^{bk}$ term, which is itself the consequence of the simplification related to the removal of square root operation.

In spite of the apparently large numbers, the computational error has a limited impact on the processing. The result of the thresholding is only different for a few samples in one million and is limited to samples at the noise/signal limit which do not significantly impact on the detectability of objects nor on their measurements.

11.3.5 Background flux

- Rationale
  The background map built in section 11.3.3 is also used to subtract the contribution of the sky background from the object measurements. Given the precision requirements for the latter on board, the resolution of the background map and the limited extension of objects significantly affected by the low background (faint stars), it was deemed sufficient to account for it in the form of the object's total background flux. Combined with the count of member samples, the average background per sample can then also be derived from this value.

  The natural formula then reads:

$$\sum_{(i,j) \in object} samp_{(i,j)}^{bk} \rightarrow bkgd\_flux \tag{11.66}$$

The cardinal number of *object* – i.e. the number of member samples pertaining to the object – although not bounded in theory, is subject to the maximum latency constraint which applies to the object's processing. This in turn translates into a maximum AL size for objects around 200 TDIs with the current assumptions, that is 100 SM samples AL due to the 2×2 binning.

Variable	Meaning	Type	Domain
i	AL relative position of the sample	i	$[0; 100[$
j	AC relative position of the sample	i	$[0; 983[$
N	number of object member samples	i	$[1; 100 \times 983]$
$samp_{i,j}^{bk}$	background value at position (i,j)	f	$]935; 1961[$
$bkgd\_flux$	total background contribution to the flux	f	$]935; 1961 \times N[$

Table 11.8: Variables and assumptions for computing the background flux.

- Fixed-point arithmetics
  As the entire word length has already been exploited when computing interpolated background values in section 11.3.3, some adjustments are mandatory when summing them to either control or avoid overflow. After a first version avoiding overflow by degrading the precision, we prefer a simpler method achieving considerable improvement in precision and error analysis by using two 32-bit words to store the sum exactly. They correspond

respectively to the integer and decimal parts, according to:

$$\underbrace{bkgd\_\hat{f}lux_{int}}_{0.32.0} = \sum_{(i,j)\in object} \frac{sa\hat{m}p^{bk}_{(i,j)}}{2^{21}} \qquad (11.67)$$

$$\underbrace{bkgd\_\hat{f}lux_{dec}}_{0.0.21} = \sum_{(i,j)\in object} sa\hat{m}p^{bk}_{(i,j)} \quad \& \quad (2^{21}-1) \qquad (11.68)$$

where & corresponds to the bit-wise AND with the mask to $2^{21}-1$. Additionally, the carry operation to the integer part is performed whenever $bkgd\_flux_{dec} > 2^{21}-1$. No risk of overflow exists on the integer part since $\log_2(983\times 100)+11 = 27.58$ bits.

- Error analysis
The modified method considerably simplifies and improves the first estimate since we have straightforwardly:

$$\begin{aligned} E(bkgd\hat{\_}flux - bkgd\_flux) &= \sum_{i,j} E(sa\hat{m}p^{bk}_{i,j} - samp^{bk}_{i,j}) \\ &= 0 \quad \text{ADU} \end{aligned} \qquad (11.69)$$

with the following, also greatly improved, upper bound:

$$|bkgd\hat{\_}flux - bkgd\_flux| \le 983\times 100\times 2^{-12} \le 24 \quad \text{ADU} \qquad (11.70)$$

With $\{hpx_k\}$ the different hyperpixels over which the object extends, formula (11.66) may be rewritten as

$$\begin{aligned} bkgd\hat{\_}flux = & \sum_{hpx_k}\Big(\sum_{(i,j)\in hpx_k}(2^{10}-2^5(i+j)+i\times j)\times m\hat{o}de_{0,0} \\ & + \sum_{(i,j)\in hpx_k}(2^5 i - i\times j)\times m\hat{o}de_{W,0} + \sum_{(i,j)\in hpx_k}(2^5 j - i\times j)\times m\hat{o}de_{0,H} \\ & + \sum_{(i,j)\in hpx_k}(i\times j)\times m\hat{o}de_{W,H}\Big) \end{aligned} \qquad (11.71)$$

Noting that computational errors on samples pertaining to different hyperpixels are independent, just as those on the individual $m\hat{o}de$ values placed at the four nodes of each hyperpixel, the different terms in the expression (11.71) are independent. The variance then factors into

$$\sigma^2_{bkgd\hat{\_}flux-bkgd\_flux} = \Big[\sum_{hpx_k}\Big(\sum_{(i,j)\in hpx_k}(2(\frac{i}{2^5})^2 - \frac{i}{2^5}+2^{10})(2(\frac{j}{2^5})^2-\frac{j}{2^5}+2^{10})\Big)\Big]\sigma^2_{m\hat{o}de-mode} \qquad (11.72)$$

just as if all $sa\hat{m}p^{bk}_{i,j}$ were independent. It is difficult to derive a general estimate of this variance because it not only depends on the object's geometry but also on its position on the hyperpixel grid because $\sigma_{m\hat{o}de-mode}$ is not translation-invariant. Over an entire hyperpixel, the coefficient is worth $\frac{466489}{1024} \sim 455.56$, so the upper bound for an object of maximum extension is:

$$\frac{983\times 100}{1024}\times\frac{466489}{1024}\times\sigma^2_{m\hat{o}de-mode} = 9.300\times 10^{-4} \quad \text{ADU}^2 \qquad (11.73)$$

11.3.6 Mean background flux

- Rationale
The mean sample-wise background contribution may easily be estimated from the overall background flux and the count of member samples. The resulting value is used when filtering local maxima when performing component segmentation and is the one included in the objects' OF packets for transmission to ground.

$$\frac{bkgd\_flux}{N} \rightarrow bkgd\_mean \qquad (11.74)$$

Variable	Meaning	Type	Domain
N	count of member samples	i	$[1; 983000]$
$bkgd\_flux$	total background contribution	f	$]935; 1961 \times N[$

Table 11.9: Variables and assumptions for the mean background flux per sample.

- Fixed-point arithmetics
  The calculation is performed over the integer and decimal fields used to store the overall flux to produce a single 0.11.21 value (square brackets [] denote the modulo):

$$\hat{bkgd\_mean} = \frac{\hat{bkgd\_flux}_{int}}{N} \times 2^{21} + \frac{\hat{bkgd\_flux}_{int}[N] \times 2^{21}}{N} + \frac{\hat{bkgd\_flux}_{dec}}{N} \qquad (11.75)$$

Whenever $N < 4096$ or $\hat{bkgd\_flux}_{int} < 4096$ then $\hat{bkgd\_flux}_{int}[N] \times 2^{21}$ does not overflow and the calculation is performed directly. Since this only fails for infrequent extended objects, we can otherwise fall back to computing the two first terms via `mul32AndDivTo32` from the SoftFloat library.

- Error analysis
  The quality of the resulting estimator derives directly[20] from equations (11.70) and (11.72). In the framework imposed by the software interface, notably concerning formats for transmitting the mean background estimate between modules and to ground, the quality of the estimator is strongly constrained by the need to store it on a single 32-bit word (because of the software's 32-bit RAM), hence in 0.11.21 format. The relative precision cannot hence be maintained when dividing by the number of member samples and the error is then dominated by the effect of truncation at the 2^{-21} level.

11.3.7 Object-wise SNR

- Rationale
  A signal-to-noise ratio is estimated at the object level to allow for discarding false detections triggered by the noise's fluctuations. The computation only relies on corrected values ($sample^{cor}$) of samples having passed the sample-wise SNR test:

$$\frac{(flux - bkgd\_flux)^2}{flux + N_o \frac{bkgd\_flux}{N_b} + N_o \times RON^2(1 + \frac{No}{N_b})} >^? SNR^2 \qquad (11.76)$$

with the set of assumptions of Tab. 11.10.

Variable	Meaning	Type	Domain
$flux$	object flux: $\sum_{(i,j)\in object} samp^{cor}_{(i,j)}$	i	$]0; 65535 \times N_o]$
$bkgd\_flux$	background flux	f	$]935; 1961 \times N_o[$
N_o	number of object member samples	i	$[1; 100 \times 983]$
N_b	number of hyperpixel samples	i	1024
SNR	object-wise signal-to-noise ratio	f	$[0; 20[$
RON	standard deviation of the read-out noise	f	$[0; 20[$

Table 11.10: Variables and assumptions for computing the object-wise SNR.

This formula – extracted from [10] (p. 263) – has mainly been inherited from older version of Pyxis. Indeed, formula (11.76) contains terms accounting for the error on the background contribution which would be exact were the latter estimated with a mean or a median. As such, it is essentially obsolete and should be replaced. Accordingly, no efforts were invested in converting it to fixed-point and the calculation is still performed with floating-point arithmetics.

[20]It is worth insisting that we are not here looking at N realisations of the same random variable but at one realisation of the random variable which corresponds to the mean value of N other ones.

11.3.8 Barycentre

- Rationale

  The location of each object is estimated as the barycentre of the member samples. The computation uses corrected values ($sample^{cor}$) of samples having passed the sample-wise SNR test:

$$\frac{\sum_{(i,j)\in object}(samp^{cor}_{i,j} \times i)}{\sum_{(i,j)\in object} samp^{cor}_{i,j}} \rightarrow pos^{obj}_{AL} \tag{11.77}$$

$$\frac{\sum_{(i,j)\in object}(samp^{cor}_{i,j} \times j)}{\sum_{(i,j)\in object} samp^{cor}_{i,j}} \rightarrow pos^{obj}_{AC} \tag{11.78}$$

with the set of assumptions of Tab. 11.11.

Variable	Meaning	Type	Domain
$samp^{cor}_{i,j}$	corrected sample value at position (i,j) (in ADU without the offset)	i	$[0;65535]$
i	AL coordinate of the sample (in the BB)	i	$[0;200[$
j	AC coordinate of the sample (in the BB)	i	$[0;1966[$
pos^{obj}_{AL}	AL position of the object	f	$[0;200[$
pos^{obj}_{AC}	AC position of the object	f	$[0;1966[$
N_{sat}	number of saturated samples	i	$[0;98300[$

Table 11.11: Variables and assumptions for the computation of the barycentre.

In an effort to reduce the computational load associated to saturated objects (processing bursts on the software side), the computation of the barycentre is limited to the set of saturated samples if any.

$$\frac{\sum_{(i,j)\in saturatedregion} i}{N_{sat}} \rightarrow pos^{obj}_{AL} \tag{11.79}$$

$$\frac{\sum_{(i,j)\in saturatedregion} j}{N_{sat}} \rightarrow pos^{obj}_{AC} \tag{11.80}$$

The downside of this simplification is that the resulting estimator – the iso-barycentre of the saturated region – has a high variance which translates the fact that the number of samples is too limited to properly account for the truncation noise corresponding to the pixel grid.

- Fixed-point arithmetics

  Although fixed-point arithmetics have been crafted for this computation, involving handling of numerical overflow and block-based computation (via partial barycentres), the special handling of bright objects has rendered these intricacies of questionable interest. Indeed, if the degradation of the quality of the estimator can be coped with – especially as far as windowing and control of the satellite's attitude are concerned – then bright objects being also the more extended ones are handled with (11.79) and (11.80) without risk of overflow. Hence overflow in the sums of (11.77) and (11.78) is infrequent as it requires the presence of a very extended object yet never reaching saturation.

  Considering the intrinsic quality of the barycentre as a location estimator, only 8 bits are allocated for the storage of the sub-pixel precision. An additional bit is used to perform round-off and avoid the truncation bias[21]. The final calculation, performed via a call to `mul32AndDivTo32` to handle the infrequent overflow on the first term of the numerator is then:

$$pos^{obj}_{AL} = \frac{\frac{2^9 \times \sum_{(i,j)\in object}(samp^{cor}_{i,j} \times i)}{\sum_{(i,j)\in object} samp^{cor}_{i,j}} + 1}{2} \tag{11.81}$$

$$pos^{obj}_{AC} = \frac{\frac{2^9 \times \sum_{(i,j)\in object}(samp^{cor}_{i,j} \times j)}{\sum_{(i,j)\in object} samp^{cor}_{i,j}} + 1}{2} \tag{11.82}$$

- Error analysis

  Overflow is avoided in almost all cases so, following the analysis in section 11.2.2, the errors spring from truncation. Further analysis would require characterising the distribution of the first terms in (11.81) and (11.82).

[21] The AL and AC coordinates being positive numbers, the bias may be systematically corrected by the application of (11.6).

11.4 Conclusion

The formulae developed in this chapter together with the corresponding numerical analysis meet the expectations corresponding to the driving considerations introduced in section 11.1: reproducibility of results across all platforms, precise control over numerical errors and compatibility with hardware. As such they are valid for hardware implantation (as chapter 12 will show) and provide the expected precision in software.

Emphasis has been placed on numerical quality both are regards setting upper bounds to numerical errors and for what concerns the production of unbiased estimates. While this is perfectly fine from a scientific point of view, it is worth noting here that it has a cost as far as implantation is concerned. It should be kept in mind that the rather high number of bits retained for the various operands as well as the additional steps introduced for bias correction translate into logic resources. As figure 8.1 in chapter 8 shows, the area occupied by operators increase exponentially with word length and incrementers are non-negligible contributions to the total. Should this latter aspect turn out to be critical, the procedures described here could be re-optimised with resources in mind, though necessarily at the expense of quality.

Chapter 12

Processing pipeline

Contents

12.1 Introduction

This chapter presents the implementation of the sample-based operations carried out for detecting the objects of interest in real-time. As such, it builds upon the descriptions of the underlying algorithms: pre-calibration (chapter 4), background estimation (chapter 5) as well as the simple object model (chapter 6). Other prerequisites include the numerical analysis driving the fixed-point implementation of all numerical stages (chapter 11) and the FPGA's overall architecture (chapter 9) which provides the facilities recalled in Tab. 12.3.

Briefly put, the specifications for this processing are to:

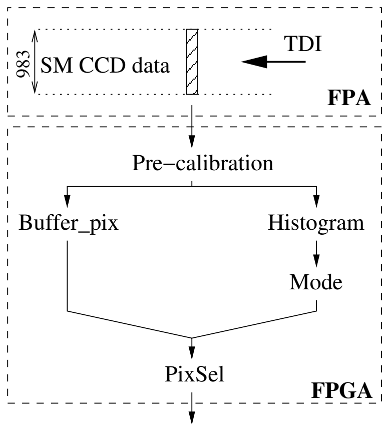
1. process the input data coherently with the acquisition rate: 983 16-bit sample values every TDI,
2. calibrate the image data on input,
3. estimate the local sky background,
4. threshold samples based on the estimated background and calibrated noise properties (uploaded from ground),
5. output the list of retained samples[1]: value (16 bits), position (AL and AC) and background estimate (32 bits).

As explained in chapter 9, two different FPGA families are targeted by this implementation: the first corresponding to the die present on our test platform (a ProASIC3E600), the second to the devices which could be flown on board the satellite (from the RTAX-S/SL family). The flash-based ProASIC3E being the official recommended prototyping FPGAs for the latter one-time-programmable ones and the same development environment being used for both

[1]This version of the pipeline is restricted to the simplified architecture. Besides a few architectural modifications (section 9.2.1), the complete one would additionally require to label connected components and manage accesses to SRAM0.

(Libero), the necessary adjustments for passing from one to the other are reduced. Being at a prototyping stage, the algorithms described in this document are primarily tuned for the ProASIC3E both in terms of available macros and resources and the items envisaged to require modification are discussed individually.

As opposed to the architecture adopted for Pyxis which favoured a functional description, the pre-calibration and the sample-based part of GD are here considered as forming a single sequence of operations. Tasks corresponding to the data flow are hence rather mapped to components according to Tab. 12.1.



Task	Component
Pre-calibration of the raw samples (linear correction and dead sample replacement)	$Pre-calibration$
Buffering of calibrated samples (until the background estimates become available)	$Buffer\_pix$
Constructing histograms on a hyperpixel basis (and localising the global maxima)	$Histogram$
Determining the modes of the respective distributions (through a quadratic profile adjustment)	$Mode$
Generating the background map (through bi-linear interpolation) Selecting the samples of interest (based on a SNR criterion)	$PixSel$
Control and command	$Scheduler$

Table 12.1: Data flow and task-to-component mapping for the sample-based detection engine.

Symbol	Meaning	Symbol	Meaning
$\|\|$	or	○	negation
$\gg$	shift bits right	$\ll$	shift bits left
$+$	add	$-$	subtract
$=$	equality comparator	&	concatenate
$>$	greater than	$<$	less than

Table 12.2: Symbols used in schematics.

12.2 Design considerations

12.2.1 Introduction

While software implementations consist in instructions being executed one after another by some processor[2], the design space for hardware implementations extends all the way between massively parallel and purely sequential schemes. Parallel computing, for instance, uses several independent units working simultaneously so that operations are performed concurrently and multiple results are produced at every cycle. Sequential operation, on the opposite, relies on a single unit to produce one result as a consequence of a sequence run every cycle. These two paradigms correspond to distributing the processing over "space" or "time" respectively to save the other resource. Intermediate solutions consist in decomposing computations in a series of stages with causal relationships – so that they must be performed one after the other – but independent calculations – so that each can be entrusted to a separate unit to have the successive stages run simultaneously. This structuring is known as pipelining and opens a wide range of trade-offs between "space" and "time", that is the physical resources used and the throughput, at the expense of latency (the delay before a given result becomes available). Fig. 12.1 illustrates the three approaches.

[2]This statement is somewhat simplified as opposed to multi-threaded applications running on several processors or as opposed to modern processor architectures with multiple cores and allowing to perform multiple operations at the same time on different units of the die. Nevertheless, the cost of synchronising the computations related to the lack of sufficiently precise timing predictions induces a flow which corresponds essentially to a graph of sequential processes. Accordingly, as compared to the possibilities offered by programmable hardware, the software flow remains dominantly sequential.

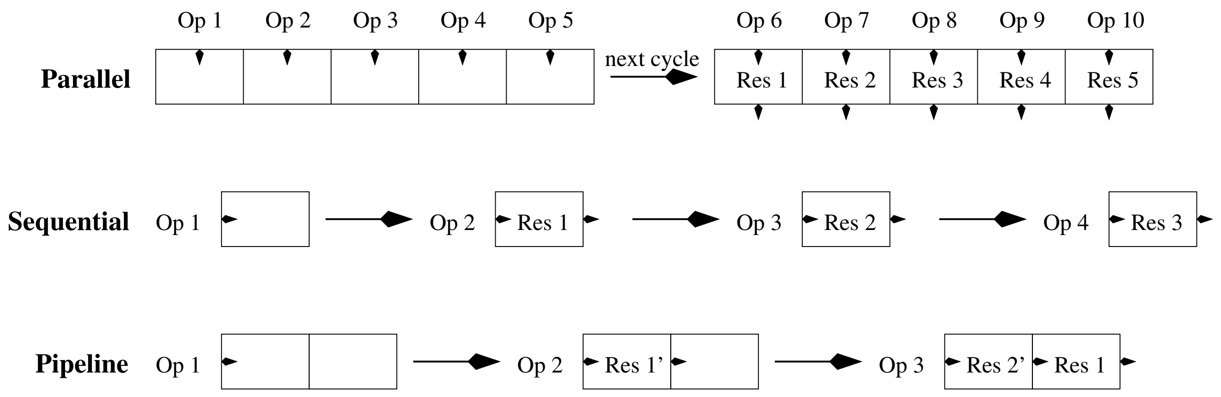


Figure 12.1: Illustration of parallel, sequential and pipelined execution schemes ("Op" stands for commanded operations and "Res" for the associated results ("Res 1' " for the intermediate results in the pipeline case), each box representing an independent processing unit).

12.2.2 Constraints

Tab. 12.3 briefly recalls the constraints which stem from the timing of CCD read-out, the selected hardware both for the target and test platforms and the overall architecture of the FPGA.

12.2.3 Pipeline

The characteristics presented in Tab 12.3 render both the massively parallel and the purely sequential approaches inadequate for the application considered here. The first because the complexity of the tasks to be carried out would imply relying on individual processing units of significant size replicated 983 times in order to process all the samples acquired within a single cycle lasting one TDI period. The corresponding needs in terms of resources would prove to be simply enormous. The other because it would correspond to performing the entire processing chain on a single sample in approximately $1\mu s$ (ie. one TDI period divided in 983 slots for each sample). This would impose very stringent timing constraints and lead to high operational frequencies and in turn to an important power consumption. Besides, from a logical point of view, neither are applicable since the treatment for each sample is not independent of the others because the background is evaluated on a regional scale.

Although anti-fuse FPGAs with a very large number of gates exist, the larger the circuit, the more important the electrical consumption and hence, the thermal dissipation – not to mention the fact that larger areas translate into more important interaction cross-sections with PPEs. The optimal "space/time" trade-off for a pipelined implementation should, hence, first and foremost fulfil the timing constraints, then do so with a minimum amount of resources.

Although the SM sample data is transferred from the FPA to the VPU in SpaceWire packets, 983 16-bit words at a time every TDI thanks to the important bandwidth available, it is preferable to have the FPGA fetch this data from the Video Buffer implanted on the VPU (which manages the asynchronous SpaceWire protocol, both more often and in smaller packets. Indeed such transactions impose relaxed constraints on both the interface and the FPGA than data bursts which require an important bandwidth and would leave the FPGA idle between transfers. A 16-bit wide interface with transfers approximately every micro-second then suffices and naturally defines the rate of the subsequent processing pipeline. It is then important to verify that the admission rate has limited variability in order to meet the hard real-time constraint related to the arrival of data on the VPU.

As mentioned above, $1\mu s$ is slow compared to the capabilities of the targeted FPGAs (advertised as capable of reaching 350 MHz) yet very demanding versus the number of tasks to carry out. A pipelined strategy allows for solving this difficulty by granting the different tasks several cycles to produce their result so long as they remain capable of admitting a new sample on input at every cycle. The latency induced is by no means a critical point for two reasons. First, because even in the worst case configuration the interval between the reception of SM2 data and their exploitation for configuring the AF1 CCD amounts to at least 400 TDIs[3]. Then, because the background estimation structurally introduces a latency corresponding to 48 TDIs compared to which the pipeline's remains negligible. The adopted structure, illustrated Fig. 12.2, is a pipeline on two levels: between the components listed in Tab. 12.1 (Pre-calibration → Buffer_pix → Histogram → Mode → PixSel) and within each.

[3]Size of the gap for the accommodation of the CCDs in the FPA.

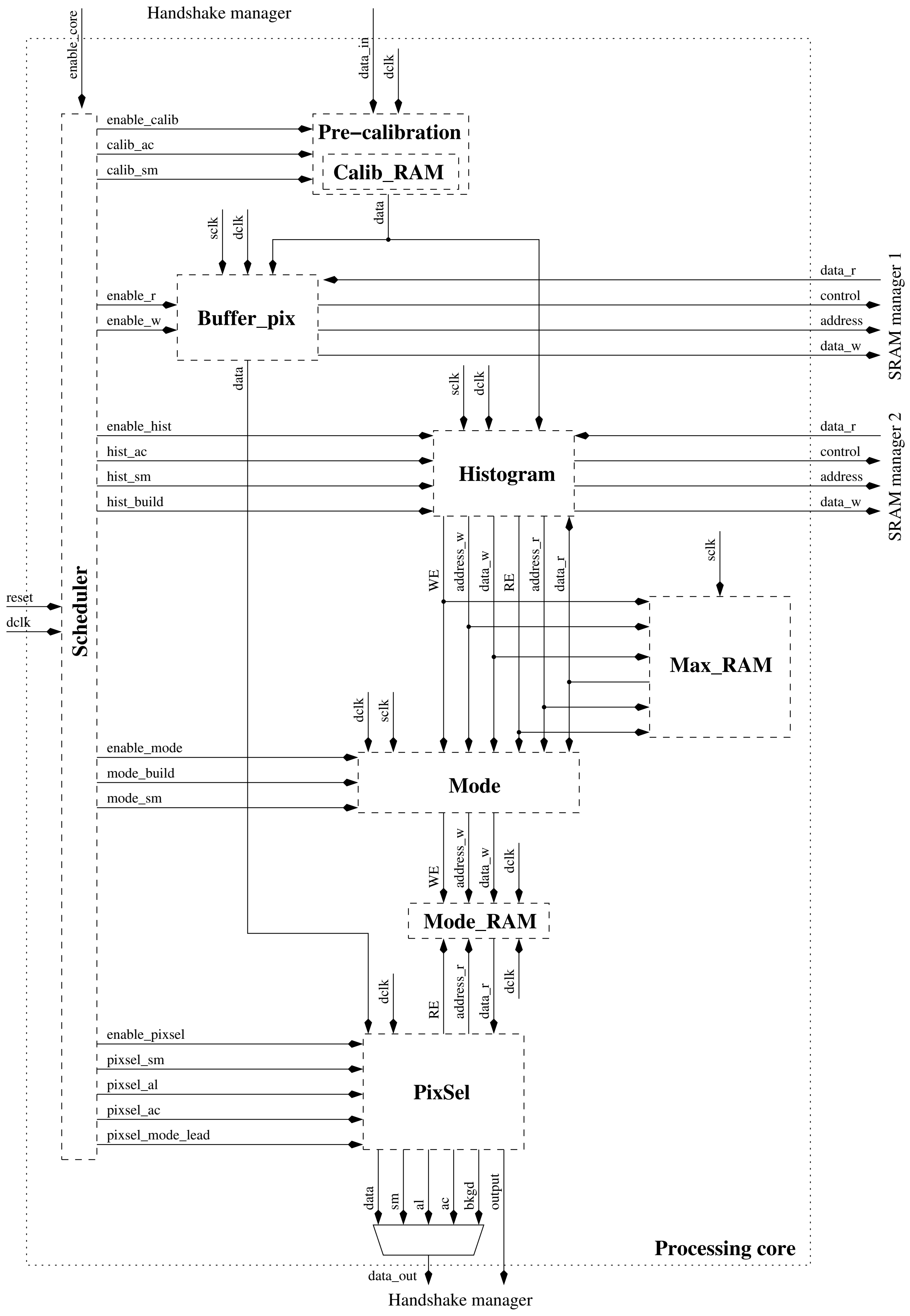


Figure 12.2: Internal structure of the processing core. For clarity, the reset network is not shown (but the signal is propagated to all components) and only input/output of clock signals are shown.

CCD read-out	
Data acquisition frequency (TDI period)	1017.5 Hz (0.9828 ms)
SM1/SM2 read-out	alternative
Data set size (ie. number of samples)	983
Data type	16-bit unsigned
RTAX-S resources (250S → 4000S) (from [59])	
System gates	250 000 → 4 000 000
Registers	1 408 → 20 160
Combinatorial	2 816 → 42 840
Flip-Flop	2 816 → 42 840
RAM	54 → 540 kbits
PLL	none
ProAsic3E 600 resources (from [60])	
System gates	600 000
D-Flip-Flop	13 824
RAM	108 kbits
PLL	6
Simplified architecture (from section 9.2.1)	
IO bus width	16 bits
IO protocol	handshake
IO maximum rate	10.5 MHz (95 ns)
External SRAM	2
Data bus width	16 bits
Address bus width	19 bits (8 MBit)
CLK	8 ns (125 MHz)
SCLK	32 ns (31.25 MHz)
DCLK	992 ns (1.008 MHz)

Table 12.3: Key design constraints.

12.2.4 Scheduling

There are many possible ways to make such a pipeline function in terms of interactions between the different stages. One possibility when the timing of individual nodes is imprecisely known or is insufficiently predictable consists in synchronising them based on the availability of results. Such a scheme was used, for instance, for the threaded implementation of GD in Pyxis 2.0. It, however, implied setting up a communication protocol at every interface, which in hardware would lead to a significant resource overhead. Fortunately, the internal scheduling of the FPGA may be determined both precisely and statically to make the timing fully predictable. The rationale for setting up the pipeline then suggests to run it synchronously with a clock governed by the availability of each new sample (DCLK).

Although one might think about making the pipeline and the components entirely DCLK-synchronous, this approach turns out not to be the simplest in terms of design, nor the most economical in terms of resources. Indeed, while it is natural to let the general pipeline be commanded by the arrival of new samples, the tasks comprise higher-frequency operations – for example, the sequence of steps necessary to format read or write requests on the external SRAMs described in chapter 10 – which, run at a lower rate, would considerably increase the depth of the tasks' pipelines and require many, possibly large, sets of registers for delay and synchronisation purposes. Accordingly, faster clocks have been introduced to carry these out at their natural frequency: SCLK for those requiring moderate speed (notably all accesses to the RAMs and SRAMs) or CLK for those calling for very fine timing (which is also the master clock from which all others are derived by division). In terms of architecture, this may be interpreted as introducing fast purely sequential units within the pipelines for performing these particular steps. Better still, whenever synchronous behaviour is not strictly required, purely combinational asynchronous paths have been introduced. With inputs and outputs respectively updated and sampled synchronously, the signals in these paths evolve to stationary states after a transient phase following a transition on the inputs. If the delay for sampling the outputs is sufficient, a complete sequence of operations may be carried out without the need for intermediate synchronisation and registers. Two examples of this can be found for Pre-calibration in section 12.3.2.

A segmented implementation was preferred to setting up a monolithic pipeline at the expense of some amount of control logic. While, indeed, a monolithic DCLK-synchronous pipeline could reduce the surrounding logic to the clock and reset distribution trees, all tasks would become operational during the first DCLK cycle. This would call for subtle initialisation of all registers, taking into account the data propagation delays through the pipeline, to ensure that they would reach the desired state upon the arrival of the first relevant data word. Additionally, from a design and debugging point of view, this approach calls for an update of all initialisation parameters downstream of the latest modification affecting the scheduling – something complex and error-prone. Instead, a segmented pipeline allows for triggering the various stages just in time for the arrival of the relevant data words and for centralising the control logic in a well-identified scheduling module, with the additional benefit of rendering independent validation of the stages possible. This segmentation is effective through the use of a hierarchy of components which only proceed through the next cycle if enabled by the Scheduler (use of an enable signal or clock gating).

12.2.5 RAM

The pipeline has memory needs both for storage and for communication. They are satisfied by relying on either internal or external resources depending on the required storage space, ease of use and availability. The details of the addressing are provided as part of the description of each component in section 12.3 but we present the overall management of access conflicts and the total usage assessment here.

The timing of accesses to the various memory resources is best defined statically to ensure their availability and share them optimally between components or processes. Indeed, whether they serve for communication or for storage only, rigorous time-sharing is the key to the fully predictable access delays necessary to achieve fixed scheduling with a minimum amount of control logic. As Tab. 12.4 shows, the available resources are not used by more that two components at any time so exclusive accesses may can be granted by distributing them based on DCLK's high or low states respectively. As a result, each component's access to the resource is exclusive between edges and only limited by the number of times the response delay may be fitted within half a DCLK period. It is worth noting, that the time slots allocated in Tab. 12.4 impose a number of constraints on the structuring of the different pipeline sequences – notably in the sense of grouping read and write accesses within a same DCLK phase or because of the introduction of delays until the memory resource can be accessed.

Resource	DCLK high		DCLK low	
	Component	Accesses	Component	Accesses
SRAM 1	Buffer_pix	1 Read / 1 Write (fetch & store)		
SRAM 2	Histogram	5 Write (4 bins & 1 reset)	Histogram	12 Read (4×3 bin populations)
`Calib_RAM`			Pre-calibration	2 Read (a_i & b_i)
`Max_RAM`	Mode	1 Read / 1 Write (fetch & reset)	Histogram	1 Read / 1 Write (fetch & store every 32 DCLKs)
`Mode_RAM`	PixSel	4 Read (interpolation nodes)	Mode	1 Write (store every 4 DCLKs)

Table 12.4: Memory operation.

The ProAsic3E chips offer a variety of RAM types which facilitate their use by either extending access possibilities or segmenting them (Tab. 12.5). The dual-port RAM offers the possibility to read and write from both ports and, hence, do so concurrently at different addresses of the same module. Pre-calibration thus fetches both calibration coefficients simultaneously from the internal RAM (section 12.3.2). On the contrary, the two-port RAM separates read and write accesses so that, when used for communication, each port may be wired to a different components with a simplified access protocol. This is the case for the accesses to the background estimates, in `MODE_RAM` in Fig. 12.2, written by Mode (section 12.3.5) and read by PixSel (section 12.3.6) for example.

The third internal RAM resource, `MAX_RAM` in Fig. 12.2, dedicated to the storage of the information relative to the histograms' global maxima (section 12.3.4), needs to be read and written both by Histogram (for updating the max data) and Mode (for the mode calculation and resetting the max data), though not at the same time according to Tab. 12.4. Some control logic must accordingly be inserted to define which component drives the RAM's port at each instant. With accesses segregated in time, this logic could well have been DCLK-synchronous, granting exclusive access to one during the low phase and to the other during the high one. Instead, for improved efficiency, the need being to sort accesses between the components when requests are made – assuming no conflicting ones are made as verified through `assert` statements – two `XOR` gates together with three multiplexers are used to connect the RAM ports as illustrated in Fig. 12.3.

Resource	Data (bits)	Address (bits)	Timing	Structure
SRAM 1	10/16	94368 (17/19)	SCLK	external
SRAM 2	16/16	126976 (17/19)	SCLK	external
ProASIC3E600				
`Calib_RAM`	15 bits	3932 (12 bits)	SCLK (Rr, Wr)	dual ports: 15 4096 $\times$ 1 blocks
`Max_RAM`	36 bits	256 (8 bits)	SCLK (Rr, Wr)	two ports: 2 256 $\times$ 18 blocks
`Mode_RAM`	22 bits	132 (8 bits)	SCLK (Rf, Wr)	two ports: 2 256 $\times$ 18 blocks
Total	19 RAM blocks (24 available)			.
RTAX-S (with EDAC)				
`Calib_RAM_A`	16 bits	1966 (11 bits)	SCLK	two ports: 16 128 $\times$ 36 blocks
`Calib_RAM_B`	16 bits	1966 (11 bits)	SCLK	two ports: 16 128 $\times$ 36 blocks
`Max_RAM`	36 bits	256 (8 bits)	SCLK	two ports: 3 256 $\times$ 18 blocks
`Mode_RAM`	22 bits	132 (8 bits)	SCLK	two ports: 2 128 $\times$ 36 blocks
Total	37 RAM blocks (36 available for 1000 & 4000, 64 for 2000)			.

Table 12.5: Memory resources and usage. The bus widths correspond to use versus available resources. As a result of out efforts to achieve a "slow" system, all memories are accessed at the SRAM rate – although the two external SRAMs are asynchronous in nature, the indicated timing corresponds to accesses times through the memory controllers (chapter 10). The RAMs in our test platform being synchronous, the timing information is complemented by the indication of the active edge (R, W stand for Read, Write and r,f stand for rising, falling). Setup and hold times can then guaranteed by changing the address or data words on the other edge.

The RTAX-S technology, on the opposite, offers a single type of two-port RAM with the same aspect ratios as for ProASIC3E chips and with optional EDAC correction [61]. This latter option, which ensures reliability of the RAM content without the need for TMR, imposes constraints on word sizes and increases response delays because of the underlying coding-decoding-scrubbing logic. With manual tuning of the modules generated by SmartGen, it is possible to achieve widths in the $[8;12] \cup [16;29] \cup [32;47]$-bit domain. Besides, it is not then possible to cascade more than 16 blocks (for the RTAX2000S) because they must physically be in the same column. As a consequence of these limitations, the dual port `CALIB_RAM` would need to be split into two 16-bit-wide two-port RAMs placed in different columns, while the `MAX_RAM` and `MODE_RAM` would transpose more naturally. With a total of 37 RAM blocks used and the need to cascade 16 blocks for the pre-calibration coefficients, only the RTAX2000S chips are found to meet our needs[4]. Regarding timing, the EDAC RAMs are shown to be operated at 10 MHz in [61]. While this is not critical as only few accesses need to be made per DCLK period, a dedicated clock would need to be introduced to manage these delays. A natural possibility would consist in further dividing the SCLK introduced for the external SRAMs or the existing FSMs could be modified to insert additional wait states to tune accesses to the RAMs.

12.3 Components

The complete pipeline illustrated Fig. 12.4 is segmented in a series of smaller ones corresponding approximately to functional blocks. Each of these stages is the object of a component declaration in VHDL, essentially for convenience in the description and configuration of the design but without impact on the final netlist since the hierarchical structure is entirely flattened to allow cross-component optimisations in the process of synthesis. This section describes the structure and operation of each of these stages.

12.3.1 Scheduler

Functional

The Scheduler is in charge of ensuring the coherence of the processing performed by the pipeline's various stages. This not only implies managing the delays related to the algorithms or to the propagation through the pipeline, but also keeping in phase with the input data as transmissions from the video buffer may fail or as synchronisation between the VPU and the FPA might cause data to be unavailable for one or more DCLK cycles. The corresponding functional specification is therefore:

1. Control the operation of each component: either freeze it or let it proceed through the next DCLK cycle:

[4]These two constraints could however be relaxed by placing the pre-calibration coefficients in one of the external SRAMs and introducing control logic to skip the accesses to the dummy mode values, thereby reducing the number of MODE_RAM addresses to 128 or 124 and the total number of RAM blocks needed to only 4, thereby rendering the use of a RTAX1000S die possible.

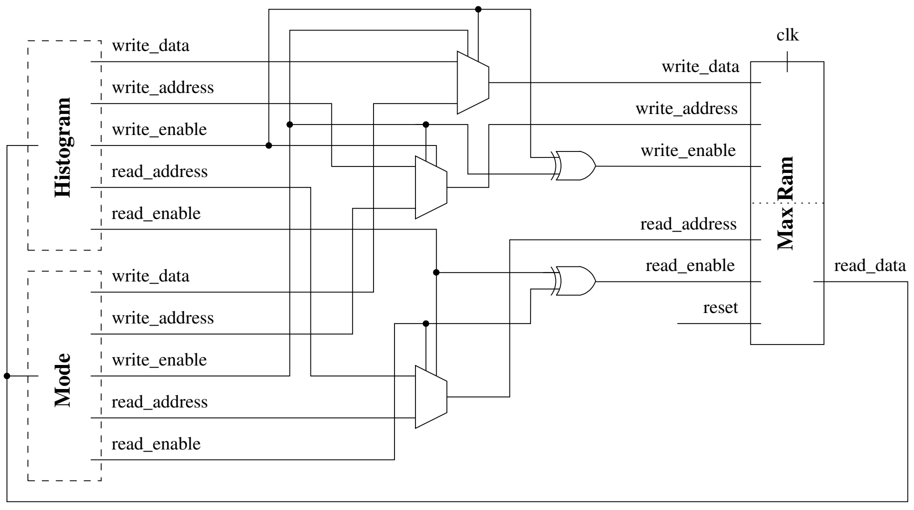


Figure 12.3: Control logic for the drivers to the MAX_RAM ports (the ports of Histogram and Mode are only incompletely represented).

(a) manage initialisation delays until relevant data propagates through the pipeline,

(b) suspend activity when handshake transactions on input or output fail,

(c) manage components not permanently in operation (Mode).

2. Provide identification for the input data and parameters at every stage of the pipeline:

(a) Pre-calibration: input sample's AC coordinate and SM origin.

(b) Buffer_pix: none.

(c) Histogram: input sample's AC coordinate and SM origin, histogram series to build and reset.

(d) Mode: cycle index, mode series to build and SM origin.

(e) PixSel: buffered sample's AL and AC coordinates and SM origin, leading mode series.

Control over the operation of the various components may be achieved in essentially two ways. Either through the determination of an `ENABLE` signal propagated to each component and used to block the synchronous processes, or via clock gating. The former is the one used so far, as can be seen on Fig. 12.2, for reason of simplicity. Yet its significant cost, in terms of the necessary feedback multiplexers introduced to maintain the values of registers, suggests to move to clock gating which would also yield an improvement in power consumption. With three clocks introduced, however, knowing how critical skew is on such signals, this evolution is postponed to a more mature stage of the design.

Discussion

The Scheduler's outputs being functions of time, or more exactly of the number of successful handshake transactions since the last reset, all may in principle be determined through combinational logic from a single general counter, provided it is only incremented when input and output communications are successful. This approach, appealing because conceptually simple, would essentially translate into a 17-bit counter (to account for the enable condition on reading the buffer and performing sample selection: $128 \times 983 = 125824$) with subsequent operations running essentially modulo 128 TDIs and some logic to derive the control signals. The amount of resources required for this latter part depends very much on the nature of the arithmetics to be carried out. Unfortunately, the AC size of the CCDs being 983 samples, many rather expensive "modulo 983" operators are to be expected in these calculations. Assuming some component were selected as an AC reference count, itself determined through one such operator, the determination of the other components' AC parameters by taking into account the propagation delay with the reference would call for adders or subtractors modulo 983. Furthermore, unless it were determined from the AC reference count above, the determination of AL coordinates would call for dividing the 17-bit counter value by 983, an expensive operation in terms of hardware.

Alternatively, a fully decentralised scheme is possible in which counters modulo 983 are maintained for each component. In this particular case, the logic generated remains simple since the operation can be based on a multiplexer

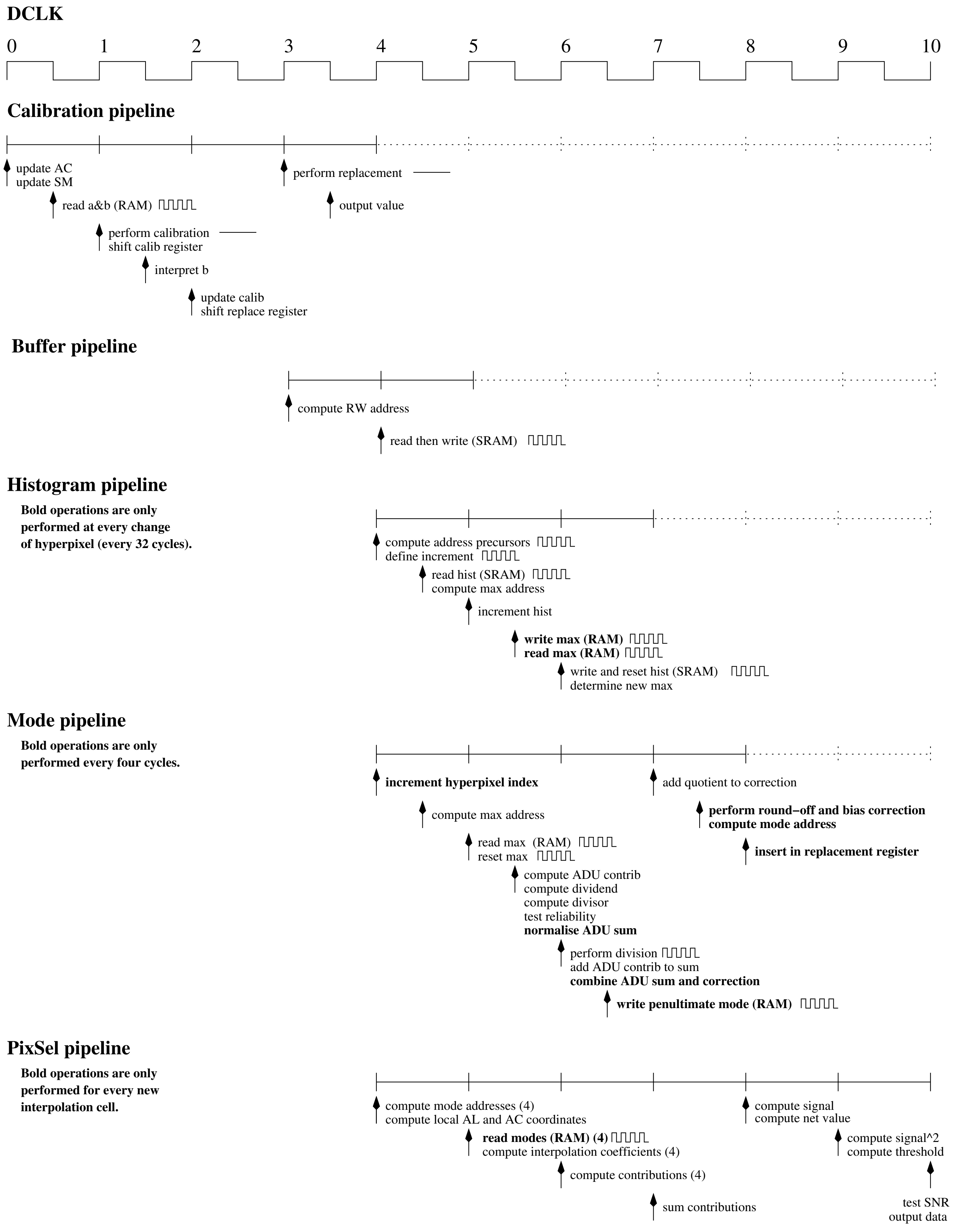


Figure 12.4: Complete core processing pipeline with propagation delays. Operations are by default DCLK-synchronous but faster ones controlled by SCLK, as discussed in section 12.2.4, are indicated with a clock waveform. Conversely, combinational paths are marked with a flat waveform representing the absence of underlying clock.

Component	Parameter	Start	Range	Period
Pre-calibration	`ENABLE_CALIB`		delay: 1 DCLK (handshake latency)	
	`CALIB_AC`	0	modulo 983 ($[0;982]$)	DCLK
	`CALIB_SM`	0	modulo 2 ($[0;1]$)	TDI
Buffer_pix	`ENABLE_W`		delay: 4 DCLK	
	(trigger write)		(handshake + Pre-calibration latency)	
	`ENABLE_R`		delay: 127 TDIs + 983 DCLK	
	(trigger read)		(two complete hyperlines in SM1)	
Histogram	`ENABLE_HIST`		delay: 4 DCLK	
			(handshake + Pre-calibration latency)	
	`HIST_AC`	0	modulo 983 ($[0;982]$)	DCLK
	`HIST_SM`	0	modulo 2 ($[0;1]$)	TDI
	`HIST_BUILD`	0	modulo 2 ($[0;1]$)	64 TDI
	(histograms to build)			
Mode	`ENABLE_MODE`		run for 131 DCLK when TDI > 2	
			and TDI $\equiv 0(mod\ 64)$ (end of hyperline in SM1)	
			or TDI $\equiv 0(mod\ 65)$ (end of hyperline in SM2)	
	`MODE_AC`	0	$[0;130]$	DCLK
	(cycle counter)			
	`MODE_SM`	0	$[0;1]$	0 when TDI $\equiv 0(mod\ 64)$
				1 when TDI $\equiv 0(mod\ 65)$
	`MODE_BUILD`	0	modulo 2 ($[0;1]$)	64 TDIs
	(modes to determine)			
PixSel	`ENABLE_PIXSEL`		delay: 127 TDIs + 983 DCLK	
			(two complete hyperlines in SM1)	
	`PIXSEL_AL`	0	modulo 32 ($[0;31]$)	TDI
	`PIXSEL_TDI`	0	modulo 128 ($[0;127]$)	TDI
	`PIXSEL_AC`	0	modulo 983 ($[0;982]$)	DCLK
	`PIXSEL_SM`	0	modulo 2 ($[0;1]$)	TDI
	`PIXSEL_MODE_LEAD`	0	modulo 2 ($[0;1]$)	64 TDIs
	(most recent mode series)			

Table 12.6: Scheduling parameters.

which depending on the comparison to a reference word[5] either leads to incrementing the counter or resetting it to 0. The expense, this time, consists in the number of counters necessary and some control logic for synchronisation but their respective dynamics may be adjusted according to each component's scheduling needs.

The approach adopted borrows much from this latter concept in the sense that multiple counters are used, except that instead of keeping components fully independent, the causal constraints imposed by the algorithmic sequence are exploited to derive some of the outputs from others to mutualize the logic. Tab. 12.7 summarises the resources used while Fig. 12.5 illustrates the resulting waveforms.

12.3.2 Pre-calibration

Functional

A pre-calibration step is introduced to enforce the validity of basic imaging assumptions on the CCD data in spite of the detectors' defects and ageing. The corresponding rationale and procedure are discussed in chapter 4, and the following functional specifications result:

1. Perform a linear correction which generalises the usual bias and offset corrections applied to CCD data (PRNU and DSNU):
   (a) use uploaded parameter LUTs for gain and bias corrections,
   (b) manage saturated sample values (saturation threshold is assumed uniform on the CCD): saturated values on input must remain saturated, handle saturation as a result of calibration.

[5]The one corresponding to the value 982.

Parameter	Resource & logic
	Pre-calibration
ENABLE_CALIB	std_logic determined from CALIB_AC then resets it.
CALIB_AC	AC counter (983 values on 10 bits).
CALIB_SM	std_logic negated when CALIB_AC is reset to 0.
	Buffer_pix
ENABLE_W	std_logic determined from HIST_AC.
ENABLE_R	std_logic determined from PIXSEL_TDI & PIXSEL_AC.
	Histogram
ENABLE_HIST	std_logic determined from HIST_AC then resets it.
HIST_AC	AC counter (983 values on 10 bits).
HIST_SM	std_logic negated when HIST_AC is reset to 0.
HIST_BUILD	std_logic negated when Mode's run for SM2 begins.
	Mode
ENABLE_MODE	std_logic determined from MODE_AC & MODE_AL counters.
MODE_START	boolean flag for synchronising MODE_AC with Histogram's: determined from MODE_AC then resets it.
MODE_AC	AC counter (983 values on 10 bits). (also serves as a cycle counter).
MODE_SM1	boolean flag to process SM1 modes: determined from MODE_AC & MODE_AL (start), determined from MODE_AC (stop).
MODE_SM2	boolean flag to process SM2 modes: determined from MODE_AC (start and stop),
MODE_SM	std_logic negated when MODE_SM2 becomes true.
MODE_BUILD	std_logic negated when MODE_SM1 becomes false.
	PixSel
ENABLE_PIXSEL	std_logic determined from PIXSEL_AC & PIXSEL_AL.
PIXSEL_START	boolean flag for synchronising PIXSEL_AC with Histogram's: determined from PIXSEL_AC then resets it.
PIXSEL_AL	AL counter (128 values on 7 bits): incremented when PIXSEL_AC counter is reset to 0. (bits 1 to 6 are output as AL position within the hyperline).
PIXSEL_AC	AC counter (983 values on 10 bits).
PIXSEL_SM	std_logic negated when PIXSEL_AC is reset to 0.
PIXSEL_MODE_LEAD	std_logic negated copy of HIST_BUILD.

Table 12.7: Scheduler's resources and logic (within a DCLK-synchronous process).

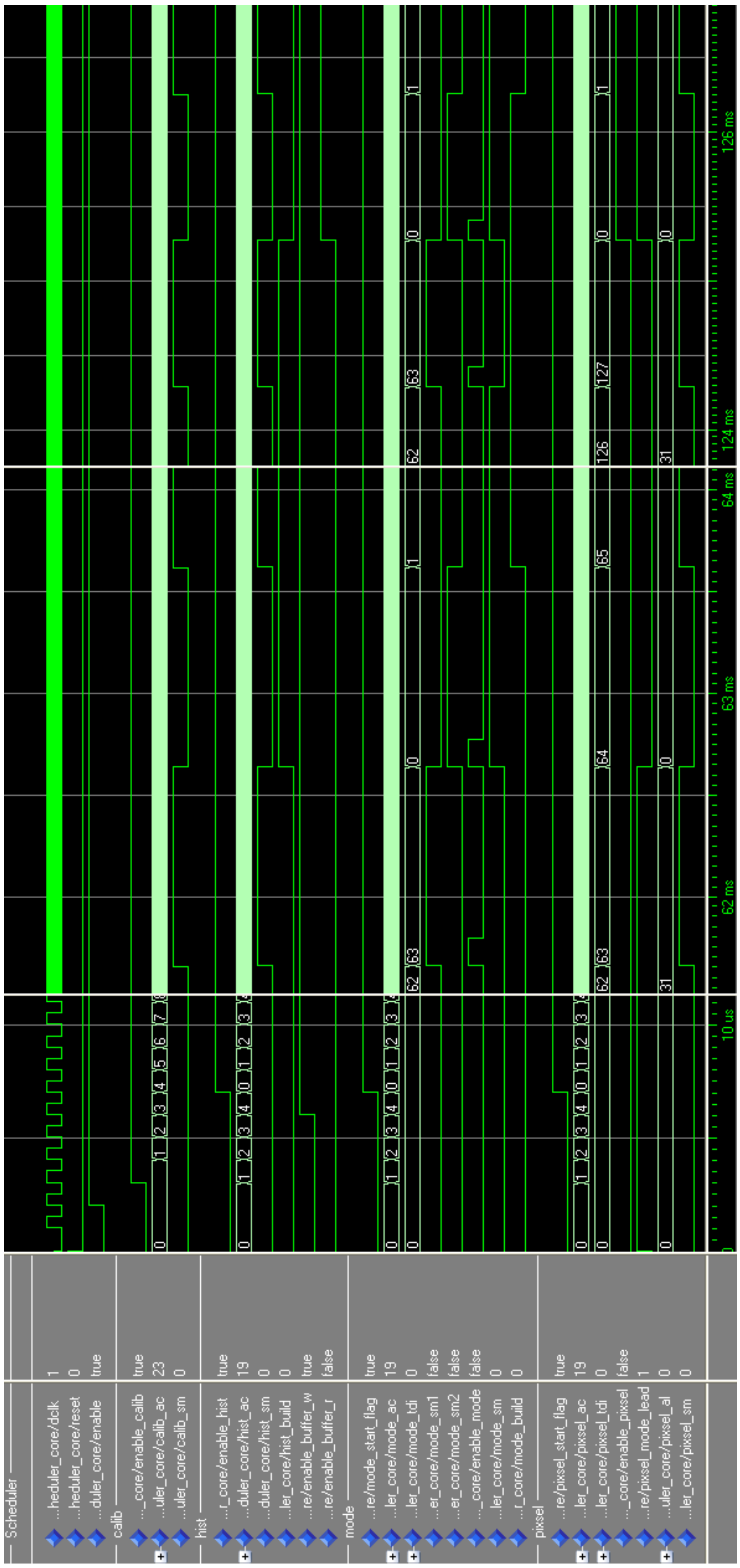


Figure 12.5: Waveform of Scheduler's signals. Three fragments of the chronology are shown: the initialisation phase, the computation of the first series of modes for SM1 and SM2 respectively and the second series with the activation of PixSel.

(c) manage the risk of underflow (mostly in cases when the calibration parameters are outdated versus the state of the CCD).

2. Replace unreliable values with estimates derived from neighbouring samples[6]:

   (a) use calibrated values as a basis for the estimates,

   (b) manage all possible dead sample configurations to ensure "graceful degradation" (as per ECSS).

3. Use the same LUTs for calibration and identification of unreliable samples (uploaded from ground).

Beside the general principles exposed in chapter 4, the details of the computation allowing to obtain ADU-precise values for the entire dynamics of the detectors are presented in chapter 11 so that the focus here is exclusively placed on deriving an RTL architecture for these computations.

RAM organisation

As a result of the analysis exposed in chapter 11 and to maintain bit-wise correspondence between the hardware and the software implementation (Pyxis), values of the calibration coefficients $(a_i, b_i)_{i \in \{0,\ldots,982\}}$ are restricted to 15 signed bits to ensure that the result of the computation does not exceed 32 bits (software constraint). The range in which calculations are performed could nevertheless be easily extended in hardware were the need to manage larger effects to arise. Tab. 12.8 summarises the properties of the resulting data set.

	a_i	b_i
	Signed	Signed
Number of bits	15	15
Number of entries per line	983	983
Number of CCDs	2	2

Table 12.8: Pre-calibration coefficients.

The addressing may then be very simply reduced to the concatenation of the SM identifier, the AC position of the sample (provided by the Scheduler through Pre-calibration's ports) and a bit statically defined for either a or b (Fig. 12.6). Thus, the coefficients' address words are formed directly by wiring the proper signals together without the need for logic. Further streamlining of RAM accesses is obtained by noting that read accesses dominate largely[7], so that both coefficients may be stored in the same module and read out simultaneously by relying on a dual port RAM. Such an organisation, as it ensures equal read times, simplifies the placement and routing to and from the RAM module on the ProAsic3E chip (but is by no means mandatory).

The resulting data set is not consecutive in memory due to unused values for AC $\in \{983, \ldots, 1023\}$. While this might imply a performance loss in software where pre-fetch operations are scheduled to fill the cache and reduce average access times, this is not a concern here since all operations on both the internal and external RAM are performed just in time and manually.

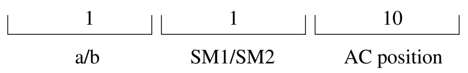


Figure 12.6: Structure of the calibration coefficients' address words (12 bits).

Pipeline

The above functional specifications clearly suggest to structure Pre-calibration as a pipeline composed of two processes: linear calibration and sample replacement. Inserting a sufficient amount of registers in between allows for making their respective timings independent at the cost of a small increase of resources and latency. As Fig. 12.7 illustrates it, a single DCLK period is sufficient to that end. The timing constraints being easily met with the technology used and to limit the overall depth of the pipeline, the linear calibration and replacement mechanisms themselves have been implemented combinationally, with stabilised outputs sampled DCLK-synchronously.

[6]Border effects are expected to be taken into account when defining the replacement strategy on ground.
[7]Write accesses to the RAM would only occur when the coefficients are updated from ground.

As explained in chapter 11, to optimise the use made of the available memory bits for the storage of a and b, the calibration calculation consists in determining the correction to be applied to the raw value instead of directly computing $a_i \times samp^{raw} + b_i$. While the latter formulation would allow for easily marking samples values to be replaced by relying on $a_i = 0$ and giving b_i a recognisable value before testing the corrected sample value themselves, the one used does not permit altogether cancelling the sample's response because of the limited dynamics available for the correction. This calls for storing the value of b_i, or more precisely after interpreting it, the relevant three bits (for five different cases) in a two-cell shift register. Using values at the end of the positive range which are the least likely to be of use for calibration, the interpretation of b_i to produce the identifiers above reduces to a bit-wise comparison with the mask "0111111111111" concatenated with b_i's two least significant bits.

- (0 - -) – Maintain the calibrated value (other values of b_i).
- (100) – Replace by a constant ($b_i = 16380$).
- (101) – Propagate AC - 1 value ($b_i = 16381$).
- (110) – Propagate AC + 1 value ($b_i = 16382$).
- (111) – Average of AC + 1 and AC - 1 neighbours ($b_i = 16383$).

In an effort to ensure equal timing in all cases and reduce the amount of dedicated logic for each, two multiplexers are inserted to define the values of the input signals. The replacement operations themselves may then be performed through the same path in spite of their apparent diversity (Fig. 12.7). Indeed, all five cases can be brought under the same formulation if A and B (the output of the multiplexers) are permitted to be equal: $\frac{A+B}{2}$.

12.3.3 Buffering

Functional

Due to the fact that the CCDs are read-out line per line, the estimation of the sky background on a regional scale introduces a delay before the sample values can be interpreted. This delay amounts to precisely $96 = (2 \times 32 - 16) \times 2$ TDIs because of the two SMs, the need to rely on two consecutive hyperpixels for interpolation and the offset of the interpolation grid versus the hyperpixels. Buffering of the calibrated sample values for this duration and both SMs calls for storing $2 \times 48 \times 983 \times 16 = 1509888$ bits, hence, necessarily in external memory. Functional specification for the buffering engine is then:

1. Save SM1 and SM2 samples values to the buffer after pre-calibration.
2. Fetch SM1 and SM2 samples values from the buffer for PixSel.

Pipeline

Two essentially different approaches may be adopted to fulfil the buffering needs depending on how one chooses to address data in the buffer. A first one would consist in implementing an addressing engine capable of determining them based on the samples' identification: SM origin, AL and AC coordinates. While requiring some amount of arithmetics for its implementation, this method would allow for random access to data in the buffer, in good agreement with the random access capabilities of the external SRAM. Conversely, the sequential nature of the samples' data flow and the fixed scheduling inside the FPGA make it possible to simply read and write the sample values serially, much as one would on a tape recorder. This latter option is the preferred one as it trades the unneeded flexibility of random access against much reduced and simplified logic. Addressing is then merely a matter of counters.

Two considerations allow for further streamlining the buffering process. The first, relevant to the alternative read-out of the SM CCDs every TDI and the fixed scheduling of the FPGA, implies that there is no necessity for relying on distinct storage spaces for the two SMs. Instead, the scheme may be entirely serial, meaning that samples can be stored one after the other, irrespective of their origin – the first from SM2 after the last from SM1 (from the previous TDI). This translates into reducing the number of counters by a factor two. The second is relative to the equal read and write access rates and the constant delay between them. Structuring the buffer as a circular one of exactly the required size and ensuring that the delay between read and write operations corresponds to an integer number of lines, the same address can be successively read (for extracting the previously buffered value) then written (for buffering a new value) so that a single counter (modulo the size of the circular buffer) suffices for all buffer accesses.

Operation then simply consists in incrementing the address counter (17 bits), then performing successive read and write accesses to this address through the controller (chapter 10) corresponding to the selected SRAM (SRAM 1). For maximum simplicity, the two are organised as a two-level DCLK-synchronous pipeline which updates the address

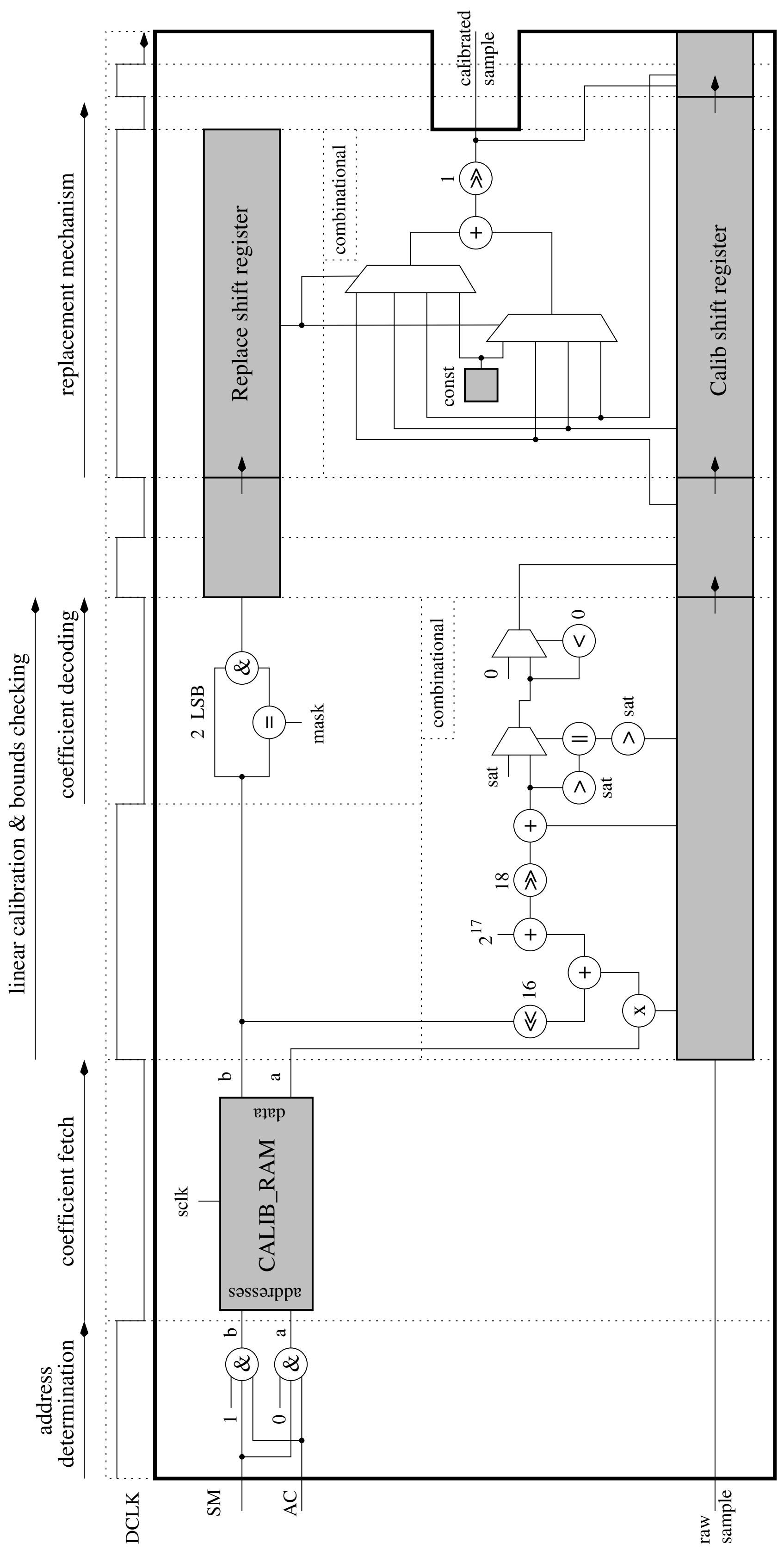


Figure 12.7: RTL representation of the Pre-calibration component and associated clocking (the logic introduced to round-off the equidistant cases ((11.23) and (11.24)) is not shown).

during the first cycle, then hands control over to a FSM driven by the SCLK for the read and write sequence. Given how ample time is available for this, accesses are performed only while DCLK is high to leave the SRAM fully available for other operations during the low part of the cycle and comfortable delays are introduced in the form of intermediate wait states to ensure all signals are stabilised. Fig. 12.8 illustrates the resulting FSM.

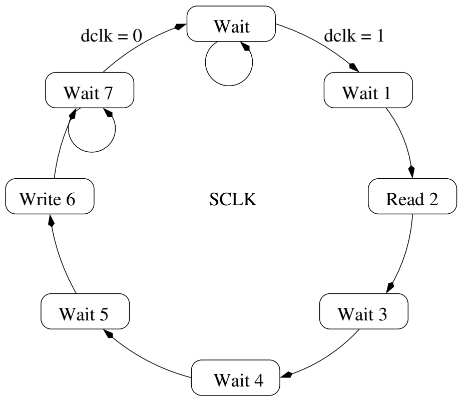


Figure 12.8: SRAM buffer access FSM (all transitions are driven by the rising edges of SCLK).

12.3.4 Histogram

Functional

The Histogram component gathers the data used for building the background statics as explained in section 5.2.1. To this end, it counts the occurrences of calibrated sample values on a regional basis, the hyperpixels, which form a 32×32 AL×AC grid starting at position $(0, 0)$. For SNR reasons, the values are lumped together in bins of four ADUs in all possible ways, so that four 4-ADU histograms are built instead of one, with unchanged storage requirements. Scanning all histograms in search for each global maximum prior to the Mode calculation would call for numerous read accesses amounting to $30 \times 1024 \times 32 = 983.94$ μs to be carried out while data from the other SM is received (one TDI). This would result in crowded accesses to the SRAM, particularly since the other SM's histograms are also being built during this interval. It is, hence, desirable – though at the cost of a notable increase of complexity – to track the maxima as the histograms are being built (max data hereafter). To further simplify the access management to the SRAM, the populations of the maximum and adjacent bins are also recorded to allow for determining modes without direct reference to the SRAM's data to, hence, grant Histogram exclusive access to it. The following functional specifications result:

1. Build sample value histograms :
   (a) identify hyperpixel from AC coordinates,
   (b) increment bins in all four histograms for sample values in the $[936; 1960]$ ADU interval,
   (c) reset histograms prior to recycling.
2. Track global maxima for all histograms:
   (a) record positions of global maxima,
   (b) record populations of global maxima and the two adjacent bins,
   (c) publish to Mode.

Although simple in principle and from an arithmetics point of view, these specifications cover the component with the most complex data management in this chapter. The need to manage 120 hyperpixels – 30 being built and 30 reset for each SM – to access both external (for histograms) and internal memory (for max data), to handle border effects, to increment histograms and finally record the maxima's positions simultaneously lead to an intricate pipeline design with, unfortunately, many special cases. During the design of this pipeline, focus was placed on achieving an exact implementation (mainly for the purpose of allowing bit-wise validation versus the software model) but it is likely that accepting a few sources of well-controlled deviations from exactitude would lead to an appreciable reduction of the underlying amount of resources by discarding some seldom used logical paths.

RAM organisation

Histograms

The combined will to base the background estimates on sufficient statistics while preserving their regional significance leads to forming two-dimensional hyperpixels. In a processing framework entirely line-based, their extending over several lines implies that all 30 must be built simultaneously, not only for one SM but for both. Each region encompassing 1024 samples and the background dynamics extending over 1024 ADUs, 60 histograms must thus be managed at the same time, each corresponding to 256×10 bits for a total of 60 kbits of data. This amount of resources greatly exceeds what can be implemented locally in the form of registers, so that storage external to the component is mandatory with the additional difficulty that the histogram bins need then be reset sequentially.

Placed into internal RAM, CLK-synchronous operation would provide $983 \times \frac{992}{8} = 121892$ cycles sufficient compared to the $256 \times 120 = 30720$ addresses to be reset[8], so that it would be possible to prepare all histograms for recycling during the TDI devoted to the other SM. Although the larger RTAX-S devices might permit that, for the ProASIC3E die mounted on our test platform dedicating 60 of the 108 kbits[9] of internal memory to the histograms is incompatible with the allocation to the calibration parameters presented in section 12.3.2 – not to mention the other intended uses for the maxima of the histograms and the background mode estimates. Conversely, the access rate to an external SRAM with a total of $983 \times \frac{992}{32} = 30473$ cycles, falls short for this. Fortunately, the greater available storage space offers the possibility to use two separate series of histograms – one being built, the other one reset – if they are exchanged between hyperlines. This is the solution retained but it is worth mentioning that substantial simplification could be achieved by placing the histograms in internal memory since as will become apparent in what follows.

With the need to record the position and population of the maxima and adjacent bins, the following operations need to be carried out with the four re-binned histograms for each input sample value:

I. Fetch the latest bin count: 4 reads (one per histogram).

II. Store the incremented bin count: 4 writes (one per histogram).

III. Fetch the adjacent bin counts: 4×2 reads.

IV. Reset one bin in the unused histogram series: 1 write.

The corresponding accesses are controlled by means of a FSM represented Fig. 12.11 and are the key driver to the overall clocking scheme discussed in chapter 9 and recalled in Tab. 12.3. Indeed, a sufficient number of SCLK cycles must fit into one DCLK period. The duration of the CLK period is also constrained at this level since it is a contributor to the SCLK period as per chapter 10 and congruence relationships are desirable to derive both the SCLK and the DCLK by division.

As for Pre-calibration in section 12.3.2, the structuring of the data in the SRAM is designed to optimise address calculations. In this case, the address word is decomposed into a series of fields before being aggregated according to needs as shown in Fig. 12.9. The padding and binning index are determined statically while the series and SM fields need only be routed from the ports. Logic is then only required for the determination of the hyperpixel index and the bin index (or position) in the histogram. Particular care is taken to optimise the resources used for these calculations. With k the index of the different histograms, (12.1) translates into mere wiring[10] while (12.2) is reduced to using a 11-bit subtractor and a bit shift[11].

$$\begin{array}{ccl} \text{Index of:} & \text{Rationale} & \text{VHDL} \\ \text{hyperpixel} & E(HIST\_AC/32) & \rightarrow HIST\_AC(9\ downto\ 5) \qquad (12.1) \\ \text{bin index} & \frac{PIX-936-k}{4} & \rightarrow (PIX(10\ downto\ 0) - offset[k]) \gg 2 \qquad (12.2) \end{array}$$

Max data

As for the histograms, the global maxima need to be tracked simultaneously for all hyperpixels and the amount of data to be recorded forbids relying on local registers. They may, however, be stored in internal memory for simplicity's sake. Given that the maximum's position in the histogram and the three bins' population counts are determined and used jointly, efficient data access can be achieved by concatenating the four pieces of information into a single word,

[8] Because of the additional incomplete hyperpixel inserted (section 12.3.4), 31×4 histograms are built per hyperline for a total of $256 \times 124 = 31744$ addresses. However, as the incomplete hyperpixel's data is never used, no reset is required for the corresponding histograms.

[9] Amounts to 62 kbits with the incomplete hyperpixel.

[10] Where $E()$ denotes the function extracting the integer part of its argument.

[11] If histograms are treated sequentially and the subtractor and bit shift shared between them, $offset[k]$ induces an additional multiplexer.

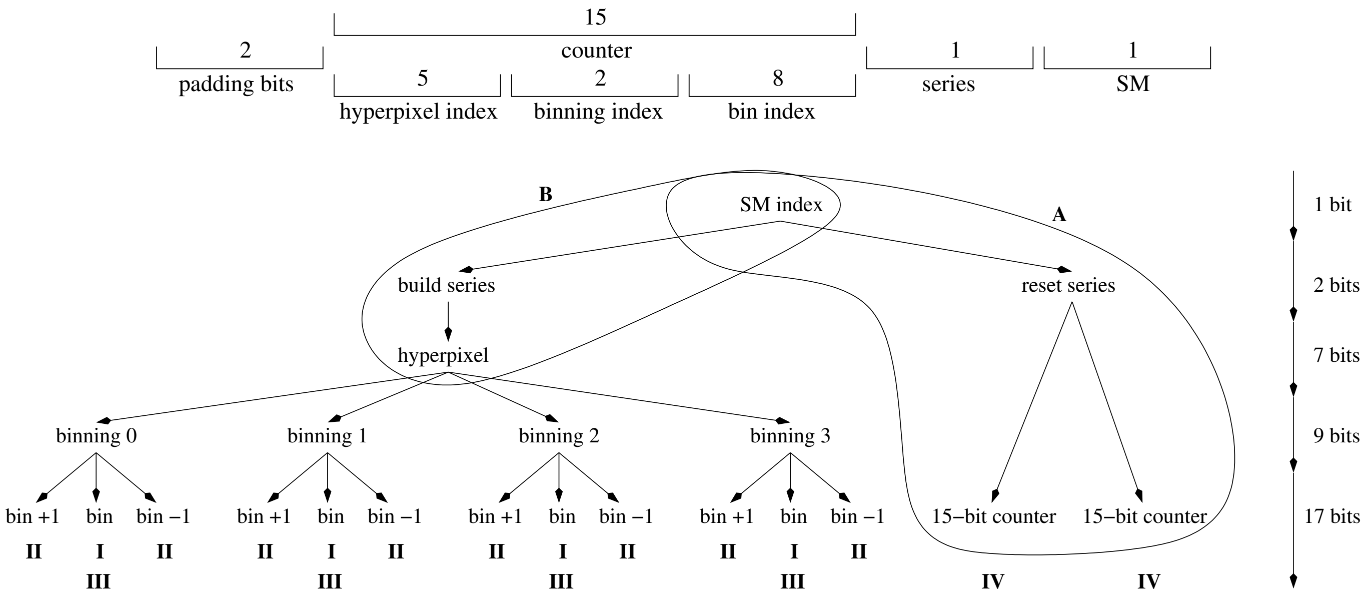


Figure 12.9: Structure of the histogram address words (top) and address construction tree (bottom). The roman literals correspond to the enumerated operations on histograms. "A" corresponds to the reset address calculation and "B" to the address precursor common to all operations for building the histograms and tracking maxima.

thus requiring only single read and write accesses to the RAM (Fig. 12.11) and reducing addressing to reusing some of the fields from the previous address calculation.

Although the volume of useful data ($30 \times 4 \times 2 = 240$ max words[12]) is rather limited, particular attention must be paid to the RAMs' aspect ratio in order for the corresponding storage space to map to the appropriate amount of resources. Naively forming max words with 8 bits for the position (the 4-ADU histograms have 256 elements) and three times 10 bits for the populations would result in 38-bit words difficult to fit into RAM modules which can be either 1, 2, 4, 9 or 18 bits wide on the ProAsic3E dies. Instead, considering that the sum of the bin populations is bounded by 1024, those of the bins adjacent to the one of maximum population are bounded by 512 so that two 9-bit fields suffice for a reduced total of 36 bits much more efficiently mapped to hardware (Tab. 12.5). Fig. 12.10 illustrates the resulting design[13].

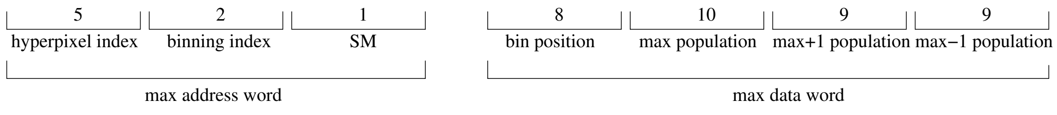


Figure 12.10: Structure of the max address and data word.

It may finally be mentioned that dummy addresses have been allocated as a convenience to ensure that memory content is not corrupted during the few first cycles following the initialisation, when no valid data has yet reached the point when writing to memory occurs. The last addresses are used for that purpose in the SRAM and among the 256 available ones in the internal `MAX_RAM`.

Pipeline

Fig. 12.12 illustrates the main tasks required for the processing. As for Pre-calibration and Buffer_pix, these have been distributed over 5 stages forming a 3-DCLK-deep pipeline. Unfortunately, this sequence seems more simple than

[12]The incomplete hyperpixel being treated as any other, it must be possible to address a total of $31 \times 4 \times 2 = 248$ max words.

[13]For completeness' sake, it is worth noting that this optimisation comes at the expense of support for two particular classes of realisations for the distribution of sample values. Indeed, the 10 bits allocated for the maximum population only allow for properly representing counts in the $[0; 1023]$ range while the 9 bits for the adjacent bins' limit the range to $[0; 511]$. The configurations leading to all the hyperpixel's samples falling within the same 4-ADU bin or between two in equal proportions would lead to an overflow of these counters and erroneous results. It would be possible to restore a conforming behaviour by introducing logic dedicated to the special interpretation of the value 0 in each of the two fields (on Mode's side) but the probability for such events to occur is so small that this resource would likely never be used.

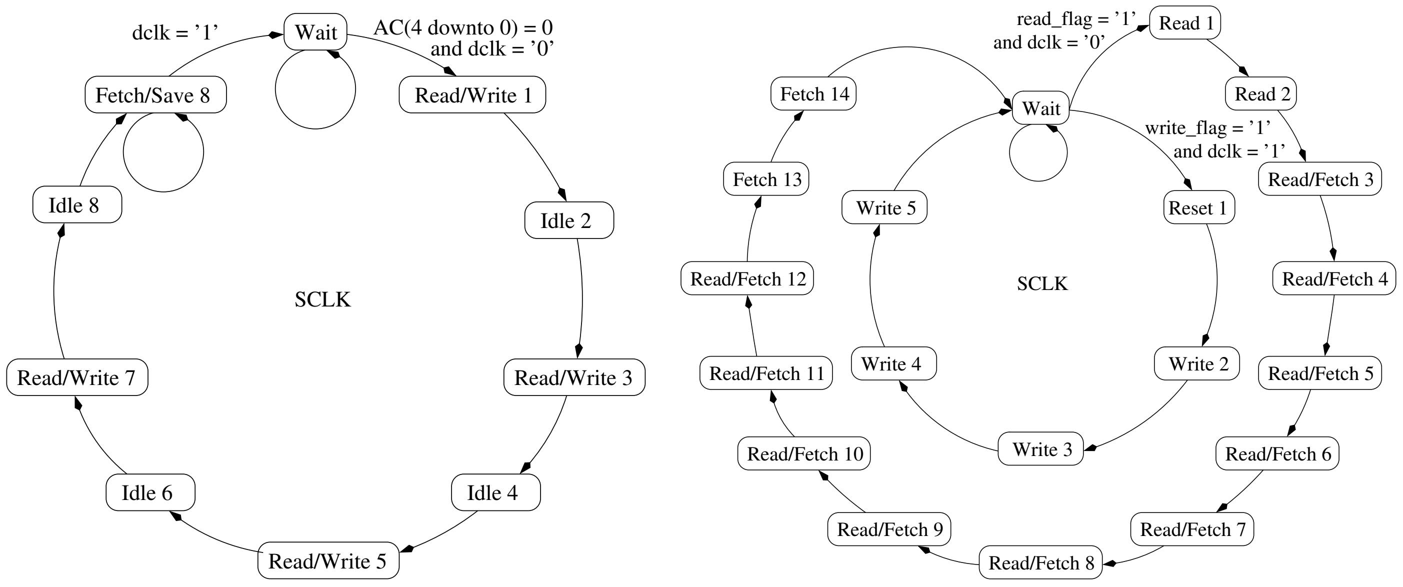


Figure 12.11: FSMs driving accesses to the internal MAX_RAM (left) and the SRAM through the SRAM controller (right). The MAX_RAM FSM has transitions triggered by falling edges of the SCLK to enforce setup and hold times (the synchronous RAM samples its inputs on rising edges). Conversely for the asynchronous SRAM edges are equivalent (the FSM is driven by rising edges).

it really is and this section discusses some intricacies successively related to data race problems, the handling of border effects and the reset strategy.

The four histograms are built in parallel through logic replicated through `for...loop` statements in VHDL. Although simplifying the organisation of the pipeline, this choice is very costly in resources, especially as far as the tracking of global maxima is concerned for it multiplies the number of operators (adders and comparators mainly). These being identical for all histograms, it might be possible to mutualize them. As discussed in section 12.2.4 and carried out further down for background interpolation in section 12.3.6 for example, sharing could be achieved, for example, by introducing an SCLK-driven pipeline handling the global maxima of each histogram in sequence, without modification of Histogram's scheduling, but at the expense of an increased complexity.

Data race

Data race difficulties occur with pipelined implementations when the processing for one element is not completely independent from that of others. While the sequential nature of operations for each element translates into the pipeline's sequence of stages and is thus statically ensured, interdependent processing may result in one stage relying on some other's result as input. Depending on the tasks to be performed, the corresponding delay may be constant or vary either in terms of which output is needed or for which stage this result is needed. Our pipelines being of limited depth and the scheduling entirely static, only the former class of problems need to be dealt with. As may well be imagined, the results which require several stages but which are used for every input are the more likely to pose such difficulties since latency then conflicts with throughput. The introduction of purely sequential units in the pipelines, as explained in section 12.2.4, makes it possible to reduce the latency by performing some calculations at a higher rate. One class of results, however, remains subject to such causality concerns, those which require, first a calculation, then storage in internal or external memory since for reasons of shared accesses to the memory, the two cannot be easily merged into a single sequence. Two such cases occur for Histogram: when incrementing and writing the histograms' bins to the SRAM and when updating and saving the max data to the internal RAM. Both are discussed in more detail here.

Based on the numbering of tasks introduced in Fig. 12.12, the sequences relative to three adjacent samples are represented in Fig. 12.13. This figure shows that the bin populations for the second sample are read from the SRAM (IV) before the values incremented as a consequence of the first have been written to it (VIII). Whenever the two samples have values within the same 4-ADU bin for one of the histograms, the read and write operations are commanded on the same address and incorrect data is read leading to incrementing the bin by only one unit instead of two. A potential third sample corresponding to the same bin, with a read-out performed before the second update but after the first has been written to the SRAM, would lead to the bin being incremented twice instead of thrice for

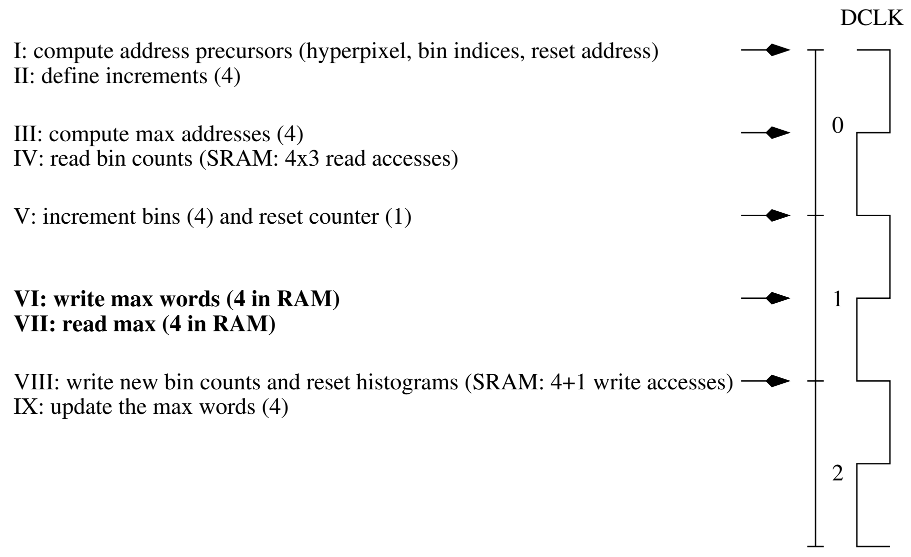

DCLK
I: compute address precursors (hyperpixel, bin indices, reset address)
II: define increments (4)
III: compute max addresses (4)
IV: read bin counts (SRAM: 4x3 read accesses)
V: increment bins (4) and reset counter (1)
VI: write max words (4 in RAM)
VII: read max (4 in RAM)
VIII: write new bin counts and reset histograms (SRAM: 4+1 write accesses)
IX: update the max words (4)
0
1
2

Figure 12.12: Sequence of the Histogram pipeline (operations typeset in bold letters are only carried out every 32 cycles).

the three samples. The conclusion is, therefore, that this data race failure would cause histogram bins' populations to be underestimated by exactly the number of pairs of consecutive samples with values differing by only three or less ADUs.

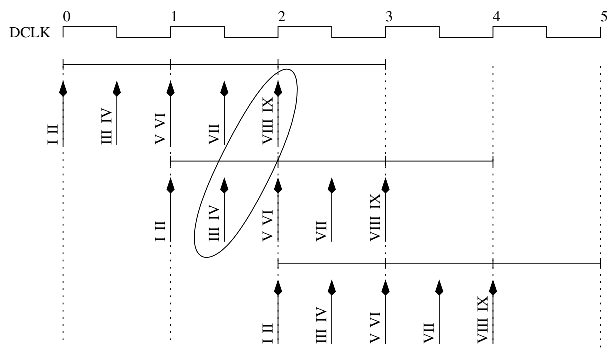


Figure 12.13: Processing for three consecutive samples in Histogram and data race problem: data is read from the histograms in external memory (IV) before their content has been updated as a result of the previous sample (VIII).

Had the histograms been build with ADU resolution, the dispersion of the input data due to the various sources of noise and the quantisation performed at the read-out level might have made this an infrequent occurrence and hence an tolerable one in terms of its impact on the resulting background estimate. In our case, however, assuming the frequent case of a low background level with Poisson statistics (λ of order 1.5 e^-), the total electronic noise with Gaussian statistics ($\sigma = 10.85\ e^-$) and a negligible dark noise ($\lambda = 0.034\ e^-$), with ADUs of order 4.07 e^-/ADU, the dispersion of the data is too small compared to the size of the bins. This effect would principally deplete the bin of maximum population relatively to the adjacent ones since, being the most likely value, it is the more subject to affect consecutive samples. With the formulae used to derive the background estimate, the adjusted parabolic profiles would then feature artificially flattened maxima leading to an additional contribution to the measurement's error in the light of the numerical analysis exposed in chapter 11.

Instead, the analysis above shows that the problem is limited to pairs of consecutive samples so that a solution, simple in principle, consists in memorising the bin's indices for one cycle to compare them to the new ones. Whenever two consecutive samples correspond to the same bin, the decision may then be taken to increment the bin's population not by one but by two units the second time. In practice, the cost relative to these special operations amounts to a series of 4 8-bit registers and comparators for identifying these cases, while for the incrementation itself, resources may be shared by introducing a register containing the value of the increment and replacing the incrementer by an adder. This latter aspect is most likely the more expensive, but the adder may be constrained to exploit the bounded nature of the increment. Besides, this additional flexibility may be exploited to handle sample values outside of the histograms' range, as explained below (border effects).

Accesses to the internal `MAX_RAM` present a similar challenge since they involve a three step sequence for fetching, updating and writing max data, with an additional constraint in this case: sharing the RAM with Mode calls for reading and writing to the RAM during the same high or low phase of DCLK (section 12.2.5). A possibility would then consist in exploiting the synchronous nature of the internal RAM for setting up a SCLK-synchronous FSM with proper delay states for performing all three within the same half DCLK period. However, due to the complex update logic for the max data (see below), this FSM would either count a large number of states or impose stringent timing constraints which are not desirable. Alternatively, noting that the max data, although potentially updated at each new sample, is relevant to the hyperpixel as a whole, a local set of registers may be used for part of its determination, then only relying on delocalised storage only when changing hyperpixels (at the beginning of the TDI line and every 32 samples) or for publishing the data to Mode. Read and write accesses to the RAM are then performed at different addresses since the former corresponds to fetching the latest max data determined during the previous TDI line for the upcoming hyperpixel and the latter to saving the max data for the one just completed. They may accordingly occur within the same DCLK phase, as is shown on Fig. 12.12.

Max data update

The main hidden intricacy in Histogram's operation lies in the decision to not only track the position of the global maximum for each histogram but also the population counts of the corresponding and adjacent bins. The multiplicity of cases requiring an update of the content of the local max data register is then a significant complication in the sense of the remark concerning special cases made previously. Tab. 12.9 recalls the conditions and logic for these changes.

Required update of max data	Boolean equation	Reference data
Maximum population change. or position change	`new_max_pop` > `max_pop` or (`new_max_pop` = `max_pop` and `new_max_pos` < `max_pos`)	
• this is not a new hyperpixel and the previous sample was in an adjacent bin	`new_hppix` = `false` and (`prev_bin` = `max_pos - 1` or `prev_bin` = `max_pos + 1`)	max: I prev: S next: I prev: I next: S
• normal update of max data		max: I prev: S next: S
Adjacent bin population change.	`bin_pos` = `max_pos - 1` or `bin_pos` = `max_pos + 1`	prev: I next: I
A new hyperpixel is begun.	`new_hppix` = `true`	max: S prev: S next: S

Table 12.9: Rationale for updating the max word, origin of the data for the maximum and adjacent bin populations ("I" and "S" stand respectively for "Internal" and "SRAM") and underlying logical conditions. The successive cases are listed in decreasing order of priority.

It can be seen that here also additional logic was inserted to counter the effect of the data race induced by the accesses to the SRAM when consecutive samples are in consecutive bins. This logic amounts to a set of four 10-bit registers, two multiplexers and three comparators for the control logic used to store the count for these bins internally instead of relying on the "not-yet-updated" SRAM value. This expense might be altogether avoided when considering that this exception only leads to one of the adjacent bins' count being underestimated by one unit in the worst case and that correctness would otherwise be restored the next time this part of the max data is updated. It is reported here and was maintained essentially to facilitate bit-wise validation of the FPGA data.

Border effects

Border effects are comparable to the special cases already discussed in the sense that any particular need induces additional and seldom used logic. In the case of Histogram, border effects are virtually everywhere, that is both in the input data due to the CCD's AC dimension and in the histograms due to their limited dynamics. More than extra logic, particular focus was placed on minimising the need for extra paths which would complicate the timing by imposing the most limiting constraint to all. Instead, the preferred approach, as for Pre-calibration in section 12.3.2, has consisted in inserting multiplexers upstream of the key steps to permit managing the border effects within the natural flow, even at the expense of a somewhat degenerate use of existing operators. The rationale for this is to trade further increase of latency against both resource and timing in the critical parts of the logical path.

1. AC dimension of the CCD.

   The CCDs' AC dimension not being a multiple of 32 samples, the entire photo-sensitive area cannot be completely covered by an integer number of hyperpixels. Accordingly, with the convention adopted for placing the hyperpixels on the CCDs, the region with AC coordinates 960 to 982 can be either regarded as forming an incomplete hyperpixel or of different size or as not being covered by a hyperpixel. The fact that this hyperpixel encompasses a reduced number of samples implies that were a background estimate derived from it, the corresponding statistics would differ from those of the others. In the frame of our efforts not to introduce selection biases to obtain a bias-free survey of stars this is not advisable. Instead the closest available background estimates, which correspond to low-pass filtering, may be propagated to these positions.

   It would then be possible, through the Scheduler and at the expense of additional control logic, to suspend Histogram's operation until relevant data is input with the start of the next TDI line from the other SM. In this case, one should proceed with caution because of the pipeline's latency. Brutally interrupting operations upon the start of the 961th DCLK cycle would leave the last hyperpixel's processing to finished only after the third sample from the next new line is received. Although not impossible to deal with, this delay would break the alignment of operations for the two SMs with the TDI periods and complicate the scheduling of the mode determination, the reset of histograms and the buffering logic. Conversely, waiting for this processing to finish imposes that Histogram function until the start of the 963th cycle and requires the allocation of memory resources for the two extra samples received from the incomplete hyperpixel: a set of four histograms in the SRAM and max data entries in the RAM. Once this resource allocated, nothing prevents Histogram from running continuously so one might save the control logic at the Scheduler level taking care to ignore the resulting data. Considering that, with the chosen addressing scheme, the extra allocated memory resource would be difficult to use otherwise and preferring to keep the scheduling simple and readable, this latter alternative which does not involve any extra logic was adopted.

2. Histogram dynamics.

   The bounded histogram dynamics generate border effects both when considering which bin to increment as a result of the input sample value and when fetching population counts from memory. Both are solved by relying on multiplexers for filling in the content of the registers respectively containing the increment to be applied (see above) and the histogram addresses. The intent is to then proceed within the normal path diverted into dummy operations.

   For values outside the histogram range the address calculation (12.2) will either overflow or underflow depending on the case but an accessible 8 bit value will result. Instead of tracking these events and introducing control logic to skip all further operations for this sample, it suffices to ensure that the corresponding data remains unmodified. Relying on the flexibility introduced by relying on an adder for the bin incrementation, this is easily achieved by adding 0 instead of 1 or 2. Precisely because the data should remain unchanged, the addresses read and written are indifferent so long as they are valid ones.

   For simplicity and sharing the resources between the four histograms, the same range is assumed in spite of their bins being offset by one ADU. This domain is naturally the intersection of all valid ranges $\bigcap_{i \in \{0 \dots 3\}} [936 + i; 1960 + i] = [939; 1960]$. Values at the extremities of the range only deserve special treatment insofar as read and write operations for the adjacent bins are concerned. A normal increment is used and the central bin is accesses thrice in lieu of its neighbours, which data-wise, respects the possibility for the global maximum to occur between the second and the penultimate bins. The five possible cases of figure are summarised in Tab. 12.10.

Reset strategy

The reset of the unused histogram series is made simple by the adopted addressing scheme, yet, letting the 15-bit counter run on indefinitely poses a synchronisation problem with the alternative SM1 and SM2 processing. Indeed, the first 983 addresses would be reset for SM1 during the first TDI, then addresses 984 to 1966 for SM2, then 1967 to 2949 for SM1 and so forth. Due to the even number of TDIs corresponding to a hyperpixel and to the fact that the maximum value the counter can hold ($2^{15} - 1 = 32767$) exceeds the number of required reset cycles (which amounts to $30 \times 1024 = 30720$), this would leave entire address intervals at variable positions without a reset.

Sample value	Case	Increment	Addresses		
(ADUs)			bin + 1	bin	bin -1
< 939	outside	0	-	-	-
939	border	1	939	939	939
$]939; 1960[$	nominal	1	bin +1	bin	bin - 1
1960	border	1	1960	1960	1960
> 1960	outside	0	-	-	-

Table 12.10: Management of samples values outside the histogram range (the encoding offset is worth 1000 so that the 0 e^- value corresponds to 1000 ADUs).

The use of two counters, respectively for SM1 and SM2 as shown on Fig. 12.9, is sufficient to ensure synchronisation. The corresponding increase in resources may be limited to a 15-bit register by sharing the incrementation mechanism between the two since they cannot be updated at the same time by construction. To further limit the amount of logic surrounding these registers, at the expense of power this time, reset is carried out for every input for a total of $32 \times 983 = 31456$ cycles[14], thus exceeding the need, and the extra cycles are left to operate on the incomplete hyperpixel.

12.3.5 Mode

Functional

With the histogram's global maxima tracked by Histogram and hence, formally speaking, the mode of the distribution already determined, the Mode component is left to carry out the parabolic profile adjustment – an operation altogether destined for refining this estimate to the sub-ADU resolution and further reduce its sensitivity to the remaining statistical noise in the histogram. Additionally, a replacement strategy is defined to counter the pollution induced by the presence of objects, either faint and in important numbers or bright and extending over large regions.

1. Determine the sub-ADU mode:
   (a) fetch then reset max data words (4),
   (b) calculate mode with fixed-point arithmetics based on the four histograms,
   (c) publish estimates to PixSel.
2. Filter unreliable histograms:
   (a) test max population to identify unreliable mode values,
   (b) perform replacement with valid values.

As opposed to Histogram above, this component features complex arithmetics but reasonably simple data management thanks to comfortable timing constraints. Indeed, the adopted scheduling (section 12.2.4) leads to updating the modes for one SM while samples from the other are being selected, hence a full TDI, corresponding to 983 DCLK periods, is available for the mode computations to complete. Additionally, all max data have already been determined at this stage so that access to the `MAX_RAM` data is only constrained by potential conflicts with Histogram (section 12.2.5). As compared to other components inserted in the overall pipeline and required to handle the corresponding data flow, the relaxed constraints at the interface allow Mode to function somewhat asynchronously or, at least, separately from the rest.

RAM organisation

This section focuses on the structure for addressing the `MODE_RAM` inserted between Mode and PixSel as a communication medium. Although Fig. 12.14 illustrates the FSM used to read max data words and reset them, the description for `MAX_RAM` can be found in section 12.3.4 and the resolution of potential access conflicts in section 12.2.5.

With 30 hyperpixels in the AC direction and the need for two series of estimates to perform the two-dimensional piecewise linear interpolation which yields a background estimate at every sample position, 60 mode estimates per SM must be made available to PixSel. With an additional one corresponding to the incomplete hyperpixel (see below) and another to a dummy address to write to until relevant data reaches the write stage, a total of 132 21-bit words

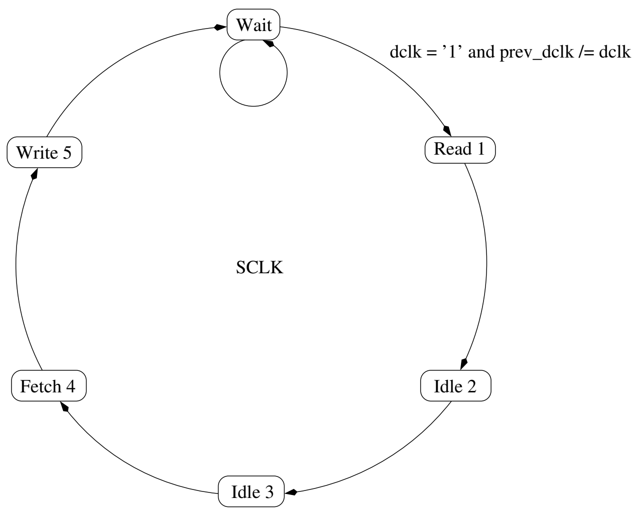


Figure 12.14: FSM driving Mode's read (for fetching max values) and write (for resetting them) accesses to the MAX_RAM. Transitions are synchronous with the falling edges of SCLK to ensure setup and hold times are respected as MAX_RAM samples its input on rising edges.

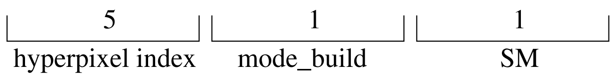


Figure 12.15: Structure of the mode address word.

(2772 bits) must be made accessible. As before, addresses are formed with minimum logic by concatenating relevant signals into a 7-bit word, as illustrated Fig. 12.15.

The `MODE_BUILD` signal, provided by the Scheduler through Mode's ports, changes for every hyperline and allows for swapping the buffers containing the mode estimates corresponding to the last two hyperlines without movement of data. The series of modes corresponding to the oldest data is by this means overwritten and will be identified by PixSel as the leading set for interpolation at the beginning of the next TDI. This delay also allows for updating the modes in place without the need for additional storage as in the case of histograms (section 12.3.4).

Pipeline

With the relaxed timing constraints, the main preoccupation driving the implantation of Mode's logic has been to save resources. This is of special importance here since the mode calculation relies on a euclidean division step for which a dedicated component must be instantiated at a significant expense in terms of logic. Processing has, accordingly, been fully pipelined so that the data follows a single logic path through Mode, in particular with only one divider. The four histograms are handled sequentially, one max data word being fetched from the `MAX_RAM` at every cycle to feed the pipeline, before combining the results every four cycles. The pipeline, represented Fig. 12.16, is then globally DCLK-synchronous with operations not fully systematic controlled through a FSM.

Replacement

On top of the mode's calculation itself, a replacement mechanism for unreliable hyperpixels is implemented in a manner similar to the one used for dead samples in the frame of pre-calibration (section 12.3.2). It follows the same principles of relying on neighbouring valid values and always overwriting irrelevant ones, which yield four cases of figure to be handled by the replacement logic: averaging neighbouring values, propagating either single valid ones (previous or next) or using a default constant. To this end, a shift register is also inserted for temporary storage until all three consecutive values have been determined.

The main difference with pre-calibration rests in the fact that while the identity of dead samples was uploaded from ground along with the strategy to be used, no such thing is possible for hyperpixels since their failure to meet the reliability criteria results from the content of the sky at that particular instant. Hence, the logic interpreting the

[14] The counter is accordingly incremented modulo 31456.

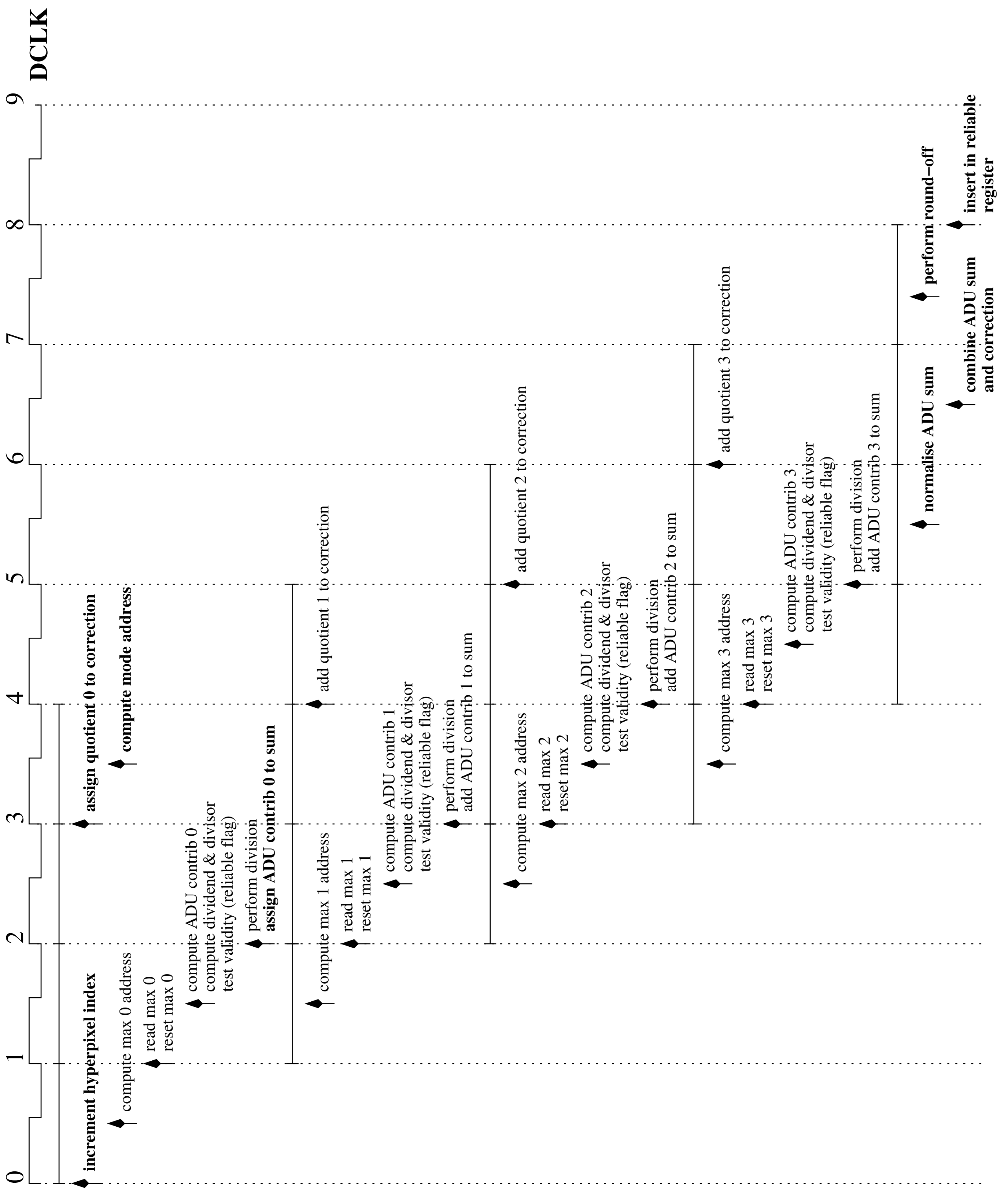


Figure 12.16: Exploded Mode pipeline (operations not carried out for every cycle are typeset with bold letters). The operations shown correspond to the determination of a single background estimate based on the max data words corresponding to the four histograms. The processing for the next estimate starts immediately after so that the each block should be replicated as soon as it ends (ie. the next "increment hyperpixel index" occurs for the rising of edge of DCLK numbered 4).

replacement cases is here replaced by tests on the bins' populations. With the decision to replace the values as soon as one of the four histograms fails the criteria, all four max word are tested sequentially – necessarily when the dividend and divisor are determined since max words are read-out every DCLK cycle. The unreliable hyperpixels are marked using a binary flag which is stored, ideally in a register shifted every four DCLK periods, until the replacement can occur. It is identical to that of pre-calibration – save for the different word lengths – in the sense that it proceeds in two stages: the first determines the content of two registers based on the reliability flags before their contents are summed and divided by two in the second, so that all cases are again handled using the same logic. Fig. 12.17 illustrates the resulting timing.

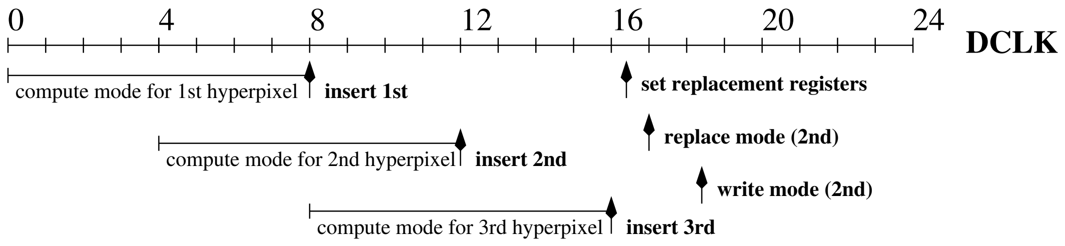

0
4
8
12
16
20
24
DCLK
compute mode for 1st hyperpixel
insert 1st
set replacement registers
replace mode (2nd)
compute mode for 2nd hyperpixel
insert 2nd
write mode (2nd)
compute mode for 3rd hyperpixel
insert 3rd

Figure 12.17: Replacement mechanism for unreliable mode values using adjacent hyperpixels. The indicated timescale is coherent with that of Fig. 12.16 and bold-faced operations are only carried out every four DCLK period.

Latency

Combining the fully pipelined calculation and the replacement procedure results in significant latency. Fig. 12.16 shows that one mode result is determined every four DCLK periods but only 8 cycles after the hyperpixel counter has been incremented. For replacement purposes an additional 4 cycles are necessary until the next mode becomes available and can be used, so that also accounting for the replacement and the access to the `MODE_RAM`, the latency totals to 14.5 DCLK cycles (can be counted on Fig. 12.17).

Accordingly, $29 \times 4 + 14.5 = 130.5$ cycles are needed to produce all 30 estimates before the end of the TDI. Noting that while the last of these estimates is being determined, data must necessarily be input to the pipeline, additional max data words are necessary. The incomplete hyperpixel provides four. Additional ones would be required for the remaining 2.5 cycles if the address did not return to 0 as a result of the overflow on the 8-bits. These, having been reset, feature a default value which, regarding the replacement mechanism, is automatically identified as unreliable. A specific condition to avoid propagating irrelevant values is thus only required for the incomplete hyperpixel.

Suspending operations immediately after the 130.5th cycle via the `ENABLE` port would stop the component in the middle of a computation. While this is not a problem functionally speaking because the data remaining in the pipeline is irrelevant, it would require a reset before the next series of modes can be determined. It is then both more simple and more efficient resource-wise to let the on-going computation finish and suspend activities before the next one, that is after 131 cycles as indicated in Tab. 12.6.

Validation

Finally, bit-wise validation of Mode's results versus Pyxis is not immediate due to the presence of the 1-bit signal used to round the equidistant cases alternatively to each of the nearest two integers. With k this binary random variable, if $sum \equiv 2^{N-13}[2^{N-12}]$, then as per chapter 11 (11.40):

$$mode = \frac{sum + 2^{N-13}}{2^{N-12}} - k \tag{12.3}$$

In practice, k is rather implemented as a flip-flop which changes state each time it is used to map values alternatively to the greater or smaller neighbouring integers. The resulting distribution fulfils the bias correction need while saving the complexity of generating a pseudo-random variable. A side effect of this simplification is that the round-off's behaviour is then history-dependent. As the processing history differs between Pyxis and the FPGA because only one SM is handled by the former and because additional operations for incomplete hyperpixels are carried out by the latter, additional logic had to be inserted for validation purposes. This logic is by no means required in the final version so its addition is configuration-dependent and the corresponding resources may be re-assigned with the help of VHDL compiler during synthesis.

Divider

Methods

S.F. Oberman and M.J. Flynn have published in [62] a review of existing approaches to hardware division which, although written with the need for division operators in FPU principally in mind, remain perfectly applicable to our fixed-point case. The five main categories they propose based on the underlying functional blocks may, from a mathematical point of view, be reduced to three classes as in [63]:

- Digit-recurrence.

  R. Michard et al. [64] refer to this method as the "paper-and-pencil" method taught in schools since the quotient is produced iteratively, digit per digit, through a succession of multiply, subtract, shift operations which also yield the remainder. The objective of euclidean division is to determine the quotient q so that $x = qd + rem$ with x the dividend, d the divisor and rem the remainder such that $0 \leq rem < d$. Writing the quotient at the jth iteration as $q[j] = \sum_{i=1}^{j} q_i r^{-i}$, in base r, and $w[j] = r^j(x - dq[j])$ the partial remainder, the following recurrence is used:

  $$\begin{cases} w[0] &= x \\ w[j+1] &= rw[j] - q_{j+1}d \end{cases} \tag{12.4}$$

  Since, the method for determining q_{j+1} is not explicit in this formulation several possibilities exist, with $r = 2$ they are:

  – restoring algorithm: $q_{j+1} = \begin{cases} 1 \text{ if } 2w[j] - d \geq 0 \\ 0 \text{ otherwise} \end{cases}$

  – non-restoring algorithm: $q_{j+1} = \begin{cases} +1 \text{ if } w[j] \geq 0 \\ -1 \text{ if } w[j] < 0 \end{cases}$

  Extension to larger radices are faced with increased complexity for comparing the partial remainder to multiples of the divisor so that SRT algorithms introduce a redundant quotient digit representation to simplify the selection of q_{j+1} and an entire class of algorithms are dedicated to "very high radix" cases in which more than 10 bits are retrieved for the quotient at each iteration [62].

- Functional iteration.

  This method chooses to regard the problem as an implicit equation, most often with the functional $f(y) = 1/y - d$ to determine the reciprocal of the divisor $(1/d)$ with the benefit of making a whole class of quotient values easily accessible through multiplication by different x. Foremost among the variety of numerical tools then available, the Newton-Raphson procedure provides quadratic convergence in determining y according to:

  $$y_{j+1} = y_j - \frac{f(y_j)}{f'(y_j)} = y_j(2 - dy_j), \tag{12.5}$$

  but other methods exist based on series expansion for instance, like Golschmidt's algorithm.

- Table look-up.

  In this case, some large amount of precomputed results are stored in RAM to limit calculations to addressing and, possibly, to a refining algorithm – for example an interpolation procedure on read-out values. The size of the memory array required for this increasing rapidly with the need for precision, these methods are often used as convergence accelerators replacing the first iterations for one of the previous schemes. Two main categories exist:

  – direct approximations which tabulate reciprocal values for divisors normalised to the $[1;2[$ interval,

  – linear approximations which store the two coefficients of the first order expansion of, for instance, $f(y) = 1/y$ in the vicinity of various values of d.

Calculation

Equation (12.6) recalls the four divisions (11.37) from chapter 11 to be carried out in our case for $i \in \{0, \ldots, 3\}$ (F_j^i denotes the population of bin j in the ith histogram).

$$\frac{\overbrace{(F^i_{max\_pos^i+1} - F^i_{max\_pos^i-1}) \times 2^{16}}^{1.9.16}}{2 \times \underbrace{(2F^i_{max\_pos^i} - F^i_{max\_pos^i-1} - F^i_{max\_pos^i+1})}_{0.12.0}} \tag{12.6}$$

As can be seen through the fixed-point formats indicated, the requirements in terms of precision are modest. Besides, the result is not needed before the end of the DCLK period as per Fig. 12.16. Even the resource constraint has been significantly reduced by sharing one divider between all four computations through the complete pipelining. The key criterion for selecting one method rather than another is then the availability of the different technologies.

Surprising as it may seem, especially in comparison to the profusion of implementations for adders or multipliers, limited numbers of dividers exist within ready-to-use libraries. In particular, Actel does not provide such functional blocks through the macro libraries for ProAsic3E or for RTAX-S dies, let alone automatic generation through the SmartGen interface in the integrated development environment (Libero).

It turns out that methodology is secondary to strict correspondence with functional needs, notably as concerns port sizes and data types – the benefits of in-depth algorithmic optimisations being completely swamped by the instantiation of many unused logical resources if ports larger than necessary are implemented. The Arénaire project's "divgen" tool [64] stands out among the VHDL models found by the possibility to finely tune the operand sizes and encoding since it is in truth a generator of VHDL models which accepts a configuration file as input[15]. Although it offers the possibility to select between either restoring, non-restoring and SRT implementations of the digit-recurrence method, only the restoring one was found to be applicable[16]. The corresponding schematic and configuration are shown Fig. 12.11.

Parameter	Value
`x_representation`	`2s_complement`
`d_representation`	`unsigned`
`q_representation`	`2s_complement`
`x_size`	`29`
`d_size`	`12`
`q_size`	`17`
`algorithm`	`restoring`

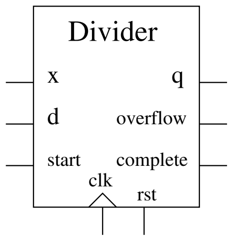


Table 12.11: Configuration and schematic for the divider generated with divgen.

Finally, the resulting component has been encapsulated in a larger one for three reasons:

- the need to perform a reset during one period between calculations and to trigger each by asserting the `start` signal for exactly one period,
- the relationship between the number of integer bits for the dividend, the divisor and the quotient is given by $i_x = i_d - i_q$, hence to obtain the proper format for the quotient, the dividend is expanded by three most significant bits,
- verification assertions, concerning overflow (theoretically impossible in our case) and completing the calculation in time for the next DCLK cycle.

The component is reported in [64] to have been operated successfully at 80 MHz on a Xilinx xcv300e VirtexE FPGA so that CLK-synchronous operation at 125 MHz would imposed timing constraints difficult to meet. Considering, on the opposite, that there is ample time for the calculation to finish, the encompassing component is clocked using SCLK. This allocates 31 cycles for the result to stabilise and the complete signal to be raised and is largely sufficient since digit-recurrence produces one correct bit per cycle and the complete quotient only contains 20 bits.

12.3.6 Pixsel

Functional

With the Scheduler in charge of synchronising all components and managing the AL and AC location counters and Buffer_pix fetching the memorised sample values from the external SRAM just in time, the prerequisites are met for the "rendez-vous" point at which the decision is made to retain each sample or not. As compared to Pyxis, in which the interpolation of the mode estimates at the sample's position and the testing for relevance are two separate functional modules, both operations are here performed jointly in an effort to avoid storage and reduce the need for communications. Hence the following specifications result for Pixsel:

[15]It is worth noting that a divider is also available in Gaisler's GRLIB IP library under the GNU GPL license (DIV32) but its operands are necessarily 32-bit wide.

[16]The non-restoring version, although expected to result in leaner logic, had unexpected behaviour on the overflow signal. The SRT one, on the other hand, requires a normalised divisor: $2^{11} \leq d < 2^{12}$ and this condition not being naturally fulfilled would call for additional logic upstream and downstream of the division proper.

1. Perform piecewise bi-linear two-dimensional interpolation to estimate the background at every sample location.

2. Test sample value vs. a SNR criterion taking the background into account and based on pre-determined noise statistics.

3. Collect output data for the retained samples: calibrated value, SM, AL and AC coordinates and associated background estimate.

Pipeline

The processing for interpolating from the background seeds (ie. the mode values stored in the `MODE_RAM`) and calculating the various terms involved in the SNR test is very favourable for a pipelined implementation in the sense that not only is each sample's completely independent from the preceding and following ones, but also that the data is used to produce results which are then used without any further reference to the initial data. These two favourable conditions imply that no data race nor synchronisation problems are to be feared, so the processing may be generously spread out in time for the benefit of a simpler, if deeper, pipeline (Fig. 12.4).

The pipeline set up here is structured after the main logical stages but it is worth noting that the simplification achieved with the relaxed timing constraints is counterbalanced by additional sets of registers for collecting the output data (Tab. 12.12). Two optimum configurations can therefore be imagined, the first with a timing finely adjusted to the response delays of the target die and hence, a pipeline of minimum depth with savings in registers, the second with a timing further relaxed to share the logical resources between the different parts of of the calculation.

As above, for Mode, it is relevant resource-wise to fully pipeline the determination of the contributions from each seed to share the underlying multiplier and adder. Compared to (5.6), formula 12.7 is more favourable resource-wise (and scheduling-wise) because it is highly homogeneous and allows the computation to run for each sample systematically. The property that made (5.6) interesting in software (moving some of the processing from the sample level to the line level) would here require introducing dedicated logic rendering the logic and the scheduling more complex.

$$samp_{i,j}^{bk} = mode_{0,0} \times (1-\frac{i}{W})(1-\frac{j}{H}) + mode_{W,0} \times \frac{i}{W}(1-\frac{j}{H}) + mode_{0,H} \times \frac{j}{H}(1-\frac{i}{W}) + mode_{W,H} \times \frac{ij}{WH} \quad (12.7)$$

Although simplicity calls for determining the four contributions in parallel, this leads to replication of the underlying logic. Considering that *mode* words are 22 bits long and that, with $(i, j) \in [0; 31]^2$, coefficients can take 1025 different values, the required 4 22×11 multipliers and 3 32-bit adders total to 3758 cells. This logic can be somewhat reduced by managing the special case $i = j = 0$ specially, that is by copying $mode_{0,0}/4$ in all four contributions directly[17], since then the 4 22×10 multipliers and 3 32-bit adders correspond to a smaller 3468 cells. Not only is it more efficient resource-wise to pipeline the execution of this calculation, but it is even mandatory in this case considering the amount of resources used. Although carried out sequentially, the need remains to produce one background estimate at every cycle because PixSel must determine the status of one sample at every DCLK cycle. Clocking this small pipeline at a faster rate than that of DCLK is hence necessary. SCLK is naturally used for this, with the potential need to insert intermediate wait states to solve the timing difficulties related to the complex arithmetics at stake. It is worth noting that no additional registers are required for this since the contributions can be incrementally aggregated to the total by the simple FSM introduced.

Register	Width (bits)	Depth	Total
Calibrated values	16	5	80
SM	1	6	6
AL	5	6	30
AC	5	6	30
Background estimate	32	2	64
			210

Table 12.12: Registers required for collecting the output data in PixSel.

[17]This condition requires an additional 32 multiplexers with a 32-bit shift as compared to the $\simeq$ 300 additional cells for the eleventh bit in the multipliers.

Local coordinates

The fixed point format adopted in chapter 11 corresponds to multiplying (12.7) by WH, thus discarding the need for divisions and shifting all results by 10 bits since $W = H = 2^5$. The i and j coordinates being local ones relevant to the interpolation cell have reduced 5-bit dynamics compared to AL and AC ones but are offset by half a hyperpixel in both directions. The AL coordinate is, naturally, managed at the Scheduler's level and available through PixSel's ports. For AC, the straightforward coordinate transform would then read:

$$j = (AC - 16)\ (mod\ 32) \qquad (12.8)$$

However, general purpose subtractors and modulo operators being rather expensive from an implementation point of view, this is not implemented as such, but rather as:

```
j <= not AC(4) & AC(3 downto 0)
```

resulting in very limited RTL logic (Fig. 12.18).

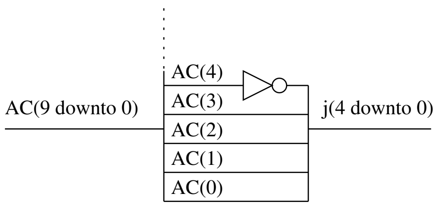


Figure 12.18: RTL logic for the determination of local interpolation AC coordinate.

Mode RAM accesses

Although the interpolation seeds could be systematically read from RAM, they only need to be updated when a new interpolation cell is entered, that is every 32 DCLK cycles. As shown on Fig. 12.4, this option has been retained with the intent to save on the number of RAM accesses which are reported to be significant contributors to the overall power consumption. Address calculations, on the other hand, run continuously although they are only seldom used. While this would be simply unthinkable in software, as leading to entire trains of instructions wasting processing time and electrical power for the production of results merely ignored, it corresponds to a trade-off between control logic and power in hardware. Considering, indeed, that instead of instructions, the processing sequence is physically represented by an interconnection of logical gates, constant inputs translate into a stationary state. Hence, from a power point of view, only the clock-driven sampling of combinational logic blocks' outputs by registers need to be accounted for. Conversely, considering logic, introduction of a specific control unit to perform the address calculation only when required – as one would in software – would call for additional resources, themselves contributing to the consumption. An intermediate and probably optimal solution would consist in using a dedicated clock for that purpose, determined at a higher level and mutualized between multiple components. In this particular case, the need for a 32-DCLK period is also found in histogram, precisely in phase opposition, so that introducing such a clock might be relevant.

Given the way the interpolation cells are offset versus the hyperpixel grid (chapter 5), new cells are begun every 32 AC samples, except for the first and last which are respectively 16 and 23 samples long. In the general case, testing the local AC coordinate for the 0 value using a 5-bit comparator is sufficient to trigger the FSM for the accesses to the `MODE_RAM`. The irregular first and last cells' sizes, on the opposite, call for a dedicated condition to read new mode values at the beginning of every TDI line. A typical example of an infrequent case generating the major part of the resources could be found here if we chose to test the input AC coordinate for this since a additional 10-bit comparator would be required. Instead, this may be detected through transitions on the SM bit at an additional cost amounting only to a 1-bit register and a 1-bit comparator.

The decision to perform only AL interpolation for the 16 first and 23 last rows implies that only two seeds are required for these interpolation cells (instead of four) and might complicate the handling of `MODE_RAM` accesses. Fortunately, using identical values in the AC direction, the two-dimensional interpolation engine degenerates into a one-dimensional AL one. Save then for some control logic filling the mode address registers differently in these cases, all samples may be handled through the same logic. Fig. 12.19 represents the FSM fetching the four mode values from the `MODE_RAM`.

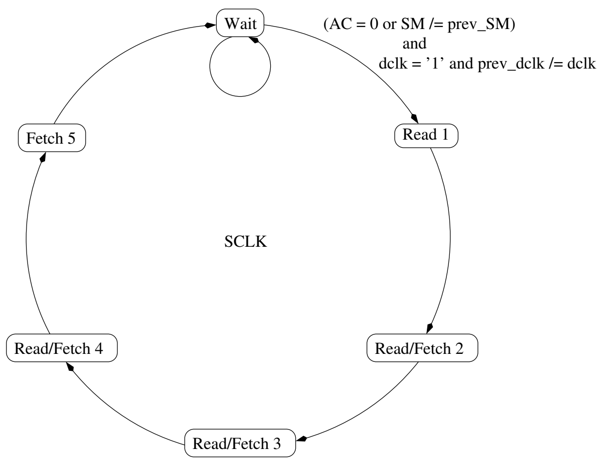


Figure 12.19: FSM driving read accesses to the internal MODE_RAM for mode data. To guarantee setup and hold times all transitions synchronous with the rising edge of the SCLK (the MODE_RAM samples its input on the falling edge for read accesses). Note that write accesses by Mode, consisting in only one request every four DCLK cycle, are simply commanded DCLK-synchronously.

12.4 Conclusion

The reformulation of the algorithms achieved in this chapter, together with their simulation, validates both the hardware-friendly nature of the algorithms, particularly as regards managing samples in their natural ordering, and the fixed-point arithmetics of chapter 11. With the decision to rely massively on pipelines a number of intricacies have been successfully solved, sometimes with counter-intuitive and apparently useless cycles or resource. The overall result, perfectly in line with the constraints which apply, favours "slow operation" through relying only on the SCLK and DCLK. This yields reduced power consumption and sensitivity to SEUs at the expense of intermediate registers and, hence, area. While this latter aspect is problematic for the ProASIC3E600 die with which the test platform is equipped, it is by no means insoluble since this part is among the smallest on the market and definitely smaller than RTAX-S ones.

Chapter 13

Synthesis & Co.

Contents

13.1 Introduction

The analyses for the different parts of the design presented in the previous chapters and the corresponding VHDL descriptions are the basis on which the subsequent steps of the design flow (section 8.3) may proceed. Two concurrent processes are involved in these, on one hand, the transformation of the description in physical terms (synthesis, place & route) and, on the other hand, their validation (functional simulation, post-synthesis simulation, post place-and-route simulation, tests on the platform).

As of this writing, not all of these steps have yet been carried out. The main reason, as already mentioned in section 9.2.1, is that the selected device (ProASIC3E 600) was found to be too exiguous for the complete design. Additionally, the “Gold” license acquired from Actel for the Libero development environment being restricted to areas smaller or equivalent to that of the ProASIC3E 600, it has not proved possible to simulate the design on larger dies. With these restraints, our efforts have concentrated on the VHDL entry and on synthesis together with the two corresponding levels of simulation (functional and post-synthesis). We hence describe in this chapter the simulations carried out for verification purposes and the results of synthesis.

Only during the last month has a full-fledged license for Libero become available to us and a Starter Kit equipped with a ProASIC3E 1500 die is expected during the forthcoming month. As to placement, routing, timing or power considerations, little more is said, in spite of their decisive importance, than that Actel’s tools converge to a solution declared valid. These results combined, although incomplete, provide good confidence that the proposed design can be fully implemented on the test platform with the ProASIC3E 1500 die and suggest that it may even be possible to fit it into the smaller 600 one provided some more efforts are invested in optimisations.

13.2 Verification

Following the ECSS terminology introduced in section 3.3, the questions of validation and verification of the hardware may be addressed separately thanks to the software prototype developed, provided the two are made equivalent at the bit level. This has been achieved with the use of fixed-point arithmetics in software also and the introduction of a single specific condition in the VHDL description of the Mode component (validation item of section 12.3.5). Balancing the software’s simpler design with the error-prone nature of hardware descriptions, validation is then performed preferably with software. Although potentially not faultless, the software then also represents a reference to which the behaviour of the hardware can be compared for its verification. With two independent software implementations (in ANSI C and in Matlab [33]), cases where the three agree can be considered valid while differences have been examined on a case by case basis.

The l54b0 test case (appendix C) has been selected for this testing because it provides a large variety of configurations due to the high object density. Extensive simulations at the functional level have been conducted up to a maximum observing time of 4.5s corresponding to $5.625.10^8$ CLK cycles. Signals at critical points have been sampled in this process and compared to their software equivalents. In the latest version, all tested bits were found to be valid by this means.

Post-synthesis inspection was also carried out to track semantic mismatches as already mentioned in section 8.3. Due to considerably slower simulation both because of the significantly more complex description of the hardware after synthesis and of performance limitations in ModelSim (related to the license owned), it has not proved possible to verify the post-synthesis design with the same amount of detail as the functional one. A sufficient number of cycles (of the order of $1.6.10^7$) have, however, been simulated to provide acceptable evidence of correct operation. Further examination will become possible with the newly acquired license for Libero and complemented by in-place debugging with “Identify” components (section 8.3) at the real system rate.

13.3 Synthesis

13.3.1 Introduction

Based on the logic instantiated for elementary components, it is possible to produce a first-order estimate of the result of synthesis area-wise. Relying on such measurements as those of Fig. 8.1, it is possible to account for arithmetic operators while inspection of the VHDL code, notably by building a census of signals and associated word lengths, allows for counting registers and evaluating the volume of control logic. The corresponding approximation has served as a guideline for the design and the implementation of the algorithms in this and the previous parts. Hence, comparing it to the netlist output by synthesis provides a quality measure for the RTL description and its interpretation by the tools. For proper interpretation of the differences between the two, the following two key factors need to be taken into account:

- Enable strategy and feedback multiplexers.
  The tasks not directly related to the control of the FPGA’s interfaces need to include logic allowing to suspend operation for an entire DCLK cycle when input or output data transfers fail (section 9.3.4). The mechanism adopted throughout chapters 10 and 12 relies on an `ENABLE` signal which must be asserted for any modification in the components’ internal state to occur. As mentioned in section 12.3.1, although conceptually simple, this solution leads to massively introducing feedback multiplexers to allow flip-flops to remain in the same state for as long as `ENABLE` is not active. This is costly both in terms of timing, since it corresponds to additional logic on the data path and in terms of resources. The corresponding cells are easily traceable in the VHDL and partially explain discrepancies between predicted and allocated resources.

- Optimisation
  Detailed static analysis is carried out as part of synthesis for optimisation purposes. This process extends over the entire design which is first entirely flattened for the purpose of allowing cross-component examination and proceeding globally. As a consequence, logic may be removed, shared or transformed, to improve on resource usage and / or timing[1] but at the expense of making low level debugging of a post-synthesis design difficult. The use of RTL coding is favourable for minimising discrepancies in this respect since the description, besides being somewhat optimised due to addressing issues at a low level, remains reasonably close to the content of the netlist. It is sufficient, however, for instance because the Symplify compiler contains a FSM compiler which optimises and re-encodes the various states based on criteria on speed and resources. The rules in Tab. 13.1 are used in the absence of user-specified attribute and lead to noticeable increases in allocated resource when the one-hot case is preferred.

The timing analysis performed during synthesis is necessarily very preliminary since the netlist cannot yet have been placed and routed[2]. Very detailed timing is not useful at this stage but will rather impact on the complexity of routing or on power consumption. However, one needs to gather sufficient evidence that a working configuration exists. Typically, reasonably negative slacks on the worst paths for each clock domain during synthesis are rather an indication that the timing constraints are well taken into account and are even considered to provide clear optimisation directives for place and route.

[1]Synplify Pro’s “retiming” option for instance trades resources against time. Although it leads the former to increase by $\simeq 5\%$, the critical times on the worst paths are relaxed by amounts up to $\simeq 80\%$ by this option. With the margin provided by using a ProASIC3E 1500 die, the results presented in this chapter correspond to this option.

[2]The number of interconnections is not known at this point (section 8.3).

Number of states	Encoding	Sequence (0 1 2 3 4 ...)
< 5	sequential	00 01 10 11
≥ 5 and ≤ 16	one-hot	0000 0001 0010 0100 1000 ...
> 16	gray	0000 0001 0011 0010 0110 ...

Table 13.1: Default FSM encoding styles used by Synplify.

As for area above, some basic general guidelines can be followed to improve the chances of designing a working system. Slow clocks are definitely a key measure taken in this sense in the previous chapters. With communications between clock domains addressed specifically (section 9.5.3), those between different synchronous processes call for special attention if modifiers are introduced on the path, like an inverter to reconcile active high and active low ports for instance. With RTL coding, it is also possible to estimate the number of logical levels separating synchronous signals to avoid imposing critical timing constraints through deep paths. At the expense of additional intermediate registers, it is for example possible to split the calculation over several clock cycles by transforming it into a simple pipeline.

13.3.2 Constraints

With the particular attention devoted to designing a "slow" pipeline, only the mandatory constraints need to be imposed for synthesis. These include the declaration of the clock signals and of the associated periods but also specifications for signals identified by synthesis as "inferred clocks" because they control flip-flops: quartz oscillator on the Starter Kit (25 ns), CLK (8 ns), SCLK (32 ns), DCLK (992 ns), `ENABLE_MODE` (992 ns), `ENABLE_HIST` (992 ns), `ENABLE_CALIB` (992 ns). The IO standards for the ports also need to provided at this stage. Based on the list in appendix A all are configured as LVCMOS 2.5 V or 3.3 V depending on their voltage. Finally, to obtain maximum confidence on the correctness of the synthesised design, the delays for the asynchronous parts of the design can be specified. Two such "max delay paths" exist in our design due to the introduction of asynchronous fragments in Pre-calibration to save on registers (Fig. 12.7). They correspond respectively to the combinational calculation of the linear transform calibration (2 paths: 992 ns) and to the replacement mechanism (3 paths: 496 ns). Given the available delay and the number of logical levels involved, this precaution is, however, somewhat unnecessary.

13.3.3 Area

The performance of the design in terms of area is relevant because it impacts on selecting a device[3], on power consumption and on its sensitivity to radiation. The first item is clearly the most important for a demonstrator like ours because of the need for margins and our feasibility assessment objective[4]. As mentioned in section 8.4.1, the ProASIC3E die with which our test platform is equipped has a radically different internal structure compared to RTAX-S devices. Beside differences in the routing technology and, as a result, in timing, the ProASICS3E chips are essentially of the "sea-of-gate" type proposing a large amount of unspecialised cells while the RTAX-S also has more specific ones (section 8.3). Some area improvements can then be expected when synthesising the design for this target.

Tab. 13.2 compares the expected amount of logic with what is allocated by synthesis for one SRAM controller. Since it is essentially a piece of control logic, the synthesised netlist is largely dominated as expected by the flip-flops represented as square boxes on Fig. 13.1.

The resources devoted to Pre-calibration can be analysed in a similar way. Tab. 13.3 provides the register / operator distribution of predicted resources. The expense related to arithmetics is here very noticeable with a 16×16-bit multiplier accounting for half of the resources.

Tab. 13.4 presents the overall results, while Tab. 13.5 approximately decomposes the figures on a structural basis.

13.3.4 Timing

Although the timing predictions made by synthesis are only very preliminary, Synplify reports as number of paths on which the corresponding constraints will be the most difficult to meet. Those critical paths are identified based on a model of the response of each cell within the selected chip and routing assumptions corresponding to a worst case. Based on these, the differences between the constraints imposed by the designer and a first estimate of delays may be measured. Tab. 13.6 collects the main figures for the worst critical paths in our design. With only positive slacks, it shows that our objective of proposing a "slow" design is met with advantages in reliability and easy routing.

[3]Another critical resource being the amount of internal RAM.
[4]Including ITAR restrictions on larger RTAX-S parts.

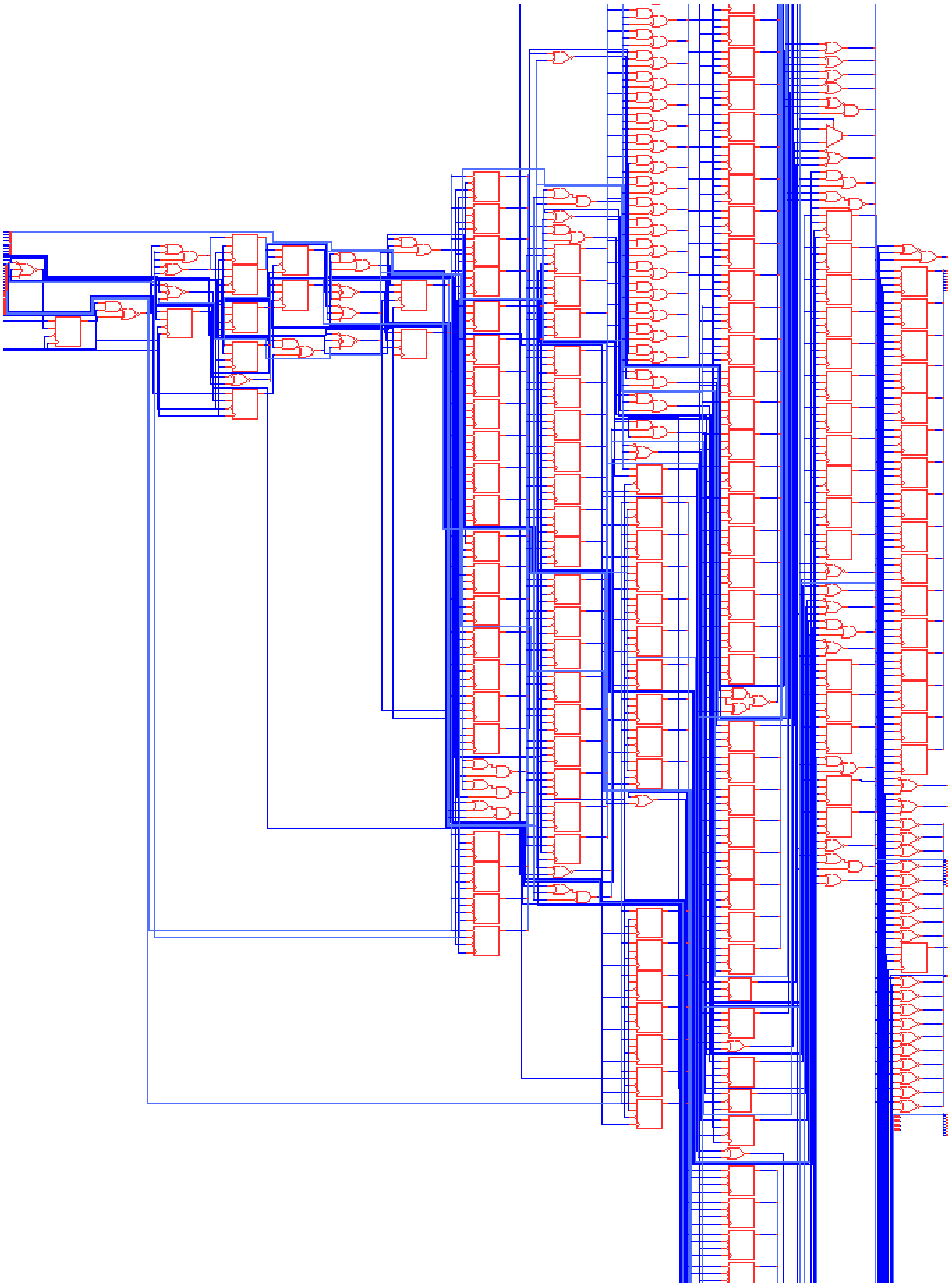


Figure 13.1: Partial view of the netlist synthesised by Synplify for one SRAM controller. The red elements are general purpose Versatiles in Actel's nomenclature and boxes represent flip-flops while the different shades of blue differentiate 1-bit wires (lighter) from networks (darker blue).

Purpose	Signals	Ports	Note
Control signals	15	8	
FSM states	6		sequential state encoding
Address buffers	4×19	2×19	
Data buffers	3×16	3×16	
Total	145	94	239 cells
Allocated by synthesis			599 cells

Table 13.2: Predicted and allocated cells for the instantiation of a single SRAM Controller with retiming enabled. The increase in resources is here notable (about 100%) as compared to without retiming because the Controller is the component where the tightest timing constraints are imposed (worst slacks are relaxed by approximately 50% also in Tab.13.6).

	Registers		Arithmetics	
Purpose	Number	Cells	Purpose	Cells
coefficients	6	76	$\times a^j$	900
addresses	3	57	$+b^j$	100
sample Values	4	64	round-off	60
replacement	3	48	correction	50
intermediate		106	average	50
Total		371		1160
Total	1531			
Synthesis	1844			

Table 13.3: Structure of predicted resource allocation for Pre-calibration.

13.4 Conclusion

With the main issue of exaggerate occupation on the ProASIC3E 600 solved by using a larger die, the previous sections establish the feasibility of the proposed design on the hardware platform envisaged. Looking back to our representativity objective also, the quality of the developments described in this part are compared both with criteria applicable to a FM (Tab. 13.7) and with ESA's quality standards (Tab. 13.8) below. Beside the manual adaptation which will be required for porting the design to a RTAX-S device, these tables show that significant efforts will have to be devoted to quality and reliability, in particular for exhaustive testing and documentation. This is not a surprise. It is even, to some extent, the very purpose of a prototype and will benefit from the upcoming availability of a database of test cases for validation.

Resource	Synthesis	ProASIC3E 600	ProASIC3E 1500
Flip-flops	14722	13 824	38400
RAM	19	24	60
IO cells	114	147	147

Table 13.4: General resource usage reported by synthesis as compared to those of the two ProASIC3E platforms.

Architecture		Processing core	
Component	Cells	Component	Cells
Framework	137	General	106
Handshake manager	329	Scheduler	327
Debug core	233	Pre-calibration	1844
Processing core	11970	Buffering	210
Controllers	2×599	Histogram	4140
Switch	137	Mode	1533
CLK division	35	Pixsel	3900
Total	14722		11970

Table 13.5: Approximate distribution of synthesised resources at a structural level.

Domain	Requested period (ns)	Slack (ns)	Logic levels	Paths
CLK	8	2.831	2	SRAM controller's `ENABLE` to IOs
		3.330	3	SRAM controller's triggering of write FSM
		1.523	3	SRAM controller's data output to the SRAM
		6.575	1	SRAM controller's data input from the SRAM
SCLK	32	27.002	2	Histogram's `MAX_RAM` access
		27.633	1	Mode's `MAX_RAM` access
DCLK	992	979.799	9	Pre-calibration's intermediate arithmetics

Table 13.6: Main critical paths identified by synthesis per clock domain (after retiming). The number of logic levels indicate how many cells are located on the path between the two consecutive points required to operate synchronously. The signals' effective periods correspond to the sum of the individual cells' delays and of propagation between them because of the negative impact of interconnections.

Category		Demonstrator	FM
Coding standards		see Tab. 13.8	
Verification		see Tab. 13.8	
Internal	radiation (RAM)	no protection	EDAC or TMR
	initialisation (RAM)	JTAG	reset & update logic (EPROM)
	macros (RAM)	uses dual-port	two-port exclusively (section 12.2.5)
	macros	PLL for main clock	external clock generation
	parameters	hard-coded	reset & update logic
	margin (resources)	none (600) $\simeq$ 100% (1500)	100% in phase B
	margin (timing)	$\simeq$ 20%	100% in phase B
	processing	missing CC-labelling and object creation	complete flow
Technology	FPGA	flash	anti-fuse or SRAM
	SRAMs	COTS	radhard or SEU-protected
	boards	crafted	PCB
	wiring	ribbon cables	PCB
	test points	flying probes	connectors
	interfaces	missing real-time software interface	complete
Execution	SRAM initialisation	$\overline{\text{debug}}$ with $\overline{\text{upload}}$ mode	reset logic
	default state	$\overline{\text{debug}}$ with $\overline{\text{upload}}$ (section 9.6.3)	nominal operation
	power on	through Starter Kit	monitored

Table 13.7: Shortcomings of the demonstrator compared to a FM configuration.

Criterion	Status
General	
VHDL-93 standard only	✓
No tools' specific types	✓
Consistent English	✓
Emphasise readability	✓
Indent with space character	✓
No generated VHDL	Smartgen Divgen
Floating point portability	✓
Implementation limitations	specified ranges
Preferred predefined operators	✓
Documentation	
Fully documented	✓
Descriptive comments	✓
Headers	incomplete
API descriptions	incomplete
Document configuration files	Srecords
Version control	subversion
Names	
No extended identifiers	✓
Recognisable active low signals	n- or $\bar{s}$
Name according to purpose	✓
Traceable signals via names	✓
Unique identifiers	component-wise
No encapsulation for renaming	✓
Signals	
MSB at left position	✓
Consistent index ordering	✓
No use of default value	✓
No number of values in type clause	symbolic
No use of enumeration order	✓
Use standardised types	✓
No global signals	✓
Minimum duplicate signals	✓
Commutative resolution	none
Associative resolution	none
No shared variables	✓
No guarded constructs	✓

Criterion	Status
Files	
`std.textio` for file access	✓
No `std.textio.input`	✓
No `std.textio.output`	✓
No file-based communication	✓
Ports	
No top-level `buffer` ports	✓
Ports in logical order	function-wise
Named-association port mapping	✓
`std_logic`-derived ports only	✓
No linkage port mode	✓
Entities & architectures	
Only `port` and `generic` in entity	✓
Assertion severity nomenclature	✗
Sensitivity lists for processes	✓
Labels for processes	✓
Labels repeated at end	✓
Identical `component` and `entity`	✓
No configuration clauses	✓
Packages & Libraries	
Preferred IEEE	✓
Packages by functionality	layer-wise
Consistent ordering	✓
Full documentation	✓
Individual test benches	✗
Distinct libraries for models	✗
No reference to `Work` library	✗
Test library for test benches	✗
Minimum `library` & `use`	debugging
Verification	
Independent verification	✗
Comparison to other model	bit-wise vs. GD
Test bench verification only	✓
Test of every executable line	✗
Test of boundary conditions	✗
Test of singularities	✗
Fast board-level model	component-level
Pull-up & pull-down modelling	✗

Table 13.8: ESA VHDL coding guidelines (from [52]) and status of the demonstrator.

Conclusion

The work presented in this thesis started with the early recognition that the autonomous detection of objects on board Gaia would represent a challenge several orders of magnitude above previously existing solutions, both scientifically and technically. The first because of the wide spectrum of objects and configurations to be handled and the need for very high detection rates and reliability, the second because of the limited time and processing resources available. The proposed scheme was begun during phase A in the frame of the OBDH working group set up by ESA which provided an understanding of Gaia's aims necessary for drafting the algorithms through close interactions with the scientists. It was then studied from an engineering point of view during the PDHE TDA which, in turn, examined it in terms of feasibility and concluded with a recommendation at using a mixed software and hardware architecture. Although a different approach was selected by the prime contractor (Astrium) when the industrial phases began, we have devoted efforts to implementing the proposed methodology under these guidelines with the intent to further validate these preliminary results with a demonstrator.

Our analysis spans the entire process from a discussion of the needs all the way to implementation details via the elaboration of algorithms. In so doing, it addresses design difficulties, be they related to the science, the image processing or the technology, not successively but in a global manner which amounts to co-design of the methods and the platform. Although rooted in previously existing approaches, the sequence of operations has been thoroughly re-examined and adapted to Gaia's constraints and aims to yield a formulation and an implementation which are original and worthy of interest as possible answers to part of the processing challenge on board Gaia.

Taking into account both the full variability of observing conditions as regards object types, configurations on the sky and instrument status and the specifications issued by ESA as part of the ITT for Gaia, a sequence of four main tasks has been proposed: calibrating sample values, estimating the sky background, identifying the objects and, finally, characterising them. Calibration comes as a first prerequisite to provide firm grounds for the image processing and ensure the reliability of measurements. The fixed-point arithmetics introduced at this level corrects the full dynamics of sample values with a probability for making ± 1 ADU errors amounting to only 8% in the worst case. A mechanism is also introduced which replaces bright or dead samples with relevant values to avoid systematic effects. As a second prerequisite to focusing on the stars themselves, the local background is studied at a regional scale. In each region, the mode of the distribution of sample values provides a measure highly robust to the pollution induced by the presence of objects yet of acceptable precision. Although difficult to fully characterise, our non-linear scheme has been tested over the full range of stellar densities and on textured backgrounds and is found to compare with the behaviour of low-pass filters but with computations both considerably reduced in volume and rendered compatible with the TDI-driven progressive acquisition of data. Relying on these estimates, the relevant samples can then be identified to achieve a data flow reduction of the order of 90% in worst density cases. A first simple object model (SOM) is introduced at this point to structure this data in CCs through a labelling algorithm designed to address the samples in the raster order resulting from the read-out.

This transition from sample-based to object-based processing also corresponds to the migration from the hardware, which performs systematic operations on a large volume of data under strict timing constraints, to the software, which is more adapted to the objects' random types and arrival times. This characterisation engine is devised to reduce type I errors (false positives) based on the previous processing which has minimised type II ones (false negatives). It filters out false detections related to noise, stars fainter than the limit of interest and PPEs prior to providing a finer description of objects based on a second object model (FOM) which allows for identifying components in compound systems and for producing the final measurements expected by the rest of the on-board processing chain. In spite of the yet incomplete scientific validation, integration of our scheme to Pyxis and Gibis and the rather large testing resulting from their use by the Gaia community confirms that the objectives are well understood and that the prototype meets the scientists' expectations. Detection probabilities above 95% are obtained not only for isolated objects but also in more difficult configurations where they crowd together as a result of stellar density or complex physical structures like DMSs. Simultaneously, the false detection rate is maintained below the level of one every 100000 samples, even at the maximum density specified to industry, and robustness to PPEs allows for continuing operation with negligible performance degradation under fluences as high as 100 particles/cm^2/s, thereby limiting the downtime and ensuring the validity of the content of the telemetry flow.

Technically speaking, the introduction of dedicated hardware, by opening a wide range of structural solutions and allowing a precise allocation of resources, contributes to solving the processing bottleneck identified during the PDHE TDA. The constraints on power consumption and area have led to relying on a pipeline whose successive steps employ clocks adjusted to the complexity of the calculations to be performed. This, in turn, relaxes timing and routing requirements. With an emphasis on control rather than on arithmetics in spite of our having clearly favoured accuracy throughout, it achieves calibration of input values, estimation of the sky background and selection of relevant samples with as much resources as would be required by 4 32×32-bit multipliers.

The software and hardware descriptions have been extensively verified at the bit level and can be considered as sufficiently mature for a review at the PDR level [48] and for the selection and procurement of parts. Targeting only detection, the resource usage (area, RAM) points to relying on a RTAX-S 1000 die for what concerns the FPGA and a Leon processor for the software while, accounting for the other on-board tasks, increased performance can be obtained with a larger RTAX-S FPGA (2000 or 4000) or using Maxwell's SCS750 board.

Beside the developments presented in this text, the experience and the insight gathered on the subject has allowed us to contribute to the project in return. For Gaia specifically, in the frame of the PDH-S contract, we have participated to the PDHE TDA and the preparation of the specifications during phase A[5] before supporting Astrium's team on the needs and on the algorithms during phase B[6]. More generally, our endeavours have also been reported in the frame of the ALGOL[7] ACI as contributions to the field of highly constrained embedded computing.

As of this writing, a first round of testing (see appendix A) has been carried out on the available test platform for the validation of the FPGA's external interfaces. Limited adjustments in the logic and more important ones in the supporting PCBs have shown the way to a second generation of PCBs intended to allow reliable operation and to extend the existing simplified description to the complete configuration (with the introduction of SRAM0 and the availability of a Starter Kit equipped with a ProASCI3E 1500 die). Finally, with the upcoming availability of a representative set of test cases for validation, it will also be of interest to compare the performances of our demonstrator to those of the industrial system planned. Additionally, beside optimisations on logic and power, it will then become possible, based on these firm ground, to extend the demonstrator by coupling the FPGA with a real-time software engine running on a Mac G3 [8] which has been reserved for the occasion.

In spite of our focus on Gaia, the resulting R&D has a wider relevance, in particular, to the next generation of satellites which involve ever-increasing acquisition capabilities and are confronted to limited transmission bandwidths. Autonomous analysis capabilities for the intelligent control of the detectors as well as the management of the resulting data, as described in this thesis, are bound to become an integral part of the architectures for missions to come [65].

[5] 5 technical notes as first author, 3 as collaborator.

[6] 11 technical notes as first author, 2 as collaborator

[7] 3 technical notes as first author.

[8] Equipped with a PPC750 processor.

Appendices

Appendix A

Test platform and setup

This appendix provides an overview of the design and setup of the analog test platform presented in section 8.4. The schematics, mostly copied from [66], represent it as originally designed by F. Rigaud[1]. A concise history of tests carried out to validate the system's external interfaces (data in (DI) and data out (DO) handshake protocols) together with some of the difficulties encountered and the adaptations which had to be made on the platform as a consequence are also included.

Schematics

Instead of full integration on a single board, several ones have been introduced as a means to provide flexibility for development and test purposes. For each board, the underlying schematic and layout are represented:

- Fig. A.1 and A.1 for the main board (C1) connecting the FPGA's pins on Actel's Starter Kit with the two SRAMs on one hand and the interface with the PC on the other hand,
- Fig. A.5 for the two small boards on which ISSI's SRAMs have been soldered,
- Fig. A.3 for the PCB (C2) supporting the interface with the PC with its voltage adaptations and active terminations.

Finally, Tab. A.1, A.2 and A.3 provide the correspondence between pins and signals.

Tests and setup

The tests conducted for validating the external interfaces have met with many difficulties. They have shown that for validation purposes, little can be taken for granted and meticulous examination of each and every aspect of the system must be the rule. This should begin with the verification of the PCBs' environment since errors in the documentation of the FPGA's Starter Kit or non-conformant behaviour of Adlink's IO board have been found to exist. It is only once this basis firmly established that the difficulties related to the system itself can begin to be tackled.

As on our test platform malfunction can result from the logic or the circuitry, or any combination of the two, all means of inspecting the behaviour of the two separately have proven precious in tracking down the underlying causes. In this respect, simulation at a variety of levels, and particularly after place and route, has been a prerequisite to reprogramming the device for although it cannot guarantee the logic to be faultless (especially for what concerns synchronisation issues between the different clocks), the errors it allows to identify are very real. We have, unfortunately, been missing the equivalent for validating the PCBs (a spectrum analyser working in the MHz frequency range) and to inspect their frequential response and to assess the quality of the decoupling between their various parts. We have, nevertheless inspected the behaviour of the system with the help of the "test receiver" (the PC receiving the data output by the FPGA (section 9.2.1), of a 60 MHz oscilloscope and a 68-channel 4 GHz logic analyser.

Foremost among the issues identified are problems related to the presence of multiple power supplies, common impedance coupling and to ground bounce which have been found to result in cross-talk between signals or to disrupt even the FPGA's internal logic. The adaptations conducted on the platform to reduce these effects (the suppression of active components, the reduction of currents and slew and efforts to decouple different parts of the circuits) if improving its functional behaviour, have not however solved its intrinsic sensitivity. The following list chronologically summarises the testing conducted and the modifications carried out to fully understand the causes of the difficulties encountered in an effort to make the platform as fully operational as possible (notably for testing accesses to the SRAMs and the FPGA's logic) and to avoid repeating the design errors in the preparation of a second version.

[1]In particular, the SPASs' full 19-bit address and 16-bit data buses are connected to the FPGA.

1. Modified the DO FSM logic to ensure data are always stable when `DI_REQ` is asserted.

2. Modified pin configuration for `DO_REQ` (routed to pin 34 instead of 32 as declared in Tab. A.1).

3. Modified the IO standard for DO transmissions (`DO_REQ` and all `PB` signals): changed from LVCMOS 2.5V to LVCMOS 3.3V to achieve coherent values between those published by the IO board and those seen by the FPGA (as a consequence all interfaces follow the LVCMOS 3.3V standard).

4. Glitches on control signals (`Reset`, `DO_REQ`, `DO_ACK`, `DI_REQ`, `DI_ACK`) when performing DI and DO exchanges simultaneously:

   - Correlated voltage fluctuations on `Vref1` and `Vref2` leading to cross-talk between signals: removal of active terminations `RP1` and `RP2`.
   - Removed `R1` and `R4` to reduce the impedance on the output of the operational amplifiers.
   - `U1:A` (TL462CN) failed to function reliably as a voltage follower (probably due to currents exceeding the part's specification): first replaced by another operational amplifier, then removed (see item 8).
   - Tested using an external 3.3V power supply instead of the Starter Kit's: resulted in a reduction of the amount of glitches on the `Reset` signal (most sensitive to cross-talk).
   - Tested at various load levels (currents drawn by transitions on the data bits of DO and DI): at constant load (constant data or constant number of bits in "high" state, eg. oscillating between "0101010101010101" and "1010101010101010") the system was functional. Conversely, at maximum load (data switching between "000000000000000" and "1111111111111111"), the system was unstable: each successful transaction impedes exchanges during the next DCLK period (see item 14).
   - Removed all active terminations (including on the IO board at this stage, see item 7).
   - Added decoupling capacitors in the vicinity of power pins (a fine characterisation of the decoupling could not be achieved but would deserve attention because the small capacitors used could resonate with larger ones on the PCBs).

5. Reduced drive strength (currents) on the FPGA's output ports (12 mA to 6 mA) to suppress overshoots on the `DO_REQ` signal.

6. Improved the synchronisation of clocks (between SCLK and DCLK) via the introduction of a constraint on delays between rising edges (as part of synthesis).

7. Used the IO board's active terminations on DI transactions.

8. Removed all remaining active components:

   - Replaced `U1`, `U2` and `U3` (SN7404) by Zener diodes and resistors for voltage adaptations following [67] (see item 9).
   - Stopped using the IO board's power supply (1 remaining from the Starter Kit to power the FPGA).

9. Tuned the system versus the IO board's behaviour:

   - `DO_REQ` only followed the specification if `DO_ACK` was asserted for approximately 64 ns (otherwise `DO_REQ` was reasserted before `DO_ACK` is deasserted).
   - Modified the FSM in charge of controlling `DO_ACK` to rely on 3 states to assert `DO_ACK` during 2 SCLK periods.
   - Adjusted the resistors coupled with the Zener diodes (47 Ω instead of 220 Ω): slow edges on `DI_ACK` caused only 2 transactions to be carried out during one DCLK period instead of 4.
   - Tested adjusting levels through resistors: would yield faster edges and allow for adapting impedances but the Zener diodes in blocking reflections should lead to cleaner signals.

10. Introduced a configurable number of `BUFD` macros in the VHDL sources on the paths to output data ports to de-synchronise switching signals [71] (tested only in post place-and-route simulation).

11. Improved the synchronisation between DCLK and `DI_REQ` assertion for better readability of the protocols (suppressed an intermediate data buffer introducing a delay).

12. Introduced VHDL code for configuring output data based on few bits in the input to test the behaviour of the system versus different content on input and output with reduced coupling between the two (previously defaulted to copying the input on output): system functional up to 7/8 of maximum load.

13. Modified pin assignments following [68], [69] and [70]:

    - Switched `PB6` and `PB7`.
    - Introduced pseudo-ground pins by forcing the unused `DO_TRIG` and `DI_TRIG` to a "strong" 0V (24 mA drive strength with high slew).

14. Examined the Starter Kit's power supply:

    - Configured the voltage of banks 4 and 5 to 3.3V with `SW8` and `SW9` (the Starter Kit's documentation is erroneous in this respect as placing the switches on position "4" results in 2.5V instead of 3.3V).
    - Observed the ground voltage under different loads: activity on the output data bus DI caused the ground voltage to fluctuate (glitches) and increase in the manner of a ramp signal. The `DO_TRIG` pin showed that the FPGA's internal ground level was also affected, thereby explaining why the logic behaved unreliably under high loads.
    - Improved the routing of the ground circuitry to separate current loops, reduce the coupling with signals and the impedances (2 separate grounds remaining: one from the Starter Kit, one from the IO board).

15. Reduced the number of probes connected from the logic analyser and the oscilloscope: the probes finely alter the electrical behaviour.

16. Reconfigured the FPGA output pins:

    - Lowered the drive strength to 2 mA.
    - Increased the capacitors to 150 pF (instead of 35 pF by default).
    - Reduced the slew to low.

17. Added ferrites on the connections to ground and on the ribbon cables: smoothed voltage fluctuations on the ground and improves reliability (normal operation only occasionally disrupted at maximum load).

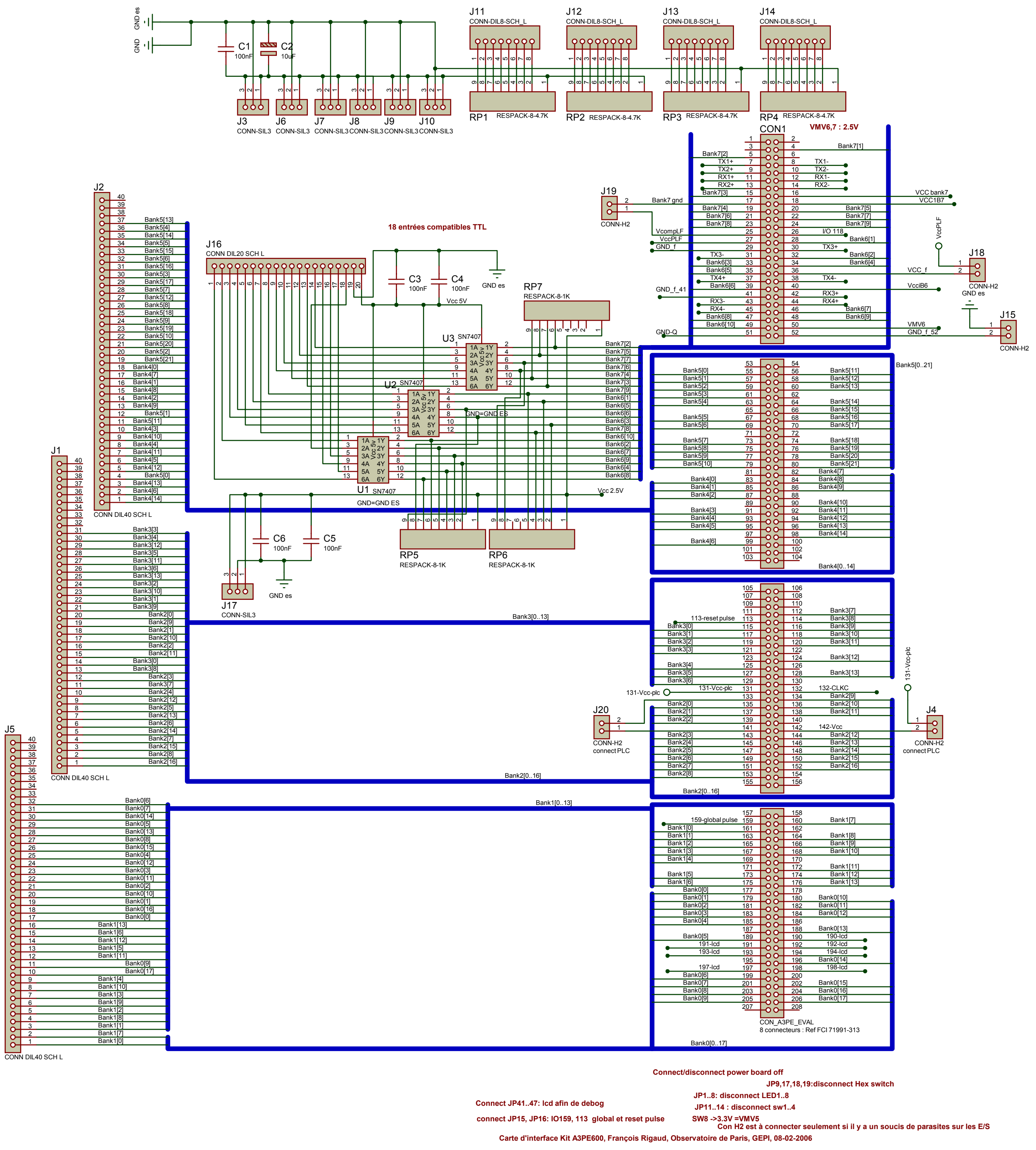

18 entrées compatibles TTL
CON1
VMV6,7 : 2.5V
CON_A3PE_EVAL
8 connecteurs : Ref FCI 71991-313
Connect/disconnect power board off
JP9,17,18,19:disconnect Hex switch
JP1..8: disconnect LED1..8
JP11..14 : disconnect sw1..4
SW8 ->3.3V =VMV5
Con H2 est à connecter seulement si il y a un soucis de parasites sur les E/S
Connect JP41..47: lcd afin de debog
connect JP15, JP16: IO159, 113 global et reset pulse
Carte d'interface Kit A3PE600, François Rigaud, Observatoire de Paris, GEPI, 08-02-2006

Figure A.1: Original schematic of the intermediate PCB (C1) introduced between the FPGA and the SRAM and PC interface.

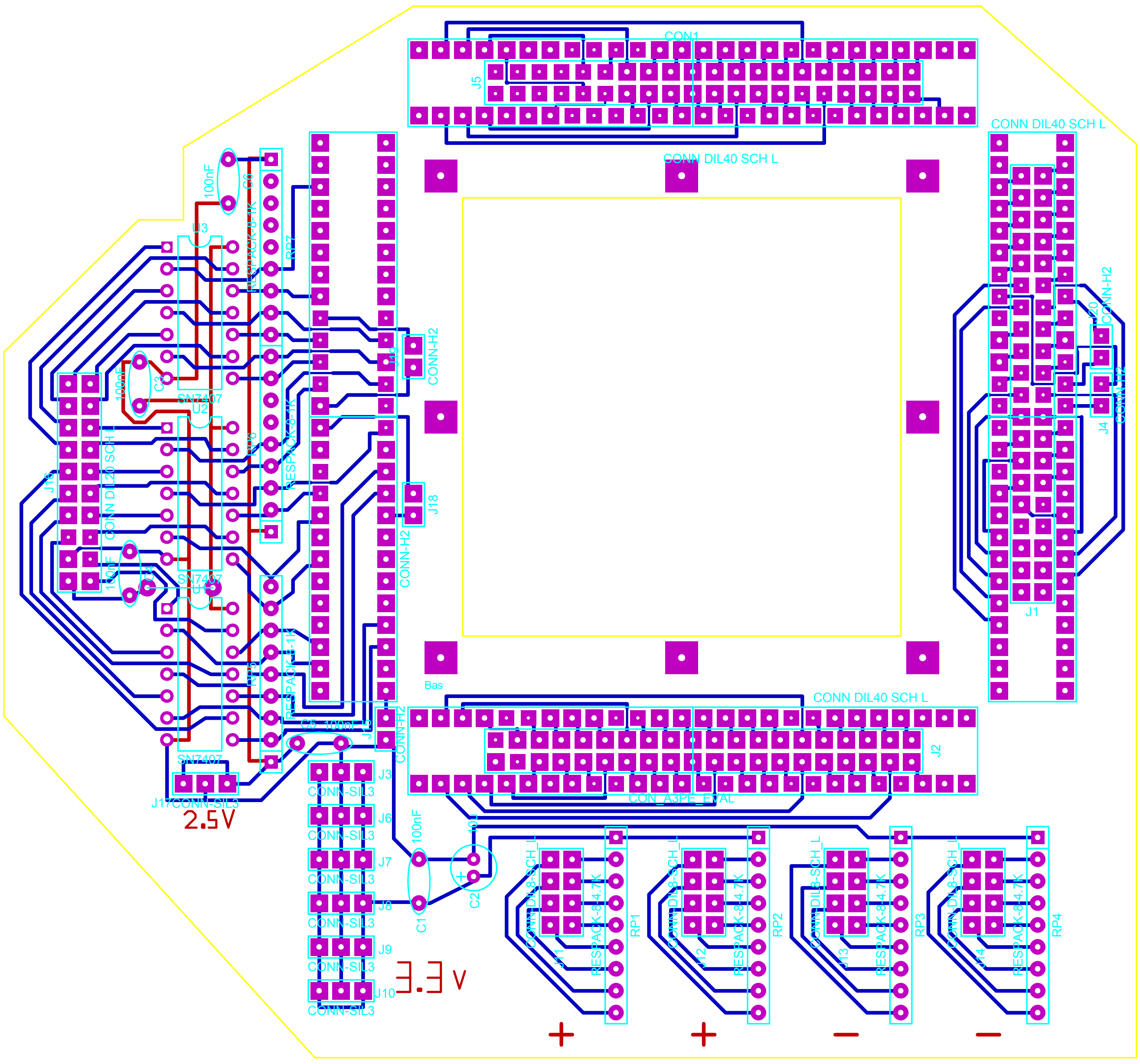


Figure A.2: Original layout of the intermediate PCB (C1) introduced between the FPGA and the SRAM and PC interface.

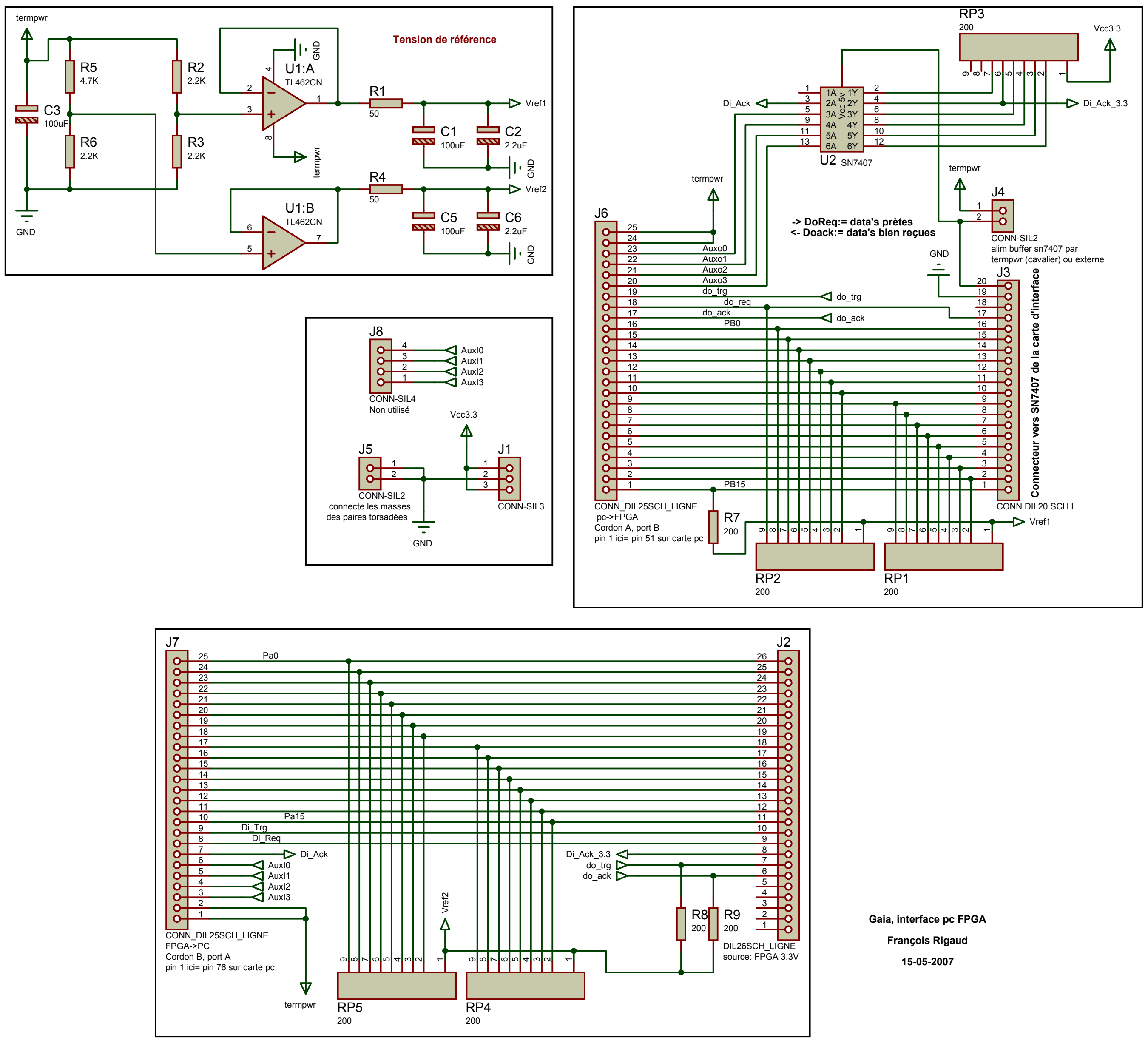


Figure A.3: Original schematic of the PCB (C2) for the interface between the FPGA and the PC with its voltage adaptations and active terminations.

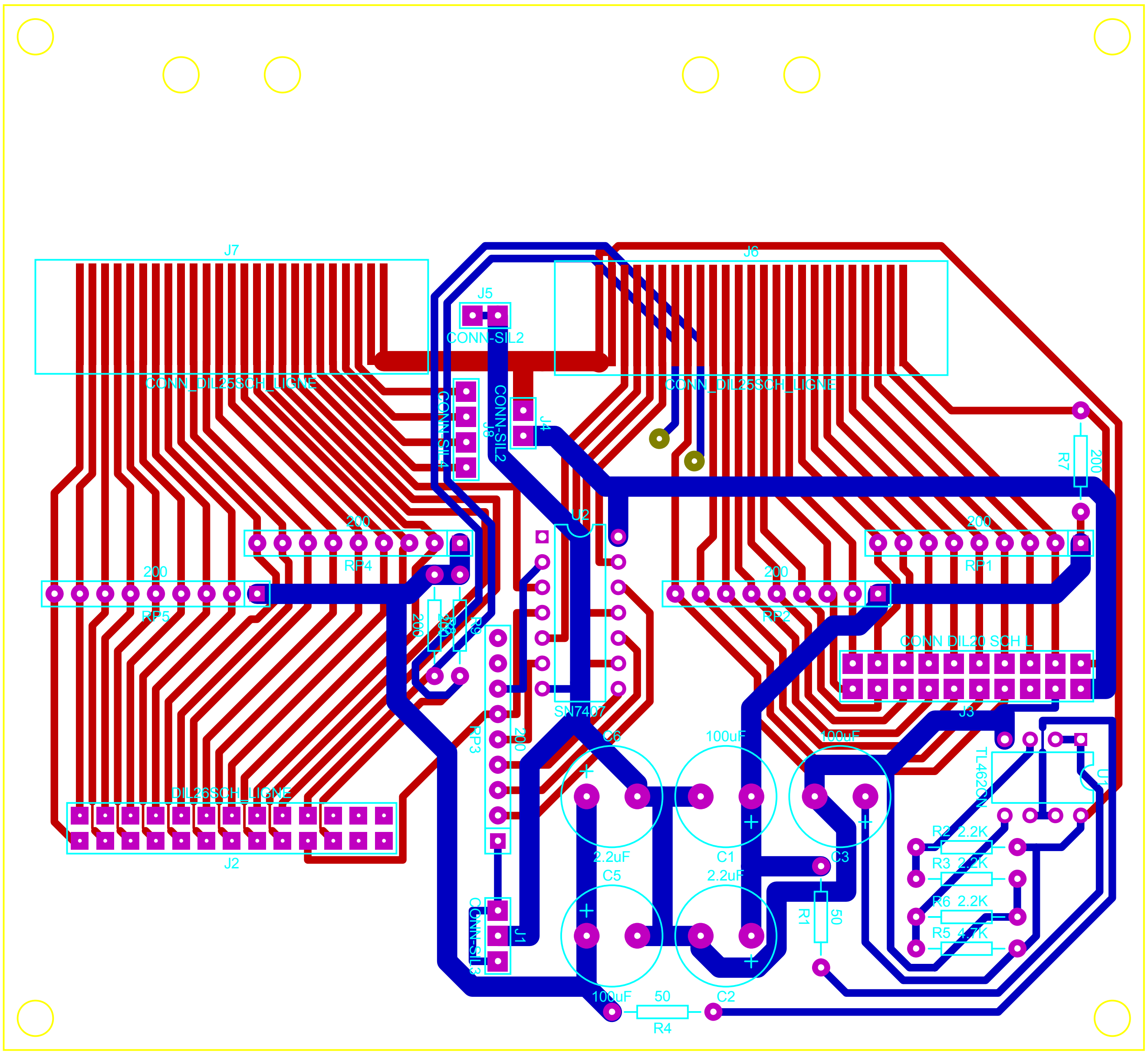
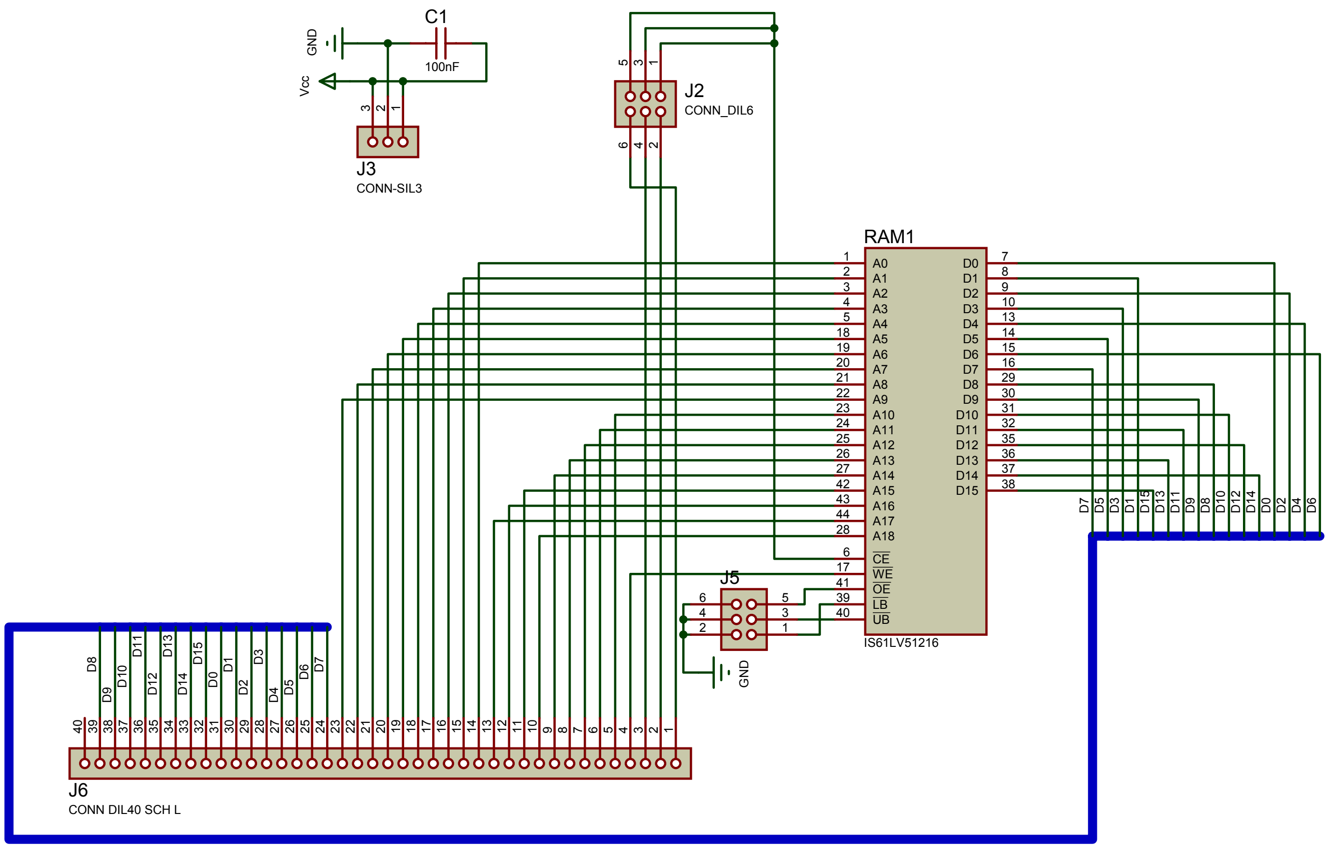


Figure A.4: Original layout of the PCB (C2) for the interface between the FPGA and the PC with its voltage adaptations and active terminations.



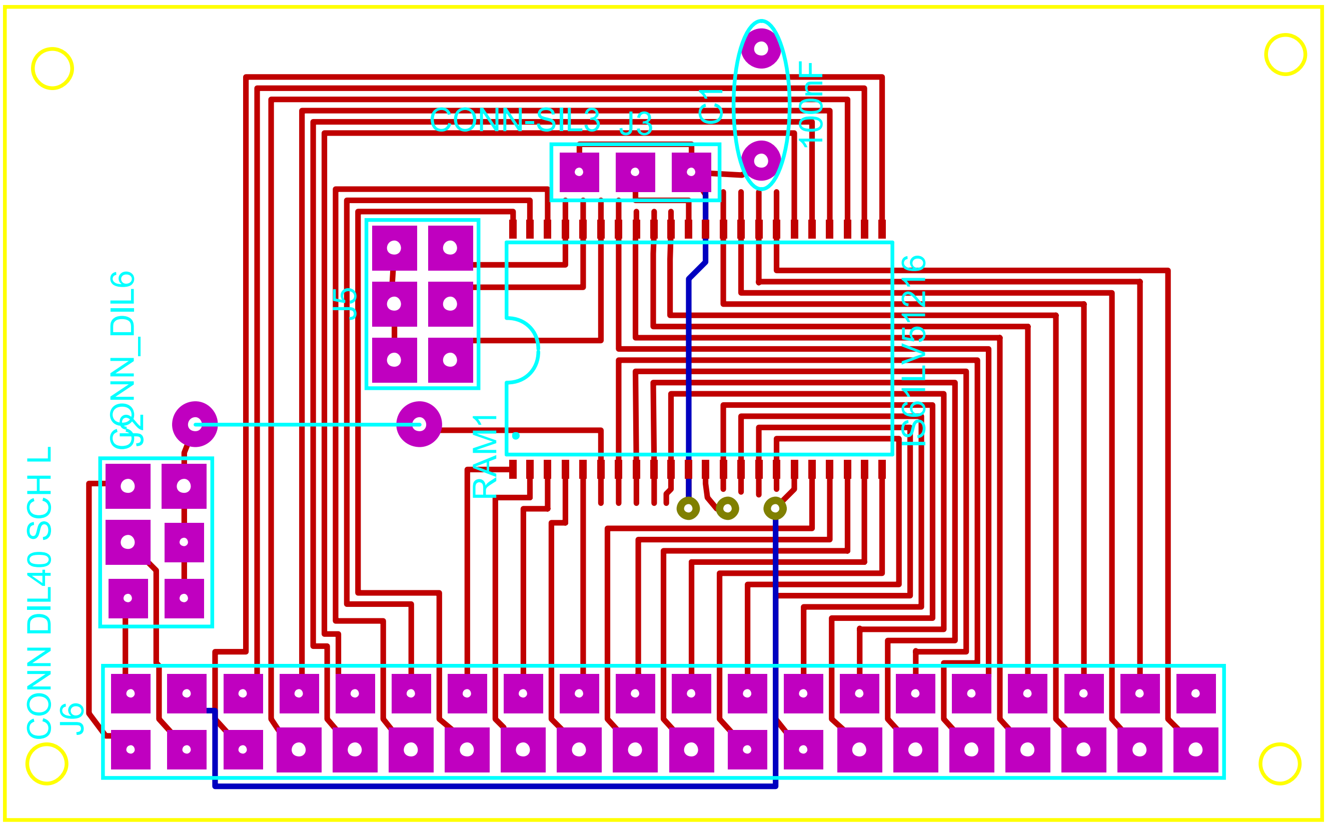


Figure A.5: Top: original schematic of the interface PCB between the FPGA and an SRAM part. Bottom: original layout of the PCB.

Signal	Source		Destination		Voltage (V)			Note	
	Entity	Pin	Entity	Pin	Source	Destination	Adaptation	Bank	Note
Do_Ack	FPGA	63	PC	scsci100, 67	3.3	3.3		5	handshake in
Do_Req	PC	scsci100, 68	FPGA	32	5	3.3	C1	6	handshake in
Do_Trg	FPGA	64	PC	scsci100, 69	3.3	3.3		5	grounded
Aux_Do3	PC	scsci100, 70	FPGA	aux yellow	NC				unused
Aux_Do2	PC	scsci100, 71	FPGA	aux green: 152	5	3.3	C2	2	upload
Aux_Do1	PC	scsci100, 72	FPGA	aux blue: 153	5	3.3	C2	2	debug
Aux_Do0	PC	scsci100, 73	FPGA	aux purple: 113	5	3.3	C2	3	reset
Di_Ack	PC	scsci100, 82	FPGA	67	5	3.3	C2	5	handshake out
Di_Req	FPGA	66	PC	scsci100, 83	3.3	3.3		5	handshake out
Di_Trg	FPGA	69	PC	scsci100, 84	3.3	3.3			grounded
Aux_Di3	FPGA	C2, J8, 1	PC	scsci100, 78	NC				unused
Aux_Di2	FPGA	C2, J8, 2	PC	scsci100, 79	NC				unused
Aux_Di1	FPGA	C2, J8, 3	PC	scsci100, 80	NC				unused
Aux_Di0	FPGA	C2, J8, 4	PC	scsci100, 81	NC				unused
PB15	PC	scsci100, 51	FPGA	47	5	3.3	C1	6	data_in (MSB)
PB14	PC	scsci100, 52	FPGA	49	5	3.3	C1	6	data_in
PB13	PC	scsci100, 53	FPGA	39	5	3.3	C1	6	data_in
PB12	PC	scsci100, 54	FPGA	35	5	3.3	C1	6	data_in
PB11	PC	scsci100, 55	FPGA	33	5	3.3	C1	6	data_in
PB10	PC	scsci100, 56	FPGA	28	5	3.3	C1	6	data_in
PB9	PC	scsci100, 57	FPGA	23	5	3.3	C1	7	data_in
PB8	PC	scsci100, 58	FPGA	24	5	3.3	C1	7	data_in
PB6	PC	scsci100, 59	FPGA	22	5	3.3	C1	7	data_in
PB7	PC	scsci100, 60	FPGA	21	5	3.3	C1	7	data_in
PB5	PC	scsci100, 61	FPGA	19	5	3.3	C1	7	data_in
PB4	PC	scsci100, 62	FPGA	20	5	3.3	C1	7	data_in
PB3	PC	scsci100, 63	FPGA	15	5	3.3	C1	7	data_in
PB2	PC	scsci100, 64	FPGA	5	5	3.3	C1	7	data_in
PB1	PC	scsci100, 65	FPGA	48	5	3.3	C1	6	data_in
PB0	PC	scsci100, 66	FPGA	46	5	3.3	C1	6	data_in (LSB)
PA15	FPGA	68	PC	scsci100, 85	3.3	3.3		5	data_out (MSB)
PA14	FPGA	61	PC	scsci100, 86	3.3	3.3		5	data_out
PA13	FPGA	70	PC	scsci100, 87	3.3	3.3		5	data_out
PA12	FPGA	73	PC	scsci100, 88	3.3	3.3		5	data_out
PA11	FPGA	58	PC	scsci100, 89	3.3	3.3		5	data_out
PA10	FPGA	75	PC	scsci100, 90	3.3	3.3		5	data_out
PA9	FPGA	74	PC	scsci100, 91	3.3	3.3		5	data_out
PA8	FPGA	77	PC	scsci100, 92	3.3	3.3		5	data_out
PA7	FPGA	76	PC	scsci100, 93	3.3	3.3		5	data_out
PA6	FPGA	79	PC	scsci100, 94	3.3	3.3		5	data_out

Table A.1: Pin assignment matrix after testing (NC stands for Not Connected).

Signal	Source		Destination		Voltage (V)			Note	
	Entity	Pin	Entity	Pin	Source	Destination	Adaptation	Bank	Note
PA5	FPGA	78	PC	scsci100, 95	3.3	3.3		5	data_out
PA4	FPGA	59	PC	scsci100, 96	3.3	3.3		5	data_out
PA3	FPGA	80	PC	scsci100, 97	3.3	3.3		5	data_out
PA2	FPGA	83	PC	scsci100, 98	3.3	3.3		4	data_out
PA1	FPGA	82	PC	scsci100, 99	3.3	3.3		4	data_out
PA0	FPGA	85	PC	scsci100, 100	3.3	3.3		4	data_out (LSB)
R1D0	FPGA	121	RAM1		3.3	3.3		3	data (in/out)
R1D1	FPGA	125	RAM1		3.3	3.3		3	data (in/out)
R1D2	FPGA	124	RAM1		3.3	3.3		3	data (in/out)
R1D3	FPGA	127	RAM1		3.3	3.3		3	data (in/out)
R1D4	FPGA	120	RAM1		3.3	3.3		3	data (in/out)
R1D5	FPGA	129	RAM1		3.3	3.3		3	data (in/out)
R1D6	FPGA	128	RAM1		3.3	3.3		3	data (in/out)
R1D7	FPGA	119	RAM1		3.3	3.3		3	data (in/out)
R1D8	FPGA	93	RAM1		3.3	3.3		4	data (in/out)
R1D9	FPGA	92	RAM1		3.3	3.3		4	data (in/out)
R1D10	FPGA	95	RAM1		3.3	3.3		4	data (in/out)
R1D11	FPGA	94	RAM1		3.3	3.3		4	data (in/out)
R1D12	FPGA	55	RAM1		3.3	3.3		5	data (in/out)
R1D13	FPGA	96	RAM1		3.3	3.3		4	data (in/out)
R1D14	FPGA	99	RAM1		3.3	3.3		4	data (in/out)
R1D15	FPGA	98	RAM1		3.3	3.3		4	data (in/out)
R1A0	FPGA	115	RAM1		3.3	3.3		3	address
R1A1	FPGA	138	RAM1		3.3	3.3		2	address
R1A2	FPGA	139	RAM1		3.3	3.3		2	address
R1A3	FPGA	136	RAM1		3.3	3.3		2	address
R1A4	FPGA	137	RAM1		3.3	3.3		2	address
R1A5	FPGA	134	RAM1		3.3	3.3		2	address
R1A6	FPGA	135	RAM1		3.3	3.3		2	address
R1A7	FPGA	116	RAM1		3.3	3.3		3	address
R1A8	FPGA	117	RAM1		3.3	3.3		3	address
R1A9	FPGA	118	RAM1		3.3	3.3		3	address
R1A10	FPGA	148	RAM1		3.3	3.3		2	address
R1A11	FPGA	149	RAM1		3.3	3.3		2	address
R1A12	FPGA	146	RAM1		3.3	3.3		2	address
R1A13	FPGA	147	RAM1		3.3	3.3		2	address
R1A14	FPGA	144	RAM1		3.3	3.3		2	address
R1A15	FPGA	112	RAM1		3.3	3.3		3	address
R1A16	FPGA	143	RAM1		3.3	3.3		2	address
R1A17	FPGA	114	RAM1		3.3	3.3		3	address

Table A.2: Pin assignment matrix after testing (continued from A.1).

Signal	Source		Destination		Voltage (V)			Note	
	Entity	Pin	Entity	Pin	Source	Destination	Adaptation	Bank	Note
R1A18	FPGA	145	RAM1		3.3	3.3		2	address
R1CE	FPGA	150	RAM1		3.3	3.3		2	$\overline{\text{CE}}$
R1WE	FPGA	151	RAM1		3.3	3.3		2	$\overline{\text{WE}}$
R2D0	FPGA	206	RAM2		3.3	3.3		0	data (in/out)
R2D1	FPGA	205	RAM2		3.3	3.3		0	data (in/out)
R2D2	FPGA	172	RAM2		3.3	3.3		1	data (in/out)
R2D3	FPGA	173	RAM2		3.3	3.3		1	data (in/out)
R2D4	FPGA	174	RAM2		3.3	3.3		1	data (in/out)
R2D5	FPGA	175	RAM2		3.3	3.3		1	data (in/out)
R2D6	FPGA	176	RAM2		3.3	3.3		1	data (in/out)
R2D7	FPGA	177	RAM2		3.3	3.3		0	data (in/out)
R2D8	FPGA	160	RAM2		3.3	3.3		1	data (in/out)
R2D9	FPGA	163	RAM2		3.3	3.3		1	data (in/out)
R2D10	FPGA	164	RAM2		3.3	3.3		1	data (in/out)
R2D11	FPGA	165	RAM2		3.3	3.3		1	data (in/out)
R2D12	FPGA	166	RAM2		3.3	3.3		1	data (in/out)
R2D13	FPGA	167	RAM2		3.3	3.3		1	data (in/out)
R2D14	FPGA	168	RAM2		3.3	3.3		1	data (in/out)
R2D15	FPGA	169	RAM2		3.3	3.3		1	data (in/out)
R2A0	FPGA	203	RAM2		3.3	3.3		0	address
R2A1	FPGA	202	RAM2		3.3	3.3		0	address
R2A2	FPGA	185	RAM2		3.3	3.3		0	address
R2A3	FPGA	184	RAM2		3.3	3.3		0	address
R2A4	FPGA	183	RAM2		3.3	3.3		0	address
R2A5	FPGA	182	RAM2		3.3	3.3		0	address
R2A6	FPGA	181	RAM2		3.3	3.3		0	address
R2A7	FPGA	180	RAM2		3.3	3.3		0	address
R2A8	FPGA	179	RAM2		3.3	3.3		0	address
R2A9	FPGA	204	RAM2		3.3	3.3		0	address
R2A10	FPGA	86	RAM2		3.3	3.3		4	address
R2A11	FPGA	57	RAM2		3.3	3.3		5	address
R2A12	FPGA	56	RAM2		3.3	3.3		5	address
R2A13	FPGA	91	RAM2		3.3	3.3		4	address
R2A14	FPGA	199	RAM2		3.3	3.3		0	address
R2A15	FPGA	196	RAM2		3.3	3.3		0	address
R2A16	FPGA	189	RAM2		3.3	3.3		0	address
R2A17	FPGA	188	RAM2		3.3	3.3		0	address
R2A18	FPGA	201	RAM2		3.3	3.3		0	address
R2CE	FPGA	84	RAM2		3.3	3.3		4	$\overline{\text{CE}}$
R2WE	FPGA	87	RAM2		3.3	3.3		4	$\overline{\text{WE}}$

Table A.3: Pin assignment matrix after testing (continued from Tab. A.2).

Appendix B

Parameters

We summarise in this appendix the entire set of modifiable parameters used by the processing for the sample-based hardware part (Tab. B.2) and for the object-based software part (Tab. B.3).

Name	Type
U16	unsigned short int (16 bits)
S16	signed short int (16 bits)
U32	unsigned int (32 bits)
S32	signed int (32 bits)
F32	floating point (32 bits)

Table B.1: Type/name correspondence.

Type	Size	Name	Value	Description
			Pre-calibration	
U16	1	`NB_PIX_FWC`	63535 ADU	depends on FWCs and ADC
U16	1	`NB_MIN_ADU`	0 ADU	minimum admissible sample value
U16	1	`DEFAULT_BKGD`	1001 ADU	replacement constant (sample value)
S16	983	$\{a_j\}$	0	gain correction $\in]-2^{-4};2^{-4}[$ (in 1.0.18 format)
S16	983	$\{b_j\}$	0	offset correction $\in]-2^{12};2^{12}[$ (in 1.12.2 format)
			Sample-based parameters (SOM)	
U32	1	`SNR1_TH`	214320 (2.2)	sample-wise SNR-threshold (in 14.18 format)
S32	1	`RON_CORR_PIX`	-206854656	RON term for sample-wise SNR-thresholding
U32	1	`GAIN`	1551892 (5.92)	encoding gain (in 14.18 format)
S32	1	`OFFSET_ADU`	1000 ADU	encoding offset: value corresponding to 0 e^-
U32	1	`MODE_BIAS`	-902 (-0.44043 ADU)	background bias correction (1.20.11 format)
U16	1	`HISTOGRAM_TH`	350	reliability threshold for histogram bins
U32	1	`DEFAULT_BKGD_VALUE`	2048679 (1000.3318 ADU)	default background value for replacement (in 0.21.11 format)

Table B.2: Parameters for the sample-based processing (in hardware) with types and values used in this text.

Type	Size	Name	Value	Description
			Faint object-related parameters (FOM)	
U16	1	MIN_SMP_SELECTED	2	threshold on number of samples to discard faint objects
U32	1	MIN_FLUX_SELECTED	44 ADU	threshold on flux to discard faint objects (without the offset)
U32	1	SNR2_TH	17626563 (14.36)	object-wise SNR threshold (in 0.14.18 format)
S32	1	RON_CORR_OBJ	-2530128116	RON correction term for the object-wise SNR-threshold
U16	1	MAX_AL_LATENCY	350	maximum allowed latency (700 TDIs)
U32	3	MAG_CLASS_THRESHOLD	80 1800 8000 ADU	flux thresholds for forming the priority queues
			Component segmentation parameters (FOM)	
U16	1	MIN_DEBLEND_SIZE_AL	3	minimum AL sample size for attempting component segmentation.
U16	1	MIN_DEBLEND_SIZE_AC	6	minimum AC sample size for attempting component segmentation.
U16	1	NB_MARKER_CLASSES	12	number of magnitude classes for filtering the markers
U16	1	NB_ENVELOPE_PIECES	3	number of pieces for the piece-wise false marker envelopes
U16	1	MARKER_ALIGN_THRESH	2	threshold for considering markers aligned (in AL or AC).
U32	11	MARKER_CLASS_THRESHOLD		thresholds between magnitude classes of markers (ADU)
U32	12×10	MARKER_ENVELOPE_AL		parameters for AL piece-wise linear envelopes.
U32	12×10	MARKER_ENVELOPE_AC		parameters for the AC piece-wise linear envelopes.
			PPE rejection parameters (FOM	
U32	1	INTERIOR_FLUX_TH	4000 ADU	minimum flux for testing the number of interior samples (without the offset)
U32	1	INTERIOR_SLOPE_TH	2500 ADU	interior energy per sample threshold for rejecting PPEs (without the offset)
U16	1	EDGE_GRAD_TH	1085 ADU	threshold on the average edge value
U32	1	COSMIC_FLUX_TEST	1040	threshold for the flux/surface test
U32	1	COSMIC_SURFACE_TEST	220	surface threshold for the flux/surface test
U16	1	MISALIGN_FLUX1	1626 ADU	low flux threshold for the misalignment test (without the offset)
U16	1	MISALIGN_FLUX2	17000 ADU	high flux threshold for the misalignment test (without the offset)
F32	1	MISALIGN_MIN	0.03125	maximum misalignment for high flux objects
F32	1	MISALIGN_MAX	0.8	maximum misalignment for low flux objects
U16	1	MISALIGN_TH1	500	gain term for misalignment threshold
U16	1	MISALIGN_TH2	1000	flux offset for misalignment threshold
F32	1	PPE_DMS_EDGE_TH	0.875	edge sample ratio for separating PPE and DMS failing misalignment

Table B.3: Parameters for the object-based processing (in software) with types and values used in this text.

Appendix C

Simulations

This appendix provides the main test cases representative of the different densities on the sky. These have been produced in support to the PDHE TDA to evaluate the behaviour of the platform. The simulations in Fig. C.1, C.2, C.3, C.4 have been produced with Gibis 2 for the 2002 baseline [34] with the Besançon Galaxy model [40]. The star catalogue of Fig. C.5, on the opposite, is based on real observation data with Hubble's WFPC2 camera which features an angular resolution comparable to Gaia and has been resimulated with Gibis [72]. All contain a uniform background at the level of 22.35 mag/deg^2, typical PPE fluences (of the order of 8 particles/cm^2/s) and single stars only (the DMS are only visual ones related to the high density).

Test cases related to density

Stars per deg^2	Figure	Description
3200	C.1	lowest density close to the galactic pole (l=74, b=74)
22250	C.2	average sky density (l=74, b=15)
195000	C.3	representative density in the galaxy disk (l=74, b=0)
609000	C.4	design density case (l=54, b=0)
$3.31.10^6$	C.5	maximum density case in Baade's windows

Test cases for calibrating the filtering of markers (section 7.3.3)

Number of stars	G range	Description
9	[6; 7[	
18	[7; 8[	
46	[8; 9[	
122	[9; 10[	
254	[10; 11[	
233	[11; 12[	
557	[12; 13[	
565	[13; 14[	Isolated stars positionned on a grid with random magnitude,
2453	[14; 15[	color and sub-sample locations.
2401	[15; 16[	
3274	[16; 17[	
3317	[17; 18[	
3318	[18; 19[	
3340	[19; 20[	
3311	[20; 21[	
3338	[21; 22[	

Test cases for measuring the performance of component segmentation (section 7.3.4)

Number of stars	Name	Description
2590	bin-1	
2612	bin-2	
2576	bin-3	
2638	bin-4	
2638	bin-5	
2560	bin-6	
2560	bin-7	
2540	bin-8	
2620	bin-9	
2652	bin-10	Isolated double stars positionned on a grid with random magnitude,
258	bin-11	sub-sample locations and color for the primary, and random separation,
258	bin-12	orientation, magnitude difference, sub-sample location and color for
254	bin-13	the secondary.
260	bin-14	
262	bin-15	
256	bin-16	
254	bin-17	
252	bin-18	
256	bin-19	
260	bin-20	
254	bin-21	
256	bin-22	
258	bin-23	
168	bin-24	
1708	bin-25	

Table C.1: Description of the simulations

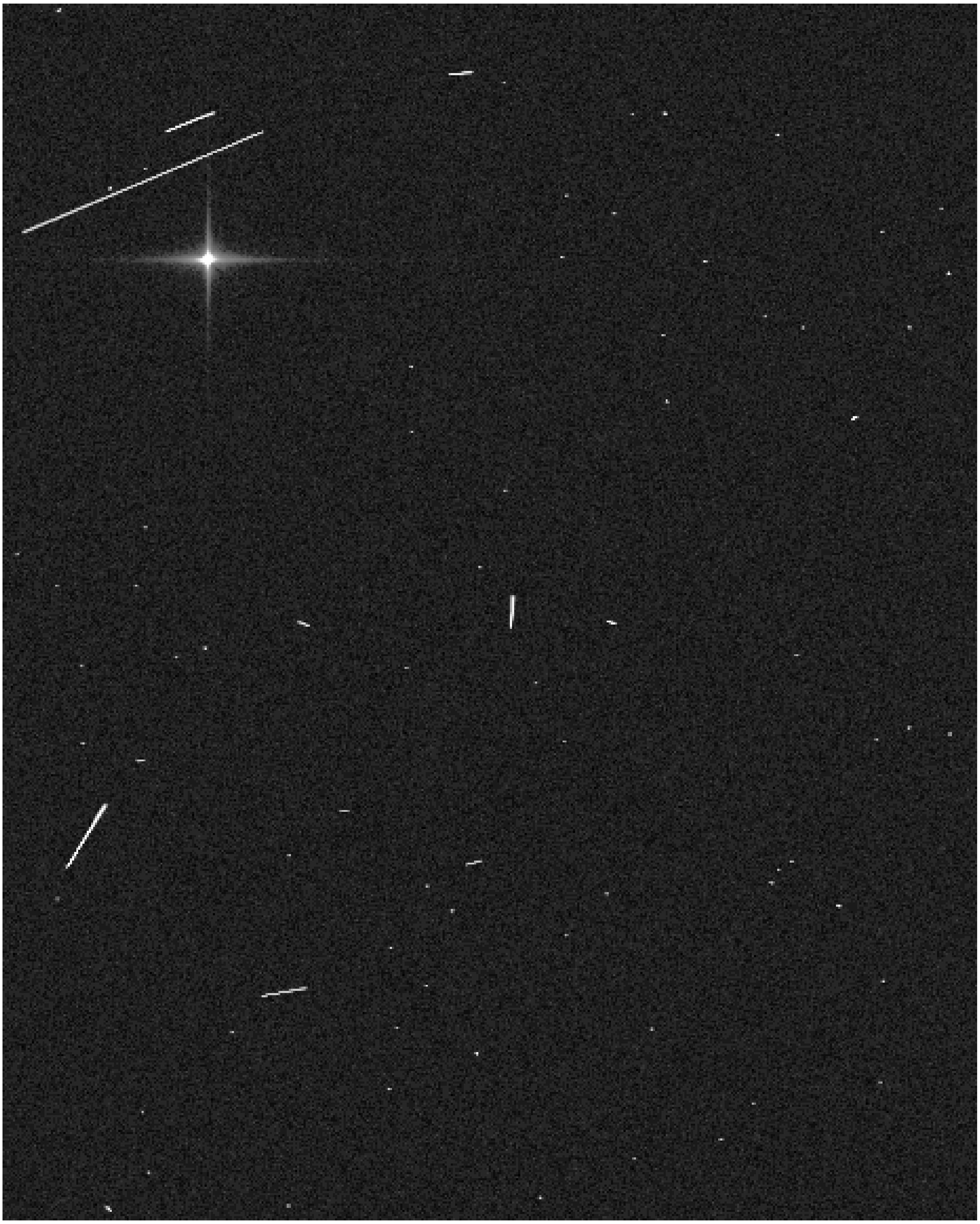


Figure C.1: The minimum density test case (3200 stars/deg^2) at galactic coordinates $l = 74$ and $b = 74$ (close to the galactic pole). The snapshot is approximately 400 AL$\times$ 525 AC SM samples.

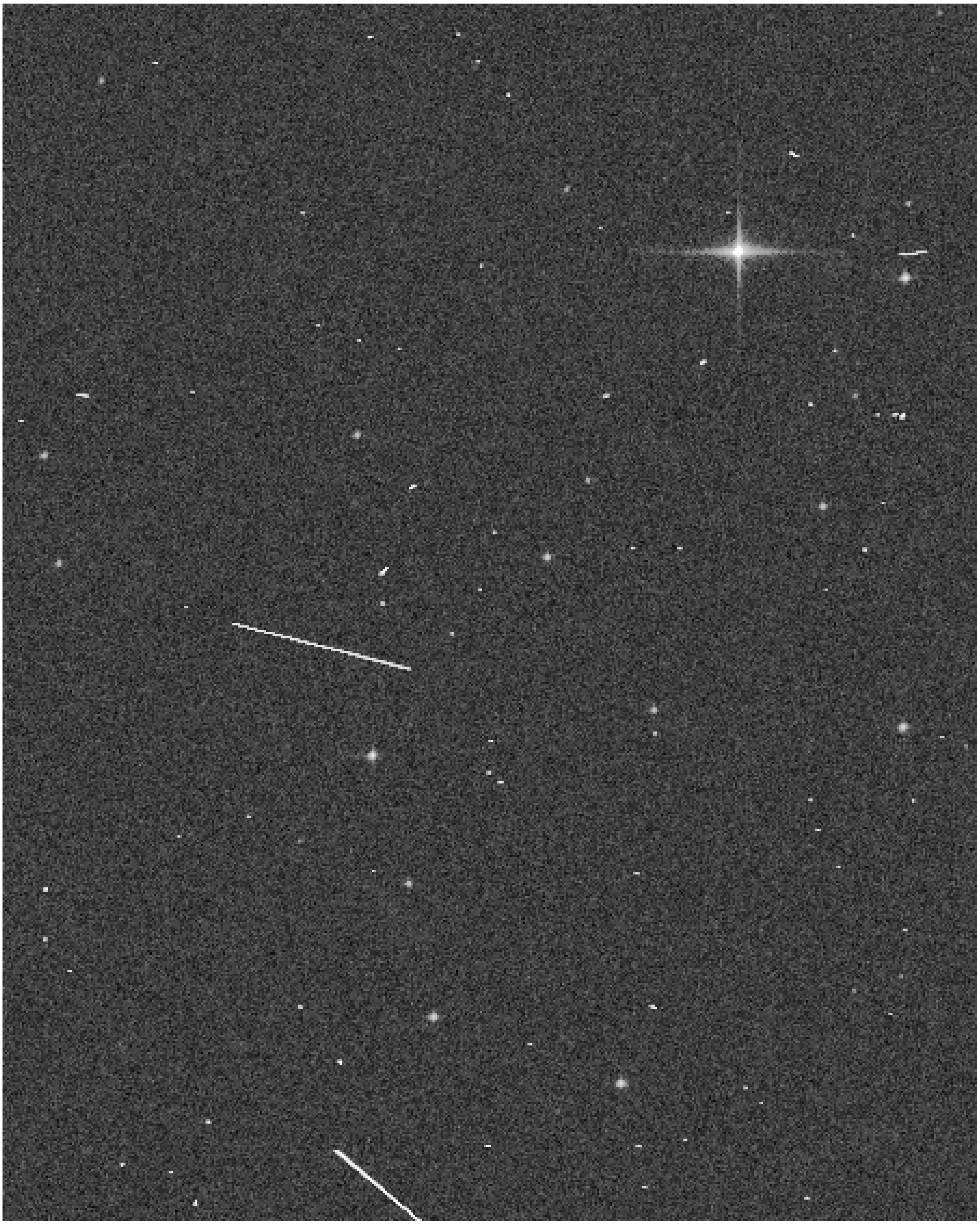


Figure C.2: The average density test case (22250 stars/deg^2) at galactic coordinates $l = 74$ and $b = 15$. The snapshot is approximately 400 AL× 525 AC SM samples.

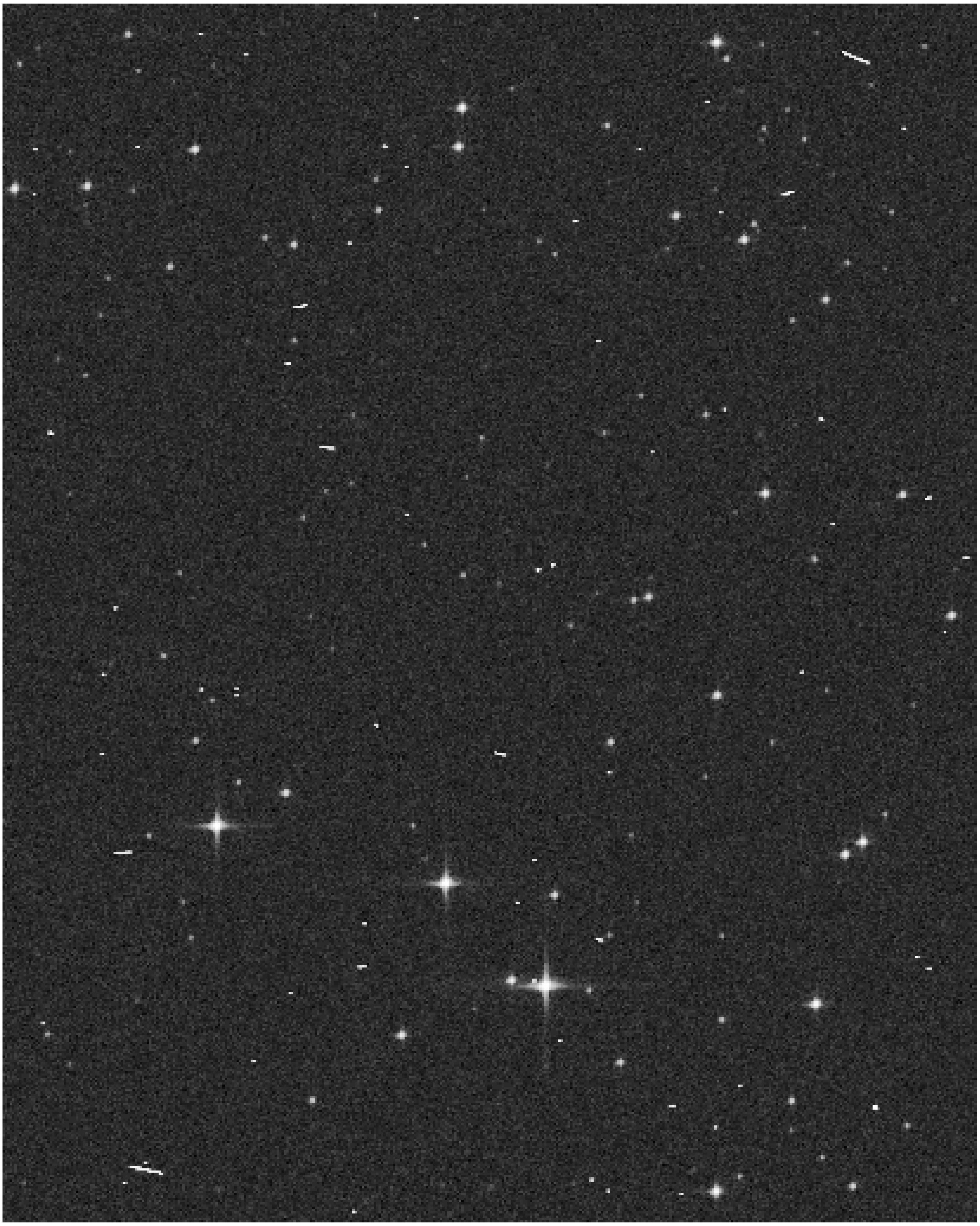


Figure C.3: The average density test case (195000 stars/deg^2) at galactic coordinates $l = 74$ and $b = 0$ (in the disk and away from the centre). The snapshot is approximately 400 AL$\times$ 525 AC SM samples.

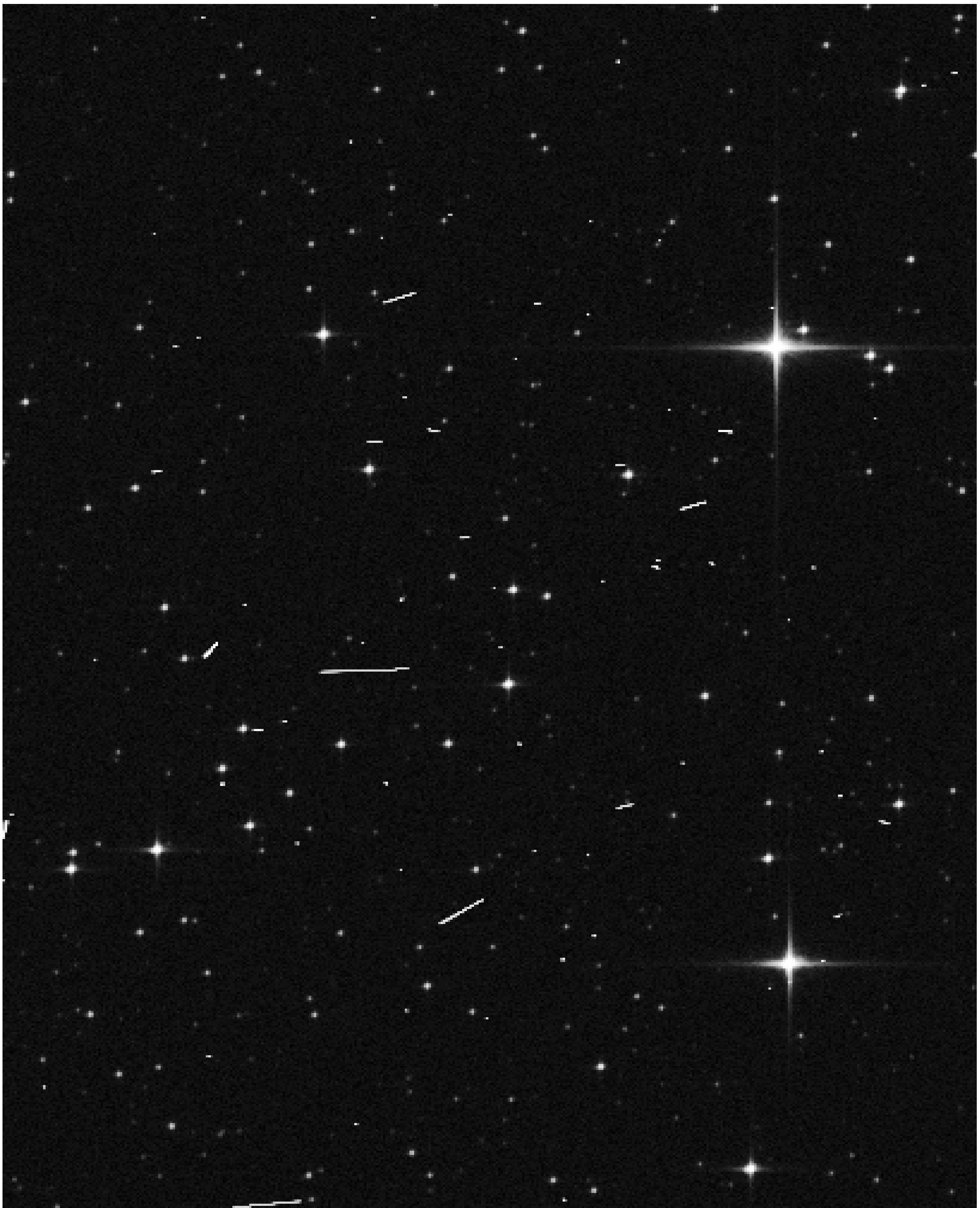


Figure C.4: The design density test case (609000 stars/deg^2) at galactic coordinates $l = 54$ and $b = 0$ (in the disk). The snapshot is approximately 400 AL× 525 AC SM samples.

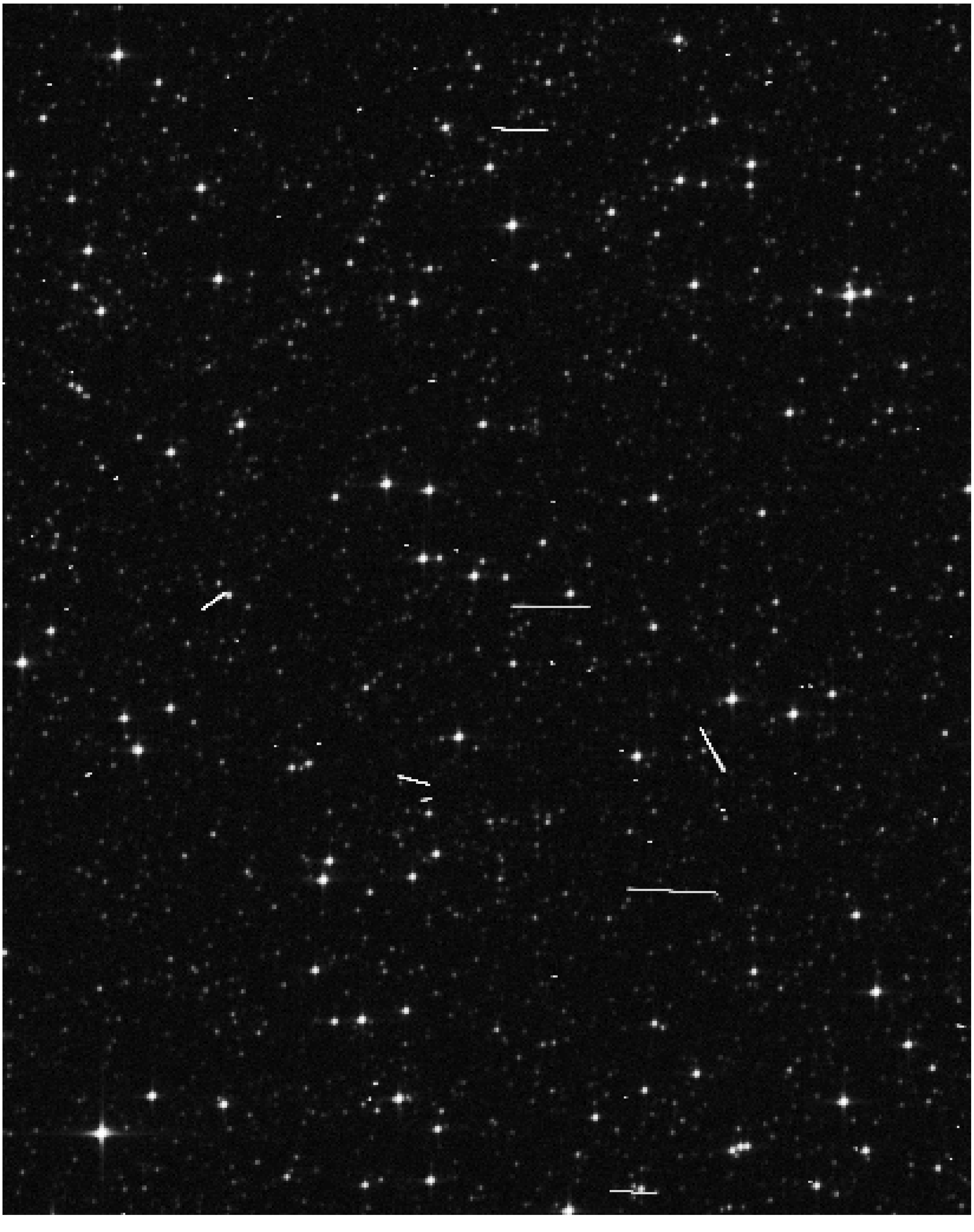


Figure C.5: The maximum density test case ($3.31.10^6$ stars/deg^2) in one of Baade's windows providing visibility through the galactic centre. The snapshot is approximately 400 AL$\times$ 525 AC SM samples.

Bibliography

Selected publications

[SM1] F. Arenou, C. Babusiaux, F. Chéreau, and S. Mignot. The gaia on-board scientific data handling. In C. Turon, K. S. O'Flaherty, and M. A. C. Perryman, editors, *Proceedings of the Symposium "The Three-Dimensional Universe with Gaia", 4-7 October 2004, Observatoire de Paris-Meudon, France*, volume ESA SP-576, page 335, 2005. http://www.rssd.esa.int/SA/GAIA/docs/Gaia_2004_Proceedings/Gaia_2004_Proceedings_335.pdf.

[SM2] S. Mignot, P. Laporte, and F. Rigaud. An efficient detection prototype based on a mixed architecture for gaia. In *Proc. EUSIPCO 2007*. Eurasip, 2007.

[SM3] S. Mignot. Sram controller. Technical Report ALGOL-4100-TN-01, Observatoire de Paris, 2007.

[SM4] S. Mignot. Pipelined implementation of sample-based processing for object detection on board gaia. Technical Report ALGOL-4100-TN-02, Observatoire de Paris, 2007.

[SM5] S. Mignot. Internal architecture of the fpga. Technical Report ALGOL-4100-DD-01, Observatoire de Paris, 2007.

[SM6] S. Mignot. Review of the "vpu algorithms requirements specifications" & "gaia video processing algorithms description". Technical Report PDH2-1500-SP-02, Observatoire de Paris, 2007.

[SM7] S. Mignot. On-board software validation philosophy. Technical Report PDH2-1100-SP-01, Observatoire de Paris, 2006.

[SM8] S. Mignot. Review of the "algorithms performances specification". Technical Report PDH2-1500-SP-001.02, Observatoire de Paris, 2006.

[SM9] S. Mignot. Explanations on the data priority scheme. Technical Report PDH2-2220-TN-01, Observatoire de Paris, 2007.

[SM10] C. Babusiaux F. Arenou and S. Mignot. Identification of test cases for the validation of the on-board algorithms. Technical Report PDH2-3110-SP-002.1 (GAIA-CH-TN-OPM-FA-048-1), Observatoire de Paris, 2007.

[SM11] S. Mignot and C. Babusiaux. Review of "fpass - tests and validation". Technical Report PDH2-3200-SP-02, Observatoire de Paris, 2007.

[SM12] S. Mignot and C. Babusiaux. Review of the "definition of reference test cases to be generated in the frame of the tda compression study". Technical Report PDH2-3110-SP-001, Observatoire de Paris, 2007.

[SM13] S. Mignot and C. Babusiaux. Review of specifications for the static fpa simulator. Technical Report PDH2-3200-SP-01, Observatoire de Paris, 2006.

[SM14] S. Mignot. Fixed point formulae for pyxis. Technical Report PDH2-1610-TN-01, Observatoire de Paris, 2006.

[SM15] S. Mignot. Component segmentation in pyxis 2.4.1. Technical Report PDH2-1610-TN-04, Observatoire de Paris, 2006.

[SM16] S. Mignot. On-board pre-calibration. Technical Report PDH2-1610-TN-05, Observatoire de Paris, 2006.

[SM17] S. Mignot. A hardware-oriented connected component labeling algorithm. Technical Report PDH2-1610-TN-06, Observatoire de Paris, 2006.

[SM18] S. Mignot. Architecture of pyxis 2.4.1 – io of main algorithms in astro. Technical Report PDH2-1610-DD-01, Observatoire de Paris, 2006.

[SM19] F. Arenou and S. Mignot. Pyxis 2.4 processing evaluation. Technical Report PDH-1100-TN-002 (GAIA-CH-TN-OPM-FA-042-1), Observatoire de Paris, 2005.

[SM20] F. Arenou, F. Chéreau, and S. Mignot. Scientific specifications for the vpu software urd. Technical Report PDH-1100-DD-002 (GAIA-CH-TN-OPM-FA-039-1), Observatoire de Paris, 2005.

[SM21] S. Mignot. Memory usage of pyxis 2.4. Technical Report PDH-2520-TN-01, Observatoire de Paris, 2005.

[SM22] S. Mignot. Contribution to the pdhs tda final presentation. Technical report, Observatoire de Paris, 2005.

[SM23] F. Arenou, C. Babusiaux, and S. Mignot. About selection and priorities on-board gaia. Technical Report OBD-CoCo-012 (GAIA-CH-TN-OPM-FA-026-1), Observatoire de Paris, 2004.

[SM24] S. Mignot. Binning strategy in asm. Technical Report OBD-SM-01, Observatoire de Paris, 2002.

References

[1] M. J. Irwin. Automatic analysis of crowded fields. *MNRAS*, 214(214):575–604, 1985.

[2] M. J. Irwin and V. Trimble. Automated star counts in the southern globular cluster ngc 6809 (m55). *Astrophysical Journal*, 89:83, 1984.

[3] E. Bertin and S. Arnouts. Sextractor: Software for source extraction. Technical report, 1996.

[4] F. Patat. A robust algorithm for sky background computation in CCD images. *Astronomy & Astrophysics*, 401:797–807, April 2003.

[5] F. Meyer. Un algorithm optimal de ligne de partage des eaux. *Proc. RFIA*, 1991.

[6] J.C. et al. Klein. Hardware implementation of the watershed zone algorithm based on a hierarchical queue structure. *IEEE. workshop on nonlinear signal and image processing*, June 1995.

[7] L Lindegren, M.A.C Perryman, U. Bastian, and et al. Gaia – global astrometric interferometer for astrophysics, proposal for a cornerstone mission concept. Technical report, Lund Observatory, 1993.

[8] L Lindegren and M.A.C Perryman. The gaia mission: internal report submitted to the esa horizon 200+ survey committee. Technical report, Lund Observatory, 1994.

[9] *The Hipparcos and Tycho Catalogues*, volume ESA SP-1200. ESA, 1997.

[10] ESA. Gaia: Composition, formation and evolution of the galaxy. concept and technology study report. Technical Report ESA-SCI(2000)4, ESA, 2000.

[11] W. J. Jin, I. Platais, and M. A. C. Perryman. *A Giant Step: from Milli- to Micro-arcsecond Astrometry.* Cambridge Journals, 2007.

[12] L Lindegren and M.A.C Perryman. A small interferometer in space for global astrometry. *IAU Symposium 166: Astronomical and astrophysical objectives of sub-milliarcsecond optical astrometry*, page 337, 1995.

[13] L Lindegren, M.A.C Perryman, and S. Loiseau. Global astrometric interferometer for astrophysics (gaia). *SPIE 2477: Spaceborne Interferometry II*, pages 91–103, 1995.

[14] Gaia Project Team. Mission requirements document. Technical Report GAIA-EST-RD-00553, ESA, 2005.

[15] M. Hechler, J. Cobos, and M. Bello-Mora. Orbits around l2 for the first, planck and gaia astronomy missions. Technical Report 1999MHNOCODE, Mission analysis section, ESOC, ESA, 1999.

[16] F. Mignard and R. Drimmel. Proposal for the gaia data processing. Technical report, Data Processing and Analysis Consortium, 2007.

[17] A. Short and et al. Gaia astrometric ccds and focal plane. *SPIE 5902: Focal Plane Arrays for Space Telescopes II*, 2005.

[18] D. Assemat, M. Avignon, and C. Correcher. *Cours de technologie spatiale - Techniques et technologies des véhicules spatiaux*. Centre National d Études Spatiales, 2006.

[19] Project Team. Detail procurement specification for the gaia flight model astro af ccds (issue 1, revision 4). Technical report, ESA, 2006.

[20] F. Chéreau. Propagaia, the on-board propagation module for gaia. Technical Report PDH2-1610-TN-02 (GAIA-CH-TN-OPM-FC-009), Observatoire de Paris, 2006.

[21] J. Chaussard. On-board selection algorithm for gaia. Technical Report OBD-JC-001, Observatoire de Paris, 2003.

[22] ECSS. Space engineering – space environment. Technical Report ECSS-E-10-04A, ESA, 2000.

[23] ECSS. Space product assurance – electrical, electronic and electromechanical components. Technical Report ECSS-Q-60B, ESA, 2007.

[24] ECSS. Space product assurance – procurement of printed circuit boards. Technical Report ECSS-Q-70-11A, ESA, 2001.

[25] EADS Astrium GmbH PDHE Team and Laben. Gaia pdhs video processing unit (final tda presentation). Technical report, EADS Astrium GmbH, 2005.

[26] ECSS. Space engineering - software. Technical Report ECSS-E-40A, European Cooperation for Space Standardization, 1999.

[27] P. Ambruster and W. Gasti. On-board payload data processing systems – on-board networks for future missions. In *Proc. EUSIPCO 2002*, pages 577–580. Eurasip, 2002.

[28] G. Aranci, L. Maltecca, and R. Ranieri. The Onboard Data Handling Subsystem for XMM/Integral Space Missions. In T.-D. Guyenne, editor, *Data Systems in Aerospace - DASIA 97*, volume 409 of *ESA Special Publication*, pages 265–+, August 1997.

[29] S Kraft and et al. On the concept of a highly integrated payload suite for use in future planetary missions: the example of the bepicolombo mercury planetary orbiter. *Acta astronautica*, Vol. 59:823–833, 2006.

[30] F. Rombeck and T. Fichna. Spaceborne mass memory systems provided by astrium gmbh - esa workshop on spacecraft data. Technical report, Astrium GmbH, 2003.

[31] ECSS. Space product assurance – materials, mechanical parts and processes. Technical Report ECSS-Q-70B, ESA, 2004.

[32] PDHE team. Gaia pdhe validation report. Technical Report GAIA.ASG.VR.0001, EADS Astrium GmbH, 2005.

[33] P. Laporte. Pyxis transfert towards an fpga algorithm (issue 2). Technical Report ALGOL-2100-DD-01, Observatoire de Paris, 2007.

[34] C. Babusiaux. The gaia instrument and basic image simulator. In C. Turon, K.S. O'Flaherty, and M.A.C. Perryman, editors, *Proceedings of the Symposium "The Three-Dimensional Universe with Gaia", 4-7 October 2004, Observatoire de Paris-Meudon, France*, pages 409–412. ESA SP-576, 2005.

[35] E. Oseret. évaluation de méthodes de traitement du signal pour la détection dans le projet gaia - rapport de stage de dea. Technical report, PRiSM, Université de Versailles Saint Quentin, 2004.

[36] F Rué and A. Bijaoui. A multiscale vision model adapted to the astronomical images. *Signal Processing*, 46:345 – 362, 1995.

[37] C. Babusiaux. Swa - sliding window algorithm - v1.0 users guide. Technical report, Observatoire de Paris, 1999.

[38] A. Bijaoui. Sky background estimation and application. *Astronomy and Astrophysics*, 84:81–84, 1980.

[39] A.N Cox and et al. *Allen's Astrophysical Quantities - fourth edition.* Springer-Verlag, 2000.

[40] A. C. Robin, C. Reylée, S. Picaud, and B. Debray. A Galactic Model as a Useful Tool for Virtual Observatories. In P. J. Quinn and K. M. Górski, editors, *Toward an International Virtual Observatory*, pages 309–+, 2004.

[41] M. B. M. B. Dillencourt, H. Samet, and M. Tamminen. A general approach to connected component labeling for arbitrary image representations. *Journal of ACM*, April 1992.

[42] EADS Astrium GmbH PDHE Team. Vpu detailed specification. Technical Report GP-ASG-RS-0004, EADS-Astrium GmbH, 2003.

[43] L. Vincent. Algorithmes morphologiques à base de files d'attentes et de lacets, extension aux graphs. *PhD thesis (ENSMP)*, 1990.

[44] S. Beucher and C. Lantuejoul. Use of watershed in contour detection. *Proc. International Workshop on Image Processing*, Sept 1979.

[45] U. Bastian. Reference systems, conventions and notations for gaia. Technical Report GAIA-ARI-BAS-003, ARI Heidelberg, 2003.

[46] G. Flandin. Algorithms performances specification. Technical Report GAIA.ASF.SP.PLM.00064, EADS-Astrium, 2006.

[47] S. Provost. Gaia video processing algorithms (vpa) validation test plan. Technical Report GAIA.ASF.TCN.PLM.00136, EADS-Astrium, 2007.

[48] Office of Logic Design. Space vehicle design criteria, spaceborne digital computer systems. Technical Report NASA-SP-8070, NASA, 1971.

[49] Y. Meyer. *Électronique numérique.* École d'Ingénieurs de l'Arc Jurassien, 2003.

[50] R. Ramin. A comparison of radiation-hard and radiation-tolerant fpgas for space applications. Technical report, NASA, 2004.

[51] ECSS. Space product assurance – asic and fpga development. Technical Report ECSS-Q-60-02A, ESA, 2007.

[52] P. Sinander. Vhdl modelling guidelines. Technical Report ASIC/001, ESA (onboard data division), 1994.

[53] Actel. Prototyping for rtax-s and rtax-sl (application note). Technical report, Actel, 2007.

[54] P. Chambers. The ten commandments of excellent design. Technical report, VLSI technology, 1997.

[55] C.E. Cummings, D. Mills, and S. Golson. Asynchronous and synchronous reset design techniques. *Synopsys User Group (SNUG)*, Boston 2003.

[56] Whiteley S.R. Resolution of differences between x86 linux/glibc floating-point to integer conversion behavior relative to other platforms (http://www.wrcad.com/linux_numerics.txt). Technical report, Whiteley Research Incorporated, 2001.

[57] Standards Committee of the IEEE Computer Society. Ieee standard for binary floating-point arithmetic. Technical Report 754-1985, IEEE, 1985.

[58] D.E. Knuth. *The art of computer programming (Seminumerical algorithms).* Addison-Welsey Publishing Company, 1981.

[59] Actel. Rtax-s/sl radtolerant fpgas (datasheet). Technical report, Actel, 2007.

[60] Actel. Proasic3e flash family fpgas (datasheet). Technical report, Actel, 2007.

[61] Actel. Using edac ram for radtolerant rtax-s fpgas and axcelerator fpgas (application note). Technical report, Actel, 2006.

[62] S.F. Oberman and M.J. Flynn. Division algorithms and implementations. *IEEE transactions on computers*, Vol. 46:833–854, 1997.

[63] N. Boullis and A. Tisserand. On digit-recurrence division algorithms for self-times circuits – rapport de recherche. Technical report, INRIA, 2001.

[64] R. Michard, A. Tisserand, and N. Veyrat-Charvillon. divgen user's and reference manual. Technical report, Arénaire project – LIP, CNRS, ENSL, INRIA, UCBL, 2005.

[65] P. Armbruster. Payload data handling system and breadboarding. Technical report, ESTEC, 2003.

[66] F. Rigaud. Gaia - description de la plateforme de simulation. Technical report, Observatoire de Paris, 2007.

[67] Actel. Interfacing proasicplus fpgas with 5v input signals (application note). Technical report, 2004.

[68] Actel. Board-level considerations (application note). Technical report, 2006.

[69] Actel. Proasic3/e sso and pin placement guidelines. Technical report, 2008.

[70] Actel. Simultaneous switching noise and signal integrity (application note). Technical report, 2006.

[71] Actel. Using the bufd and invd delay macros (technical brief). Technical report, 2001.

[72] C. Babusiaux and Arenou. F. The baade window data set. Technical Report PDH-4310-TN-01, Observatoire de Paris, 2005.

Résumé

La mission Pierre Angulaire de l'ESA, Gaia, dont le lancement est prévu en décembre 2011 et qui est présentement en phase de conception détaillée sous la direction de EADS Astrium SAS, doit répondre au défi de recenser les étoiles sur la voûte céleste. Limité seulement par leurs magnitudes[1], ce catalogue contiendra approximativement un milliard d'étoiles, approximativement un pourcent de la Voie Lactée. Dans la lignée du satellite HiPParCoS, Gaia va balayer continûment le ciel grâce à la combinaison d'une rotation sur lui-même et de la précession de cet axe, pour collecter des données dans deux champs de vue distants. Combinées sur un même plan focal, ces observations doivent permettre d'élucider la composition, la formation et l'évolution de la galaxie à travers une astrométrie absolue et la caractérisation astrophysique des étoiles. À la différence de son prédécesseur, Gaia se trouvera en orbite autour du second point de Lagrange du système Terre-Soleil, résultat d'un compromis entre stabilité thermique et dynamique d'une part et la bande passante pour la transmission des données vers la Terre d'autre part. De même, les photo-multiplicateurs se trouvent remplacés par 106 CCDs de 4500×1966 pixels fonctionnant en mode TDI. Ce mode décale les photo-électrons au sein de la matrice du CCD de façon synchrone avec le déplacement des sources sur le plan focal (suite à la rotation du satellite) et produit des images continues s'étalant sur les 5 ans de la mission au rythme de plusieurs gigabits par seconde.

L'impossibilité de télécharger un tel volume de données appelle des traitements élaborés à bord pour effectuer une sélection sur le contenu et mettre un oeuvre un schéma complet de gestion et de compression des données tel que décrit dans [65]. Cet effort commence au niveau de l'acquisition elle même en ne lisant que partiellement les CCDs autour des étoiles situées dans la gamme de magnitude d'intérêt ($[6; 20]$) dans un ciel autrement essentiellement vide. Comme celui-ci n'est qu'imparfaitement connu au niveau de sensibilité et de résolution de Gaia, l'utilisation d'un catalogue, comme pour HiPParCoS, est impossible et doit être remplacée par une détection autonome afin de produire un relevé non biaisé. Les sources sont donc détectées lorsqu'elles traversent les repéreurs d'étoiles (SM 1 et 2 pour les deux champs de vue, voir Fig. C.6). Afin de discriminer les rayons cosmiques et d'assurer l'asservissement de l'attitude, elles sont confirmées lors de leur traversée du premier CCD dédié à l'astrométrie (AF1) avant d'être suivies dans les suivants. Finalement, l'information collectée est stockée puis transmise vers le sol par ordre de priorité pour garantir que le comportement du système ne se dégrade qu'harmonieusement lorsque les ressources à bord s'avèrent insuffisantes.

[1]La magnitude est liée au flux par: $m = m_0 - 2.5\log(f)$ et décroît donc lorsque le flux augmente.

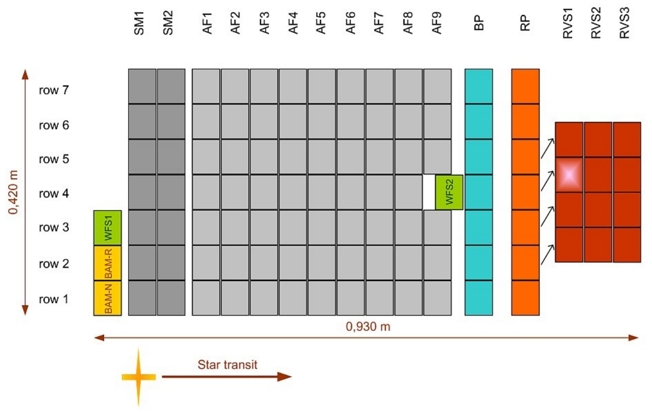


Figure C.6: Le plan focal de Gaia et ses 106 CCDs.

Durant la phase A du projet, l'aspect critique de ces opérations quant aux objectifs de la mission a conduit l'un des groupes de travail mis en place par l'ESA à prototyper la détection des sources pour évaluer le niveau de performance scientifique accessible et fournir une base pour la rédaction des spécifications. Ce modèle a ensuite évolué pour couvrir l'ensemble des opérations attenantes à l'acquisition (Pyxis) et a servi de base pour une étude conduite par Astrium GmbH sur la faisabilité et le dimensionnement de l'électronique à bord. Celle-ci, grâce à un démonstrateur logiciel temps-réel s'exécutant sous VxWorks sur une carte Maxwell (correspondant aux performances maximales attendues pour du matériel qualifié spatial), a montré qu'une architecture mixte s'appuyant sur du logiciel et de l'électronique dédiée serait nécessaire pour atteindre les performances visées [32]. Notre schéma de détection a aussi été l'objet d'une implémentation matérielle simplifiée par Astrium lors de leur réponse à l'appel d'offre émis par l'ESA pour Gaia. Ces deux résultats ayant validé la faisabilité de l'approche proposée, nous avons entrepris la conception d'un démonstrateur complet dans le cadre d'une ACI "Masse de données" (ALGOL).

Après une présentation des contraintes qui s'appliquent à bord et leur prise en compte au cœur même de la conception, nous décrirons la structure globale des traitements et les algorithmes employés en mettant l'accent sur les problèmes d'implémentation avant de conclure par une brève évaluation des performances.

Conception

Les spécifications scientifiques sont naturellement au coeur du processus de conception. Bien qu'elles puissent se résumer au simple besoin de détecter toutes les sources et de les mesurer en termes de magnitude et de position avec un minimum de fausses détections, cette formulation dissimulerait le fait qu'elles peuvent différer d'un rapport 400 000 en flux et qu'il est nécessaire d'être robuste aux effets liés à:

- leur nature physique (principalement couleur et extension angulaire),
- leur environnement (étoiles proches, fond de ciel et densités variables avec des pics à 120 fois la moyenne),
- la variété des conditions d'observation (radiations, vieillissement des détecteurs, attitude du satellite),
- sans parler des évolutions du concept de l'instrument pendant les premières phases du projet.

Pour l'ensemble de ces raisons, les méthodes d'analyse locale ou s'appuyant sur la reconnaissance d'un motif ont été écartées au regard de la difficulté de mettre en place un système homogène, et donc de valider la multiplicité des cas particuliers devant nécessairement survenir. À la place, s'inspirant de la séquence proposée par M. Irwin [1] et créditée d'une application massive à l'analyse de plaques photographiques, nous avons préféré une approche générique s'appuyant sur un étage de segmentation non paramétrique pour former, dans un premier temps, des objets au contenu suffisamment riche pour, dans un second temps, les caractériser plus finement avant de les sélectionner.

La détection réalise une compression des données en substituant une description abstraite à la liste exhaustive des pixels et réduit, de ce fait, le flux de données. Il est cependant nécessaire, à cette fin, de lire intégralement les CCDs SM et donc, pour la détection, de tenir les 3,8 MBytes/s de données (correspondant à 1966 pixels codés sur 16 bits lus à la cadence de 1.018 kHz). Ce flux de données et le besoin d'accéder aux pixels d'une manière cohérente avec la connectivité choisie suggère de s'appuyer sur une implémentation matérielle pour accéder à la mémoire de façon parfaitement systématique et éviter de dépendre du fonctionnement des caches qui dégrade sévèrement les performances d'une implémentation logicielle dans notre cas [32]. Inversement, bénéficiant de la réduction du volume de données obtenue en ne conservant que quelques pour cents des pixels sous les pires conditions de densité et prenant en compte la variabilité intrinsèque des opérations de caractérisation des objets, une approche logicielle est préférable à ce second niveau.

Une telle partition attribue les opérations pixeliques au matériel et celles relevant des objets au logiciel en accord avec l'importance du flux de données et la diversité des opérations à conduire. Elle offre une opportunité supplémentaire d'optimisation, en le lisant pas les CCD SM pixel par pixel mais plutôt par bloc de 2×2 (échantillons dans la suite). Ce compromis entre la résolution angulaire et le niveau de signal permet d'améliorer la complétude des résultats et de diviser le flux de donnés. SM1 et SM2 peuvent alors être lus alternativement ce qui permet de les traiter avec les mêmes ressources matérielle et logicielle.

Les transferts depuis les CCDs étant régis par la seule période TDI, un dispositif temps-réel "dur" permet de s'affranchir du besoin d'un buffer intermédiaire dont la taille irait rapidement croissante avec la latence. En sortie, par contre, le flux est naturellement dépendant de la survenue d'objets et n'est soumis à une contrainte "dure" qu'au moment de la configuration ou de la lecture du CCD AF1. Bien que de la marge doive être conservée pour les autres tâches concourant à cette opération, le délai lié à la traversée SM1-AF1 permet de ne mettre en œuvre qu'une approche temps-réel "molle" à ce niveau.

Plate-forme

L'équilibre obtenu se situe entre volume et complexité. Le matériel a la responsabilité de la gestion de beaucoup de données mais n'effectue dessus que des traitements systématiques et bas niveau dans le but d'identifier la petite fraction de données utiles sur laquelle le logiciel effectue ensuite des analyses plus sophistiquées et moins prévisibles. L'électronique dédiée est parfaitement adapté à ce titre puisqu'elle permet de manipuler les données ligne à ligne, voire échantillon par échantillon selon un schéma prédéterminé. Suite au besoin de partager le temps processeur avec d'autres tâches et de préserver des marges très significatives, la partie logicielle requiert de hautes performances [32] de sorte que la carte Maxwell SCS740A reste la seule option envisageable aujourd'hui.

Au delà des contraintes qui touchent au lancement et au coût, l'électronique en général doit supporter la transition vers le vide spatial et les radiations. En orbite, ces dernières consistent en un régime permanent qui se superpose à des particules de très hautes énergies. Les systèmes de traitement de l'information doivent donc s'appuyer sur des équipements spécifiquement conçus pour ces conditions de fonctionnement, à travers un blindage (composants anti-fusibles qui ne sont pas reprogrammables et atteignent des densités inférieures à ce qui existe au sol), des méthodes de détection et de correction d'erreurs (EDAC) et de la redondance assortie d'une logique de vote. Le raisonnement suivi consiste à dire que si, suite à ces particules, des changements impromptus d'état ne peuvent être évités, ils restent localisés et peu fréquents. Les stratégies précédentes tentent de rétablir un état valide par le biais d'une sur-information, au niveau du codage, au niveau de la porte logique ou encore du sous-système. D'un point de vue logique, et malgré ces protections, le système doit rester capable de quitter des états théoriquement impossibles mais potentiellement accessibles pour revenir à un fonctionnement nominal, par exemple pour les machines à états finis.

Partant de cette analyse, une plate-forme de test a été conçue en équilibrant l'objectif de représentativité avec

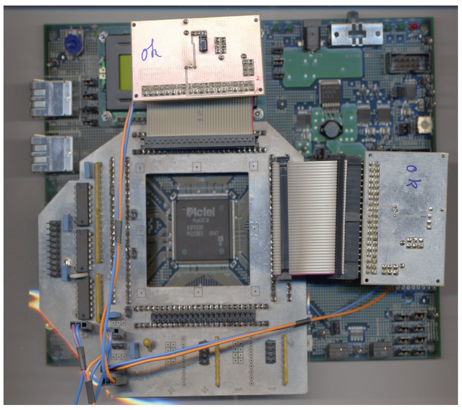


Figure C.7: La plate-forme matérielle à base de FPGA avec ses deux SRAMs externes.

les besoins liés au développement. Avec pour point de mire la technologie anti-fusible RTAX robuste aux radiations d'Actel, un FPGA à base de mémoire Flash a été choisi chez le même fabricant pour être reprogrammable tout en assurant la portabilité des développements grâce à l'utilisation du même environnement de développement. Pour des raisons de simplicité, un "kit de démarrage" équipé d'un ProASIC3E 600 a été choisi. Deux RAMs statiques, pour éviter d'avoir à les rafraîchir périodiquement, et de hautes performances d'un megaoctet (IS61LV51216 de chez ISSI), montées sur des cartes secondaires afin d'offrir la possibilité de modifier l'affectation des broches, lui ont été adjointes pour permettre des accès simultanés à cette ressource externe . Bien que plus rapides que leurs équivalents pour l'espace, ces composants asynchrones ne sont employés qu'à des fréquences de fonctionnement réalistes par rapport à un système spatial. Enfin, les interfaces externes s'appuient sur des entrées et des sorties via des canaux de 16 bits de large réalisant plusieurs transactions "handshake" par cycle pour atteindre 80 Mbit/s vers un PC qui simule le buffer vidéo en entrée et collecte les résultats des traitements au niveau échantillon en sortie.

Séquence des traitements

Une vue générale de la séquence est illustrée par la figure C.8. Chacune des étapes est décrite dans cette section mais à des fins de brièveté et pour mieux rendre la diversité de la conception et de l'implémentation, différents point de vue sont adoptés à chaque fois au prix de l'exhaustivité. D'un point de vue global, avec une période TDI qui dure 0.9828 ms et 983 échantillons à traiter par cycle, les traiter en série ne permettrait que d'allouer une maigre microseconde à chacun. Avec le besoin de recourir à des accès mémoire externes, ceci n'est guère réaliste et, à la place, les tâches de la figure C.8 sont implémentées sur la forme d'un pipeline dont chaque étape est elle-même un pipeline pour satisfaire la contrainte temps-réel "dure" en entrée et exploiter la latence disponible en sortie. Inversement, les opérations qui n'ont pas besoin d'être exécutées pour chaque échantillon sont commandées par un

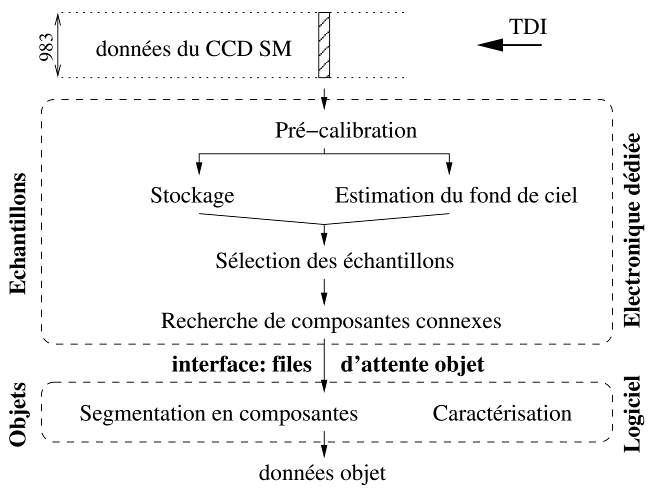

983
données du CCD SM
TDI
Pré−calibration
Stockage
Estimation du fond de ciel
Sélection des échantillons
Recherche de composantes connexes
Echantillons
Electronique dédiée
interface: files d'attente objet
Segmentation en composantes
Caractérisation
Objets
Logiciel
données objet

Figure C.8: Représentation haut niveau de la séquence des traitements.

chef d'orchestre et, au besoin, différées au cycle suivant (pendant que l'autre CCD SM est en train d'être lu).

Trois horloges différentes sont introduites en support à ce schéma, une qui correspond à l'arrivée d'un nouvel échantillon (DCLK), une introduite pour gérer les délais introduits par les temps de réponse des SRAMs (SCLK) et enfin, une horloge principale (CLK) qui fixe la cadence des opérations synchrones les plus rapides dans le FPGA et à partir de laquelle les autres sont obtenues par division. La fréquence d'arrivée des échantillons étant de l'ordre de 1 MHz et un maximum de 14 cycles SCLK étant requis par échantillon pour des accès en lecture sur la mémoire externe, les trois périodes retenues sont respectivement 8 ns, 32 ns et 992 ns pour CLK, SCLK et DCLK.

Calibration

Malgré une spécification de très haute qualité pour les CCDs, le faible rendement de production et le nombre élevé de composants requis impliquent que des défauts seront présents non seulement par suite du vieillissement ou des radiations mais aussi directement après approvisionnement. En conséquence, un premier étage de calibration a été introduit pour répondre aux besoins de:

- corriger les défauts locaux pour satisfaire les hypothèses faites par les algorithmes sur les propriétés images (pixels chauds ou morts),
- normaliser le niveau de réponse au sein et entre les CCDs pour permettre une sélection non biaisée des objets (variations de sensibilité (PRNU), biais (DSNU), éclairement non uniforme),
- compenser ou régulariser les effets dûs au vieillissement pour assurer une dégradation graduelle des performances au fil du temps.

Une transformation linéaire qui généralise les méthodes de calibration usuelles au sol est appliquée à chaque échantillon. Ses coefficients sont déterminés au sol puis mis à jour périodiquement. Puisque les valeurs des échantillons sont discrètes (en ADUs), cette correction se borne à restaurer la valeur qui correspondrait à un CCD parfait de mêmes caractéristiques. Comme les FPGAs susceptibles d'être employés dans l'espace ne disposent pas de module de calcul en virgule flottante, cette opération est réalisée en virgule fixe. Réalisant un compromis entre produire des valeurs erronées et les besoins de stockage pour les coefficients, la formule suivante est employée:

$$s_i^c = s_i^r + ((a_i \times s_i^r + (b_i \ll 16) \pm 2^{17}) \gg 18) \qquad \text{(C.1)}$$

où $\ll$ et $\gg$ sont des opérateurs de décalage de bits, a_i et b_i sont les coefficients correctifs et s_i^r la valeur lue sur le CCD à la position i. Cette formule calcule la correction, l'arrondit à l'ADU le plus proche et, au prix d'un peu plus d'arithmétique, permet le stockage des coefficients sur 16 bits seulement (a_i au format[2] 1.0.18 si le PRNU n'excède pas ±6.25% et b_i en 1.13.2). Les 983 positions possibles pour les échantillons pour les CCDs SM1 et SM2 conduisent à un total de 7864 octets qu'il est encore possible de stocker dans la mémoire interne du FPGA. L'implémentation en pipeline visible figure C.9 impose, en plus, la saturation à une valeur fixe et effectue le remplacement des échantillons défectueux, suivant la valeur de b_i, avec la moyenne des deux voisins, ou en propageant l'une de leurs valeurs ou encore avec une constante prédéterminée représentant le niveau de ciel le plus probable (d'où la présence de deux registres à décalage contenant les valeurs des échantillons et les b_i).

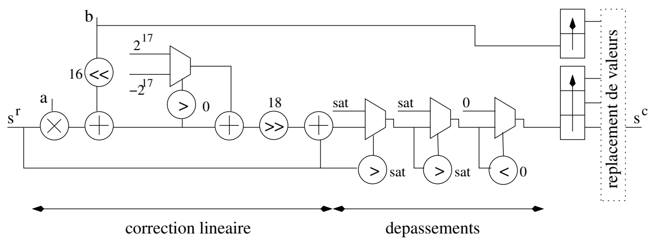


Figure C.9: Pipeline de calibration.

Fond de ciel

Mesurer le fond de ciel localement permet d'appliquer ensuite rigoureusement les différents critères quantitatifs puisque les quantités mesurées peuvent être rendues propres aux étoiles suite à la correction de la contribution de leur environnement. Si ceci est un pré-requis pour produire des mesures non biaisées, en particulier celles relatives à la magnitude pour la sélection des objets, cela permet aussi d'éviter que la segmentation de l'image en objets ne produise de fausses détections dans les régions de formation d'étoiles, dans les nébuleuses ou par suite de la lumière zodiacale ou du fond diffus galactique.

[2]Ce format décrit l'allocation des bits: signe.partie entière.partie décimale.

Suivant l'exemple de M. Irwin, le fond de ciel est évalué à une échelle régionale sur des régions composées de 32×32 échantillons qui représentent un compromis entre le bruit, la pollution par les étoiles, la qualité de la statistique et le délai introduit avant que les échantillons d'intérêt puissent être sélectionnés. Pour des raisons à la fois de robustesse et de simplicité calculatoire, le mode de la distribution est déterminé sur chacune de ces régions et un estimateur de variance minimale est obtenu en considérant des bins de 4 ADUs de large qui équilibrent le bruit statistique et la résolution en valeur. Par conséquent, 4 histogrammes tronqués au domaine dans lequel les fluctuations de fond sont attendues sont construits en parallèle (dans la mémoire externe) et 4 valeurs du mode sont obtenues par l'ajustement d'un profil parabolique [4] avant d'être combinées. De plus, les valeurs obtenues pour les régions dominées par les objets sont replacées par celles de régions voisines suite à un test sur la population du bin correspondant au mode d'une manière analogue au mécanisme mis en place pour le replacement des échantillons défectueux lors de la calibration. Pour un cas correspondant une densité maximale d'étoiles sur un fond uniforme, la valeur mesurée surestime le fond réel de $\sim$ 0.4 ADUs, typiquement à cause de l'asymétrie induite par le nombre d'objets faibles, mais demeure précise avec une dispersion inférieure à 0.2 ADUs.

Les valeurs régionales sont ensuite interpolées pour produire une carte linéaire par morceau couvrant l'intégralité du CCD (figure C.10) qui contient principalement les basses fréquences et avec une latence de 48×2 TDIs considérablement réduite par rapport à un filtre passe-bas de performance comparable. Afin de répartir au mieux la charge de calcul et de réduire les besoins de stockage, cette dernière opération n'est effectuée que juste à temps pour le test de sélection des échantillons. Grâce au fait d'avoir choisi des régions dont les dimensions sont des puissances de deux, l'interpolation est simplement la somme de quatre termes pondérés par des entiers codés sur 5 bits.

Sélection des échantillons

S'appuyant sur la théorie de l'information, un simple seuil sur le SNR est adopté pour identifier les échantillons porteurs d'information et réduire le flux de données. La contribution du fond de ciel est soustraite du signal et le bruit est considéré comme résultant de deux processus indépendants: le bruit de photon et le bruit électronique (calibré au sol et incluant la chaîne de lecture et la conversion analogue-digital) modélisés respectivement comme des variables de Poisson et de Gauss (centrée de variance σ_{RON}^2). Avec des valeurs exprimées en électrons, le test serait:

$$\frac{|s_i^c - bkgd|}{\sqrt{s_i^c + \sigma_{RON}^2}} >^? SNR \qquad \text{(C.2)}$$

mais pour plus de simplicité il est préférable d'évaluer plutôt:

$$\underbrace{(s_i^c - bkgd)^2}_{0.14.18} >^? \underbrace{SNR^2}_{0.14.18} \times (\underbrace{s_i^c}_{0.16.0} + \underbrace{\sigma_{RON}^2}_{0.14.18}). \qquad \text{(C.3)}$$

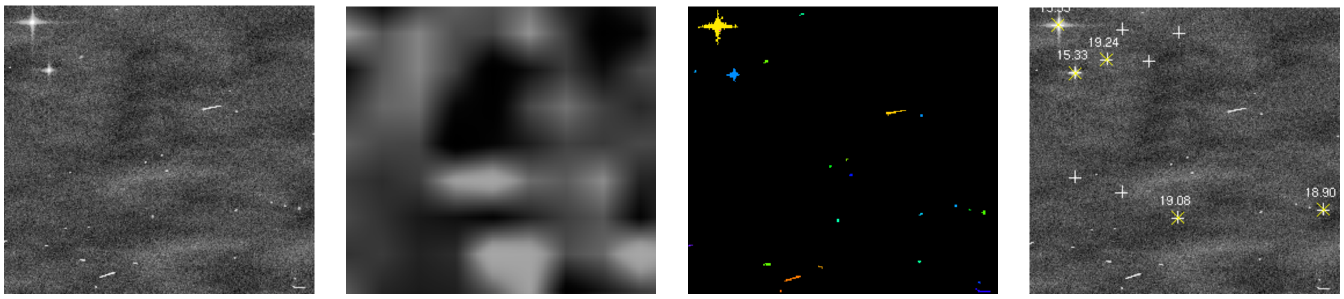


Figure C.10: Les différentes phases de détection dans un champ synthétique avec un fond de ciel présentant des textures (basé sur la nébuleuse d'Orion) et des étoiles faibles: image brute, carte du fond (avec un contraste augmenté), CCs et résultats (catalogue d'entrée avec des "+" blanches, objets détectés avec des "x" jaunes). Tous les objets d'intérêt sont détectés à l'exclusion des rayons cosmiques et des objets plus faibles que la magnitude limite.

Bien que significativement simplifié par l'élimination du quotient et du radical sous cette forme mise au carré, le test peut être encore optimisé en notant qu'un calcul détaillé n'est nécessaire qu'au voisinage du seuil. À la suite d'une analyse portant les propriétés image et les paramètres d'encodage, pour des SNR dans l'intervalle $[0; 6]$, la procédure suivant s'avère préférable:

1. calculer $s_i^c - bkgd$ au format 1.22.9,
2. tester le signe: si négatif rejeter l'échantillon,
3. sinon, tester la partie entière: si $\geq$ 128 ADUs conserver l'échantillon,
4. sinon, effectuer le calcul complet.

Avec un nombre de bits limité à 32 pour réduire le coût en ressources matérielles et pour assurer la compatibilité avec l'implémentation logicielle, cette stratégie permet de faire le meilleur usage possible de la précision disponible. L'erreur de calcul sur (C.3) est majorée par 0.133 ADU^2 et ne conduit à retenir à tort que quelques échantillons par million sans impact notable sur les performances.

Identification des composantes connexes

Si les étapes précédentes permettent la réduction du volume de données, ces dernières ne sont pas encore structurées à ce niveau des traitements. L'étiquetage des composantes connexes, en tant que méthode générant des paquets de données indépendants par croissance de région, assure la transition entre les traitements au niveau des échantillons et ceux applicables aux objets.

Les méthodes naturelles propagent géodésiquement les étiquettes jusqu'à ce que tous les échantillons des classes d'équivalence liées à la relation de connexité aient été atteints. Dans notre cas, cependant, le besoin de fonctionner de façon synchrone avec le TDI impose des contraintes d'adressage qui appellent plutôt une approche s'appuyant sur une succession d'étiquetage-fusion-ré-étiquetage [41]. Avec l'ordre induit sur les échantillons, une CC de géométrie complexe peut fort bien commencer par deux échantillons non contigus. Comme à ce moment précis il n'est pas possible de déterminer s'ils sont connexes par arc parce que l'arc peut fort bien se situer dans une portion de l'image non encore lue, des étiquettes différentes leurs sont nécessairement associées (étiquetage). Ensuite, lorsque l'horizon progresse et que le chemin qui les relie devient apparent, les deux parties de la même CC peuvent être fusionnées (fusion) et les deux étiquettes déclarées équivalentes (ré-étiquetage).

Diverses optimisations sont possibles afin de recycler les étiquettes rapidement mais, dans notre cas, la difficulté réside principalement dans leur gestion et de celle des structures de données par l'électronique. Pour simplifier celle-ci au maximum, un ensemble de tables sont allouées dans la mémoire interne ou externe suivant leurs tailles. Pour obtenir une complexité fixe, les étiquettes sont organisées en hiérarchies qui favorise la largeur par rapport à la profondeur de façon à accéder à leurs racines respectives par un nombre minimum d'opérations. Ceci permet de maintenir simplement la structure arborescente suite à la fusion de deux CCs.

Files d'objets

Les objets formés lors de l'étape précédente sont ensuite insérés dans un certain nombre de files d'attentes en fonction de l'information disponible sur leur magnitude. Implémentées sous forme d'une mémoire partagée entre le FPGA et le processeur, ces files d'attentes servent d'interface, temporairement d'espace de stockage, jusqu'à ce que les objets puissent être caractérisés et permettent à cette dernière opération d'être conduite par ordre de priorité. Les traitements logiciels peuvent alors extraire les objets par ordre croissant de magnitude, de sorte que, si les ressources de calcul viennent a manquer momentanément, les objets les plus brillants ont précédence tandis que les autres demeurent en attente soit d'être traités si leur latence maximale n'a pas déjà été atteinte, soit d'être effacés après notification au superviseur (PDHU).

Segmentation en composantes

Les données transférées à la partie logicielle résultent d'une segmentation conçue pour ne dépendre que d'un nombre minimum d'hypothèses, par soucis de simplicité et de robustesse quant à la conception de l'électronique. Elles représentent des ensembles qui contiennent toute l'information nécessaire pour effectuer des mesures sur les objets et ensuite décider s'ils doivent être observés et comment, mais elles n'indiquent en rien combien de sources sont sous-jacentes. Si ceci n'est pas un problème pour une majorité de CCs faibles pour lesquelles la magnitude peut bien se satisfaire d'être simplement globale (puisqu'elles sont convenablement observées par l'intermédiaire d'une unique fenêtre dans les CCDs à suivre), ce n'est pas le cas pour celles plus brillantes qui requièrent qu'une fenêtre soit dédiée à chacune des composantes.

Un modèle plus fin doit être introduit pour identifier ces composantes. Les méthodes qui tentent un dé-mélange de ces dernières demeurent trop gourmandes pour notre application mais suggèrent de simplifier cette décomposition à travers une approximation qui assigne simplement chaque échantillon à son principal contributeur. Suite à la concentration de l'énergie dans le lobe principale de la figure de diffraction, ceci conduit à partitionner les CCs en domaines connexes suivant les maxima locaux et des frontières qui s'appuient sur les minima d'énergie.

Avec quelques adaptations mineures, la transformée par ligne de partage de eaux, avec des marqueurs construits à partir des maxima locaux, fournit une solution tant efficace qu'élégante et en complexité linéaire à ce problème. En effet, à cause de la nature additive du signal optique, la présence d'un compagnon séparé, même faible, se traduit nécessairement par un surcroît d'énergie à sa position. Le problème de sur-segmentation est évité en filtrant ces maxima secondaires par comparaison avec des seuils dépendants de la séparation. Les paires de maxima sont envisagées tour à tour et le plus faible des deux est comparé aux niveaux attendus suite à la figure de diffraction du plus brillant en prenant en compte les lobes secondaires et le bruit. Pour des performances maximales, l'implémentation logicielle suit les mêmes lignes directrices que celle, matérielle, proposée dans [6].

Caractérisation des objets

L'objectif de la caractérisation des objets est double: d'une part, déterminer la nature des sources pour rejeter celles qui ne sont pas d'intérêt et les artefacts et, d'autre part, estimer la magnitude et la position des objets pour permettre leur observation ultérieure. En pratique, pour faire un usage optimal des ressources de calcul, les descripteurs sont organisés en une cascade adaptative dont la complexité va croissante, de sorte que la segmentation en composantes n'est tentée que pour les CCs de grande taille et seulement après que leur intérêt ait été confirmé.

Trois catégories d'objets parasites sont efficacement rejetés à ce niveau pour réduire la charge sur le système et préserver les ressources pour les observations en AF1: les fausses détections liées au bruit, les étoiles plus faibles que la magnitude d'intérêt limite et les rayons cosmiques. Les deux premières sont aisément éliminées, avec une marge pour le bruit de Poisson, en imposant successivement un nombre minimum d'échantillons, puis un seuil sur le flux et enfin sur le SNR globalement pour l'objet.

Les rayons cosmiques représentent un problème plus difficile parce que les motifs varient avec la quantité d'énergie déposée et l'angle d'incidence sur le CCD. Ceux de faible énergie et les particules secondaires sont les plus problématiques à cause de leur forte ressemblance avec des étoiles faibles. Le tri effectué ici se limite, volontairement, à ceux pour lesquels l'identification est aisée de façon à maintenir faibles les taux de faux positifs et de faux négatifs pour éviter les biais de détection. La répétition d'une part de ces opérations en AF1 (confirmation) permettra de trancher les cas plus incertains. Quatre critères s'appuyant sur des considérations de géométrie et de densité d'énergie sont utilisés.

Finalement, les mesures de position (barycentre des échantillons) et de flux (somme des échantillons) après soustraction du fond de ciel représentent la sortie de la détection.

Performances

Un prototype logiciel complet a été développé pour permettre tôt une évaluation des performances et comme outil pour la vérification de l'implémentation matérielle. Ce modèle, représentatif au bit près, puisque s'appuyant sur le même formulaire d'arithmétique en virgule fixe, permet de valider plus aisément la solution proposée en s'appuyant sur trois cas de densité:

1. la densité moyenne ("l74b15": 25 000 étoiles/deg^2),
2. la densité de design qui est le cas limite spécifié à l'industrie ("l54b0": 600 000 étoiles/deg^2)
3. et la densité maximale sur le ciel qui correspond au centre galactique ("Baade": 3.10^6 étoiles/deg^2),

simulés avec l'outil développé par la communauté scientifique impliquée dans Gaia [34]. Les résultats sont présentés sur la figure C.11 avec un taux de fausses détections qui s'élève à 1 tous les 5.10^6, 110 000 et 11 500 échantillons respectivement. Les performances sont au niveau d'exigence des spécifications qui requièrent d'observer tous les objets d'intérêt lors de 95% de leurs transits jusqu'à la densité de design. La figure C.11 illustre aussi combien la densité est un facteur limitant par suite de la superposition des objets qui rend plus difficile leurs détections même avec le schéma de segmentation élaboré ici. Alors que la précision sur la localisation est d'importance limitée puisque les observations sont contraintes par la grille des pixels, celle sur la magnitude a un impact fondamental sur la gestion des données à bord. Produire un catalogue complet à la magnitude 20 requiert, en effet, d'adopter des marges vis-à-vis

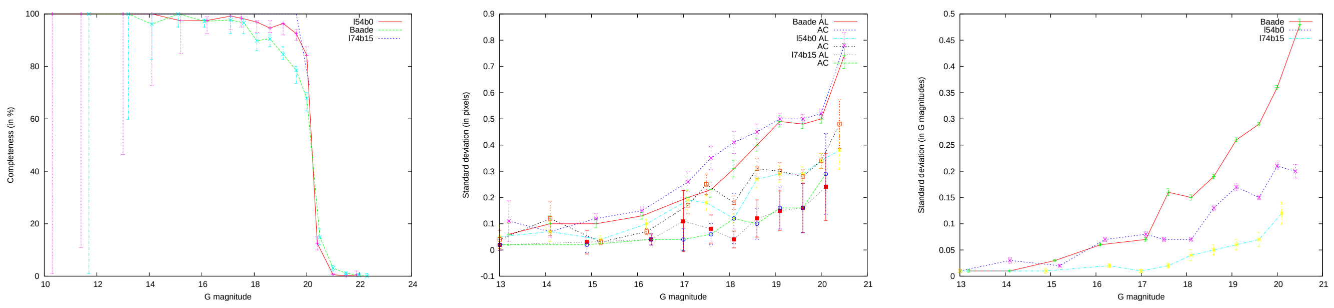


Figure C.11: Complétude et écarts types pour la localisation (dans la direction du balayage (AL) et orthogonalement (AC)) et l'estimation de magnitude (les barres d'erreur sont formelles à 3σ et omises pour la complétude dans le cas l74b15 par suite du nombre très faible d'objets).

des erreurs de mesure et, comme le nombre déjà important d'objets double entre la magnitude 20 et 21, des erreurs importantes conduiraient à noyer les données utiles sous les indésirables.

Du point de vue de l'électronique, les développements ont été limités à la portion qui s'étend de la calibration jusqu'à la sélection des échantillons, repoussant l'identification des CC et la mise en place des files d'attentes pour les objets à un stade ultérieur (à cause du besoin de disposer d'un système logiciel temps-réel pour lire la mémoire placée à l'interface et pour recycler les ressources allouées aux objets). L'ensemble est décrit dans un VHDL proche des recommandations de style faites par l'ESA et a été vérifié au bit près sur des simulations fonctionnelles s'étendant sur plusieurs centaines de millions de cycles et après synthèse. Le choix d'organiser les traitements en pipeline permet un taux de réutilisation élevé des ressources logiques de sorte que le tout occupe une surface comparable à 4 multiplieurs 32×32 bits en dépit de l'accent mis sur la précision et le contrôle des biais. La puce ProASIC3E 600 choisie s'avère cependant un peu exigüe et est avantageusement remplacée par la ProASIC3E 1500 fournissant une marge de 100% et présageant le choix d'un RTAX-S 1000. Enfin, le cadencement fin des différentes parties du traitement se traduit par un schéma pour lequel les contraintes de temps et de routage sont aisément satisfaites, tout en promettant de bonnes performances en termes de consommation électrique.

Conclusions et perspectives

L'approche présentée répond au besoin d'un système à haut rendement assurant une détection non biaisée des objets pour la constitution d'un catalogue statistiquement représentatif de la galaxie. Elle a joué un rôle important durant la phase A du projet en établissant le niveau de performance accessible, en montrant combien le besoin en puissance de traitement représente un défi et en fournissant une base pour certains choix architecturaux liés à la charge utile. Bien qu'une autre approche ait été choisie par le maître d'œuvre en phase B, le démonstrateur complet que nous avons développé valide non seulement la R&D sous-jacente mais fournit aussi un modèle alternatif pour la phase de validation à venir et un exemple d'ingénierie pour des missions à venir confrontées à un besoin similaire quant à l'acquisition et la gestion des données à bord.

Abstract

ESA's cornerstone mission Gaia aims at building a star catalogue limited only by their magnitudes. The expected billion objects must be detected on board before they can be observed and the scientific and technical requirements make this an engineering challenge. We have devised a prototype to assess achievable performances and assist in sizing the on-board electronics (PDHE TDA). It is based on a sequence of four tasks: calibrating the incoming data from the CCDs, estimating the sky background, identifying the objects and, finally, characterising them to command subsequent observations. Although inspired by previous similar studies (APM, Sextractor), this approach has been thoroughly revisited and finely adapted to Gaia. Following the recommendations of the PDHE TDA, a mixed implementation is proposed which deals with the important data flow and the hard real-time constraints in hardware (FPGA) and entrusts more complex or variable processing to software. The segmentation also corresponds to subdividing the previous operations in pixel-based and object-based domains. Our demonstrator shows that the scientific specifications are met in terms of completeness, of precision and of robustness to the variety of observing conditions while, technically speaking, our pipeline, optimised for area and power consumption, allows for identifying a target technology. Our model has not been retained for the industrial phases of Gaia but, beside its recognised usefulness in the project, represents R&D for the forthcoming generation of satellites.

Résumé

La mission Pierre Angulaire de l'ESA, Gaia, doit bâtir un catalogue d'étoiles limité seulement par leurs magnitudes. Ce milliard d'objets doit être détecté à bord pour pouvoir être observé et les exigences scientifiques et techniques font de ce problème un défi d'ingénierie. Nous avons élaboré un prototype pour estimer les performances accessibles et servir au dimensionnement de l'électronique à bord (TDA PDHE). Il s'appuie sur une séquence de quatre tâches: la calibration des données issues des CCDs, l'estimation du fond de ciel, l'identification des objets et, enfin, leur caractérisation pour commander les observations elles-mêmes. Bien qu'inspirée par des études antérieures (APM, Sextractor), cette approche a été intégralement révisée et adaptée aux spécificités de Gaia. Suite aux recommandations du TDA PDHE, une implémentation mixte est proposée qui traite les volumes de données importants et soumis aux contraintes de temps-réel "dures" avec de l'électronique dédiée (FPGA) et réalise les traitements complexes ou variables via du logiciel. Cette partition correspond aussi à subdiviser les opérations précédentes en un domaine pixel et un domaine objet. Notre démonstrateur montre que les attentes scientifiques sont satisfaites en termes de complétude, de précision et de robustesse à la diversité des configurations. Techniquement parlant, notre pipeline, optimisé quant à la surface et la consommation électrique, permet l'identification d'une technologie cible. Notre modèle n'a pas été retenu pour les phases industrielles de Gaia mais, outre son utilité avérée dans le projet, représente une R&D pour la génération de satellites à venir.