UNIVERSITY COLLEGE LONDON

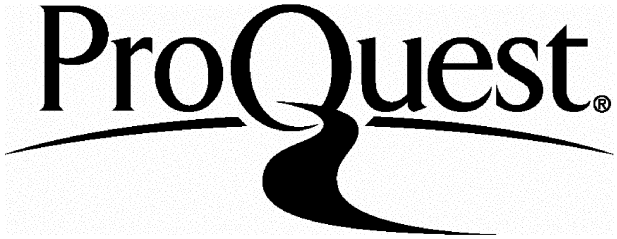
"Real-Time Algorithms for Optimal CCD Data Reduction in High Energy Astronomy"

Stephen James Welch

A thesis submitted to the University of London for the degree of Doctor of Philosophy

Mullard Space Science Laboratory, Department of Space & Climate Physics,
University College London.

August 2001



Abstract

This thesis presents novel and reusable algorithms and philosophies for the reduction of data produced by CCD detectors used for space astronomy. Some of the techniques described can be extended to other two-dimensional data sets, and all of them have relevance beyond the particular spacecraft on which they are currently being used.

The author began the work described in this thesis in January 1995, looking at ways in which the data produced from a spectroscopic instrument on the XMM-Newton spacecraft could be reduced sufficiently to fit into the comparatively meagre telemetry bandwidth available to it. The work was also constrained by the use of a processor system with many fewer resources available than ideal, but chosen for its reliability and tolerance to radiation, both important factors in a ten-year mission.

Chapter one introduces the need for spacecraft onboard data reduction, and the XMM-Newton spacecraft, and its instruments. Chapter two focusses on the principles of operation of CCDs, briefly considering the sources of noise that affect them in use. Chapter three examines the mechanics of the onboard software designed by the author, and arguments are made for trading data quality against data quantity. Chapter four describes the construction of a software, standalone instrument simulator able to quantify the quality of the existing onboard software, provide feedback to settings used, and analyse the impact of future

modifications. Chapter five presents results from the testing of the onboard software and early data from the commissioning phase of XMM-Newton. The thesis concludes with some suggestions for further improvements to the onboard software, and hints at possible applications to other observational scenarios involving large data-sets.

Table of Contents

Chapter 5

Chapter 6

List of Figures

Chapter 1

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Chapter 6

All figures are by the author, except: 1.1, 1.6 courtesy Dr. J. W. A. den Herder, SRON; 1.2, 1.4, 1.5 courtesy ESA; 1.4 courtesy Dr. F. Paerels, UCB; 5.12, 5.13 courtesy Dr. C. P. de Vries, SRON.

List of Tables

Chapter 1

Chapter 2

Chapter 3

Chapter 5

Chapter 6

Chapter 1: Introduction

The X-ray astrophysical observatory XMM-Newton is the largest scientific spacecraft launched by the European Space Agency (ESA), and is producing X-ray imaging and spectroscopic data of unprecedented quality. The author of this thesis was responsible for the on-board data pre-processing software for the Reflection Grating Spectrometer (RGS) instruments on this spacecraft.

The function of this software is to dramatically reduce the amount of data generated by each RGS, in order to fit it into the meagre telemetry allocation—6Kbits/second per spectrometer. This critical software had to be:

- accurate—any errors or ambiguities introduced by the onboard preprocessing might compromise the integrity of the science data;
- reliable, and written to stringent ESA software engineering standards (ESA, 1991);
- able to run on a processor of limited horsepower, and with little memory;
- able to make the best use of the telemetry bandwidth.

Ideally, all of the raw data from each RGS would be available at the ground segment, but owing to the constraint on telemetry bandwidth (the reasons for which are explored in appendix 1), this is rarely possible. That the raw data cannot be analysed in different ways after parts of it have been discarded before transmission, places strong responsibilities on the on-board pre-processing:

- too much data removed, or imperfect use of the telemetry bandwidth will impact the effective collecting area of the telescope;

- incorrect reconstruction of X-ray events with their energies spread across more than one pixel will affect the energy resolution of the instrument;
- unremoved 'bad' or 'hot' pixels may be wrongly interpreted as spectral features in the source;
- imperfectly deleted cosmic ray charge fragments may be misconstrued X-ray charge deposits;
- unstable software could harm long exposures (up to 65k seconds in the early phases (Erd, 2001), and now up to 130k seconds), or even cause programmatic difficulties.

This thesis is about the methods used by the author to achieve the demanding results required of software of this criticality.

This chapter gives some description of the XMM-Newton spacecraft, and of the RGS instrument in particular, with some quantifying of the disparity between the amount of data generated by the instrument and the telemetry bandwidth available to it. The next chapter describes the basic operation of a charge-coupled device (CCD) detector, and how these devices can behave when used for X-ray astronomy.

The main output of the author's work—the current RGS on-board software—is described in chapter 3, and includes some novel approaches and techniques which may find application elsewhere. Chapter 4 describes a standalone instrument simulator in software, created by the author for use when denied access to the hardware development system, and ultimately developed into a deliverable item which could be used for analysing such raw data as is available from the RGS, with the results used to feedback values for parameter setting to

the onboard software. A description of the testing of the software, and of the early flight results can be found in chapter 5, with conclusions and some suggestions for future work in chapter 6.

1.0 The XMM-Newton Spacecraft

1.0.1 Spacecraft Configuration

This observatory class satellite has the largest aperture ever flown for an X-ray mission, comprising three separate nests of fifty-eight mirror shells forming three individual telescopes (Jansen *et al*, 2001). At the focal point of each telescope is a CCD camera designed for imaging and medium resolution spectroscopy in X-rays. These are the 'European Photon Imaging Camera' (EPIC) CCDs, but in the case of only one of the telescopes do the CCDs receive the whole flux gathered by the mirrors. For the other two, the mirror shells are followed by a reflection grating array which picks off a little less than half of the flux, dispersing the photons according to their energy.

The individual gratings which form each of the two assemblies are arranged on a toroidal Rowland surface, and give an additional off-axis focus for the mirrors, this time for spectroscopic observation (Kahn *et al*, 1996). At this focus is the RGS Focal Camera (RFC)—a strip of CCDs arranged along the dispersion axis, and these, along with their associated electronics and gratings form an RGS. Two views of this arrangement are shown schematically in figure 1.1.

Details of the international consortium responsible for the complete RGS instrument chain can be found in appendix 4.

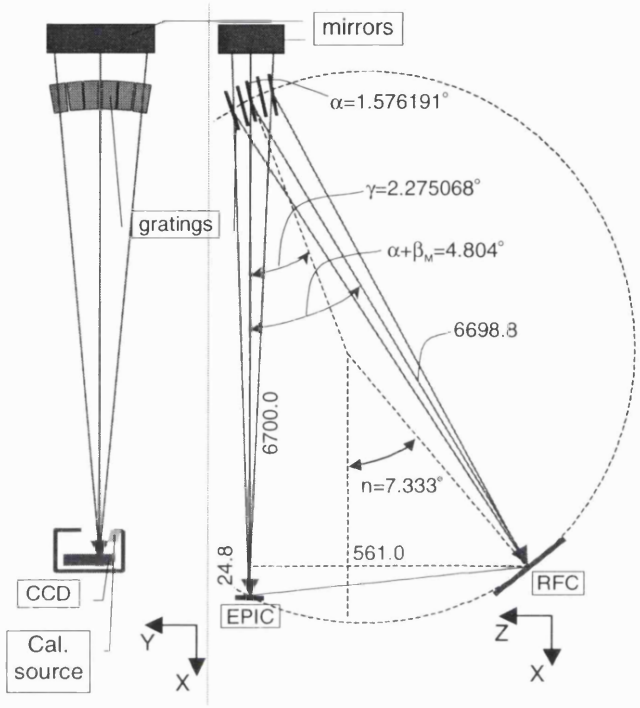


Figure 1.1: *Arrangement of mirrors, gratings and CCDs in the RGS. X-rays arrive at the top of the diagram, and a few key linear dimensions are shown with units in mm. The right-hand part of the illustration shows the splitting of the incoming Y-ray beam between EPIC and RGS foci. α, β and γ are the angle of incidence, the blaze angle and the graze angle on the facets of the grating and are shown in more detail in figure 1.6. The left-hand part of the illustration is an anti-clockwise rotation, out of the page (note the Y, X and Z, X arrows), looking now end-on to the CCD bench. Also indicated here are four internal in-flight calibration sources, producing fluorescent emission at 1487eV (aluminium) and 676.8eV (fluorine). Unless otherwise noted, numerical details are all taken from den Herder et al, 2000).*

In addition to the X-ray instruments, the observatory includes an optical telescope to cover the optical and UV regions of the spectrum (1500–6000Å). This telescope—the 'Optical Monitor' (OM)—is a Ritchey Chrétien instrument with an aperture of 30cm, and some additional optics formed in the detector window. The spectrum coverage of the XMM observatory is, then, from about 1Å to about 6,000Å, with a gap in the region from 100Å to 1500Å (coinciding with the wavelength range which is subject to a high degree of absorption by the interstellar medium). The OM will also be used for the real-time optical identification of X-ray sources, and having this telescope onboard makes XMM the first truly multi wavelength observatory.

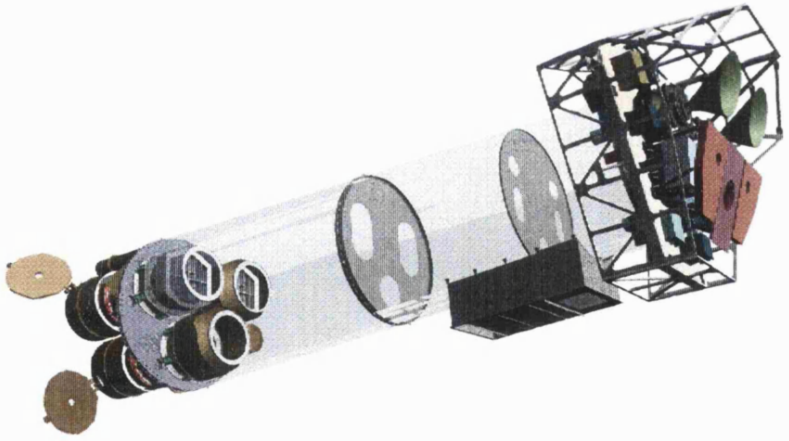


Figure 1.2: *Three dimensional schematic of the XMM-Newton Spacecraft. X-rays enter from the left hand side of the diagram, where the three mirror modules can be seen with their protective doors open, and grating assemblies visible behind two of them. To the right of the illustration is the instrument platform which supports the radiators, and the detectors and associated electronics.*

1.0.2 Spacecraft Orbit and Operations

The orbit of XMM-Newton has been chosen to allow the five CCD cameras to be cooled to temperatures as low as –100°C using only passive radiators, and to allow long uninterrupted pointings at sources of interest. The 48 hour highly elliptical orbit is inclined at –40°, having a perigee of 7000km, an apogee of 114,000km, and operations are limited to altitudes above 60,000km when the spacecraft is clear of the Earth's radiation belts. Allowing for the times when the Earth's magnetotail impinges even at the operational altitudes, the instruments are operated for about 65% of the time in the periods of lowest background radiation (Erd, 2001). XMM-Newton can cover most of the sky, excluding only a region near the Sun, the solar aspect angle being fixed by the baffle design and attitude control tolerance to within ± 20° of a perpendicular to the telescope axis.

Three groundstations are currently in use, located in Perth, Kourou and from February 2001, Santiago. Between them they allow uninterrupted exposures of up 135,000 seconds. The telemetry stream is structured using variable length packets: packets of science data from the instruments are merged with instrumental housekeeping data packets and other spacecraft data packets, aggregating to a total data rate of 64Kbits per second.

1.1 The RGS Instrument

The two RGS instrument chains on XMM are designed to give a previously unattained sensitivity combined with high energy resolution in the 'soft' X-ray region (5–35Å, 2.5–0.4keV). This region is of interest because it contains many

prominent features from nearly all ion stages of many abundant elements, including the K-shell line and continuum transitions of carbon, nitrogen, oxygen, magnesium and silicon; and the L-shell transitions of silicon, sulphur, argon, calcium, nickel and iron (Paerels *et al*, 1998).

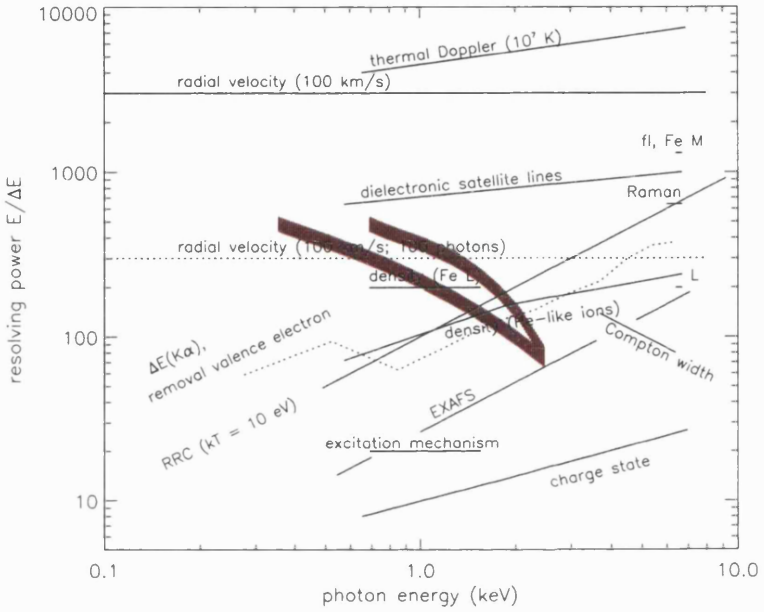


Figure 1.3: *Plot of resolving power vs. photon energy showing the figures necessary to perform various spectroscopic diagnostics. Actual performance of the instrument in flight for first and second orders is overlaid in red, second order being the higher of the two bands.*

The RGS instrument can detect many of these features—in emission or absorption—for a large variety of stellar, interstellar and extragalactic plasmas. The RGS instrument provides the first ever opportunity for determining important physical parameters of these plasmas, such as electron temperatures, density distributions, ion and elemental abundances, mass motions and the nature of the ambient radiation field, and the resolving powers needed to perform these (and

other) diagnostics are shown in figure 1.3, plotted as a function of wavelength and with the actual in-flight performance of the RGS over-plotted in red.

To meet these targets, then, the important criteria for the spectrometer are sensitivity and resolving power (den Herder *et al* 2001). To achieve a suitable count-rate of photons from the faintest sources of interest the telescope mirrors are designed for high throughput: i.e. large effective surface area. Figure 1.4 shows the impressive effective area that has been achieved on XMM relative to other X-ray missions.

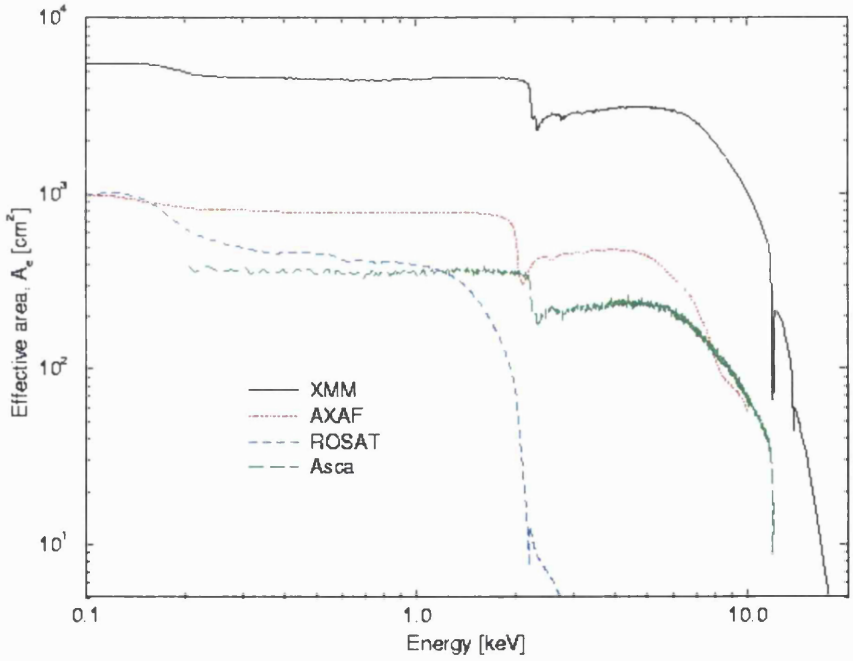


Figure 1.4: *Effective area of the XMM-Newton mirrors plotted as a function of energy, and shown relative to equivalent plots from other X-ray missions.*

The mirrors have a significant collecting area from 0.1 to 10keV, but this is further influenced by the responses of the instruments themselves, as shown in figure 1.5.

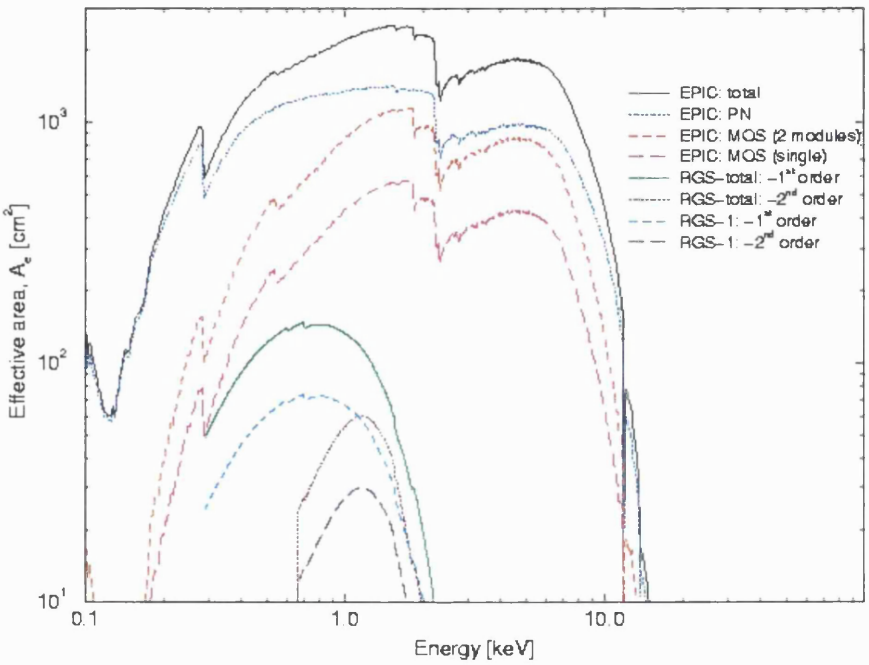


Figure 1.5: *Net effective area of the mirrors when folded in with the responses of the individual instruments. Notice the abrupt reduction in effective area seen by the EPIC MOS cameras compared with the EPIC PN, this being of course due to the sharing of the flux with the RGS.*

Owing to the engineering constraints involved in making the mirror shells (Chambure *et al* 1997), the resulting array is somewhat 'moderate' in resolution, contributing about 15 arc-seconds of 'blur'. Aside from looking at higher spectral orders, the resolving power of a spectrometer with a particular mirror accuracy can be improved by increasing the line density of the grating. The target results for the RGS case could not be met using traditional transmission gratings. The

resolving power of the RGS was set at that required to separate the three lines of He-like triplets of the most abundant elements in the spectral band, including O VII—a useful plasma density diagnostic. The targetted first-order resolving power range for the RGS was approximately 100 – 500. For an $E/\Delta E$ of, say, 250 and a mirror blur of 15", this implies a need for a dispersion angle of 15 x 250/ 3600 ≈ 1.04°, and hence a grating period in excess of 20,000.

Happily, owing to the way in which the effective period of a reflection grating is projected rather than physical, densities of this sort can be achieved. By orientating the grating at a grazing incidence to the incoming beam the physical grating period is increased by a factor of:

$$\frac{d}{\sin\alpha}$$

In the case of the RGS gratings, the average period of 646 lines/mm becomes, at an angle of incidence of 1.58°, approximately 24,000 lines/mm. This, combined with the mirror blur gives a resolution element of 250μm x 2 mm in the dispersion/cross-dispersion axes.

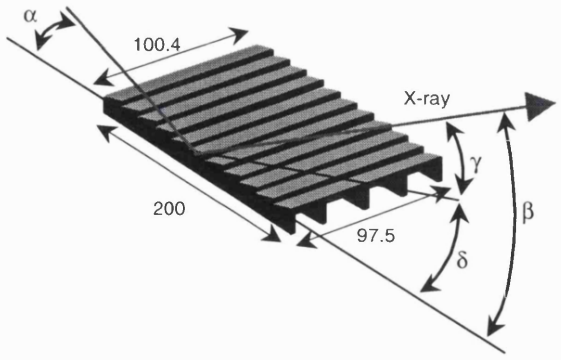


Figure 1.6: *Schematic of an RGS grating showing key dimensions (mm) and angles. α is the angle of incidence, β is the diffraction angle for the blaze wavelength, γ is the graze angle of the facets on the grating, and δ is the blaze angle.*

The 182 individual reflection gratings that form the reflection grating array (RGA) in each of the RGS chains are set at grazing incidence to the beam in the 'classical' configuration, in which the incident and diffracted rays lie in a plane which is perpendicular to the grating grooves (see figure 1.6), and are mounted in the otherwise unused space between the mirrors and the focal plane. The X-ray beam from the mirrors is still converging at this point, but this departure from parallelism is easily corrected by varying the line spacing on the gratings. Even if the entire flux from the mirrors was available, increasing the packing density of the gratings beyond a certain point will lead to them self-vignetting, hence it is practical to include imaging detectors at the principal telescope focus also. Other aberrations are eliminated by centering the gratings on a type of Rowland circle orientating them such that each grating makes the same angle α with respect to the incident beam that intersects its centre.

If the telescope focus lies on the Rowland circle as well, then rays diffracted from the centres of the gratings will converge to a single spectroscopic focus also located on the circle and aberrations are eliminated, aside from some astigmatism in the cross-dispersion direction. This arrangement works well at all wavelengths as long as the spectroscopic detectors are arrayed tangentially to the circle.

The dispersion axes for the two separate RGS instruments are parallel in order to provide an extra degree of redundancy: ie in addition to having the two RGSs seeing the same source, a spectral feature of interest could be moved to another CCD in the event of a failure. In practice, however, the loss of efficiency engendered by this (about 50%) makes it less likely to be useful as long as there is

a functioning CCD in the necessary position on one or other of the CCD benches. The X-ray spectrum is imaged as a narrow band, and nine back-illuminated CCDs are used to cover it, with the position of the X-ray photon on the detector along the dispersion direction giving the wavelength. The different spectral orders are overlaid on the CCD surface, but these can be separated by the intrinsic CCD energy resolution, as will be seen in chapter 2.

1.1.1 Instrument Configuration

The 'detector end' of the RGS divides into three sections, and these are illustrated in figure 1.7. The 'RFC' is the RGS Focal camera, and comprises the cooled CCD bench and associated 'front-end' conditioning electronics. The 'RAE' is the RGS analogue electronics. This is a separate unit providing further signal conditioning and monitoring, the analogue-to-digital conversion and the clock sequence generator.

This last, though more strictly a 'digital' device, is part of the RAE for logistical reasons. Interfacing to the spacecraft data and power buses is the 'RDE', the RGS digital electronics. This unit contains the processor systems for instrument control and data handling, the instrument power supply, and the thermal regulation circuitry.

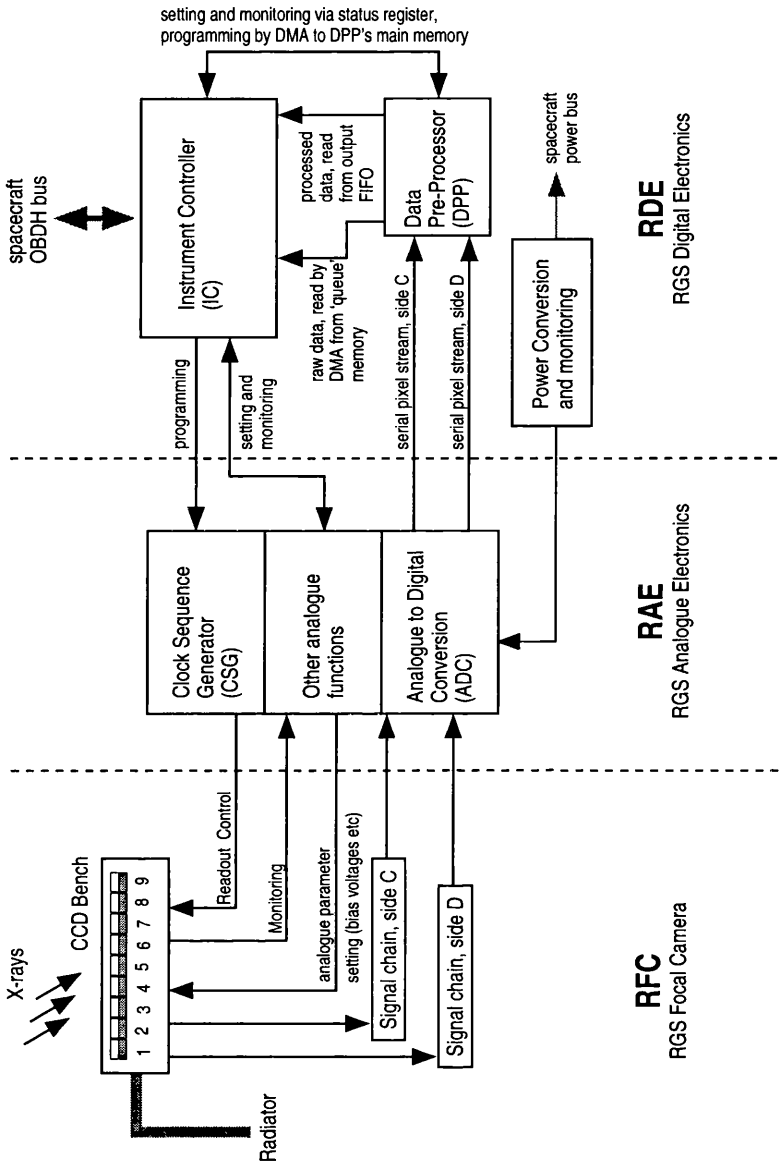


Figure 1.7: *Simplified block diagram of the configuration of the key components of the RGS Instrument at the detector end. The three sections of the diagram follow the division of the system into separate units, with the RFC at the 'front-end', and separate digital and analogue electronics sections. For obvious optical reasons there is only one CCD bench per RGS chain, and there are limited opportunities for redundancy within the RAE owing to signal constraints. Each RGS does, however, include two RDEs with one as a cold-redundant spare.*

1.1.1.1 CCD Camera Analogue Electronics

The 'analogue electronics' of each RGS chain (known as the 'RAE') comprise the conditioning electronics for the nine CCDs and the readout amplifiers and requisite signal clocking circuits and multiplexers.

The RGS uses large format CCDs—1024 x 384 pixels of 27μm square— developed by EEV/Marconi, with special enhancements such as improved output amplifier performance. The CCDs are 'frame transfer' types (of which more in 2.0.4.1), and each has two readout nodes. Of the four potential read-out nodes available (A, B, C, D), only C and D are employed, each connected to an independent electronic signal chain. Use of two nodes in this way makes it possible to read the CCD out faster, but also provides warm redundancy in the event of a failure in one or other chain. The CCDs are arranged as shown in figure 1.8.

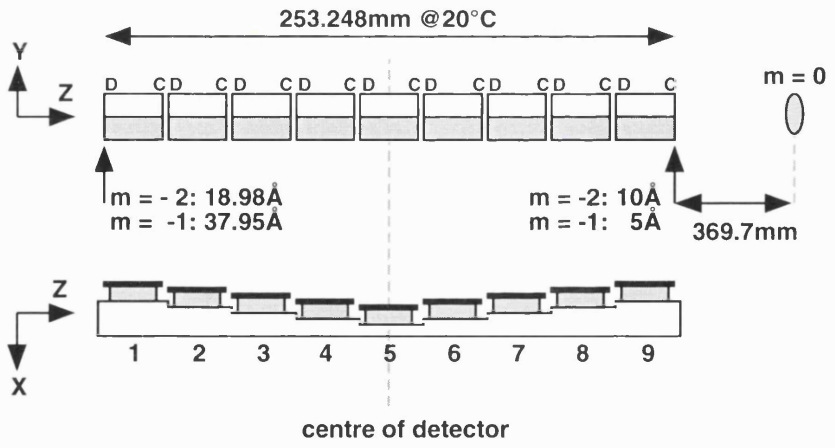


Figure 1.8: Disposition of the CCDs in the RGS CCD bench. The CCDs in each view are aligned: in the upper view the 9 CCDs can be seen, each divided into image (grey) and storage sections. The dead-space between each CCD is 0.5mm. Also shown in this view is the position of zero order, some 369.7mm to the right. The lower view looks at the CCDs 'edge-on'. The CCD chip is the thick black line, and each is mounted at a step up (or down) from its neighbour such that the whole set follow an arc on the Rowland circle at the RGS's focus.

Throughout the rest of this thesis pixels will be classified as arriving from 'side C' or 'side D', and it is often convenient (and realistic) to imagine the CCD bench comprising eighteen CCDs rather than nine. A further benefit accruing from the dual readout node arrangement is that settings for analogue parameters such as gain can be optimised on a per CCD half basis.

Parameter	value
Typical event size	1.3 – 1.7 x 27μm pixels (i.e. 1x1 OCB)
Partial event fraction	~5%
Dark current	0.02 – 0.08 electrons/pixel/second at -80°C
Parallel CTI	$<10^{-5}$ per transfer
Serial CTI	$< 3 \times 10^{-5}$ per transfer
Readout noise	5 – 6 electrons per pixel
Quantum Efficiency	>80% for E = 0.35 – 2 keV

Table 1.1: *Summary of key parameters of the RGS CCDs. The figures result from ground calibration tests on the flight CCD benches, and are further explained in Chapter 2.*

The readout noise associated with the signal at each output node is reduced by the use of 'correlated double sampling' (CDS), whereby a sample of the noise present on the output node immediately after the reset is sampled and then subtracted from the readout of the signal itself, which is stable at the same node after a known time period. The nine CCDs share a single subsequent signal chain for each of nodes C and D, with the outputs from the CDS front-end amplifiers multiplexed to 12-bit analogue-to-digital conversion. The gain and bias settings of the output amplifiers are configured such that one analogue-to-digital-unit (ADU) equates to 1eV, and this remains true across all nine CCDs. Clock and

serial data chains for the two signal paths are then entirely independent and are merged at the front-end of the DPP, which sees the pixels arriving two every 15μs (ie effectively a pixel every 7.5μs when read out is from two nodes).

1.1.1.2 The Digital Electronics

The RGS digital electronics (RDE) comprise the instrument controller (IC), the data pre-processor (DPP), the local dc-to-dc power convertor for the whole instrument, and some monitoring for this power. Of particular interest here are the IC and the DPP, both processor systems based on the Plessey MA31750—a 16-bit microprocessor built to execute the MIL1750 instruction set.

As its name suggests, the IC is at the heart of the instrument, and is responsible for:

- interfacing the instrument to the spacecraft on-board data handling bus (OBDH), receiving and distributing command information;
- storing settings for the 'adjustable' parts of the system, and programming them as appropriate;
- controlling the temperature of the CCD bench;
- collecting housekeeping values from the various parts of the system and forming the information into packets;
- collecting pixel data from the DPP and forming it into packets;
- loading software and data into the DPP;
- conducting exposures, setting parameters, starting and stopping sequences and so on, as appropriate.

The DPP design is a hybrid system in which the DPP has a 'hardware' front end ending in the 'input' first-in first-out buffer (FIFO), and then a software con-

nection to take data from this input FIFO, and write it to the 'output' FIFO, having performed any of various options of additional processing *en route.*

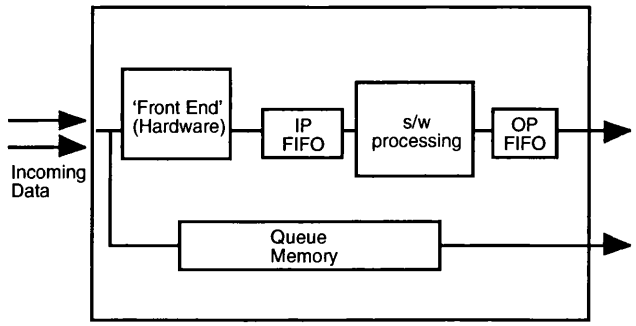


Figure 1.9: *Data paths through the DPP. The telemetry allocation of the RGS instrument does not permit the transmission of every pixel, and so the data is reduced in this pre-processor using a mixture of hardware and software. Since raw data is sometimes necessary, a method is included whereby selected CCD data may be stored, untouched, in a queue for fast download or for trickling into the telemetry stream as bandwidth permits. This raw data is known interchangeably as 'queue data' or 'diagnostic data'.*

The front end merges the data from the two readout chains, attaches a counter to each energy and rejects pixels below a programmable threshold. A separate route for the pixel energies is via the queue memory, where raw data can be passed to the telemetry with no processing applied, downloaded in blocks in one go, or trickled into the telemetry stream to the ground segment as bandwidth permits.

As far as the processor in the DPP is concerned, it is very much an embedded system insofar as the cpu has no knowledge about the outside world: even to the extent that the designer eschewed the use of 'port addressing' — the usual method of interfacing to items outside of the processor — choosing instead to insert all external entities (i.e. status registers, FIFOs etc.) as items in the memory

map. In the case of the MA31750 cpu memory mapping is particularly advantageous for speed as it uses typically one third of the number of clock-cycles per access. The use of the compiler and optimiser is made more complex by this method, however, since it gives the appearance that apparently redundant repeated reads and writes are being conducted — for example when polling the status register to monitor the setting of a particular bit.

The IC operates the DPP by programming its hardware front end with certain parameters at the beginning of each readout:

- CCD number, generally referred to as the 'CCD ID'
- number of readout nodes
- lower threshold to apply to each of the two nodes

The ID of the CCD being read out is important to the software part of the DPP, as it is used to select parameters from a bank, also programmed by the IC and described in 3.1.1.2. The number of nodes (can be 1 or 2) determines how the remaining data will be written into the input FIFO. Finally, it is valuable to have separate thresholds for each CCD node as they can form part of the 'tuning' used to optimise the signal quality from each node. The input FIFO of the DPP is seen by the software as 48 bits wide (three transfers on the 16 bit data bus), and the functions of these are detailed in table 1.2.

The DPP output FIFO is 16 bits wide and 1024 words deep. As will be seen in 3.1.1.3, the most a DPP generated word will need to use is 12 bits, and so the remaining 4 are free for use as 'tags' identifying them to subsequent processors, e.g. "I am a y-coordinate", "I am an x-coordinate etc.".

bit number	function
0 – 3	CCD ID, bit 0 to bit 3
4 – 15	Pixel energy, bit 0 to bit 11
16 – 34	Pixel count, bit 0 to bit 18
35	Pixel count bit 19, or side flag (set to '0' for a pixel originating from side C)
36	start-of-CCD bit
37 – 47	not used

Table 1.2: *Bit allocations in the DPP input FIFO. On detection of the start of CCD bit (bit 36) the software knows that a new CCD readout has started and will use the arrival of this to trigger various actions. Note that this bit simply appears in the data stream in the FIFO along with the CCD ID, and is not a special signal. This FIFO is in fact physically only 36 bits wide, being formed from registers and 9 bit devices, mapped onto the 16 bit wide data bus. The input FIFO is effectively 1024 48 bit words deep.*

Input and output 'flow' is outside of the control of the DPP. If the input rate to the front-end exceeds the rate at which the DPP processing software can drain the input FIFO, then incoming data will be lost. In the event that the hardware fails to complete a write to its FIFO it will increment a 'lost event counter' which is monitored by the IC. Owing to engineering constraints, this counter rolls over at 10 bits, and so the number of lost events cannot be measured from this (because a low number of lost events may or may not include multiples of 1024). The purpose of this counter is to flag to the operator that some parameters need to be changed, and frames with an associated non-zero lost event count are

viewed with suspicion. The drain rate of the output FIFO is governed by the IC, which is in turn driven by capacity on the spacecraft bus.

1.2 Quantifying the Need for On-Board Data Reduction

A fuller treatment of the operation of a CCD will be found in chapter 2, but allowing, for the moment, that the pixel data is read from the CCD serially, then the raw output of the CCDs can be considered a one-dimensional array of energy values, where the position of an element in the list implies its original coordinate on the CCD. Thus with knowledge of the readout pattern (i.e. such and such starting pixel, so many lines by so many columns) simply transmitting the contents of the array to the next stage of processing is enough for it to be able to deduce the two-dimensional coordinates to go with the energies. The total number of bits for transmission of the raw data, N, is then the number bits of energy resolution, n, times the total number of pixels:

$$N = n \times \text{number of lines} \times \text{number of columns}$$

In the case that not all of the pixels are wanted—perhaps there are some known to be faulty, or there are a significant number of pixels containing only noise—we can delete these unwanted pixels at the 'front-end'; however, in this case we no longer consider only the energy values of the pixels, but must also transmit information about where each pixel appears in the list. This means that we must be certain of the value of removing pixels before transmission, since below some threshold there will be a net increase in data. Taking the example of a readout pattern 100 rows by 100 columns, and with an energy resolution of 10 bits, then the total number of bits to be transmitted will be:

$$100 \times 100 \times 10 = 100{,}000$$

Including the x and y coordinates implies that we need a further 7 bits ($2^6 < 100 < 2^7$) for each coordinate:

$$100 \times 100 \times (10 + 7 + 7) = 240{,}000$$

suggesting the rule-of-thumb result that the break-even point between sending the data in its raw form, and removing some data but adding extra information in the form of coordinates typically occurs at below half of the raw number.

The RGS CCD bench comprises 9 CCDs, each of which is 1024 pixels by 384 pixel, making a total of 393,216 pixels. The energy resolution of the digital-to-analogue conversion is 12 bits. Were the instrument to attempt to telemeter every single pixel to the ground station, each CCD would therefore require:

$$393{,}216 \times 12 = 4.5\text{Mbits}$$

The integration time for this readout pattern is 3.94 seconds, implying the need for something over 1Mbits/sec. from the telemetry allocation—even without including any overhead from packetising the data or including any housekeeping information about the instrument. Since the baseline telemetry allocation to the instrument is 6KBits/second, it is clear that some on-board processing is required. What about the implication of the extra bits that need to be added? Assuming a worst case where every pixel energy is sent, but now has to be accompanied by an index number, then a further 19 bits will be needed to encode the highest number, giving:

$$393{,}216 \times (12 + 19) = 11.625 \text{ Mbits}$$

implying that we need to remove more than 61% of the pixels before we find merit in processing them on-board. In the case of the RGS, this decision is made easy since with a detector looking at dispersed spectra, the number of 'wanted' pixels is likely to be small in comparison with the number containing only noise.

In fact, there are typically fewer than 40 valid pixels per CCD readout, and the art of separating these pixels from the other signals from the CCD is the key purpose behind the work carried out by the author.

In situations where the data is not random, algorithms for noiseless data compression can be very effective at reducing the bandwidth requirements, though implementing these algorithms will typically require more processor 'horsepower' than available to the DPP. It was originally envisaged that the DPP would contain no cpu whatsoever, but only discrete logic. As will be discussed in 3.1, the DPP was to become the hybrid system described, with a relatively slow processor (<1MIP), and the task for this processor was still pre-processing, and not compression. Data compression techniques are employed in the IC (Al Janabi, 1999), but these are quite separate from the work of the DPP, which is very much a 'preprocessor'.

Elsewhere on XMM, the EPIC 'MOS' cameras also employ data preprocessing, but using a system based around a fast application-specific integrated circuit (Turner *et al*, 2001). The EPIC preprocessing system sorts the raw image from the CCD by using a pattern recognition technique which is also effective at removing cosmic ray deposits.

1.3 Preprocessing Requirements

In any scientific spacecraft it is the target of the engineering design to return as much as possible of the 'raw' data to the ground segment, since once data is discarded it is lost forever, and therefore unavailable to reprocessing by new techniques.

A number of requirements were defined for the processing in the DPP:

- Thresholding: three levels required:

  (i) 'rejection' or 'lower' threshold to provide a lower level for pixel energy values in order to forward only pixels which are above the noise floor (of which more in chapter 2);

  (ii) 'acceptance' level to allow for some finesse in applying the lower threshold, by defining a further low threshold that can be applied only to certain classes of pixels, specifically those isolated pixels where we can be certain that the measured energy level cannot be boosted by aggregating it with an adjacent pixel, which is possibly part of the same X-ray event;

  (iii) 'upper' level to reject pixels whose energy is beyond that which could be deposited by a high-energy X-ray photon of the type of interest.
- Bad pixel removal: any pixels known in advance to contain unwanted data, for whatever reason, should be removed by the DPP;
- Split Event Identification: As will be seen in chapter 2, it is a feature of the type of CCDs used, and the energy level of the photons impinging on them, that the charge resulting from a particular X-ray may be spread across more than one pixel. At the time of the writing of the RGS requirements (1995/6) there was not yet a formal definition of how this information would be used, but it was at least clear that the DPP needed this identification in order to:

  (i) identify single events for acceptance thresholding purposes;

(ii) identify single events for rejection purposes—ie keep only the single events when the source is bright enough to saturate the telemetry;

(iii) identify split-events for onboard reconstruction purposes, with any pixels that are reconstructed also flagged for identification in the ground segment.

At the time the author took on the responsibility of designing the DPP software, three generic modes for the instrument had been specified (Branduardi-Raymont, 1996):

- "Diagnostic Mode" —all pixel data is forwarded to the instrument controller via the queue memory with no software involvement (the DPP cpu is held in 'reset'). For the RGS, the term 'Diagnostic data' is synonymous with the term 'raw data', ie each pixel clocked out of the CCD is forwarded for transmission to the ground segment without any onboard processing applied;
- "Spectroscopy Mode" —The main purposes of this mode are served by the additional software processing available, so the IC will primarily take its data from the output FIFO, optionally interleaving diagnostic data from the queue memory as bandwidth permits. Two sub-versions of spectroscopy were identified: 'basic' and 'high-event rate' (HER). Lower thresholding and bad pixel rejection are common to both, with HER the home for other functions available, with inclusion of acceptance thresholding baselined, but not yet a clear specification of how the other options might be adopted;
- "High Time Resolution"—in this mode a special clock sequence is employed which reads 74 lines at a time from the region where the spec-

trum is expected to fall, summing them into a single row (ie over the y-axis). Pixels from this row are processed by the DPP in the same way to those acquired in basic spectroscopy. Allowing for the potential for spectral accumulations to overlap owing to pointing jitter, the effective time resolution for the instrument in this mode is approximately 20 milliseconds when repeatedly reading from a single CCD

The processing options and combinations, and how they could be achieved within the hardware constraints (memory and cpu horsepower) are considered at length in chapter 3.

1.4 Summary: the need for the onboard processing software

Capacity on the spacecraft data bus is a precious commodity, and so the DPP must ideally never keep the IC waiting for data in the output FIFO, and since the X-ray photons are similarly precious, the DPP must also keep the input FIFO from becoming full. Were the CCD frame easily subdivided into X-ray events and noise pixels this would not be a challenge, and a simple thresholding technique would be enough to strip the unwanted pixels. This is of course not the case, and a real frame includes pixels at energies more or less across the whole range of possible values because:

- X-ray photons may have their energy spread across more than one pixel in various ways, thus reducing the pulse height
- X-rays may be only partially absorbed, losing some of their energy to regions in the CCD outside of the charge collection wells: this will reduce the pulse-height of both single events and split events;

- There are other sources of energy in the pixel wells than X-rays and noise: cosmic rays and minimum ionising particles will leave footprints not necessarily contained within a single pixel and though some of these will be outside of the expected energy window, there will be fragments which are not. These fragments may appear within the X-ray energy discrimination window, or abut other fragments which could be misleadingly aggregated to form spurious events.

The result of these factors is that though a front end threshold of some sort is necessary, placing this too high will eat into the set of pixels which may contain some useful information. Some further pixel rejection can be accomplished by applying the 'acceptance' threshold as mentioned above, but this may still leave more pixels behind than can be fitted into the telemetry. Simply raising these low energy thresholds until the data rate fits into the bandwidth is a rather blunt instrument, with serious consequences on efficiency and energy resolution—and will be particularly ineffective when looking at faint sources where the loss of detail may make it impossible to separate the source from the background—as will be seen in chapter 5.

The preprocessing software must therefore be able to interpret what it finds when looking at the data itself: correctly identifying items that can be deleted without leaving any ambiguous remnant; reconstructing all of the components that make an event of some sort, such that appropriate thresholding can be applied to all of them; and locating any remaining reconstructed events correctly such that the wavelength calibration of the RFC is not degraded.

Chapter 2: CCD Detectors

The celestial sources of interest to X-ray astronomers produce photons which though high in energy are few in number in the vicinity of the Earth, and it is typical for X-ray detectors to suffer from a poor signal-to-noise ratio (Fraser, 1989). Various spaceborne detectors have been used in X-ray astronomy, particularly the proportional counter, but with its ease of use and improved intrinsic energy resolution (requiring only about 3.65eV to generate an electron-hole pair in silicon) the CCD is becoming widely used in this field. Chapter 2 reviews the principle of operation of a CCD, and further explores the interactions in the semiconductor material with respect to photons at X-ray energy levels, but also cosmic rays and minimum ionising particles, both of which contribute significantly to the background seen by the RGS and strongly influenced the design of the onboard software.

2.0 Basic CCD Operation

2.0.1 Charge Storage

A CCD is formed from a matrix of metal-oxide semiconductor (MOS) capacitors. The construction of such a cell is illustrated in figure 2.1.

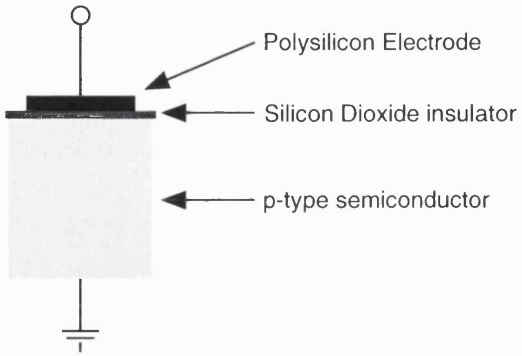


Figure 2.1: *A schematic of the construction of a MOS capacitor*

With no potential difference across the p-type material in figure 2.1, there will be an even distribution of the majority carriers, or 'holes'. On application of an increasingly positive voltage (of the order of a few volts) at the electrode, the holes will be repelled from the region immediately below, creating a depletion zone.

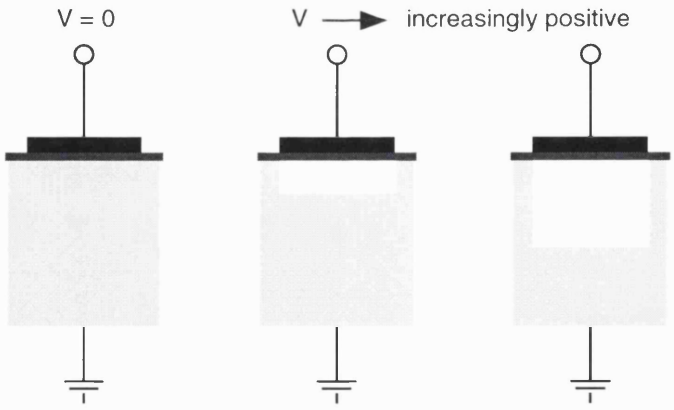


Figure 2.2: *A sequence of schematics of the MOS cell introduced in figure 2.1, showing the increase in magnitude of the depletion zone (shown in white) as the majority carriers are swept from below an electrode at an increasingly positive potential.*

The potential difference at the semiconductor/insulator interface is called the surface potential. Dependent on the number of electrons available, there will be a threshold where the surface potential becomes sufficiently positive that enough electrons are attracted to this region to form an inversion layer. This is the 'threshold voltage'. Below the threshold voltage the depth of the depletion zone is proportional to the magnitude of the voltage applied to the electrode, as shown in figure 2.2.

The magnitude of the threshold voltage, and hence the potential depth of the depletion zone, can be increased by cooling the CCD. These depletion zones are commonly called 'potential wells'.

2.0.2 Charge Transfer

Extending the concept of a semiconductor charge well, the next consideration is the question of how to 'read' any charge collected in it. The complexity implied by an attempt to connect to each well individually would militate against greater numbers of wells, which is the opposite of the target. There is however, an arrangement of these MOS charge wells which can permit an entire CCD to be read out serially from a single node. The mechanism of 'charge transfer' is used to effect this moving of data, and is handled this way. Consider two adjacent wells, as in figure 2.3:

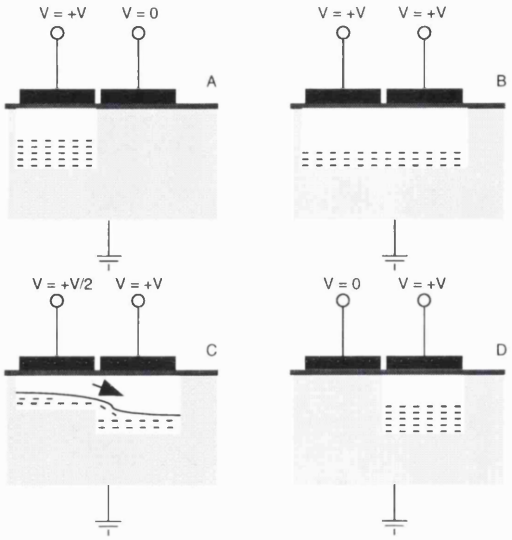


Figure 2.3: Charge Transfer between MOS cells. (A) A quantity of charge is stored in the potential well to the left. (B) Increasing the voltage on the right-hand terminal creates a potential well below it, and the charge is now distributed between the two. (C) The voltage on the left-hand terminal is now reduced, and as its potential well collapses the charge flows towards the right-hand well until the transfer is complete (D).

Continuing the analogy to water implied by the use of the term 'well', it can be seen in figure 2.3 that charge can be first shared between wells, and ultimately transferred between them. This diagram is of course a schematic only: in the physical case the electrons are collected directly under the terminal.

The most common technique for arranging the wells, and that employed by the RGS, is that of the three-phase system. In this arrangement, each CCD picture element ('pixel') is formed from three adjacent wells. Electrodes are now classified into three types, and all of each type are connected together, as shown in figure 2.4.

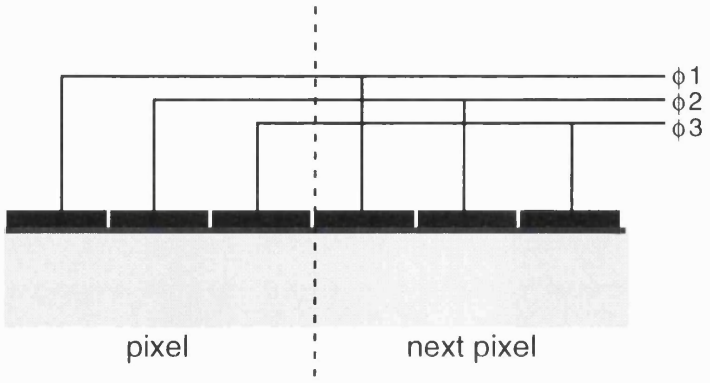


Figure 2.4: Section from a CCD surface showing the construction of individual pixels from three electrodes, wired to a three phase driving system.

It can be seen that the data in both 'pixel' and 'next pixel' can be moved simultaneously, and in either direction. Ways in which the three phases of signals can be 'clocked' to give the different results are considered in 2.0.4.

2.0.3 Buried Channel CCDs

The discussion so far has been focused on 'surface channel' CCDs, so called because the charge storage and movement occur in the region at the surface of the semiconductor material abutting the insulator. Simplest to make and to understand, this type of construction does have some drawbacks. It is a feature of the silicon/silicon-dioxide interface that there are regions which will attract charge very quickly but will be 'reluctant' to release it: these are 'fast surface states', and their existence limits the speed at which the stored charge can be moved.

The solution to this problem came with the invention of the 'bulk channel' or 'buried channel' CCD (Walden *et al*, 1972). In this arrangement the charge storage area is buried in the bulk semiconductor material—i.e. away from the surface. This shift of the depletion area is achieved by interposing a layer of n-type semiconductor between the insulator and the p-type, and reversing the way that the voltage biasing is handled.

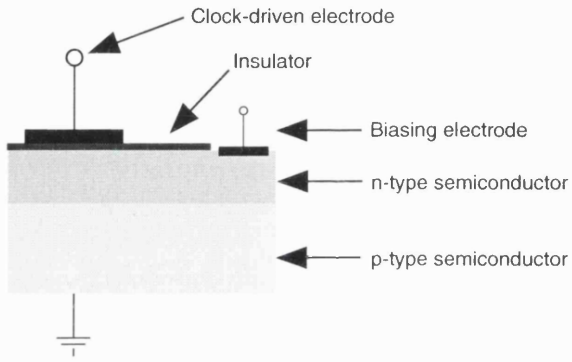


***Figure** 2.5: The arrangement of semiconductor materials and electrodes in a buried channel MOS-cell*

With the biasing electrode in the n-type material made more positive than the clock-drive electrode, a depletion region will form under the insulator in the same way as in the case of the surface channel type. As this region grows, it will cause the formation of a depletion region in the p-type material below. This new depletion region grows in two directions from the p-n junction. Figure 2.6 shows the depletion regions growing out as the reverse bias is increased.

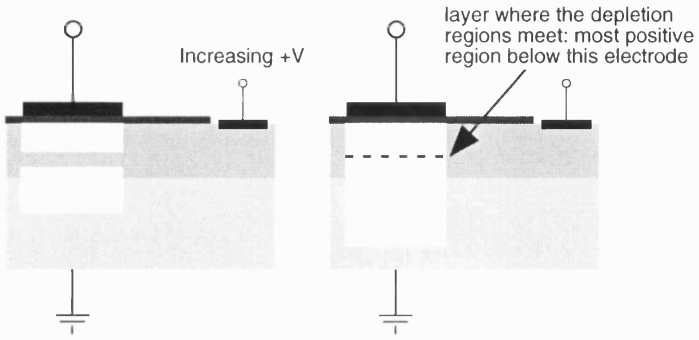


Figure 2.6: Growth of depletion regions in a buried channel MOS-device

The dotted line represents the layer where the two regions meet. Once this point is reached, making the bias voltage more positive will have no further effect, and this region will be the most positive below that electrode. This, then, is the layer where the electrons are attracted to, and being somewhat below the silicon/insulator interface is not subject to the fast surface states. A disadvantage of the buried channel arrangement is its reduced charge capacity: a surface channel type providing 3–4 times more (Howell, 2000).

In terms of drive, interface and control, surface channel and buried channel CCDs behave the same way, but a further key advantage of the buried type is its significantly higher efficiency at the charge transfer process: in even a relatively small CCD the pixel furthest away from the readout node may undergo hundreds of transfers. This quality of 'charge transfer efficiency' will be considered at more length in 2.1.1.

2.0.4 Clock Sequences

Any CCD type will require sequences of voltage changes on the electrodes forming the pixels to move the charge that they contain to elsewhere on the CCD,

and eventually to an output node. Continuing the discussion with the three-phase CCD, a plot of the waveforms on the three groups of electrodes looks like this:

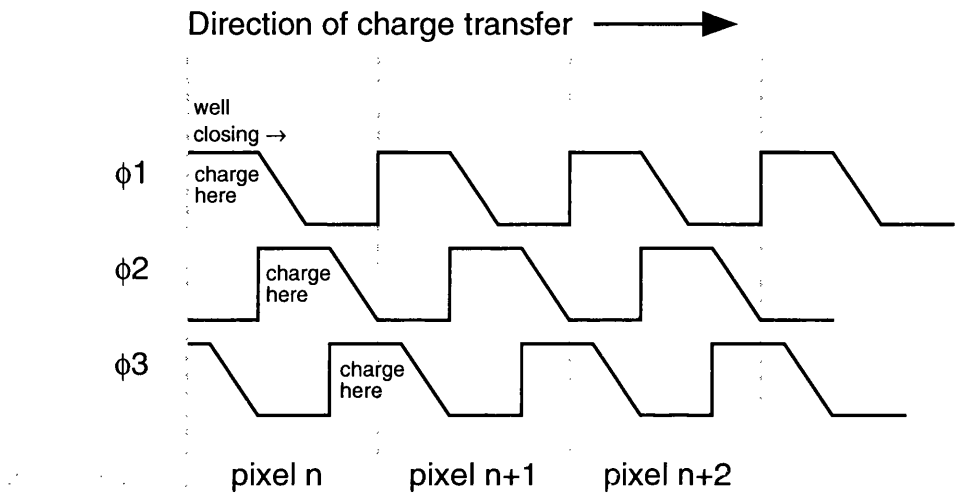


Figure 2.7: *Timing diagram showing the phase relationship between the three electrodes forming a pixel. The rise and fall times of the edges will be 'tuned' according to the behaviour of the CCD in use.*

The timing diagram in figure 2.7 shows how the charge in a certain pixel can be moved along to a neighbouring pixel: the contents of pixel n moving to the position of pixel n+1, while pixel n+1's contents are shifted to pixel n+2, and so on. It is possible, in a three-phase CCD, to shift the pixels in either direction and the motivations for some of the shifting strategies are considered in the following sub-sections. A particular series of signals to achieve a particular shifting pattern is called a 'clock sequence'.

2.0.4.1 Frame Transfer

Mechanical shutters being undesirable in the spacecraft case owing to their limited reliability, the 'frame transfer' CCD is often preferred. In this construction the imaging area of the CCD is joined by a 'storage' area which is of the same geometry, but which is masked from incoming photons.

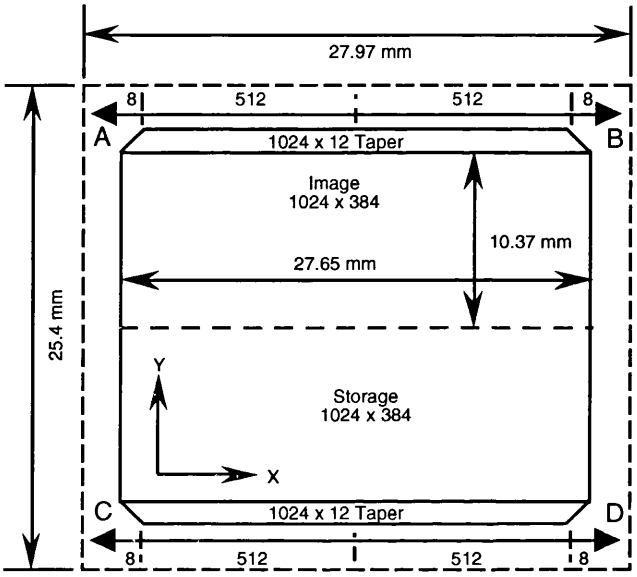


Figure 2.8: Schematic of the CCDs used in the RGS instrument, showing the dimensions, the arrangement of 'image' and 'storage' sections, and the positions of all of the potential readout nodes (A, B, C, D).

Figure 2.8 shows a schematic of the CCDs used in the RGS instrument. After an exposure time, the contents of the image section are shifted rapidly into the storage section, which can then be read out during the following integration period. Since it is the reading out which is relatively slow, the shifting of the completed image can in this fashion be achieved as quickly as possible, minimising the chance of any photons arriving during this operation and adding spurious events to the frame. Assuming for the moment that the CCD is being read out

from a single node, then the readout operation continues after the block-shift from image to storage, by horizontally shifting the bottom row one pixel at a time out of the readout node. Once the row is empty, the contents of the storage section are block-shifted down by one row, and this next row is read out pixel by pixel. To enhance the maximum speed at which the storage section can be read out, and to provide some useful redundancy, the RGS CCDs are read out from two nodes on the bottom row. These nodes are called 'C' and 'D' (see figure 2.8), and each is used to clock out its respective half of the image. This adds complexity to the subsequent processing since one side of the image will need to be reversed before the two halves are rejoined, but it adds the option of changing some of the associated analogue electronics parameters, such as the gain of the output amplifiers. A further advantage of the use of multiple readout nodes in the spacecraft case is the implied enhancement to the reliability, since in the event of failure of one node the entire CCD can be clocked through the other.

2.0.4.2 Windowing and Coordinate Deduction

In order to save telemetry, storage space or processing time, there may be occasions when it is desirable to read out less than the whole area of the CCD. This can be practical if, for example, it is known that a particular feature is in a certain place on the CCD. In this case, then a portion—or 'window'—can be selected, and the clock sequence adjusted such that pixels outside of the window are suppressed, and only those within the window are sent through the readout node. This raises a question about coordinate deduction. Considering the case of a CCD read through a single node, then the analogue output of each pixel can be digitised, such that the data stream becomes a sequence of n-bit energy values. At

some subsequent processing electronics this serial data stream can be converted back to an image of the surface of the CCD by deducing the coordinates of each pixel from its position in the stream. To do this, the readout pattern must be known—i.e. how many pixels there are in each of the x- and y-axes. When 'windowing', care will be needed that since the first arriving pixel from the windowed region will then have a coordinate (0,0) not representative of its physical location, the offset must be added in at some point.

2.0.4.3 On-Chip Binning

A CCD has a certain pixel size determined by the physical size of the construction used to fabricate it. This minimum physical size—27μm square in the RGS case (Bootsma *et al*, 2000)— can by adjusted to be apparently larger by the process of 'on-chip binning'. In this technique, the clock sequence is so modified that the charge from particular groups of pixels is merged, with the result that the CCD appears to have fewer, larger pixels. Though this binning will aggregate the electrons due to dark current, the net effect on signal to noise ratio will still be a marked improvement owing to the reduction in read-out noise. The appropriateness of the size of a bin is relative to the resolution element: in the RGS case with a resolution element of 250μm in the dispersion direction, then a 27μm bin size is massively oversampling, hence the default operation of the CCDs in 3x3 OCB. At 1 electron per physical bin due to dark current, and a read-out noise of 5 or 6 electrons, it is clear that there is a significant improvement in the signal-to-noise performance gained by reducing the oversampling.

Taking '3x3' on-chip binning as an example, nine pixels in a square are merged to become a single pixel. The binning is achieved in this case by conduct-

ing three line-shifts into the readout register, and then three serial shifts into the output node instead of only one.

2.1 Influences on CCD Performance

2.1.1 Charge Transfer Efficiency

Charge transfer efficiency is the measure of the ability of the CCD to transfer charge from one pixel to the next. Efficiency can be reduced through unwanted recombination, or through charge being left behind during the transfer. This 'smeared' charge is not 'lost', but it nevertheless reduces the magnitude of the signal, and will, in the case of a spectroscopic instrument, for example, lead to a worsening of the resolution. Very high efficiencies are critical to the success of the operation of a CCD owing to the large number of transfers necessary: in the case of the CCDs used on the RGS, if read though a single node, then the pixel furthest away from the output node will have to be transferred 1024 + 768 = 1792 times (this is the number of pixels in the x-axis, plus the total number of pixels in the y-axis in the image and storage sections). Clearly losing even a single-digit number of electrons in each transfer will be problematic. The measured CTE for the RGS CCDs is typically better than 0.99999 (den Herder *et al*, 2001). The CTE is not uniform across the surface of the CCD, and in the RGS case there is a small deterioration in this factor near the side edges (i.e. the y-axis) of the CCDs, and this is attributed to stresses put in the structure when the individual CCD chips are guillotined from the wafer on which they are fabricated.

It is not expected that the current CTE performance of the RGS CCDs will remain as accumulating radiation damage will tend to worsen this parameter,

and degrading CTE is something which may call for adjustments to the onboard pre-processing algorithms, and this is considered later in the thesis.

2.1.2 Quantum Efficiency

An important measure determining the sensitivity of a CCD to the photons incident upon it is its 'quantum efficiency' (QE). The overall QE of the CCD accounts for: reflection losses at the CCD surface; losses in the gate electrodes and their insulators; loss of carriers in the silicon substrate and insufficient photon absorption in the CCD material. Absorption in the gate and insulator material can be avoided by presenting the opposite face of the CCD to the incoming flux: this is 'back-illumination'. Care is taken in the fabrication process for back-illuminated devices that the rear face is not positively charged: for example, the process of oxidation on the silicon will cause that surface to acquire a positive charge. There are several methods for dealing with this, the most common of which involves doping the rear of the CCD with a thin layer of boron to make it rich in holes, and a development of this technique is used for the back-illuminated CCDs used in the RGS (van den Berg *et al*, 1996). At X-ray energies, the photon absorption depth increases with energy. As the depletion depth of a typical CCD is about 10μm this gives a lower QE for high energy photons. An enhanced QE for high energy X-rays can be obtained by the use of 'deep-depletion' CCDs. Whereas an 'ordinary' CCD employs silicon with a resistivity of 10–50 Ω-cm, a deep-depletion CCD uses silicon with a much higher resistivity (>1500 Ω-cm in the RGS case), increasing the depletion depth from around 10μm to around 40μm, proportionately increasing the quantum efficiency—a significant factor in the X-ray region.

This relatively complex to fabricate technique may yield to new methods whereby there is a return to front-illumination, where the front surface is coated with a photoemissive layer: this then converts the incoming radiation to a suitable wavelength and implied path length to get the maximum QE out of the CCD below it.

2.2 Sources of Noise in CCD Detectors

An understanding of the sources of noise is important to the techniques which must be employed to minimise their impact. Using the term 'noise' in the sense of unwanted data mixed in with wanted data, the sources of noise can be classified three ways: those inherent to the CCD and its associated electronics; those which result from other extraneous sources of charge in the CCD wells; and those erroneous artefacts left behind by the data processing itself.

2.2.1 Inherent Noise

Employing as it does a number of analogue electronic techniques, a CCD system will be subject to a number of noise sources, and these affect the dynamic range of the device by limiting the lowest signal that can be resolved. For the purpose of this thesis, the key outcomes of these types of noise are that they mean that every pixel will have an energy, even those which have not seen an incoming photon, and that this noise will need to be considered when adding pixels together to reconstruct photon energy split among them. A specific type of noise problem associated with CCDs is that of 'dark current'. At room temperature silicon will thermally generate between 10^{14} and 10^{16} electron-hole pairs per cubic centimetre per second (Beynon and Lamb, 1980 p45). In the case of the buried

channel CCD, then, these majority carriers can be attracted into a potential well and become otherwise indistinguishable from those generated by photoelectric emission. Being a thermal effect this can be ameliorated by cooling the CCD, yielding an exponential improvement since the temperature dependence is the same as that of the intrinsic carrier concentration. If integration times are to be extended, then clearly the dark current must be suppressed. A further complication arising for the downstream processing is that the level of thermal generation is not uniform over the volume of the CCD and this can lead to patterns in the CCD output.

2.2.2 Extrinsic Sources of Noise

In the context of an astronomical detector, extrinsic noise can be considered as the different types of stray light, and charge deposits resulting from ionising radiation.

2.2.2.1 Stray and Diffuse Light

As mentioned in 2.1.2, the CCD is sensitive to a wide energy range, and this must be controlled, as in any astronomical application. In the RGS, optical and ultra-violet light can be focussed or scattered onto the detector array. This means that aside from taking the obvious mechanical precautions against internal reflections etc., some coating must be applied to the CCDs in order to limit their sensitivity to optical/UV photons. This 'bandwidth' limiting of the CCD's photon sensitivity is directly analogous to the precautions typically taken in the design of electronic amplifiers. Allowing for off-axis sources, the effective field-of-view of the RGS is 4°, meaning that it will contain at least one magnitude 4 or 5 star in

every observation, thus optical sensitivity is reduced by coating the surfaces of the CCDs with a layer of aluminium, of variable thickness—ranging from 45 to 68nm according to the position of the CCD in the array (Bootsma *et al*, 2000). The optical throughput of the mirrors decreases with increasing off-axis angle, hence the coating can be thinner in the direction of increasing first order X-ray wavelength on the detector array.

The thickness of the aluminium layer is also enough to entirely remove sensitivity to the diffuse continuum background in optical, ultra-violet and extreme ultra-violet. No problem has been found with stray light in orbit, and restrictions on the maximum optical brightness of a star contained within the RGS field-of-view has been relaxed to magnitude 2.

2.2.2.2 Relativistic Particles

More problematic for the RGS are cosmic rays, a particular feature of which is their ability to leave complex splashes of charge on the CCD such that simple thresholding techniques are inappropriate—if only part of the charge deposited by a cosmic ray is removed, then the remaining components will appear to be isolated, or part of simpler events, and as such will contribute to the general background. As will be seen later, removing this cosmic ray background was to prove critical for RGS.

The ability of a material to remove energy from radiation per unit density per unit path length is its 'stopping power'. The stopping power for silicon for protons with a momentum of 3 GeV/c is 1.6MeV g^{-1} cm^2. In the RGS case, then, where 25μm < CCD thickness < 35μm, this translates to a minimum energy loss of 12keV. Particles with higher or lower momenta will deposit increasing

amounts of energy (levelling off at a factor of two in the former case), implying that energy deposits from these relativistic particles will be easily discriminated by their energy. However, as in the cosmic ray case, these ionisations in the field-free region of the CCD may have their charge spread across more than one pixel, and simply applying a threshold will give an ambiguous result.

2.2.3 Sources of noise due to split- and partial-events

In the previous section it has already been indicated that there is a class of 'noise' whose origin is more properly expressed as a processing error. Important in any data processing, these error artefacts are critical if they result from space-craft onboard processing which leaves no possibility of return to the unprocessed state.

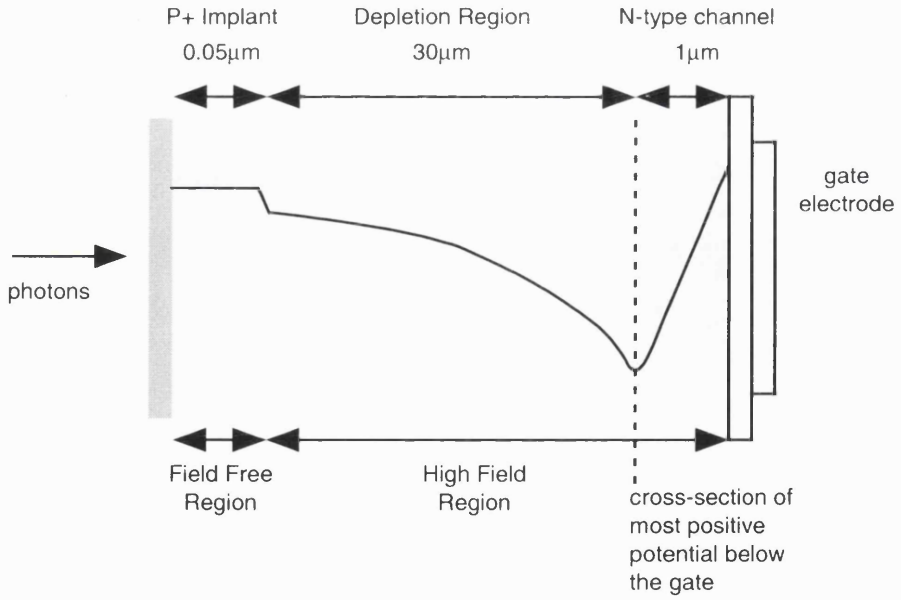


Figure 2.9: Cross-section through a back-illuminated CCD showing the elements which it comprises, shown not to scale for clarity. Also shown is the variation of potential within the CCD.

It can be easily imagined that a cosmic ray or relativistic particle might leave

charge in more than one potential well, but genuine X-ray events may also deposit their charge in this way.

Even in a back illuminated CCD there will be a region where there is no field exerted by the potential applied to the gate electrode. This 'field-free' region is of increasing relevance as the energy of the X-ray photon reduces, since a greater proportion of the photons will be absorbed here. Electrons here will thermally diffuse in both directions and may either get caught in the fielded region and hence drawn to the well, or may reach the oxide layer where they will recombine. Two effects result from this diffusion. The first is that the originally small charge cloud diameter (typically less than one micron) can spread far enough by the time it reaches the depletion region that the charge will be split across more than one well—this is a 'split-event'. Split events can potentially be reconstructed, but care is needed in the decision making process—as shall be seen in chapter 3—otherwise the results will be misleading. The second possible outcome of charge diffusion in the field-free region is that if charge is lost by recombination in the oxide layer then the total energy in the pixel (or pixels) will be below the true value inferred from the energy of the incoming photon (Owens and McCarthy, 1995). This, then, is a 'partial' event.

In a dispersive spectrometer such as the RGS where the pixel energy is used for separation of the spectral orders, there is accordingly less criticality in the precise energy recorded. Partial events, however, being unreconstructable will broaden the peak in the distribution of energies by adding to the accumulation on the low energy side.

2.3 Radiation Damage in CCDs

There are two types of radiation damage to which spaceborne electronic devices may fall prey: ionisation damage and displacement damage.

Ionisation damage may be caused by gamma rays, or charged particles (Jansen and den Herder, 1995), and may be considered simplistically as the deposition of charge in places where it is not wanted. This may be harmless if occurring in a conductive region, but is more serious if occurring in an insulating region—such as those in a CCD's gate structure. Charge accrued in such an insulator will have an effect on the conductivity of the underlying semiconductor material, and will have the effect, in a CCD pixel, of increasing the dark current. Pixels affected in this way may appear as 'hot', i.e. will always contain an energy above the average noise floor, and the effect may be long-lasting or may improve again through mechanisms which are not completely understood (Gabriel, 2001). The effects of ionising radiation are minimised if there is no potential difference across an insulated region (i.e. it is 'off') since the low mobility of holes in an insulator increases the probability of electron-hole pairs recombining within the insulator itself. In the presence of an electric field, the electrons will be more rapidly swept from the affected region tending to leave it more positively charged.

Displacement damage occurs when heavy particles strike the CCD with sufficient energy to move atoms in the semiconductor from their positions in the crystal lattice. The damaged site will form a charge trap, leading to a degradation in the CTE for that pixel (Howell, 2000). Lattice damage tends to be permanent, usually only improved by annealing at higher temperatures. The RGS CCD benches include provision for heating the CCDs to +80°C for this purpose: not as

a regular operational action, but as a potential option for recovering CCD performance late in the mission.

2.4 The RGS Focal Camera

The RFC comprises 9 CCDs of the type illustrated in figure 2.8, arranged as shown in figure 1.8. These CCDs were specially made for the RGS project (Bootsma *et al*, 1996), having their readout register split in two to permit simultaneous read-out from two separate nodes, where there are individually bias-able low noise FET amplifiers. The length of the arc on the Rowland circle implied by the desired ~30Å spectral coverage of the CCD bench is 253mm, and with a pixel size of 27μm the dispersion becomes 30Å/253mm/27μm ≈ 3.2mÅ per pixel.
Each CCD is 1024 pixels wide and is designed to be edge-buttable to minimise the 'dead' region along the dispersion axis, but this still amounts to approximately 0.5mm between each device (den Herder *et al*, 2001a), or about 60mÅ.

The potential resolution of the CCD bench must be considered in the context of the optical resolution also, and this can best be done by looking at the line spread function (LSF). The LSF is a convolution of the responses of the mirrors and the gratings: i.e. such properties as the alignment and flatness of the gratings, the quality of the figuring on the mirrors, scattering on support structures and so on. The precise value of the LSF is dependent in wavelength, but it has been measured to be of the order of 250μm wide in the dispersion direction, and this makes it practical to reduce the CCD sampling rate by operating the CCD in a 3x3 on-chip binning mode (2.0.4.3), thus using virtual pixels 81μm on a side, or about 9.6mÅ in the dispersion axis. Though integrating the quantity of dark cur-

rent, the on-chip binning yields a significant reduction in the read-out noise since there is now only one read-out per 9 pixels.

The design of the on-chip output amplifier is subject to a number of trade-offs between output impedance and read-out speed. For the RGS this amplifier has a responsivity of 4μV/electron and a noise of 4.5 electrons r.m.s. for a 3μs sample time. The final design provides an energy resolution in 1x1 OCB of approximately 160eV at 5Å, improving to 130eV at 35Å. As a direct consequence of the reduced noise content when performing on-chip binning, at 3x3 OCB these figures become 140eV and 100eV respectively. The primary need for photon energy measurement in the RGS is for the separation of the spectral orders:

First order range	5Å (2.5keV)	35Å (0.35keV)
Second Order range	2.5Å (5keV)	18.5Å (0.7keV)
Difference	2.5keV	0.35keV

Table 2.1: *Implication of overlapping spectral orders on the surface of an RGS CCD. The smallest energy difference that needs to be resolved is 350eV.*

and though this requires an energy resolution of 'only', say, 250eV allowing for some margin of error, the design target has been to improve this parameter to its maximum to assist in separating the X-ray photons from the background noise. This works by using an energy discrimination window around the pixels according to their x-position on the CCD surface: this position implies a particular energy value for the result of an X-ray photon from an on-axis source, as will be seen in chapter 5.

2.4.1 Sources of Background for the RGS

A CCD frame from the RFC will show pixels containing energy from a number of sources, even after removing those due apparently only to dark current. Separation of dispersed X-ray photons belonging to the on-axis source from other off-axis X-ray sources, and from altogether different sources is a challenge. Some steps are taken in the design of the RFC and in the flight operations to minimise the number of unwanted events, and some steps are applied in the data processing to minimise their impact. The main contributors to the unwanted background are low energy electrons; medium energy electrons; minimum ionising particles, gamma rays and cosmic rays.

Low energy electrons cannot be discriminated from X-rays, and the primary flux of these is very high (averaging about 10^5 /cm^2/sec.), and they are therefore kept from the CCD bench by magnetic deflection at the mirrors and careful baffling of the RFC's field-of-view. The design of the RGS achieves fewer than 6×10^{-3} of these electrons /cm^2/second/keV at the position of the RFC. However, the majority of the remaining electrons observed in the CCD are the result of gamma ray interactions in the CCD itself, and hence cannot be prevented.

Medium energy electrons may be discriminated by their energy, but secondary emission radiation created by them may not. Fluxes for these are shown in table 2.2. The primary source for secondary emission is fluorescence in the aluminium detector housing, and the housing is therefore plated with gold in order to suppress this.

Energy Interval (MeV)	Flux (electrons/sq. cm/sec./steradian)
0.03 – 0.5	7000
0.05 – 1.0	8.5
1.0 – 5.0	2.1
5.0 – 10.0	0.04

Table 2.2: *Measured medium energy electron fluxes (Danner, 1993)*

Minimum Ionising Particles (MIPs) and Cosmic Rays (relativistic protons, electrons and ions) may deposit energy above the energy band of the RGS. MIPs deposit energy in silicon according to a 'Landau' distribution: this is an asymmetric probability density function having a narrow peak and a long tail towards positive values owing to the small number of collisions, each with a small probability of transferring a comparatively large amount of energy (Landau, 1944). The peak of this distribution is at 0.23 keV/μm, and is, in thin absorbers, broadened by a Gaussian distribution with $\sigma = 0.4\sqrt{d}$ keV, d being the depth measured in μm. This would apparently make MIPs still easy to reject applying the dispersion relation but for the added implications of pixel size and fielded and field-free regions. Recalling figure 2.9, there is a region in the CCD which is not covered by a field, and electrons deposited here will radially diffuse—some to the unfielded back surface, and some to the fielded region below a pixel or several pixels. Adding to this is the possibility that the angle of incidence on the CCD surface may be anything from orthogonal to grazing and hence the particle may anyway leave a trail, as shown in figure 2.10.

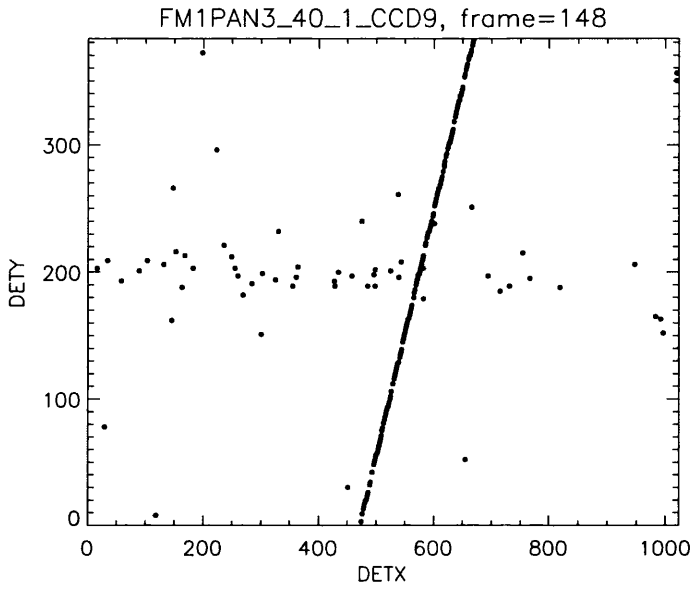


Figure 2.10: *Cosmic Ray track on frame of CCD data taken during ground testing of the RGS flight CCDs. The dispersed X-ray photons from the multi-wavelength target at the test facility can be seen forming an approximate horizontal band in the centre of the frame. Note that the cosmic ray track is not continuous.*

It is possible that some components of a cosmic ray trace will not be above the lower energy threshold set, meaning that after thresholding the cosmic ray track will not necessarily be continuous, as is further shown in figure 2.11.

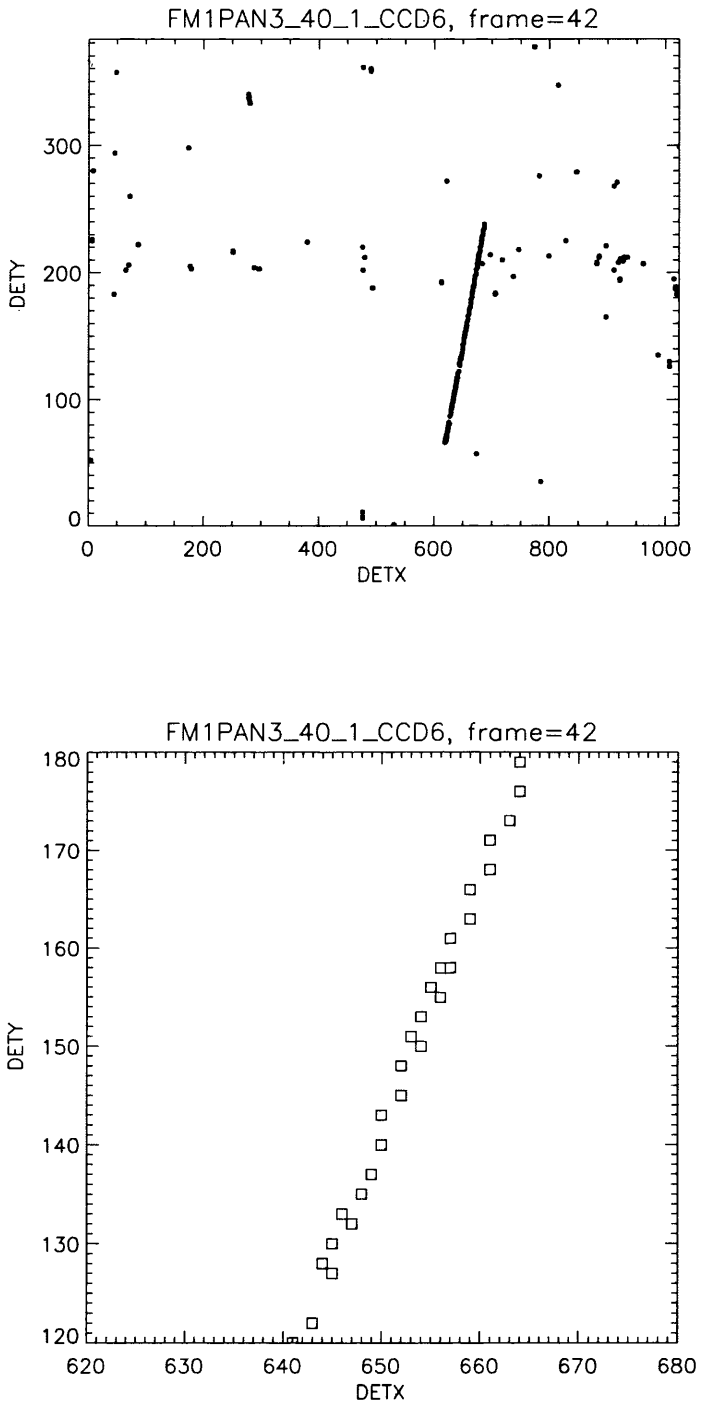


Figure 2.11: *Two views of another cosmic ray track visible on in data taken during the same ground testing campaign.The shorter, apparently more continuous track (top panel) can be seen, on zooming in, to be comprised of a number of pixels not necessarily connected to each other.*

2.4.2 The Challenge of Onboard Preprocessing

A raw, unprocessed frame of CCD data from the RGS RFC is of little direct use to an experimenter. Such a frame will contain:

- pixels which contain only electronic noise
- bad pixels—i.e. pixels fraudulently containing no energy ('cold') or more energy than they should ('hot')
- X-rays from off axis sources—again these may be split or partial events
- X-rays from the diffuse X-ray background
- stray-light from optical/UV/EUV sources within the field of view
- events from cosmic rays or MIPs or medium energy electrons, or secondary emission caused by these
- X-rays from the on-axis source, but which may be only partial events or split events

and the whole frame is likely to need correction for instrumental effects, calibrations and so on. A number of the key science related CCD parameters a listed in table 2.3.

Parameter	Value
eV per electron-hole pair	3.65 (varies slightly with λ)
ADC resolution	12 bits, 0–4095 ADU
ADU to Energy relation	1ADU = 1eV
Channel to λ relation	9.6 mÅ per 3x3 pixel bin
Energy resolution in 3x3 at 5Å/35Å	140eV / 100eV
Typical lower/acceptance threshold values	200/220 ADU

Table 2.3: Summary of key science related CCD parameters

There is, then, a sequence of processing required to select the real information from each frame, and given the telemetry constraints it is necessary to start this processing onboard the spacecraft.The major concern about data removed at this point in the stream is that it is irrecoverable. This means that aside from the general engineering difficulties and constraints on the writing of the onboard preprocessing software, great care is required at every stage to ensure that the algorithms and techniques employed do not compromise the science that can be extracted: i.e. it is not enough just to squeeze as much data as possible into the telemetry stream in order to minimise impact on effective area—the data that remains after pre-processing must be as far as possible unambiguous, and relevant to the experiment.

Chapter 3: The RGS Onboard Software

Following the description of the RGS hardware on chapter one, and the elements of operation of a CCD detector introduced in chapter two, this chapter charts the development of the onboard data pre-processing software ultimately flown on the RGS instrument on XMM-Newton. The chapter opens with a review of the requirements of the onboard data processing software, resolving the functions and processes into the names which were applied to them, with detailed descriptions following. Some discussion is given where appropriate on how the requirements educed, and significant algorithms are detailed with generic flowcharts to aid their employment elsewhere. The functional requirements and the expositions of their detailed design are presented somewhat as if they were all clear from the beginning of the work. This, of course, was not the case as even the requirements matured and evolved over the years of instrument development, but they are described in their final form for clarity.

3.0 Onboard Software Data Pre-Processing Requirements

Certain of the facilities—coordinate deduction, lower thresholding etc.—which the software must provide were clear from the outset, and required by any mode which was defined. Other features, however, needed to be collected into logical groups, and this was done by the author, with the result that a menu of five modes was made available:

- 'Basic Spectroscopy';
- 'Spectroscopy with High Event Rate' (abbreviated to 'HER');
- 'Spectroscopy with Single Event Selection' (abbreviated to 'SES');
- 'Spectroscopy with Split Event Reconstruction' (abbreviated to 'SER');
- 'Spectroscopy with High Time Resolution' (abbreviated to 'HTR').

These modes formalised the general capabilities called for in the "Instrument Modes and Telemetry Rates" document (Branduardi-Raymont, 1996) whose author had sought to itemise the individual techniques which might be employed to match the data production with the telemetry bandwidth. Each of these five modes is described, starting with some further general requirements common to each.

3.0.1 General Requirements and design solutions

Along with the basics of thresholding and coordinate deduction, all of the software processing modes were required to have the ability to:

- reject individual pixels, entire columns of pixels, and regions of pixels, according to data uploadable from the ground segment. This function was termed 'Hot Items Processing' (HIP) by the author, and is available to all of the software processing modes;
- communicate their running/hung status to the instrument controller. This requirement could have been implemented in the DPP hardware, but the need for it became apparent while the hardware design was being finalised for flight, and so a software solution proposed by the

author was preferred. The basis for the idea was that since the software would be constantly writing data to its associated hardware output FIFO, the instrument controller could monitor this FIFO to ensure that it doesn't become empty for longer than a specified interval. At times where there is no pixel data for the DPP to process, the software could periodically write a 'null' pixel to its output. This became known as the 'heartbeat'.

3.0.2 Basic Spectroscopy

In this mode, the DPP hardware front end will have applied a lower threshold to the incoming pixel data streams, supplying to the software section an energy and an associated pixel count. The software must then:

- deduce the coordinates of the pixel;
- reject pixels above a programmed upper threshold;
- reject pixels that are either 'hot', or part of a hot region or hot column;
- keep statistics relating to the numbers of pixels above the lower threshold and above the upper threshold;
- periodically write a null pixel to the output FIFO when idle (i.e. no data arriving at the input FIFO).

3.0.3 Spectroscopy with High Event Rate (HER)

This mode adds to the functionality of basic spectroscopy by allowing 'single pixel acceptance thresholding', whereby a further user programmable threshold can be applied to isolated pixels. Recall that these isolated pixels cannot be part of a split-event, and whether they are the result of electronic noise or partial

events, they are of no use in the spectroscopic analysis and can be discarded in the event that the generated data rate is otherwise too high. Statistics for the number of pixels discarded as below the acceptance threshold are now kept.

3.0.4 Spectroscopy with Single Event Selection (SES)

'Single event selection' spectroscopy is the complement to the 'high event rate' spectroscopy described in 3.03. In this mode, only single pixels above the 'acceptance' threshold are telemetered to the ground segment. This is useful when looking at a very bright source for which there will otherwise be too many pixels to fit the telemetry bandwidth. To emphasise the difference, whereas HER mode transmits all of the data except those isolated pixels below the acceptance threshold, SES mode also deletes those pixels with any kind of neighbour; irrespective of their energy value.

3.0.5 Spectroscopy with Split Event Reconstruction (SER)

Sitting in between HER and SES in terms of the amount of data discarded onboard, spectroscopy with split event reconstruction adds the ability for the onboard software to analyse the morphology of connected pixels, and according to some rules, combine them into a single pixel, thus saving telemetry bandwidth. The upper- and acceptance thresholds are applied to the reconstructed events.

3.0.6 High Time Resolution

Since the seventy-four rows of the y-coordinate will have been compressed by the read-out sequence to a single value, the software processing for this mode

is somewhat similar to that of basic spectroscopy, but only the x-coordinates need to be resolved. Upper thresholding and hot items processing are applied, and only the remaining stream of x-coordinate plus energy values are written to the output FIFO.

3.1 Detailed Design of the Onboard Software

In the earliest descriptions of the DPP subsystem, it was assumed that all of the processing would be handled by hardware. There were various influences on this hardware development, but by early 1995 it had become clear that a processor based solution would be needed to meet the reliability, power, cost and geometric constraints. In short, there were no field-programmable gate arrays available at the time which would meet the quality assurance specifications, application-specific integrated circuits would be too expensive, and there would be insufficient space and power for the requisite discrete logic. With the Plessey MA31750 in use elsewhere onboard XMM, this was chosen. An estimate of whether one or two processors would be needed was performed by Chun, 1996b, from which a final plan emerged to use a single processor design with the front end lower thresholding and counting in discrete logic, leaving the processor at a point in the data flow where it could handle the more elaborate steps without being overwhelmed. There being a local Ada cross-compiler available it was decided that this could be used if it produced fast enough object code, and that this would aid the design cycle.

The availability of an (expensive) Ada compiler occurred since the use of Ada was specified by the product and quality assurance requirements of the project for the instrument controller software. The DPP software could possibly have

avoided this formal requirement, being an embedded system, but it was considered at least worth comparing the Ada generated code with the 'hand-written' assembler. Some experiments were conducted by Chun (1996b) that led him to suggest that the Ada compiler would produce code comparable in size and speed to that produced using an assembler. These experiments, however, did no more than read data from one FIFO and write it to another with no other processing—a task so simple that any high-level language ought to be able to match a low-level one.

Based on this work, and the work of the designer of the DPP hardware (Rees, 1997) the final system was built as the hybrid as described in chapter 1, and using a single processor.

3.1.1 General Considerations

Before looking at the first attempt at a basic spectroscopy, some general points needed to be addressed, specifically: the coordinate system to be used; the parameters that needed to be specified to the software from the ground segment; and a method for identifying the data in the output FIFO.

3.1.1.1 Coordinate System

The first point considered here was whether something other than a traditionally incrementing count might be a good idea: i.e. could some coordinate system be found which would simplify the processing of the data? Various exotic ideas were experimented with, such as considering the centre of the CCD read out area as pixel '0' and counting out from it in a spiral, but no scheme provided a uniformly better system, and most ended up needing more bits to define them,

which automatically excluded them. The final plan, then, was to count the first arriving pixel from a readout node as (0,0) and count from there in the most obvious way, incrementing the x-coordinate until the end of that row, then resetting the x counter and incrementing the y-coordinate by one. For a CCD readout from two nodes, each node's first arriving pixel was declared (0,0) and the onus for any necessary image reversal placed on the ground segment, there being no particular value to performing this onboard.

The nine CCDs onboard were numbered 1–9, with this number called the 'CCD-ID'. CCD-ID '0' was available for use with non-CCD sourced data (e.g. test patterns injected into the data stream), and for use by the onboard software as an error trap (e.g. in the event of some parameters being specified out of range the identity of the CCD under test is set to '0'. This is considered further in 3.1.2).

3.1.1.2 Parameter Bank

A list was drawn up of the parameter words which the onboard software would need—twelve per CCD—and the allocation in memory was rounded up to sixteen words to leave the origin of the entries for each CCD on a convenient binary boundary, while allowing room for some future additions. The final list of twelve parameters was:

- X length, Y length;

  these are used to deduce the coordinates of a pixel from its count;

- X start, Y start;

  these are integers added to the coordinates deduced in the event that a 'window' of the CCD is being read out. This is required onboard to allow registration with specified hot pixels and columns;

- acceptance threshold side C, acceptance threshold side D, upper threshold side C, upper threshold side D;

  these are user programmable thresholds applied to the energies of the pixels under test;

- hot column shift size

  this is used by the hot column rejection routine to match the pixel under test to a particular segment in the column, and will be more fully explained in 3.1.4.1*;*

- hot pixel table start C, hot pixel table start D;

  these represent the origins of the pointers used by the HIP software in rejecting hot pixels specified by the ground segment, and will be more fully explained in 3.1.4.2*;*

- software mode/read-out nodes/HIP enable.

  The specification of the software mode needing only three bits, and the readout nodes and HIP enabling only a further two each, these were concatenated into one word. The software mode, also known as 'science type', only needs reading from one place since it remains unchanged per exposure. For simplicity, the tool used to generate the parameter bank includes the software mode value in the corresponding location for every CCD, but the onboard software only reads it from the location for CCD 1*.*

There are parameter sets for each of the nine CCDs, plus another set to be used with CCD 0, and they are placed in a specified memory location (starting at 7000 hex) by the ground segment via the instrument controller. Thus the onboard software can find, say, CCD 0 X length at 7002 (hex), CCD 1 X length at 7012

(hex), and so on. A small point, but having the parameters organised in a natural way was very helpful during the long design and debugging phase when one was frequently looking at raw memory dumps.

3.1.1.3 Data Tagging

Recalling that the output for this processing software is a FIFO, one word wide, a scheme was needed to identify each data item put into it for the sake of the subsequent processing. Eight different entities were identified:

- CCD Number

  This can have a value from 0–9, so maximum of four bits needed;

- x coordinate from side C, x-coordinate from side D

  two are needed to ensure that the ground segment software correctly performs any mirroring needed. If reading out from a single node with no on-chip binning, then the maximum value could be 1023, so a maximum of ten bits needed (dual node maximum being 511, or nine bits);

- y coordinate

  only one type of y-coordinate, maximum possible value is 384, hence nine bits needed;

- energy

  12-bit analogue-to-digital conversion, twelve bits needed for telemetry;

- statistics

  were the CCD read out from one node, and all pixels counted in a single statistic (for example every pixel rejected as above the programmed upper threshold) then the maximum value could be 1024 x 384 = 393,216, or 19 bits. This exceeds the 16-bit word length, so each of the statistics is split

into two words. Only one tag is used, and the twelve components that form the twelve statistic words are sent sequentially in a specified order. Maximum number of bits per word is therefore ten;

- anomaly

  <5 bits needed for the software to write in the event that it is given specious parameters to work with. As the anomaly loop can be called in several places, an index is included so that the section which is generating the report can be traced;

- null

  no bits needed—this is a null entity written to the output FIFO as a heartbeat to assure the instrument controller that the DPP software is still running. The value in the tag identifies this word to the instrument controller;

- software version ID

  to aid clarity the decimal number is represented directly in hexadecimal, thus aiding the instant identification of this number in the raw hex telemetry: for example the flight version is 19.0, and this is telemetered as 0190 (hex), needing nine bits.

With the maximum bit-consumption being twelve bits, the most significant four bits of each word were free to use as a 'tag'. Accordingly, values were agreed with the author of the instrument controller software, whose code could then identify the nature of any sixteen-bit word it finds in the output FIFO.

3.1.2 Basic Spectroscopy

This mode was composed in 'pure' Ada, with the expectation that as the complexity of the processing increased it would be possible to substitute certain routines with hand-crafted assembler if the speed could be improved by this technique. To make the best start on this, it soon became clear that there were three classes of activity needed: i.e. actions needed for every pixel, actions needed once per new CCD, and actions only needed once per exposure. Clearly, the greatest time saving would be achieved by minimising the number of actions required to process each pixel, and from the outset this was a design priority.

Figure 3.1 shows the highest level view of this structure, and figure 3.2 extends this into a flowchart.

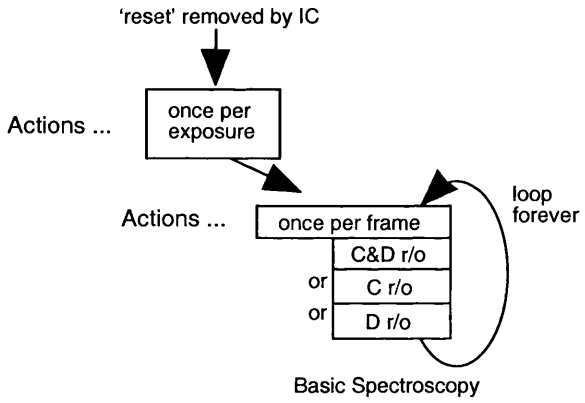


Figure 3.1: Top Level View of the layout of 'Basic Spectroscopy Mode' software.

When idle, the DPP is kept in reset by the instrument controller. When this reset is removed certain one-time-only actions are performed: all variables are initialized to their starting values; the software version number is written to the output FIFO; and the science type is selected. 'Science Type' refers to the mode of

operation, and can be one of the number of species of spectroscopy, or can be high time resolution mode. The arrangement described here was released by the author to be used with the rest of the instrument when testing at the Max Planck Institute's Panter long beam X-ray facility, and was named version 10 (see table 3.1). In this early version only 'basic' spectroscopy is supported, so any selection other than that is trapped and an anomaly loop runs periodically writing the anomaly word into the output FIFO, signalling that operator intervention is needed. There being no need to swamp the telemetry with an error notification, a counter is included to limit the number of writes to one per 50,000 (an arbitrary number to keep the frequency in the order of milliseconds).

Next, the software must wait for there to be some data in the input FIFO, and while in this part of the loop null pixels are emitted to the output FIFO as a heartbeat, with a similar periodicity to that of the science type anomaly.

	Hot Items processing	Basic spec.	HTR	HER	SES	SER
10	–	✓	–	–	–	–
11	–	✓	✓	–	–	–
12	✓	✓	✓	–	–	–
13	✓	✓	✓	✓	–	–
14	✓	✓	✓	✓	✓	–
15	✓	✓	✓	✓	✓	✓

Table 3.1: *Summary of the major version numbers of the DPP software. The scheme was proposed both as a clear statement of what each version could do, and a road-map for the order of the addition of extra features. Updated versions were identified with a suffix, originally an alphabetic character (the final version of '12', for example, was '12g'). A later requirement for the DPP to be able to identify its version number to the ground segment meant that this was exchanged for a numerical character. Versions beyond 15 represent modifications or enhancements to the basic modes as differentiated as above. The version in use onboard the spacecraft as at the time of the publication of this thesis was 19.0.*

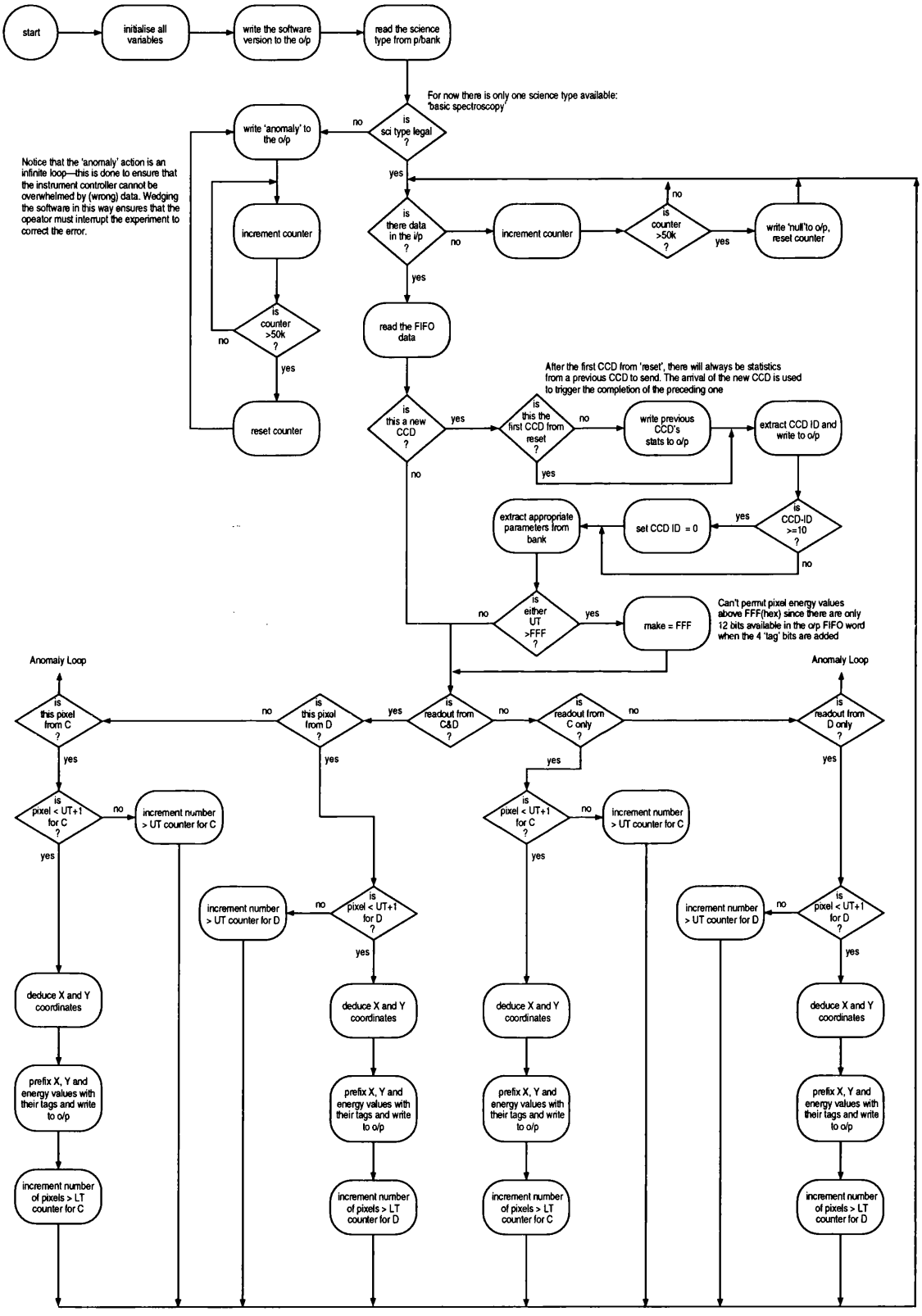


Figure 3.2: *Overall Flowchart of 'Basic Spectroscopy Mode' software*

When there is some data, it is read from the input FIFO to local variables, and the 'start-of-CCD' bit in the status register is checked. If set, then this is a new CCD and the data read from the FIFO contains the number of the CCD (known as the 'CCD-ID'). This number is immediately echoed to the output FIFO for transmission to the ground segment, and is then checked for legality. If found to be out of the range 0–9, the CCD-ID is overwritten to be 0. This is necessary since the CCD-ID is used to select a range of entries from the parameter bank, which is in turn hard-coded into a specific region of memory—reading parameters from elsewhere would not in itself be directly harmful, but would lead to unpredictable results. This outcome is avoided by constraining the CCD-ID range to be 0–9, and requiring that parameters are always loaded for all ten CCD parameter sets. Echoing the untouched CCD-ID to the ground segment gives the operator a chance to tally the ID with that programmed into the instrument controller, since that is where the error will have occurred.

The science type won't be read again till the next reset, and the code will loop forever reading data from the input FIFO. The incoming CCD-ID is used as a trigger to cause the statistics of the previous CCD frame to be output to the instrument controller, but for the first CCD readout after reset there will not be any previous readout, so this is also tested at the top of the loop. Having verified a legal CCD-ID, this number is then used to index into the parameter bank to pull out the necessary parameters, as detailed in 3.1.1.2. The upper threshold settings are checked since they must not permit an energy value above 4095, this being twelve bits, the other four in the word being needed for the data tag. Of course the digitisation of the pixel energy is anyway performed to twelve bits, making this step potentially redundant; however, in later versions of the software when

event reconstruction would be possible, the software would be able to internally assemble pixel values above twelve bits, so the test was included from the outset—it was a design philosophy that each subsection of code would be developed as if for the final version wherever possible. Note that the threshold is 'less than FFF(hex) +1': this is because the upper threshold is always applied (defending the output FIFO as mentioned, and saving cpu time switching it on and off), but one may potentially want to pass FFF(hex) as a legal value in exceptional circumstances.

According to the settings in the parameter bank for which readout nodes are in use, the software moves the current pixel down one of four similar chains (i.e. side C only, side D only, side C with both nodes read, side D with both nodes read). The next test is on the energy value, and this is done now since there would be no point expending cpu time extracting the coordinates if the pixel is above the upper threshold. If rejected, the number-of pixels-above-the-upper-threshold-side-C (or D) counter is incremented and the code loops straight round to collect the next pixel. If the current pixel is below the threshold, then the coordinates are deduced and the three entities (x-coordinate, y-coordinate, energy) are tagged and written to the output FIFO. Finally, before looping round for the next pixel, the number-of-pixels-above-the-lower-threshold-side-C (or D) counter is incremented.

Referring to the basic spectroscopy flowchart in figure 3.2, there are several details worth drawing out about the layout chosen. Firstly, each anomaly loop is separate and where there are arrows pointing to 'anomaly loop' the flowchart symbols have been omitted for clarity. The pattern of each anomaly loop is the same, but they each have their own code rather than pointing to a 'master' anom-

aly handler for two reasons: firstly to minimise the number of conditions that need to be evaluated on a per-pixel basis, and secondly to give the opportunity for each occurrence of the anomaly loop to have its own index—thus anomaly '1' indicates that an illegal science type has been selected, anomaly '2' indicates that no readout nodes are selected, and so on. Next detail: it can be seen that once it has been determined whether both nodes are being read from, or only one, the chain of operations down, say, side C node when reading from both, or side C when only reading from side C are identical. Despite this, they each have their own code because it avoids extra decisions in the program flow, and hence saves time. Making code 'inline' in this fashion is a common technique for maximising execution speed. Also, it can be seen that there are no interrupts in use, and that the code runs as a single process. It happens that the author has extensive experience in interrupt based multi-tasking software, gained in the design of the MSSL parts of the EGSE system for RGS, but despite this and the fact that Ada supports multi-tasking, the author chose to start with a single task polling based loop structure owing to the simplicity in forecasting the load on the cpu. The onboard software must of course be crash proof, and to this end it was a design goal to make the software care as little as possible about the data rate, while guaranteeing that once data is accepted by the DPP it is not lost. So, there being an interrupt available on the input FIFO indicating that it is half-full, one approach would have been to have the software read a block of data in on receipt of that interrupt and spend the time between interrupts processing that block. In the circumstance, however, that the incoming data rate exceeds the rate at which the data can be processed, it can be seen that the processor could end up being overwhelmed by servicing this input interrupt. A simpler philosophy was to posit

that data in the input FIFO and any internal software buffers is safe, and that any excess data can 'bounce' at the input causing only an increment to the hardware 'lost event' counter. A similar approach occurs at the output side. The output FIFO is emptied by the instrument controller at a rate dependent on the rate at which it can ultimately deposit the data on the spacecraft bus. So rather than wait for an interrupt to tell the DPP software that there is room in the output FIFO, at points where the software has something to write, it polls the output FIFO full flag, and if the FIFO is full it waits until there is room. At first sight this seems wasteful if the waiting time could be used for performing elaborate data reduction steps, but this was approached from the other direction. Since the output rate from the DPP is related to the telemetry allocation, there is an implied limit to the number of pixels that can be sent from the DPP, and if the intermediate processing algorithms are demonstrably fast enough to process that maximum number relative to the input which would generate it, then if the input rate exceeds that capacity, the engineering solution must ensure that data already collected cannot be lost.

In effect the telemetry allocation to the instrument has an influence on the effective area of the telescope (Welch, 1996a): if the settings of the instrument are such that too much data is generated, then eventually the instrument controller will have to stop taking data from the DPP output FIFO while it waits for the spacecraft data bus, and when this output FIFO is full the DPP software will be stopped from reading the data from the input FIFO, and when this is full any further data that arrives is lost.

3.1.3 High Time Resolution Mode

The processing of this mode is similar to that of basic spectroscopy, differing only in that the y-coordinate is not extracted, and that statistics are not kept. An implementation of this mode was produced—version 11 released September 1997—formed as an extension to version 10, see figure 3.3.

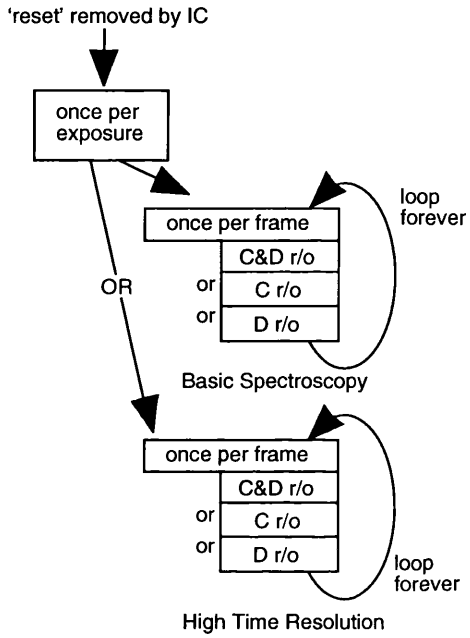


Figure 3.3: *Top Level View of DPP software version 11 showing that there is now a choice of science types, and that once selected a science type is not changeable.*

Now the value of 'science type' has a real meaning. The first lines of the software are unchanged, but there is a branch pending on the choice of high time resolution or basic spectroscopy. Figure 3.4 illustrates the program flow for HTR mode. Note that the first part of the diagram is the same as figure 3.2, shown in grey. There is a decision made according the science mode selected, and then once in the HTR loop the flow is as for basic spectroscopy, but with

the simplifications already mentioned (no y-coordinates, no statistics kept). Note that version 11 of the software adds the HTR functionality as inline code: were the individual sections of code themselves made switchable rather than the whole routine, then there would be extra decisions on a per pixel basis, which is the opposite of the intention. Take the section where the coordinates are deduced: this happens for every pixel, therefore were the extraction of the y-coordinate made dependent on the science type, then this would require a condition to be evaluated for every pixel.

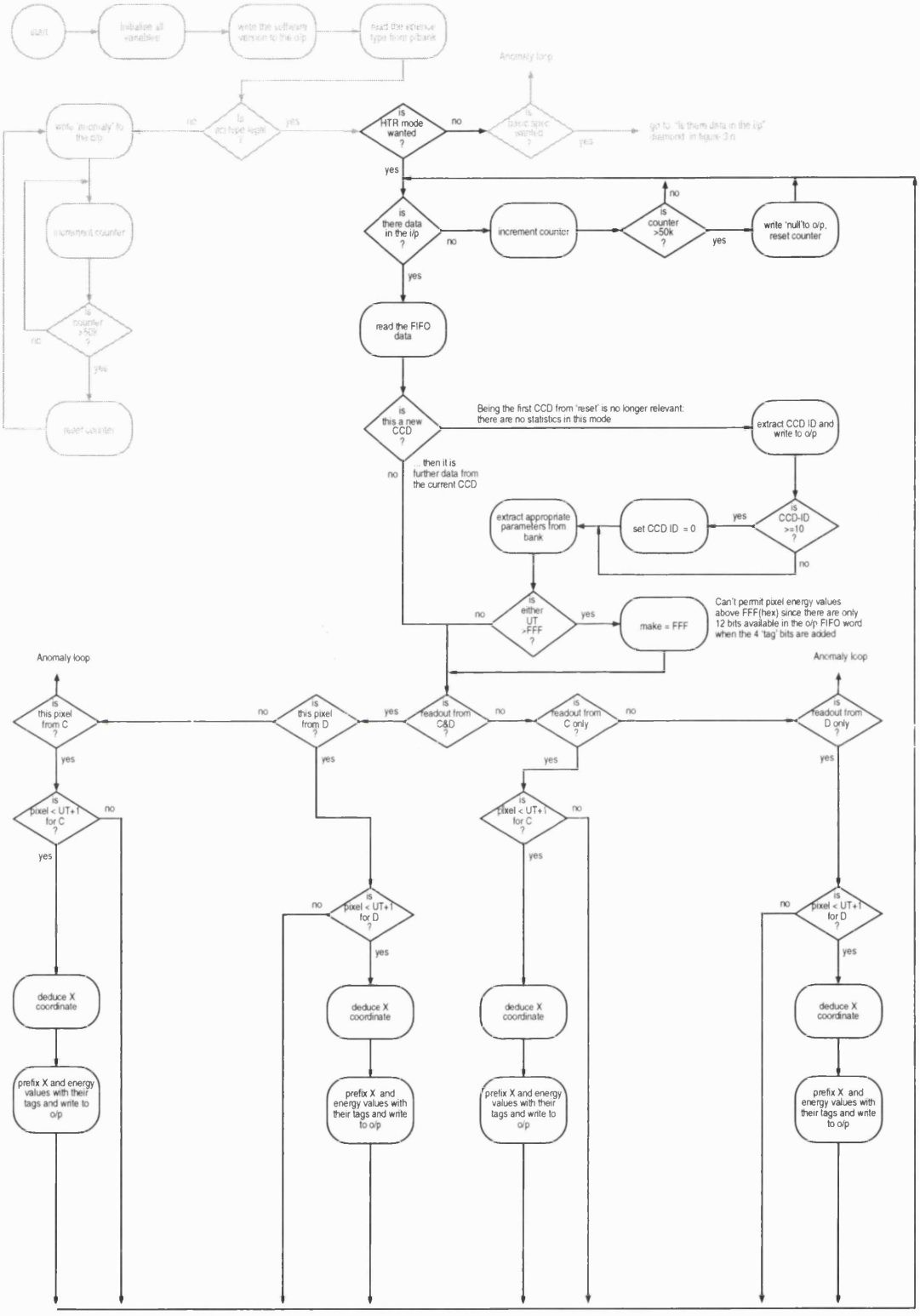


Figure 3.4: *Overall flowchart for HTR mode.*

3.1.4 Hot Items Processing

For RGS, bad pixels are identified by looking at queue memory images, or by noting the appearance of specific pixels in several contiguous frames. Once identified, hot pixels can be selected by the ground segment for removal on-board. Since the act of clocking the image section of the CCD into its storage section tends to smear the energy in the y direction, many of the hot pixels are repeated in a column. Amongst the requirements for the software in the DPP is that it be able to remove:

- individual hot pixels;
- hot columns;
- hot regions.

Collectively this function is referred to as 'hot items processing' (HIP; Welch, 2000a).

3.1.4.1 Hot Region and Column Removal

Since 'regions' are, in the case of this instrument, rectangles parallel to the edges of the CCD, it can be seen that if the columns can be broken into segments, then region removal can be accomplished by marking adjacent column segments as 'hot'. To achieve the removal of these segmented columns, a 'hot column table' is loaded into a fixed area of the DPP's memory. The DPP software considers each CCD's table as an array of 1024 x 16 booleans. Each x-coordinate, then, has its own word in the DPP's RAM, and the y-coordinate is used to select one of the bits of the 16 by a method described later. Each column can be divided in to a number of segments—up to a maximum of 16 (because there are 16 bits in the word allocated for this). Since these tables are

not expected to be changed often, no management of the tables happens onboard: the complexity of changes to the table structure is completely removed to the responsibility of the ground segment. As far as the onboard software is concerned, CCD 1 node C always starts at the beginning of the array, with node D 512 columns along, and CCD 2 side C at 1024 along and so on. Any changes of readout parameter that would imply a need to move these points then obliges the ground segment to upload new tables with the appropriate changes. This simplification is very important to the streamlining of the DPP software's processing load.

So the x-coordinate, the read out node and the CCD number are used to work out the entry in the array, but what about the y-coordinate? According to a parameter selected by the ground segment (the 'hot column shift size', an integer taken from the parameter bank), the y-coordinate is shifted to the right to leave a maximum of four bits, and this 'coarse' value of y can now be used to select which bit out of the 16 available is to be tested. Note that the actual length of the segments is a power of 2 while the range of the y-coordinate is up to 383 (384 pixels counting from 0), so there may be some unused segments, or there may be one segment which has fewer pixels than the others—this all depends on the readout parameters used.

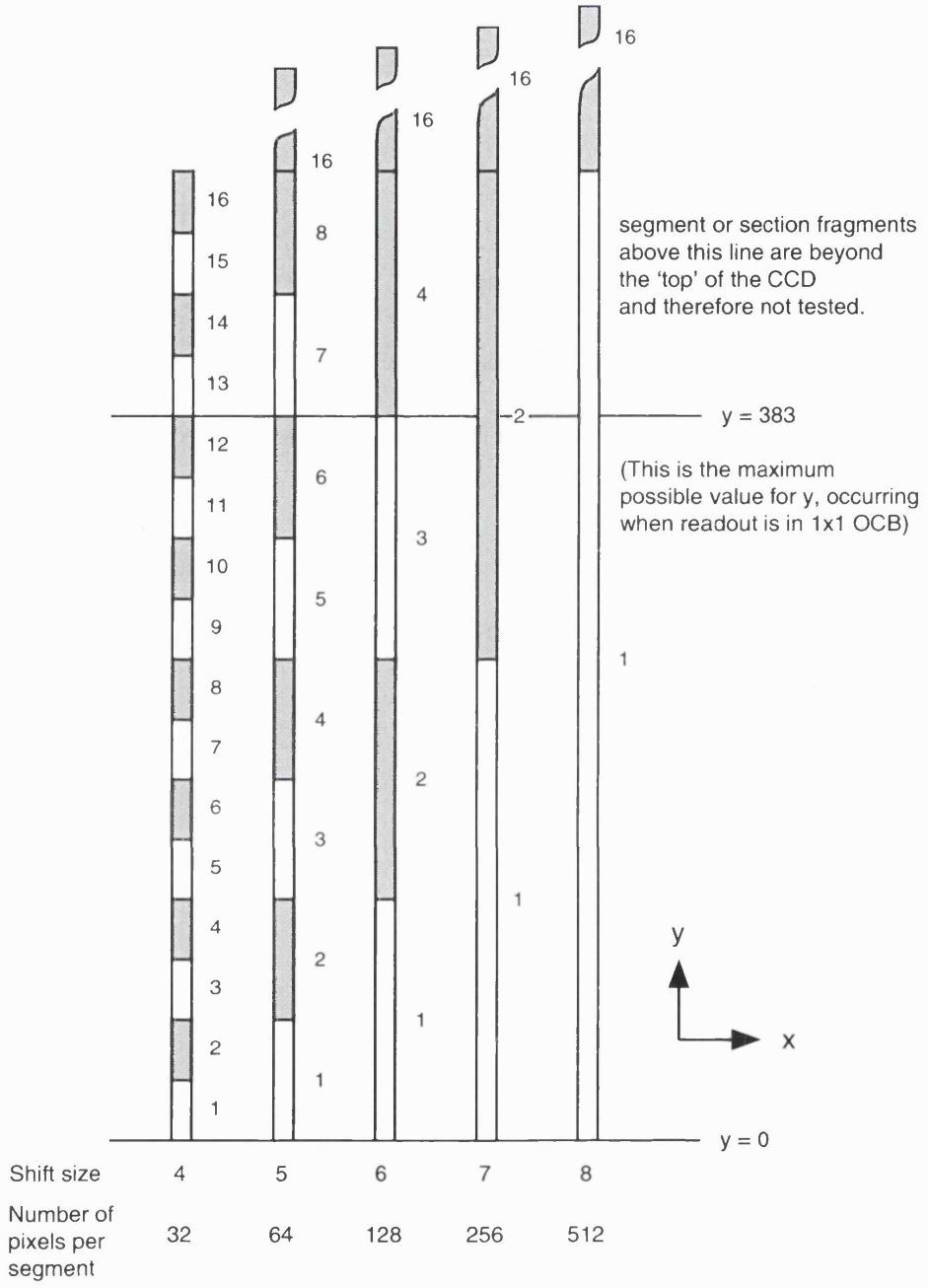


Figure 3.5: *Pictorial representation of the implication of the segmentation of hot-columns.*

The user specifiable value for this shift is called the 'hot column shift size', and ground software used to build the hot column tables can vary this to give the best fit of a column segment around a number of pixels to be removed: it can be the case that the placement of fewer, larger segments gives a better ratio of rejection of wanted/unwanted pixels.

An example is useful here. Assuming that the entire CCD is being read out, the maximum value that y can take is 383, implying that nine bits are needed for the y-coordinate. 2^9 is actually 512, and if the y-coordinate is shifted to the right by 5 bits, the remaining (most significant) 4 bits yield 16 values. 512/16=32 pixels per segment. For 384 pixels, there will therefore be 12 'full' segments and the remaining four will be beyond the 'top' of the CCD, as illustrated in figure 3.5.

In the general case, then, for a given maximum y value, 2^n bits are used, where n is just large enough to give equal to or greater than the biggest value of y:

$$2^{(\text{No. of bits in y} - \text{hot column shift size})} = \text{divisor}$$

$$2^{(n\ /\ \text{divisor})} = \text{number of pixels per segment}$$

In the RGS case, a '1' indicates that the pixel is a member of a segment of a hot column, and should be rejected. To reject a whole column, then, all 16 bits would be set to '1', and region rejection is accomplished by using adjacent column segments.

Figure 3.6 illustrates the subroutine that is used to remove the hot column segments. This function is written as a callable procedure, and the variable

'verdict_KEEP' is a boolean which the calling program later tests to determine whether to continue the processing on the supplied bin.

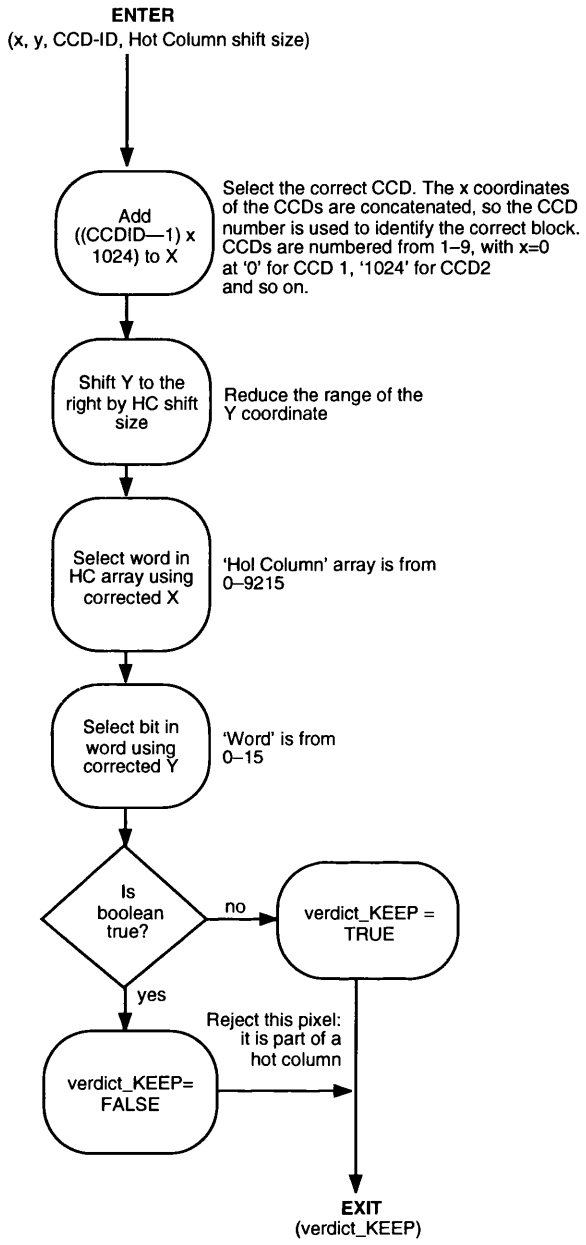


Figure 3.6: *Flowchart for the operation of the 'Hot Column' processing routine. Using information supplied by the calling program, this routine determines whether the pixel under test is part of a hot column or hot column segment and returns a boolean set to the appropriate value.*

3.1.4.2 Hot Pixel Removal

The removal of specific single pixels from the data stream presents a different challenge. With no constraints on time or memory, one approach would be to have a look-up table for every pixel, but this is far too memory intensive for the RGS DPP:

9 (CCDs) x 1 word (x-coordinate) x 1 word (y-coordinate) x 1024 x 384

= 3.38 Mwords

A better approach for this case is to have a table containing only the coordinates of those bins which are to be removed, and listing them in their proper arrival order (see figure 3.7).

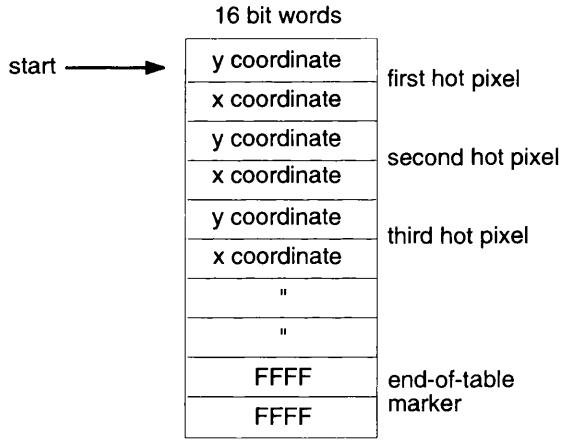


Figure 3.7: *Format of the 'Hot Pixel' table. The arrow represents the s/w pointer selecting a value from the table. The pointer is advanced to the next x-coordinate when there is a y match, and then either to the next y or back to the current one according to whether there was a match on x. The 'FFFF's are the end-of-table markers, and consist of a hexadecimal value beyond that which a legal x- or y-coordinate could reach.*

Giving the software the origin of the table for each CCD means that the tables can be of various sizes, and that for a given memory allocation the tables of different CCDs can be of different sizes. This makes better use of a given quantity of memory. A further refinement in this case to save speed and remove the necessity for range checking is to put a marker at the end of the table so that the end of the data can be identified automatically.

Again this can be a callable routine, and a first approach to this technique is illustrated in figure 3.8.

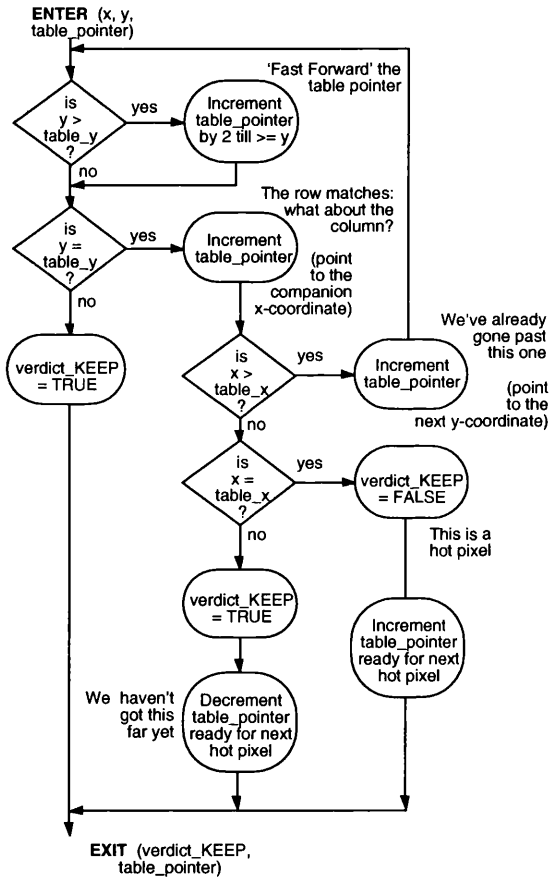


Figure 3.8: *Flowchart for the operation of the 'Hot Pixel' routine. This routine compares the coordinates of the pixel under test with those selected from a lookup table by a pointer, and includes all of the necessary pointer housekeeping to ensure that the selected entry is synchronised to the position on the CCD of that pixel.*

At the start of the routine a pointer is selecting a y-coordinate stored in the hot pixel table (figure 3.7). Now, since it is the arrival of the hot pixel that will be triggering the incrementing of the pointer, there needs to be protection against its non-arrival, so if the current pixel under test has a y value greater than that in the table, the pointer is fast forwarded until the value is the same or greater. If there is no match, then the routine is complete, but if there is a match, then the table pointer must be incremented. The pointer is now selecting an x-coordinate, and this is again tested to see if has been passed. Notice how this test includes a need to return to the y-coordinate test that the x value has not been passed. Finally, if there is a y-coordinate and an x-coordinate match, the pixel can be marked for rejection and the pointer incremented ready for the next arriving pixel. The value of FFFF (hex) is chosen as an end-of-table marker since no y-coordinate will have a value above 180 (hex): the first test will always yield a non-match and the routine will naturally exit by the quickest route.

Protection against the non-arrival of a hot pixel is included since it is possible for a hot pixel to recover itself over time, or even to flicker such that it occasionally dips below the front end lower threshold and hence never reaches the HIP software module. Were a hot pixel not to arrive, then the absence of any pointer housekeeping would lead to all of the hot pixels following the non-arriving hot pixel being kept because the algorithm would otherwise never catch up. Since this pointer housekeeping implies an overhead on every pixel test (that is including the many tests where the pixel is ultimately kept), a way was sought to reduce this load.

By the time that the HIP module was being designed (third quarter 1997), several very realistic tests had been performed with integrated RGS instruments

at the Max Planck Institute's 'Panter' long beam X-ray testing facility, and so there were plenty of representative data sets to look at (de Vries, 1999c, den Boggende 1998). Study of this data showed that for the CCDs seen by this time there was never more than one hot pixel on a given row.

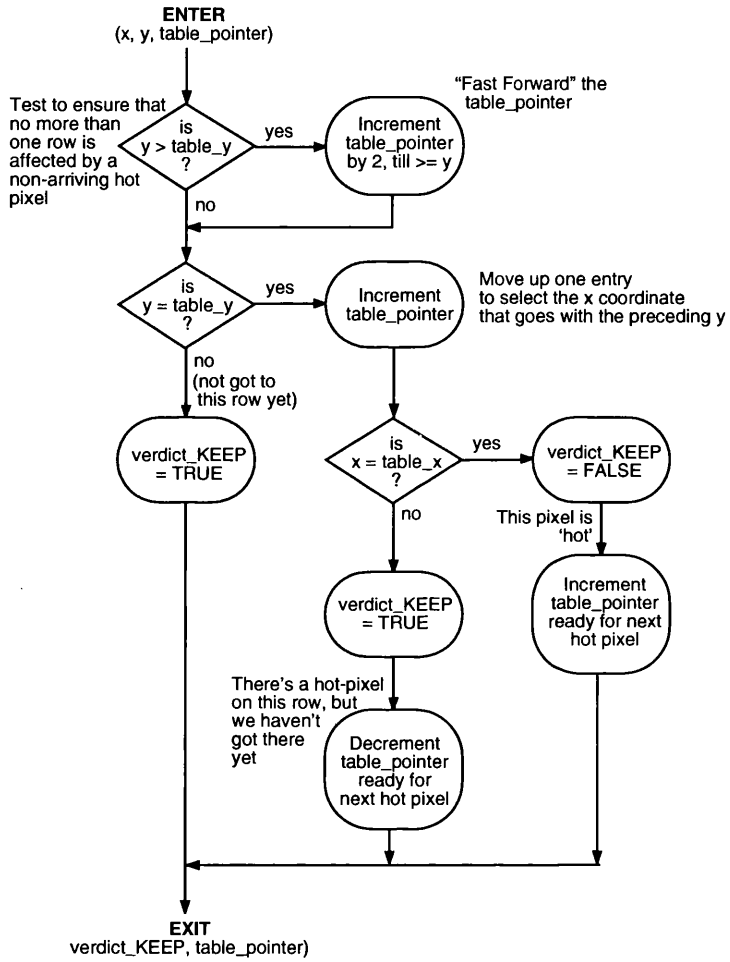


Figure 3.9: *Revised flowchart for the operation of the 'Hot Pixel' routine. In this version the pointer housekeeping is simplified and now only tests for the current row of the pixel under test, allowing pixels on the same row to be at risk of being missed, but significantly reducing the per-pixel cpu load.*

This is a useful result because resynchronising on every line is relatively simple, but resynchronising on every pixel on every line is very cpu intensive: recall

that in any pointer housekeeping strategy there is an overhead of processing even when a pixel does turn up.

Figure 3.9 illustrates the compromise chosen for the final version. There is only a test for the value of the currently selected y-coordinate compared with the y-coordinate under test. The maximum impact of a non-arriving hot pixel is that the following hot pixels on the same row will not be rejected.

3.1.4.3 Hot Items Processing: Overall Disposition

Version 12 of the onboard software including this functionality was formed by amending version 11, inserting a call to the hot items testing immediately after the coordinate deduction is completed, saving processor time in the case of rejection since no other operations will be wasted on a pixel for deletion. A requirement was placed on the author to make the calling of the hot items processing switchable. This being contrary to the target of minimising per-pixel processing, it was opposed but overruled and was included. (The per-pixel load could be avoided only by cloning the two modes to produce two more with the HIP selected, but this would be unreasonably memory intensive with so much software still to write). Switching the HIP on and off was unnecessary since with the tables set correctly nothing would be removed this way. As things turned out, however, with the flight CCDs all having hot pixels from new, the HIP deselection was never used, but the author found it convenient to save having to load the hot pixel and column tables during the testing phase... The new layout of the code is shown in figure 3.10.

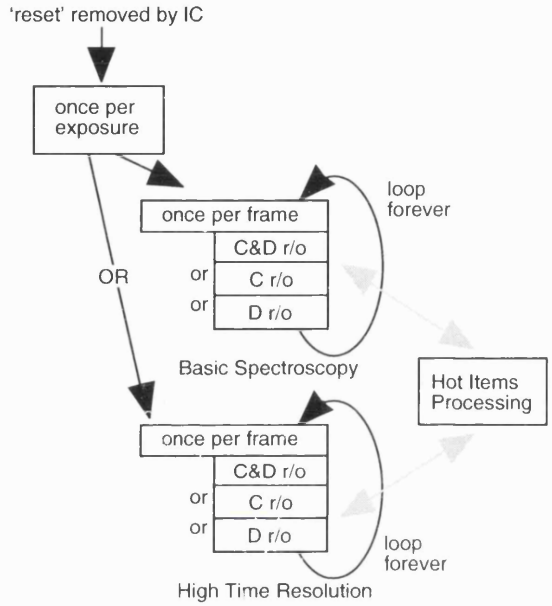


Figure 3.10: *Top-level schematic for the code layout in software version 12. Hot items processing is now available to the two science type processes, with the same code executed by both.*

Figures 3.11 and 3.12 on the following two pages show the flowcharts for basic spectroscopy and high time resolution, with the symbols from 3.2 and 3.4 shown in grey, and the new calls in highlighted in black.

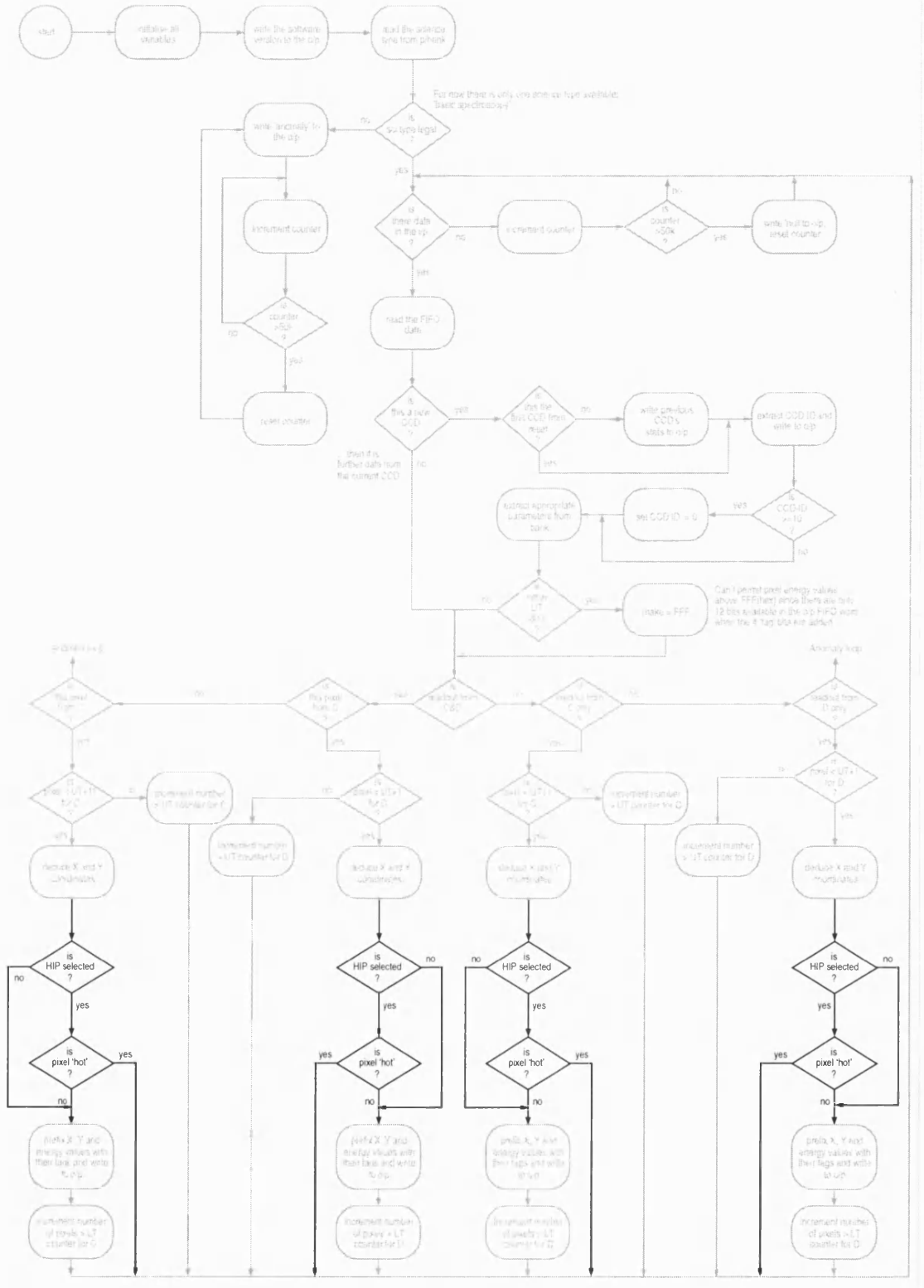


Figure 3.11: *Flowchart for basic spectroscopy, now incorporating hot items processing*

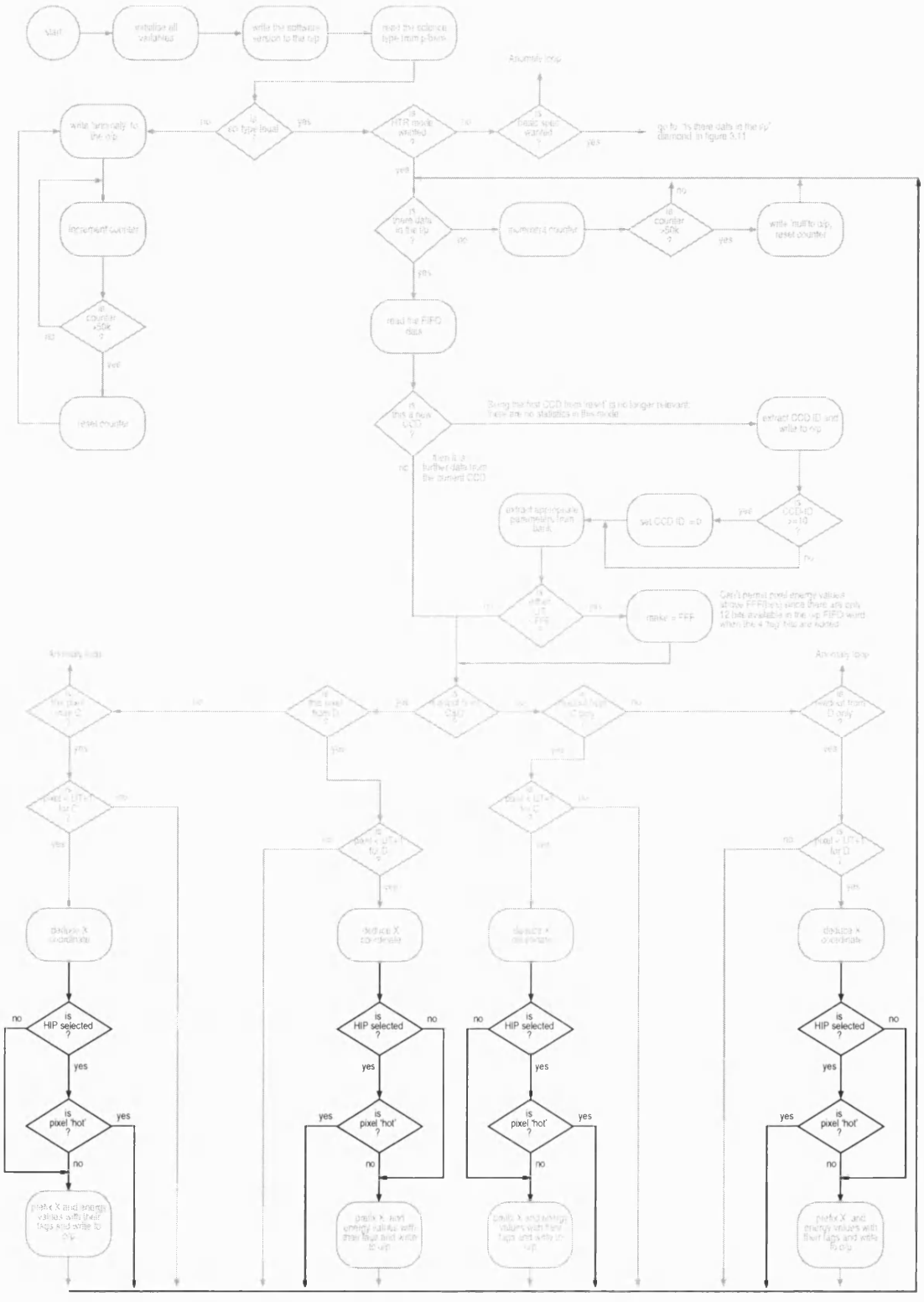


Figure 3.12: *Flowchart for HTR, now incorporating hot items processing.*

The final switching arrangement for the HIP module is such that all of the processing is selected or none, though sides C and D from the CCD can be switched independently. Inside the HIP module the hot column processing is executed first since the expectation is that many more pixels will be classed this way than as isolated hot pixels, and the hot pixel routine is only called if the pixel is not already rejected as part of a hot column or hot column segment.

Special consideration occurs within the calling software for high time resolution mode. Since each spectrum is compressed into a single line, any hot pixel or hot column will result in a hot entity at the implied x-coordinate. How can this be resolved by the onboard code without needing a separate HIP module to account for this lack of a y-coordinate? The answer is to provide the HTR program with a hard-coded y-coordinate, FFF(hex), and hot column shift size (8). The values chosen place the pixel at the furthest extent of the y-range, and in the sixteenth column segment, and only hot column processing is called. The onus is then once again on the ground segment that the correct hot column table is loaded, but with the feature that when operating in 3x3 on-chip binning mode—which is the default—the sixteenth segment is not used for anything else, and can be 'overloaded' by also containing the correct entries for HTR mode. These entries are formed by marking the sixteenth column for rejection if there is either (a) a hot column segment or (b) a hot pixel anywhere below it. Use of the sixteenth segment in this way only simplifies the construction of the hot column table in a few circumstances, and the operator must always ensure that an appropriate table is loaded for the observation to be conducted. Once again, the design of the code is such that it won't fail if supplied with an inappropriate table, but the rejection results will be unpredictable.

3.1.5 Spectroscopy with High Event Rate

Recalling the discussions in chapter one on the use of energy thresholding to remove data, and from chapter two about the potential for charge from high energy events to be diffused into more than one pixel, it can be seen that especially where the energy window around a particular X-ray energy is at the lower end of the energy scale, care will be needed in the placement of the DPP's front-end lower energy rejection threshold if fractions of reconstructable pixels are not to be erroneously discarded. The approach to HER mode is to verify that a particular pixel is isolated, and if so to apply a second low energy threshold, thus allowing somewhat more finesse in the treatment of the overall lower thresholding. This second low threshold is called the 'acceptance' threshold, and to give an idea of scale the typical settings on the RGS are around 200 for the lower threshold and 220 for the acceptance threshold (recalling that the maximum value for the 12-bit digitiser is 4095), with these numbers representing 3σ and 5σ of the system noise peak respectively.

The new requirement of HER mode, then, is that it must be able to distinguish between connected and unconnected pixels. To distinguish between the term 'pixel' which is synonymous with a physical or virtual 'bin' from the CCD surface, the term 'event' is applied to situations where the information is referenced. So a pixel is always one 'bin', but an incoming X-ray (or other high energy photon or particle) generates an 'event'. In this section we consider an algorithm for assessing the connectedness status of a pixel and how it can be folded into the existing code layout.

3.1.5.1 Connection Detection Algorithm

Firstly we must be clear what constitutes a connection. Consider figure 3.13:

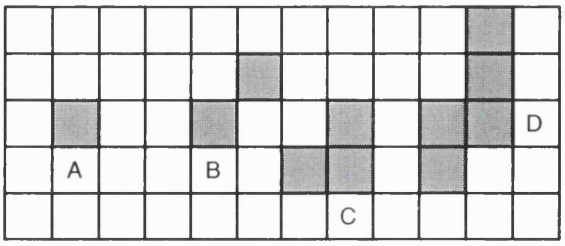


Figure 3.13: *Surface of a 12 x 5 CCD showing four of the (many) possible dispositions of charge, shown in the shaded squares. For HER and SES modes only the fact of a connection or not is relevant. For modes where split event reconstruction is conducted then there is further consideration given to the shapes of the events, and these are discussed in more detail in 3.1.7.*

Item 'A' is a 'single event', an easy classification owing to it being surrounded by a 'moat' of empty pixels. 'B' shows two pixels connected only diagonally: returning to the idea of the way that the charge can spread between pixels, these are not considered to be connected. All of the pixels in 'C' and 'D' are clearly connected. In all of the discussions in this and following sections about how the searching around the pixel 'map' is conducted, the read out is considered as the bottom left most pixel, and the direction from right to left.

The logical schematic of this is shown in figure 3.14. In this diagram the first pixel to be read out will be (0,0), having the index '1'. An important point to observe here is that the search scope for a pixel under test can neglect some of the area as it will have been already covered.

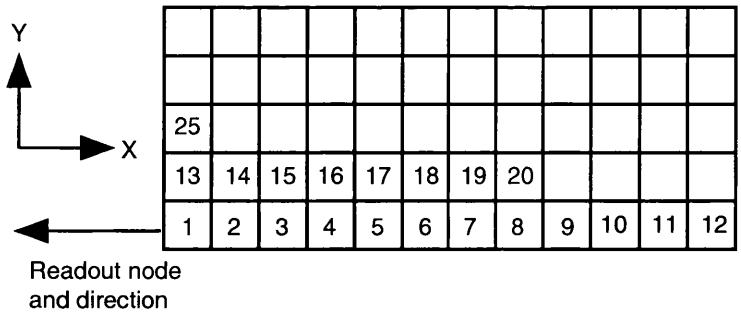


Figure 3.14: *Surface of a 12 x 5 CCD showing the readout node (bottom left hand corner) and the direction (right to left).*

Consider pixel '4' in figure 3.14. If a log of connections found is kept with each pixel then the positions 1, 2, 3, 13 and 14 can be disregarded as they will have been checked before, and 6 and 18 can be disregarded as they will be checked at a later stage. The general case for this is shown in figure 3.15.

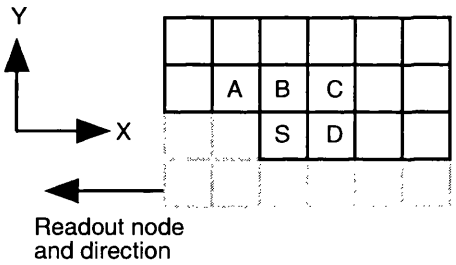


Figure 3.15: *General case for the search scope inferred by the need to identify whether a pixel is single or 'connected'. 'S' is the 'seed' pixel, or 'pixel under test. The grey boxes are previous seed pixels. The naming and positions of the adjacent pixels is chosen to harmonise with that needed later in the split event reconstruction discussions.*

The seed pixel, 'S', can be considered isolated if:

- there is no pixel in position B;
- there is no pixel in position D;
- there is no inherited connection from a previous pixel.

A routine to perform a test for pixel connectedness will need to keep a log with each pixel, and this is achieved in the RGS software by storing the pixels in an array structured like this:

pixel.energy

pixel.y_coordinate

pixel.x_coordinate

pixel.connections

The 'pixel.connections' field needs no internal structure and can simply be an integer incremented whenever a connection is found. For the purposes of this HER mode the test on this field need only concern itself about whether the value is zero or not.

How the routine is fitted into the overall structure of the code will be considered in more depth, but for now consider an array of pixels that can be treated for the high event rate requirements. At the start of the routine 'S.connections' may or may not already be zero, but the tests around it must still be executed to ensure the propagation of inheritance. First connection test is for the pixel at location 'D' since we know from the readout direction that this is the next pixel in the buffer. To aid the description of this, consider a small CCD such as figure 3.16.

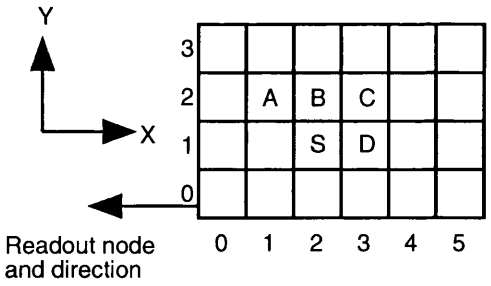


Figure 3.16: Small example CCD, four rows by six columns. There are energies at (2,1), (5,1), (2,2), and (4,2). The pixels shown in white are 'empty' (they will have been removed by the front-end lower threshold).

This CCD has a maximum of 24 pixels, but some of these have been removed by the front-end lower threshold and only four remain, as described in table 3.2.

array index	y coordinate	x coordinate	value in '.connections'	connected?
1	1	2	1	yes
2	1	5	0	no
3	2	2	1	yes
4	2	4	0	no

Table 3.2: Contents of the array in the DPP software: four entries because there are four pixels. The number shown in the '.connections' column refers to the value of that component after processing. For HER, this must be '0' if the pixel is to have the acceptance threshold applied.

From this example, it becomes clear why position 'D' is tested first. Once that position has been examined, the array pointer can be fast-forwarded past the remaining pixel on that row (array index = 2) since we know that it is too far away to be relevant. If there are no pixels in the row y = 2 (i.e. y = y + 1 in the gen-

eral case) the routine can exit, otherwise it must step through the pixels on y = 2 looking for a match with the seed pixel's x-coordinate. It can be seen that the search range along y = 2 could be limited to a plus and minus region relative to the column containing the seed pixel, but this of course would add more tests on a per operation basis and as the CCD frames are somewhat sparsely populated, the extra cpu time taken is not likely to be recouped compared with simply testing the whole row. The complete routine is detailed in figure 3.17.

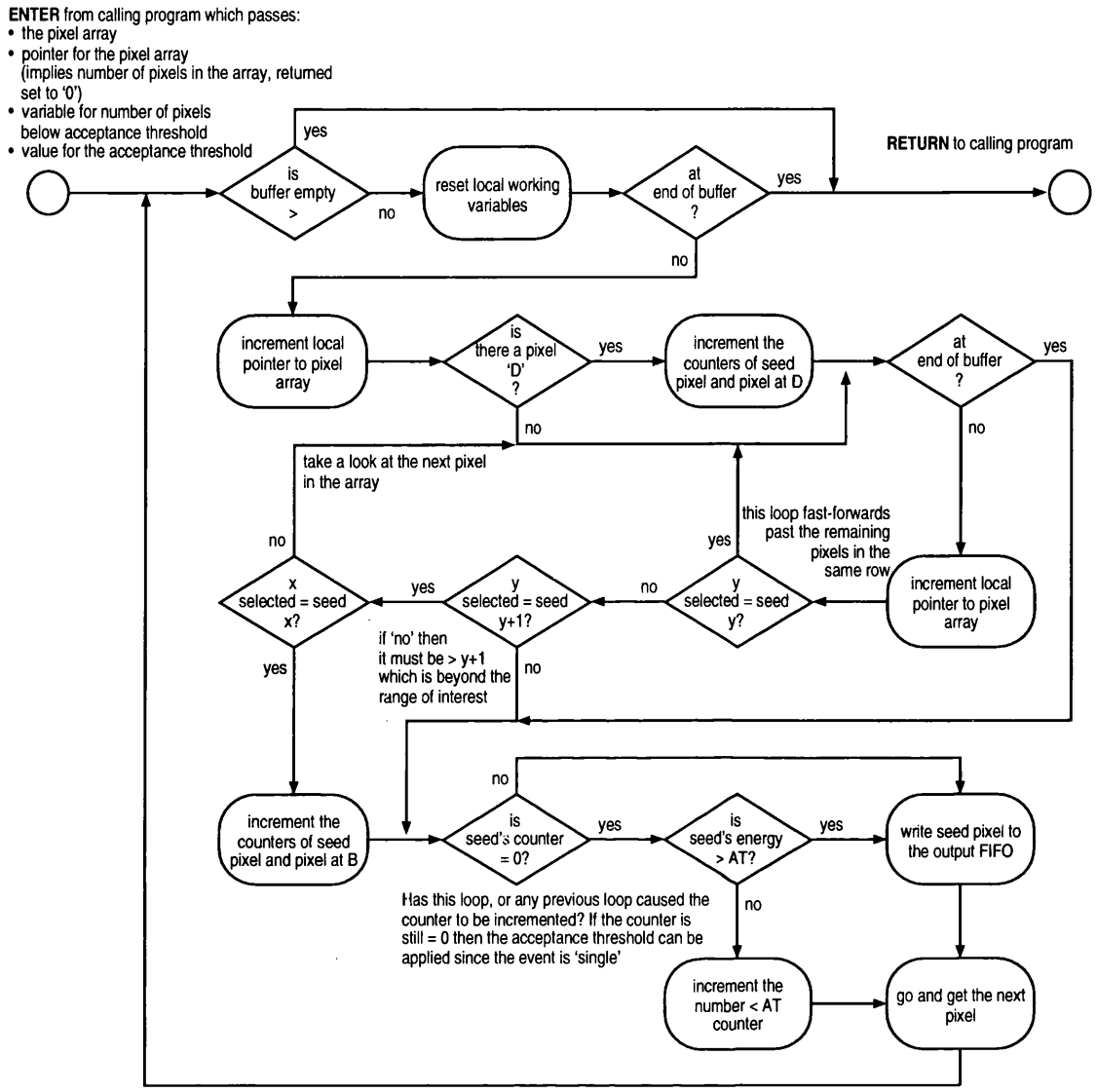


Figure 3.17: Application of the acceptance threshold in HER

3.1.5.2 Incorporating the HER Subroutine

The next question to address is how best to add this feature to the overall scheme. The existing two routines (basic spectroscopy and HTR) were operating on one pixel at a time, all the way from input FIFO to output FIFO, but to be able to perform searching in the way described, there must be a buffer for the pixels. Creating such a buffer also leads to the option of whether to continue with the

software as a single task or add a separate task: for example, were the pixels stored in a ring buffer, then one process could look after the input, more or less following the existing schemes but writing its output to the buffer, and a second task could look after the emptying of this buffer to the output FIFO via a choice of some of the more esoteric processing options to come. This concept certainly presents itself as a logical arrangement: easy to understand and modify; uses well established techniques (Burns and Welllings, 1995). But what about the impact on pixel processing speed? A major concern here is that since the output rate from the DPP is much lower than the input rate—being ultimately throttled by the telemetry allocation—the software must maintain the optimum load balancing: the instrument controller must never have to wait for a pixel (potential waste of telemetry bandwidth) and the input FIFO must never be left full if avoidable (potential for pixels to be lost). This requirement for the DPP to guarantee the processing of these aperiodically arriving events classifies it as a 'hard real-time system' (Stankovic, 1998). Time-critical aspects are typical for real-time systems of this sort, and the usual approach is to run multiple processes simultaneously under the direction of another process—the 'scheduler'. Even so, it can be difficult to guarantee that all of the deadlines will be met without the scheduler itself needing some intelligence, such as the rate-monotonic scheduler proposed by Sha and Goodenough (1990). With at most only two contexts to manage, this approach looked increasingly unwieldy for the DPP software.

Let's start by looking from the perspective of the instrument controller. At the default telemetry allocation of 6Kbits/second, allowing 20% for housekeeping overheads (Al Janabi, 1999), then in round numbers the instrument controller can handle:

$$0.8(6 \times 1024 / 16) = 307 \text{ words per second}$$

At the default on-chip binning mode of 3x3 the readout time is 0.6 seconds, this translates to:

$$0.6 \times 307 = 184 \text{ words per frame}$$

Now, a pixel requires three words (x, y and energy), and at the end of the frame there are twelve statistics words, this is:

$$(184 - 12) / 3 = 57 \text{ pixels per frame}$$

The output FIFO of the DPP has room for 1024 words, translating to

$$1024 / ((3 \times 57) + 12) = 5.5 \text{ frames}$$

In general, then, the disposition of the DPP software can afford to concentrate slightly more on handling the input than the output. Altering the on-chip binning mode has little effect since the time for each frame is also extended, and increasing the telemetry allocation to 12Kbit/second still gives the DPP room to buffer more than two CCDs in its output. If performing input and output as two separate processes, then the task scheduler can be used to allocate priority accordingly, but there is a cpu overhead in performing these changes of context, and also a memory implication on the stack size. Neither of these are automatically bad *per se*, but any development of the task handling to tune the throughput will add complexity to the software, and will also add a significant area of uncertainty when debugging any unexpected failure modes.

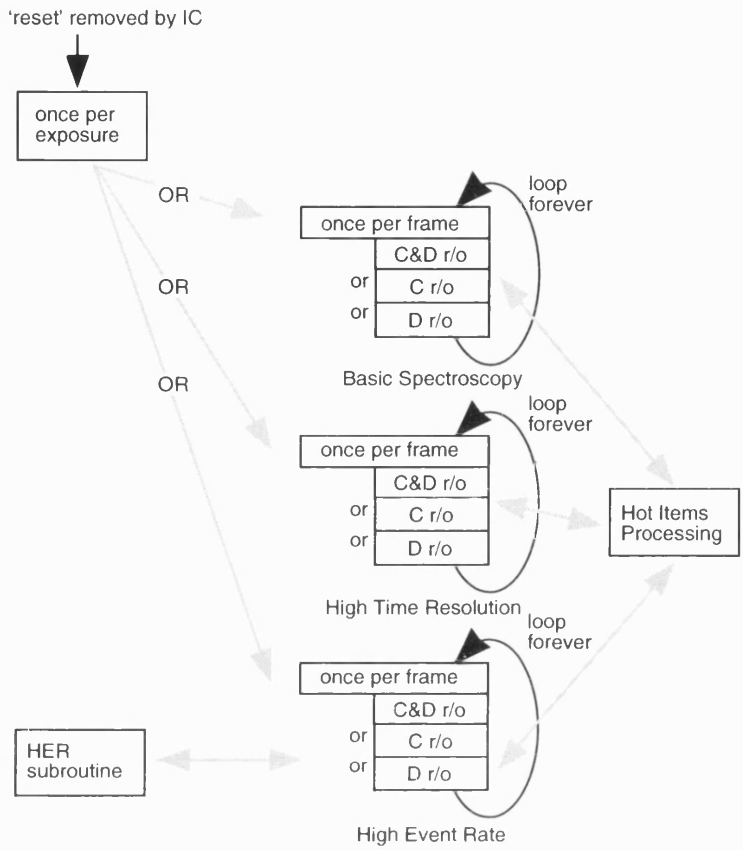


Figure 3.18: *Code layout for version 13 of the DPP software. Once again the extra functionality has been added in the form of an inline extension, using as much as possible of the already delivered software.*

The approach that the author chose for this was to use the inherent load balancing supplied for free by the input and output FIFOs, and to continue to build the software as a single task, with such switches of context that are necessary being under the control of that task. This was achieved by adopting the same 'front-end' of software as that used already, but now instead of writing pixels to an output FIFO, they are written to a memory array. This is a simple one-dimensional structured array of the form described in 3.1.5.1. How large should it be? Factors pushing to reduce the size were memory available and dwell time (i.e.

not spend too long waiting to fill up the buffer and leave the output FIFO empty). The main reason to extend the size was the desirability of incorporating the entire CCD in the buffer before the search: this would be simplest, though a scheme to store only enough of the CCD for the current search (i.e. the current row plus the next one) could be used, but that would increase complexity again.

Based on the expectation that around fifty pixels could be output per frame (den Herder 1994), the round number of one thousand was applied (in the form of separate buffers for pixels from side C or D, five hundred in each), allowing for the valid pixels to represent around 5% of those being processed. In 3x3 OCB these 'thousand' would be arriving in the 0.6 seconds taken for the entire frame, giving a delay of 0.6 seconds before the start of the processing to write the pixels to the output. As has already been seen, the worst case for this is if the instrument controller sees no queue in the output FIFO of the DPP, which now has 0.4 seconds to write the 307 words (~50 pixels) into the FIFO which the IC can drain per second. Even at only 1MIP, however, this still gives the DPP software 400 instructions per input pixel. Recall that the cpu is not idle during the filling of the buffer: in most cases in fact, most of the cpu time per pixel is consumed in the software front end performing coordinate deduction and hot items processing.

These numbers are all very approximate of course, and their purpose now, as at the time of the software writing, is to show that the sizing of the buffer at 1000 pixels was likely to be a reasonable starting point: if, as expected, the real number of pixels for processing was <1000 then the DPP software would be able to build up a queue in the output, and if the input number exceeded this and a single frame wouldn't fit, then as long as the buffer was emptied at not later than 1000 pixel intervals, the load on the DPP software

could be predicted. The drawback to this scheme in the event that an entire CCD does not fit into the buffer, is that any connections along the line of the join will not be seen: in the first section of the frame they won't have arrived yet, and they will be invisible to the second section. Of course in the 3x3 mode being discussed it is expected that the frame contents after lower thresholding will already be somewhat thinly populated, giving some statistical amelioration to this problem since there won't necessarily have been connections that would have been missed, even where there is a join. This is not a strong argument, but it is nevertheless a key example of the approach to the design of this software: were this project to be started today, then with the precise knowledge of the instrument which is now known; and with the increase in experience of the author, then some aspects would most likely be managed in a different way—this is power of hindsight. For the *ante facto* case, the author contends that it is preferable to start with simpler software that is designed to be scaled up in complexity as necessary.

Version 13 of the DPP software, incorporating the HER option, was released in April 1998 (Welch, 1997, 1998) and figure 3.18 shows the overall structure adopted for this. The author's strategy of building the extra functionality into the new version by adding inline code without modifying what had gone before fitted in well with the preferences of MSSL's collaborators on the project since it implied a reduced risk to the integrity of the code developed thus far, proven, and already extensively used in testing and calibrations. This situation was to come up frequently: new software would be made ready to use with the integrated RGS system, but the people using this system for testing mirrors and gratings and CCDs would—not unreasonably—prefer

to carry on with what they had already got. Relative to any previous version and the software routines in it, each new delivery built on the previous, proven, code to the extent of filling in the otherwise empty science-type selection: hence the external experimenters could be made less nervous about loading and using the new version since in the event of bugs found in HER, then they could revert to basic spectroscopy mode, knowing that this was unchanged.

Figure 3.19 shows the flowchart for HER mode; once again the parts of the diagram common to the previous modes are shown in grey.

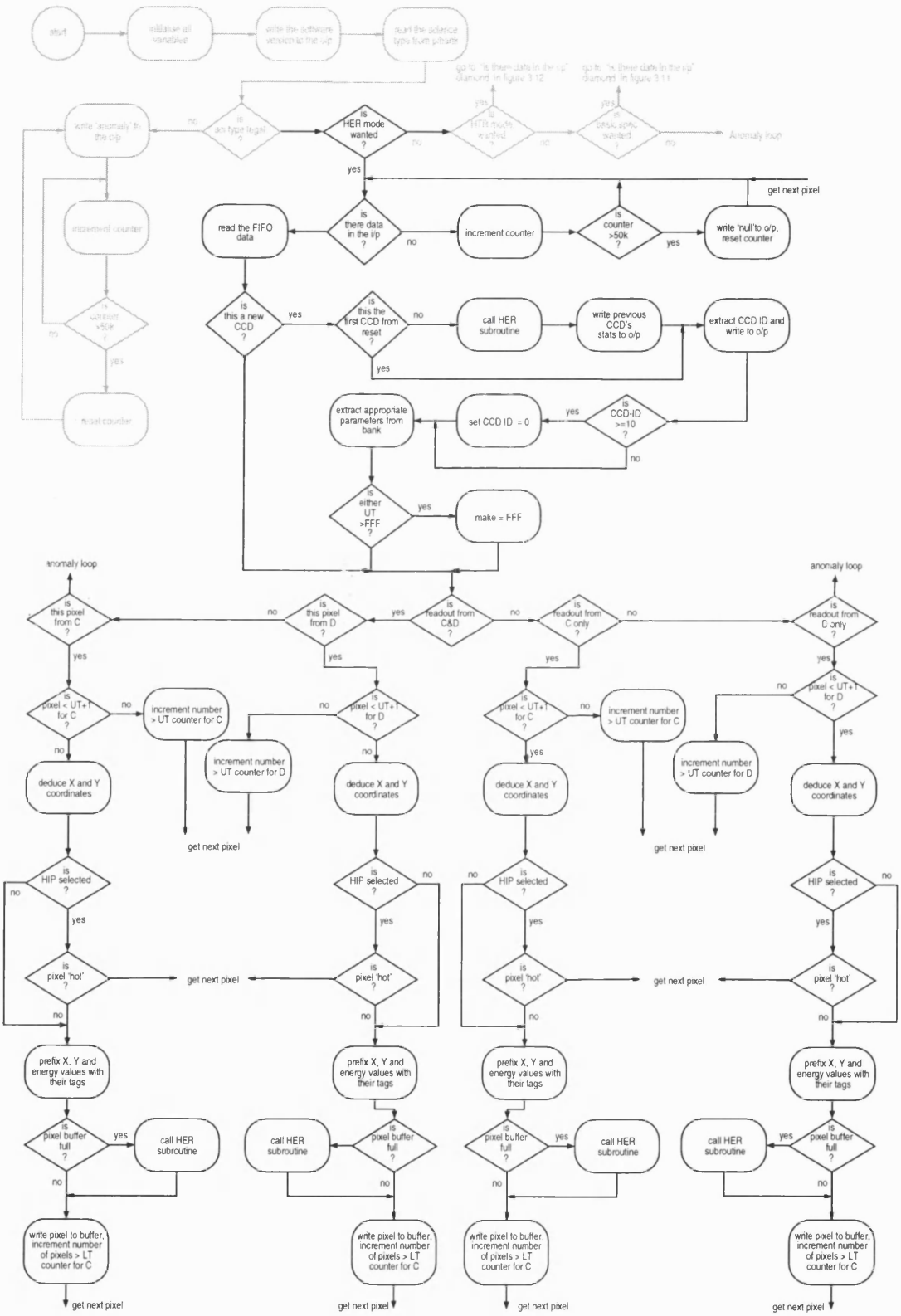


Figure 3.19: Overall flowchart for HER

3.1.6 Spectroscopy with Single Event Selection

This mode can be considered the technical complement to the HER mode discussed in 3.1.5. Conceived for use with very bright sources, SES mode adopts the same connection detection strategies as HER mode, but now only thresholds and passes the single events—any connected event is rejected automatically, irrespective of its energy. The flowchart for this subroutine is shown in figure 3.21. The underlying factors behind the layout of the code including this new function remained: from the need for speed, to the need to incorporate the new mode in such a way as to present minimal risk to the code already developed and proven. Accordingly the strategy from the implementation of HER mode was adopted, and the layout of the software updated to that shown in figure 3.20.

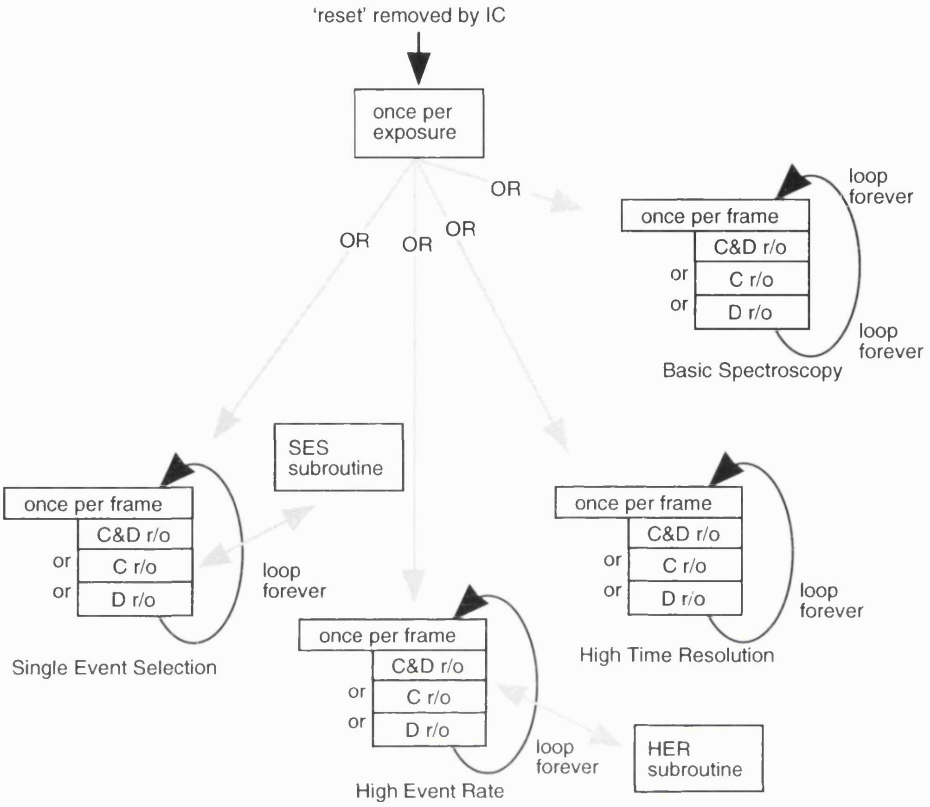


Figure 3.20: *Code layout for version 14 of the DPP software. This extends the layout from version 13: note that calls to the HIP module have been omitted for clarity, but the same module is available to each mode, as before.*

The overall flowchart incorporating the SES functionality is shown in figure 3.22.

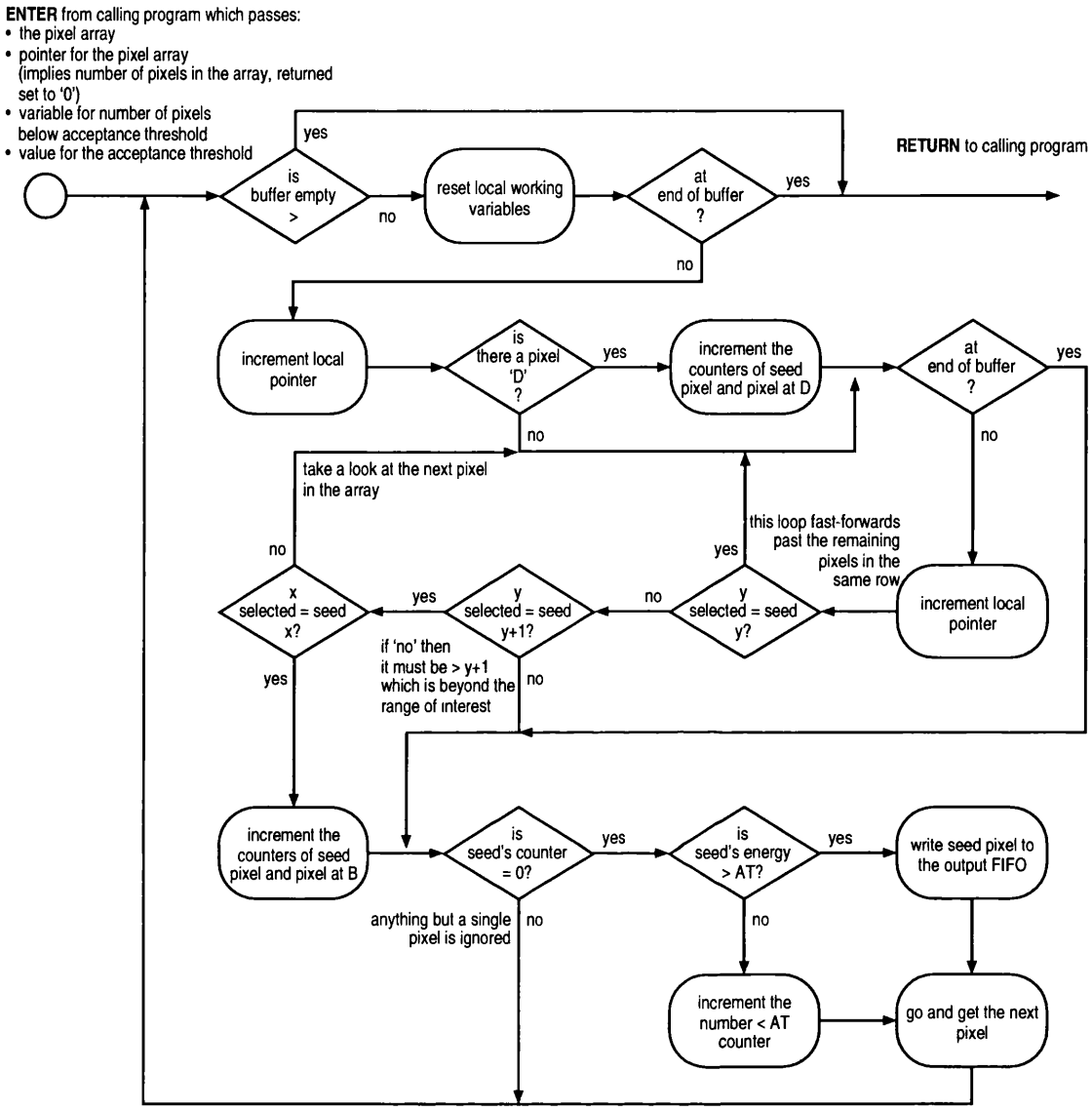


Figure 3.21: Application of the acceptance threshold in SES

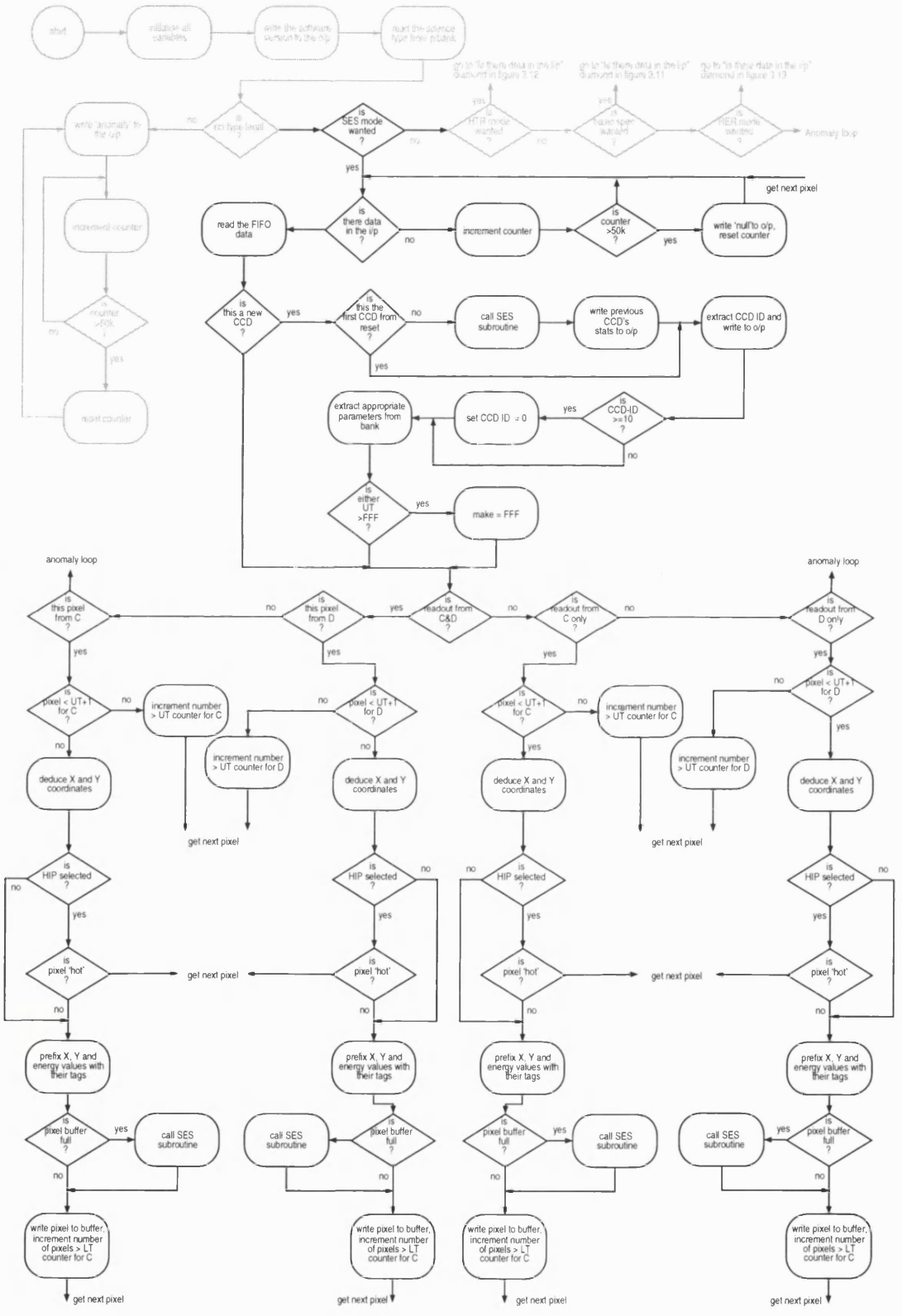


Figure 3.22: Overall flowchart for SES

3.1.7 Modes with Split Event Reconstruction

The question of how to perform event reconstruction has been an issue from the earliest steps of the XMM programme. Indeed, at one time there was a friendly competition at SRON—MSSL's collaborators on the RGS—to produce the most elegant algorithm for split event reconstruction: the most imaginative of which was written in PostScript, with the event reconstruction being performed by the printer! Some of the proposals from this competition, and those documented elsewhere for RGS (Chun 1996a, b; Philippus 1995, de Vries 1999, a, b) take little or no account taken of the time available for the processing (i.e. the cpu horsepower) or the memory space available. In the context of reversible routines to be applied on data in the ground segment this is one thing, but the onboard case was somewhat different. The ideas presented elsewhere all had problems when dealing with long trails of charge left across the CCD by cosmic rays, especially when there were several of these and they impinged on each other. No solutions were presented for this situation aside from that of discarding frames with cosmic rays (Chun *et al*, 1996).

After studying the techniques and algorithms found elsewhere, the author judged it preferable to start with a new paradigm—working up from the data quality, rather than down from the data left after the thresholding: i.e. looking at the CCD surface from the point of view of the data, rather than concentrating on an algorithm that would be able to run unterminated over the whole CCD correctly identifying all of the pixel connections, irrespective of what those connections might mean. From the discussion in chapter two, some predictions can be made about the split event morphologies by convolving what is known about the charge absorption mechanism at X-ray, with the action of the lower threshold,

and a set of eight outcomes for the most probable result from an incoming X-ray photon is proposed (figure 3.23).

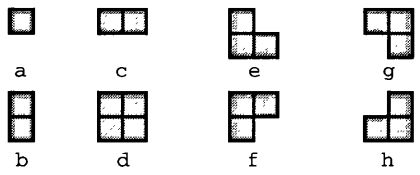


Figure 3.23: *The eight most probable energy dispositions from an X-ray photon incident on a CCD: (a) is a single event where all of the energy is contained within the boundary of a single pixel; (b) and (c) result when the photon is absorbed on a vertical or horizontal boundary between two adjacent pixels; (d–h) the photon is absorbed on the vertex between four pixels, with (e–h) the result when one of those pixels falls below the lower energy threshold.*

Any extensions to the shapes given in figure 3.23 can then have a choice of interpretations. Take the simplest case of three pixels in a row: these could be three single pixels, or a single pixel and a split event, but in this case to which event does the middle pixel belong? If the data is available on the ground then it can be reconstructed in a choice of ways, but actions onboard tend to be irreversible: put simply, if no information is discarded then no telemetry bandwidth is saved (recall that this is onboard preprocessing not compression).

The events shown in figure 3.23 are classified by the author as 'simple events', and an attempt was made to look into the impact on the telemetry of this approach (James and Welch, 1998). The analysis focussed again on calibration data taken at integrated instrument tests at the Panter test facility near Munich. The results from this showed much sensitivity to the thresholds applied, which is to be expected, but also quite clearly showed that amongst the simple events there was, for the data available, a strong bias towards single events, followed by

vertically split pairs ('b' in figure 3.23), then horizontal pairs, threes and finally the blocks of four least likely. The exact results were variable, but generally the distribution followed the order of 70%, 20%, 5%, 4%, 1%. The ratio of simple events to 'complex' events (which are events with connections which go beyond the 2x2 region) was completely dependent on the threshold setting: this was not a useful result in predicting the telemetry saving, but was a useful pointer towards some possible operational constraints on the instrument once in orbit. A commonly voiced reaction to the possibility of too much data for the telemetry bandwidth was that the lower thresholds could be raised: but this is not a good solution because there will necessarily be interesting data from the lower energy events which are near the noise floor and will be lost.

The principal advantage of reconstructing only simple events onboard is the minimising of the ambiguities in the resultant data, which can be further improved by considering what is done with the reconstructed event. For the location of the reconstructed event, two choices suggest themselves: first arriving pixel; and pixel with the highest starting energy. The former has the advantage of being simplest for the s/w to handle as it removes the need for any inspection and analysis of the individual pixel energies, but this may add an error to the position, and in the case of a spectroscopic instrument such as the RGS this could be important. Specifically, the deviation in x is important as this represents the dispersion direction of the grating assembly. Comparing the physical implications yields the result for RGS that in 1x1 or 2x2 OCB the error is smaller than the resolution of the spectrometer, but in 3x3 it is comparable with it; although it could be argued that the number of split events ought to be reduced by increasing the apparent pixel size, it is still important to avoid the possibility of such a

systematic error. This means that the location of the reconstructed event ought to be in at least the correct column. An approach based on putting the reconstructed pixel in a location within the column of highest energy weighting has promise, but it turns out that there is a further advantage to locating it in the first arriving pixel location.

Recalling the 'hot' items pixels from earlier (3.1.4), one may want to remove data that adjoins certain 'bad' pixels, for example the pixel which falls next to a pixel removed by, say, the hot pixel processing may also fall under suspicion (figure 3.24, A). Now the reconstruction location becomes important. Take the case of an event split into four such as figure 3.24, B. There are four possible sites for the reconstructed event, two of which are shown in figure 3.24, C and D.

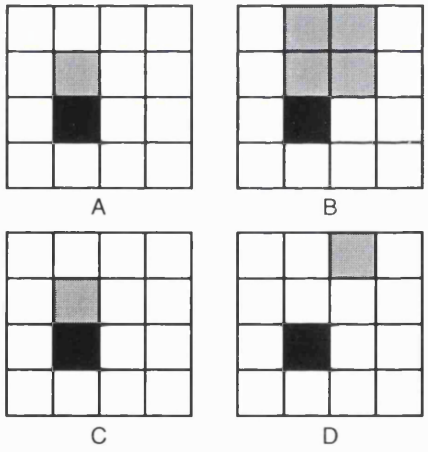


Figure 3.24: *Four occurrences around the same section of a CCD surface, near a 'bad' (shown in black). In 'A' a single event has arrived next to the hot pixel allowing its optional rejection on that basis. In 'B' the arriving event is split into four: if it is to be reconstructed the choice of which pixel should contain the energy sum becomes important, as the two examples show. In 'C' the reconstructed pixel still abuts the hot pixel, but in 'D' it appears quite distant and may therefore escape suspicion.*

It can be seen that from the ground segment's point of view, the reconstructed pixel in the first case might want rejecting where the second appears to be clear of influence of the adjacent bad pixel. How can this be overcome without increasing the amount of data sent to the ground?

To answer this we return to noise considerations. If we posit that each pixel contains a quantity of noise, then simply adding n pixels together will produce a slightly misleading result since their sum will contain n units of noise, instead of only one. Event reconstruction, then, must either take care of this noise correction onboard, or must include the number of pixels involved in an event in the data sent to the ground. For speed and flexibility the author adopts the latter approach, and does so without increasing the telemetry bandwidth by including the shape of the pixel into the data word used to describe the y-coordinate. Subtracting the four-bit data tag and allowing nine bits for the largest y value in 1x1 OCB, there are three bits available for this. These could have been used for directly stating the number of components in a reconstructed event, with '0' meaning that the event is single, '1' meaning that it comprises two components, and so on. However, if we use these three bits to indicate the shape of the event, then the number of components is implicit—and we have the option of splitting the pixel again in the ground segment if we make the origin of the event the first arriving pixel. The final disposition of this encoding as used on the RGS is shown in figure 3.25.

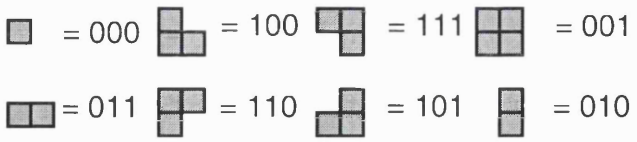


Figure 3.25: *The eight shapes counted by the RGS DPP s/w as legal for onboard reconstruction, along with their corresponding numerical value (shown in binary to avoid confusion with the internal shape encoding described later). This numerical value is inserted into three spare bits in the y-coordinate, and can be used to optionally 'unreconstruct' the event on the ground if required.*

By this method only the distribution of the energies in the individual components is lost—they will be averaged—and keeping the spatial information returns some flexibility to the further processing.

In conclusion, in the author's judgement the primary driving force on the design of the onboard processing should be the need for data quality over quantity, so that what remains can be unambiguously processed. Reconstructing only the simplest events indicated in figure 3.25, rather than a more extended set, has the drawback that the telemetry consumption is reduced by a lesser amount. If data is sparse, then this form of reconstruction can be employed, and the complex events can be telemetered as-is for later processing on the ground—and when the data quantity is too high, complex events can be deleted onboard. This last approach is close in spirit to an extension of the single-event selection mode, and was therefore termed by the author as 'enhanced single-event selection' (eSES). By the time of the software construction, the development of the ground segment was far enough advanced that adding eSES as a sixth optional mode would not have been possible, even if there were space in the DPP's memory for

it, and so the mode was developed as an alternative option able to be substituted for either the SER mode or for the SES mode (the author's preferred outcome).

3.1.7.1 Spectroscopy with Split Event Reconstruction

From the foregoing discussion the search algorithm needed for this mode is very much simplified by the short range of the connection search. Furthermore, the same data structure can be employed as for the SES and HER modes described earlier: the key here for each pixel is only whether it has a connection or not and the need for more complex pixel flagging arrangements is obviated.

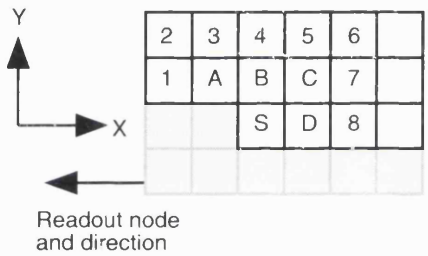


Figure 3.26: *An extension of figure 3.15. 'S' remains the seed pixel, and the region in grey is that which has gone before and of no further interest. A–D are the pixels which must be tested to define the shape of the event. Once known, some of pixels 1–8 will be tested, and the result of these tests will determine whether the event is 'simple' or 'complex'.*

It can be seen that the first part of the search algorithm can exactly follow the technique originated in section 3.1.5.1, needing only a simple way of identifying the shape which is the outcome. This is achieved by allocating each of the pixel locations A–D (figure 3.26) a binary weighted number, such that their sum yields an unambiguous integer indicating the shape, so: A = 1; B = 2; C = 4; D = 8. This

region is called the 'inner shell'. Position 'A' is only tested if there is a pixel in position 'B', and position 'C' is only tested if there is pixel in positions 'B' or 'D'. Having determined the shape of the event, certain of pixel positions 1–8, the 'outer shell', need to be examined to see if the shape is 'simple' (no connections in the outer shell) or 'complex' (there are connections in the outer shell).

Event Shape	3-bit flag for ground segment	Numerical value	outer shell components
	000	0	none
	010	2	4
	011	8	8
	100	10	4,8
	111	3	1, 3, 4
	110	6	4, 5, 7
	101	12	5, 7, 8
	001	14	4, 5, 7, 8
other	none	n/a	too complex

Table 3.3: *Summary of the codings and implied searches for event shapes within the DPP software*

The particular value of this philosophy in reducing the search load on the reconstruction can be seen in table 3.3: the most commonly occurring event (single) requires no searching beyond the inner-shell, and the next most common population (pairs) require for only one other pixel position to be checked. Even the worst case event, the block of four, requires only four outer-shell positions to be sought. The SER subroutine works on one pixel at a time from its input (a pixel array in the form described in 3.1.5.1) to its output (the hardware output FIFO). Once again the order of the execution of the functions in the SER routine has been carefully designed to ensure that the most commonly occurring events exit the routine by the quickest route. An overall flowchart for the SER subroutine is presented in figure 3.27. The same data structure was employed for this routine as for the HER and SES routines, and consideration was given to adding functionality to the word which carries the number of connections found, but this proved unnecessarily complicated. All that is needed is a zero/non-zero indication for the number of connections, and a third value to exclude the pixel altogether. This last is achieved by adding a test which looks for a value exceeding EFF(hex). This value cannot be reached by incrementing the counter when a connection is found, and by the same token there is no possibility that the FFFF(hex) word boundary can be exceeded by adding F00(hex) to the aggregate of connection increments: thus no range checking or bit manipulation are required. Since the data buffer is not in the form of a ring buffer, this means was needed to preclude pixels absorbed into a reconstructed event from being looked at again on an individual basis, without deleting them from the list until they have been seen by all of the nearby searches.

On entry into the SER subroutine the first test, then, is to see whether the pixel has already been included in a reconstructed event. Next there must be a test to see whether there is a connection to a pixel that has gone before (i.e. in the grey zone in figure 3.26). A yes/no is all that is needed for this implementation: connected events that are simple will have been reconstructed, and those that are too complex will be telemetered as-is, so a boolean can be set indicating that, though the search must continue, this pixel will not be prevented from being written to the output FIFO. This is the property of 'inheritance'. The connection searching around the inner-shell continues in much the same fashion as before, but now a local variable is increased according the pixels found (this is the A = +1 to D = +8 scheme).

Searches on pixels 'A' and 'C' are conditional on there being other pixels (B, B or D respectively) making them other than only diagonally connected. At the end of the search for the inner-shell, the 'shape register' can contain one of eight numbers, and these are used for the next branch.

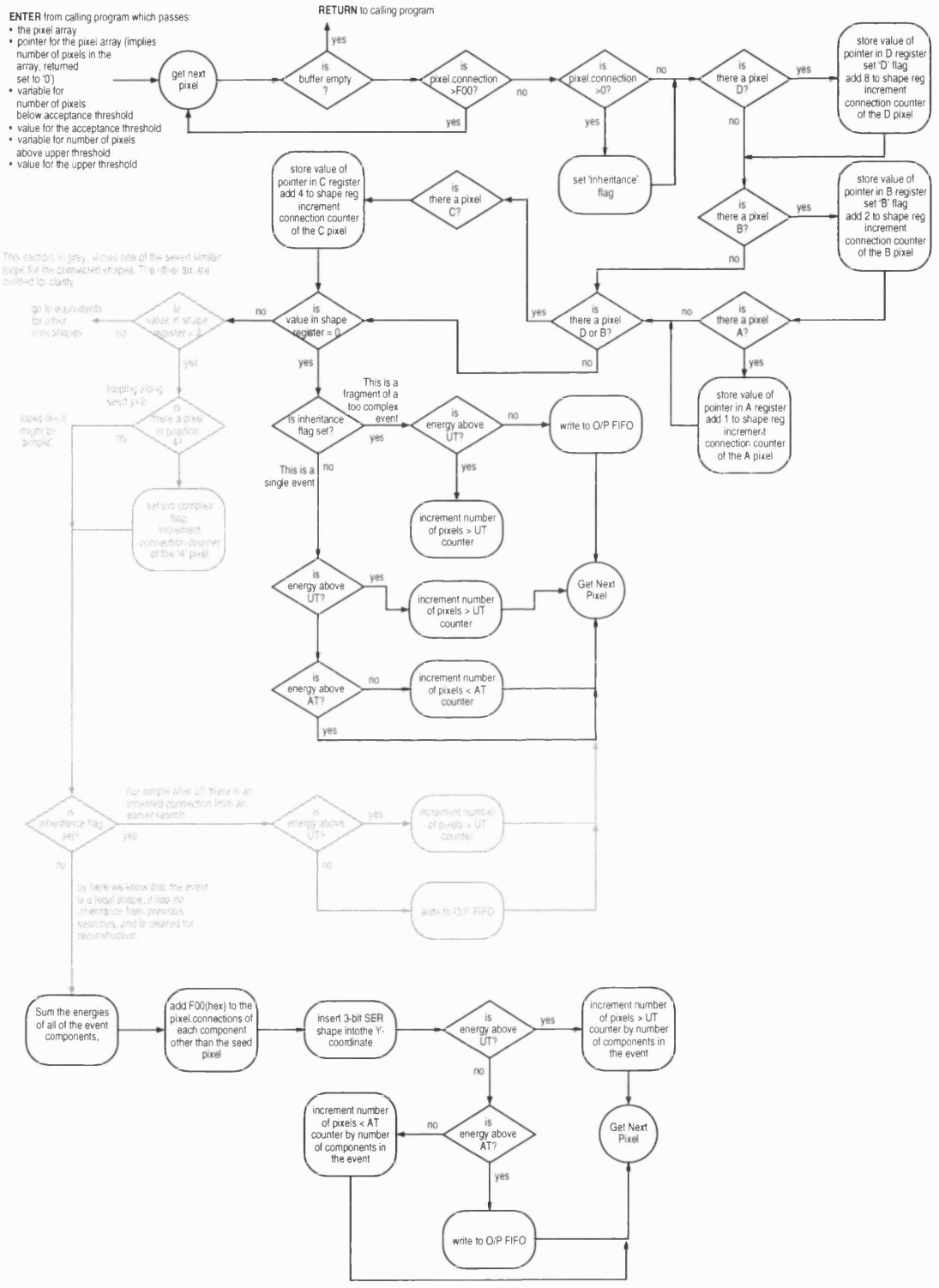


Figure 3.27: *The Split event reconstruction algorithm*

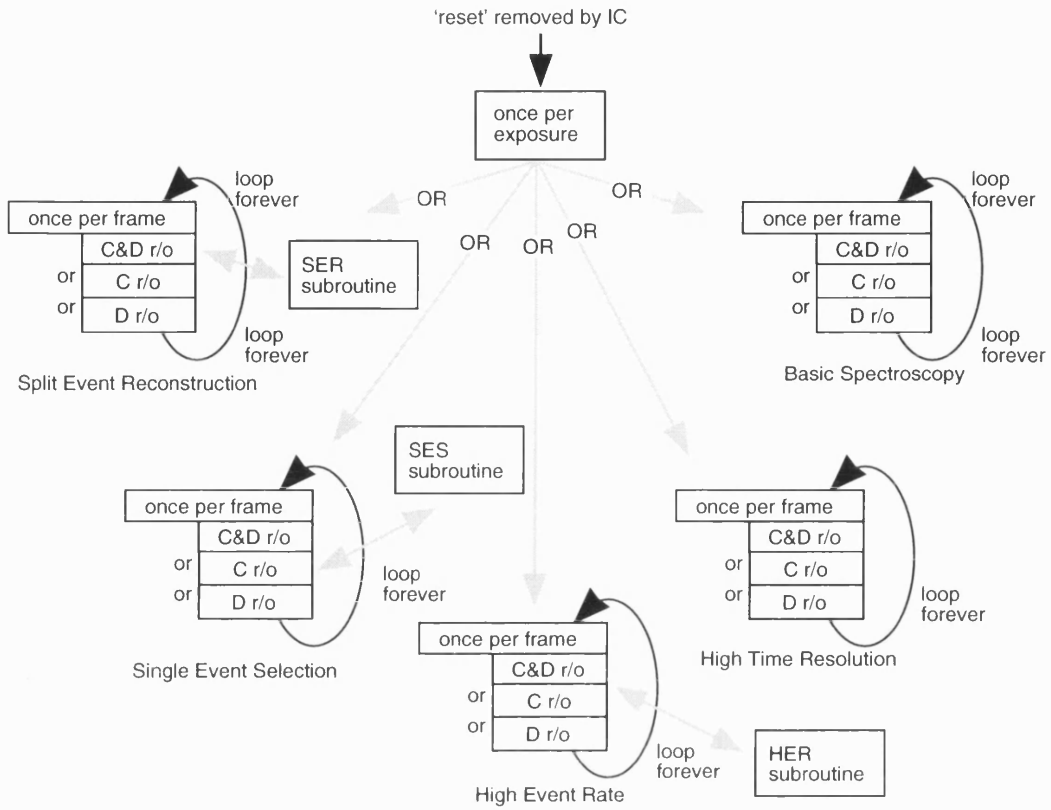


Figure 3.28: *Code layout for version 15 of the DPP software. Once again this extends the code of the previous release, with the changes confined to the new subsection, allowing it to be used externally to MSSL with minimum risk to the collaborators programmes: in the event of problems, other modes can still be accessed.*

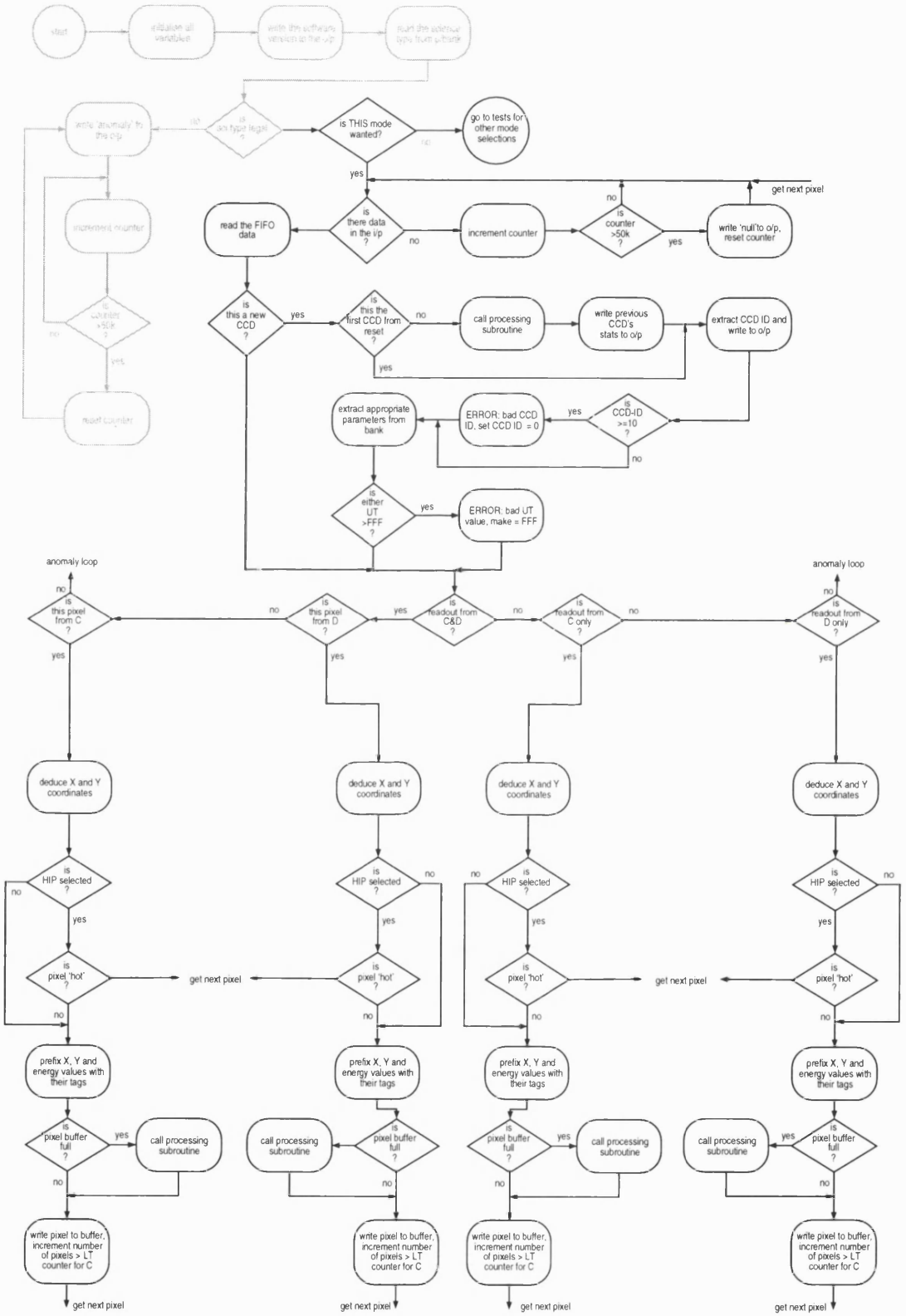


Figure 3.29: *Overall flowchart, generic case (works with SER and eSES).*

If the pixel under test yields a shape value of '0', and the inheritance flag is not set, then it is a single event and both the upper- and acceptance thresholds can be applied. If the pixel has any inherited connection, then only the upper threshold is applied. The logic which calls for this threshold to be applied even though there is a risk of removing a pixel and leaving an ambiguous remnant was rooted in the expectation that there would always be sufficient high energy pixels to consume a disproportionate amount of the telemetry allocation. A more complete solution to this trade-off is found in the eSES treatment where all 'too-complex' events are rejected.

If the pixel under test yields a legal, non-zero value (2, 3, 6, 8, 10, 12, 14) then a search for outer-shell connections can be carried out. All of these follow a similar pattern, but only that for '2' is shown for the sake of clarity. This is in grey in figure 3.27. Following a similar search technique to that used for the inner shell, the DPP software then looks for the subset of pixels in the outer shell according to the core shape (see table 3.3). If a connection is found then a boolean is set indicating that the event is too complex, and the pixel is ultimately shifted out with only the upper threshold applied. If a non-legal, non-zero value is distilled for the shape, then the pixel is also flagged as too complex and treated in the same way.

Finally, a legal non-zero event shape that has no inheritance can be reconstructed. This is made as simple as possible by adding the values of the seed pixel's energy, and each of the energies of A–D: there is no need to select a matching subset of the actual pixels as any unused values in A–D are already set to zero. All four of the inner shell components have F00(hex)

added to their count of pixel connections (this is irrelevant to the components not used, so there is again no value in adding conditional statements to be evaluated). The resulting reconstructed event can now have upper- and acceptance thresholds applied, but now the statistical counters must be incremented by the total number of components rejected, since this is now more than 1.

The overall program structure for the released version (15) of the software including SER was retained, with the SER routine being a new callable module, and with the mother routine modified only by the removal of any application of the upper thresholding prior to the call.

3.1.7.2 Spectroscopy with Enhanced Single Event Selection

This mode was developed by the author as a complement to the SER processing described in 3.1.7.1, with the code being under unit level testing at MSSL before the flight of XMM, and before it was actually wanted. The generic flowchart of figure 3.29 still applies, and the eSES processing subroutine is more closely allied to that of SER than of SES. In eSES the inner- and outer shell analysis of SER is executed, but now all complex events and pixels with inheritance are deleted.

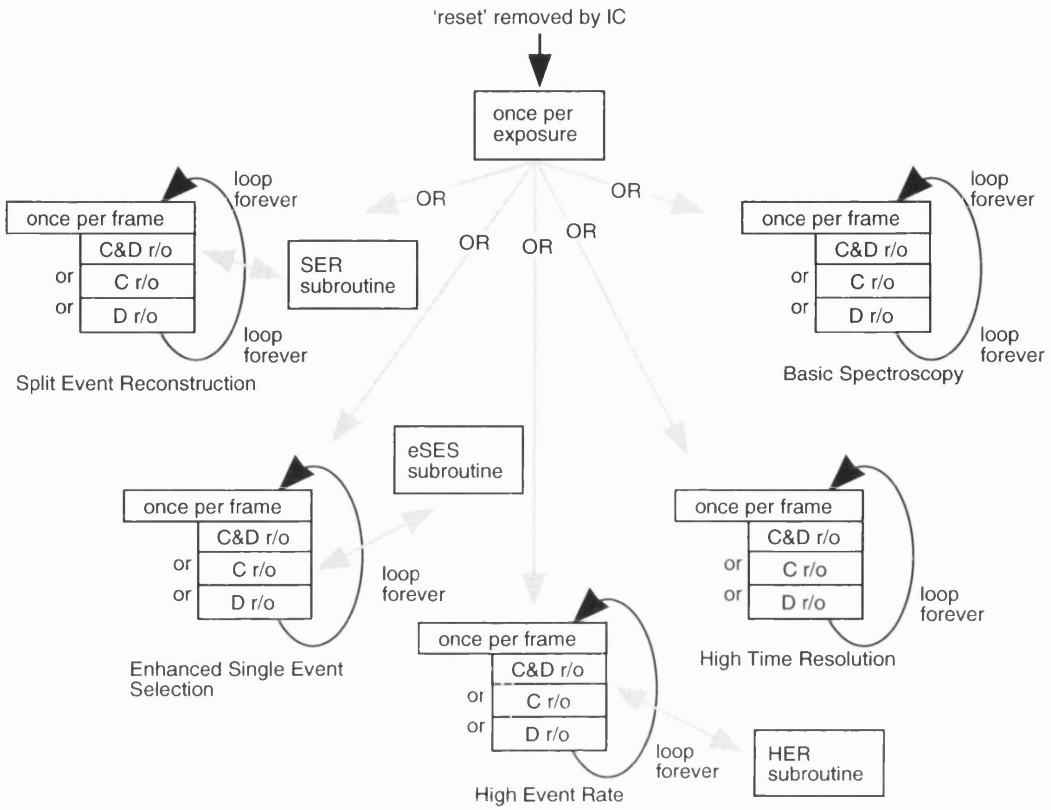


Figure 3.30: *Code layout for version 19 of the DPP software. There are no new modes available, and SES mode is replaced with the eSES routine.*

The flowchart for this new subroutine can be found in figure 3.31.

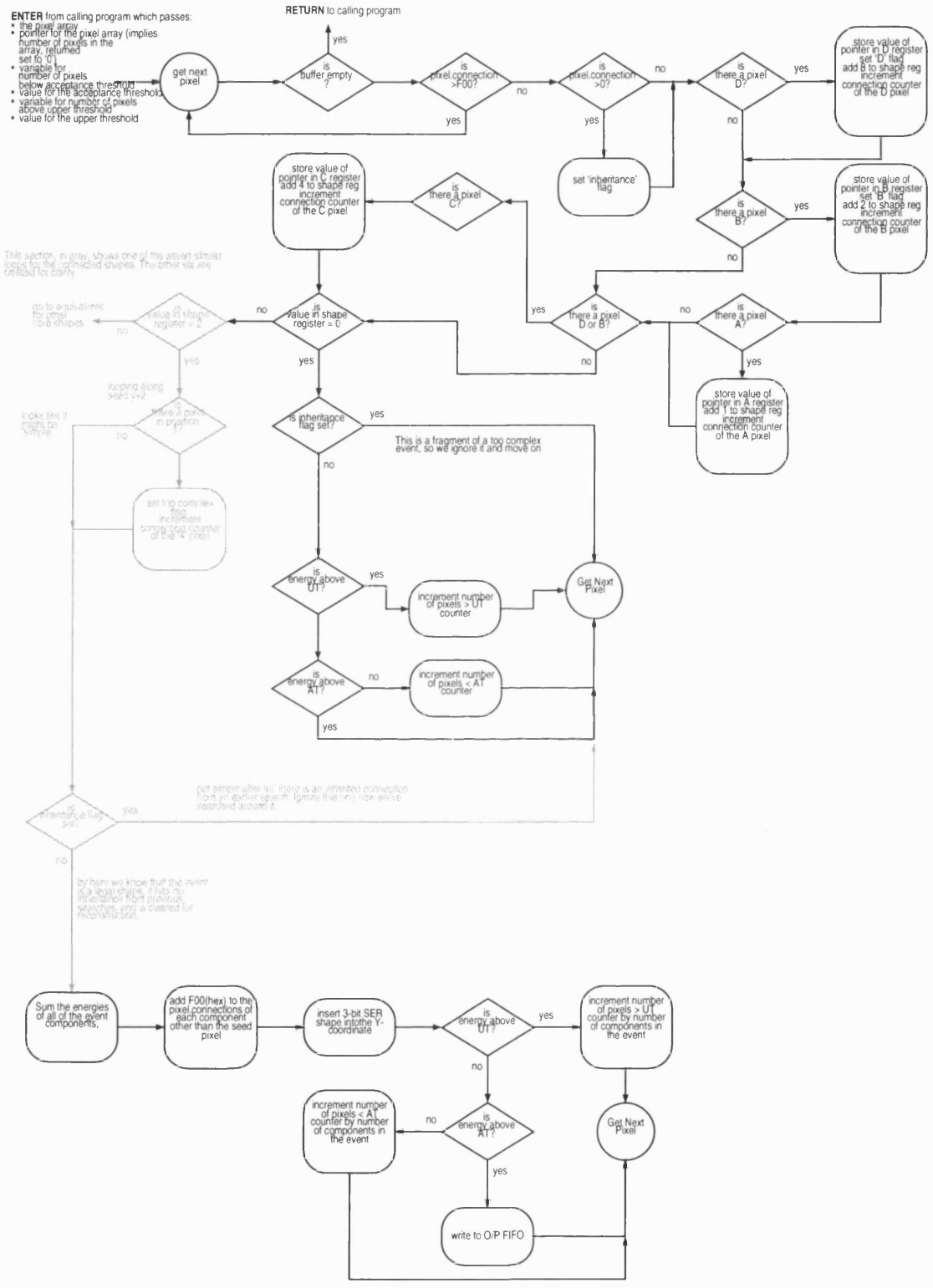


Figure 3.31: Flowchart for the eSES subroutine.

The flowchart for eSES follows the same style as the flowchart for SER (figure 3.27), with the preliminary search around the inner-shell conducted as before.

The first change is that in the case of an apparently single event, if the inheritance flag is set the pixel is deleted. A non-zero shape causes an appropriate search around the outer-shell for connections, but once this search is completed, there is no need to continue with the pixel under test unless it is simple and has no inheritance.

Version 16 of the software was selected for the early post-launch operations of the RGS. This version was an iteration of the version 15 layout already described, containing only bug-fixes. The software had already been developed beyond this point, but version 16 was the release with the most pedigree insofar as it had been in use for the most time with the integrated instrument. More information about the history of the changes of plans for default modes and so on will be discussed in chapter five, though author persisted with the developments, also creating versions 17, 18 and 19 and testing them at MSSL. These later versions contained either bug fixes and small modifications (17) or different permutations of the modules already discussed (18 and 19). After early in-orbit results indicated a significantly higher background than forecast, the science operations team rapidly requested version 19 of the code as this would—as will be seen in Chapter 5—reduce the background while permitting the lower and acceptance thresholds to be set to lower values. Version 19 was duly delivered to ESA, becoming active on the spacecraft in March 2000, with eSES as the default mode of operation.

	Hot Items processing	Basic spec.	HTR	HER	SES	SER	eSES
10	–	✓	–	–	–	–	–
11	–	✓	✓	–	–	–	–
12	✓	✓	✓	–	–	–	–
13	✓	✓	✓	✓	–	–	–
14	✓	✓	✓	✓	✓	–	–
15	✓	✓	✓	✓	✓	✓	–
16	✓	✓	✓	✓	✓	✓	–
17	✓	✓	✓	✓	✓	✓	–
18	✓	✓	✓	✓	✓	–	✓
19	✓	✓	✓	✓	–	✓	✓

Table 3.4: *Final summary of onboard software versions as created by March 2001. Versions 16 and 17 incorporate small modifications, version 18 substitutes eSES for HER mode, and version 19 substitutes eSES for SES.*

The complete source code listing for version 19 runs to 5627 lines of Ada, with no assembly language inserts required. The line count was measured by concatenating all of the individual source files, then running the following awk script:

```
/^$/{bl += 1}

/^[ \t]*--/ {c += 1}

/^[ \t]*[a-z A-Z]/
```

which removes lines which are empty, and lines starting with '--' which is the Ada comment indicator. The object code occupies 27KWords.

Chapter 4: The Software Simulator

In the early development phase of the onboard software it proved useful to develop a stand-alone software simulator for the DPP. This ultimately had several major benefits because it allowed:

- work on the onboard software to continue without access to the actual hardware, which was of course still under construction: a common situation in space engineering development;
- testing and optimisation of algorithms in another language, thus helping to trap errors of understanding or logic;
- application of the author's algorithms in the ground segment's pipeline processing system to determine the optimum settings for the onboard software.

Chapter four briefly describes the genesis of the software DPP—known as the SWDPP—from its creation as a local tool, to a more general utility that was incorporated into other software facilities.

4.0 The early software DPP for local use

4.0.1 The need for the simulator and the philosophy behind its creation

In a manner typical to the development phase of any complex project, competition for resources amongst the RGS team was frequently high. At these times, the onboard software for the DPP was last in the queue for access to the hard-

ware development systems, behind the folk developing the hardware itself and the author of the onboard instrument controller software. This is quite reasonable of course, but meant that there was potential for a lot of 'elapsed' time to be consumed from the schedule for the development of the preprocessing software. The author's strategy to combat this problem was to develop a stand-alone software DPP that could be run on a desktop computer. This would have to be a simulator for the whole of the DPP, i.e. including the hardware, that would allow testing algorithms and techniques against known data (Welch, 1995).

No modelling of the real-time behaviour of the real DPP was incorporated into the SWDPP, though the author considered methods for connecting the operation of this simulator to actual time by adding, for example, some code counters which would increment a specifiable unit of time such that forecasts could be made for the time taken to perform operations. These ideas may have provided some useful indicators, but it was decided that the value of such information relative to the cost implication on the time available for the work was small.

The purpose of this software DPP was, therefore, to take user generated pixel data files, and according to the contents of parameter, hot pixel and hot column files, generate an output file which could be inspected to verify the correct operation of the routines and algorithms employed. No account of CCD ID was taken, and each pixel data file was considered to contain a single frame of node-independent data.

4.0.2 Approach and Basic Design

The first question to address was the choice of language: this was between Ada and C. C was already well known to the author, and was available, making it a good choice from the point of view of saving development time. For Ada evaluation, the author downloaded GNAT, an open source version available on many platforms, but after some experiments concluded that the benefits of the extra Ada experience were minimal since they were mostly in irrelevant areas (e.g. file handling, screen output etc.). Furthermore, there was potentially a benefit in rewriting algorithms in a different language to help prevent misconceptions being transported.

Simple ASCII was chosen as the format for the files employed by the DPP since they are easily generated by many techniques, and are simple to read and modify, especially if organised as a single value per line, as this is directly analogous to a one dimensional array in memory, allowing the usual `fopen` methods, but also functions such as `fseek` which is especially useful when it is not possible to read an entire file into local memory. The contents of the hot pixel and column tables follow those of the 'real' DPP. Also needed is a 'parameters' file containing a list of the parameters as per the onboard code, and adding values for the lower thresholds for each CCD node. In the sense that not all of the functions of the hardware DPP are synthesised, the software DPP is an imperfect analogue. However, items such as the CCD number and the number of lost events were not relevant to the stated purpose for this software.

Earliest versions of this software were produced from late 1995, with development continuing intermittently as the requirements for the flight design were evolving, and as constraints on access to the hardware development system ebbed and flowed.

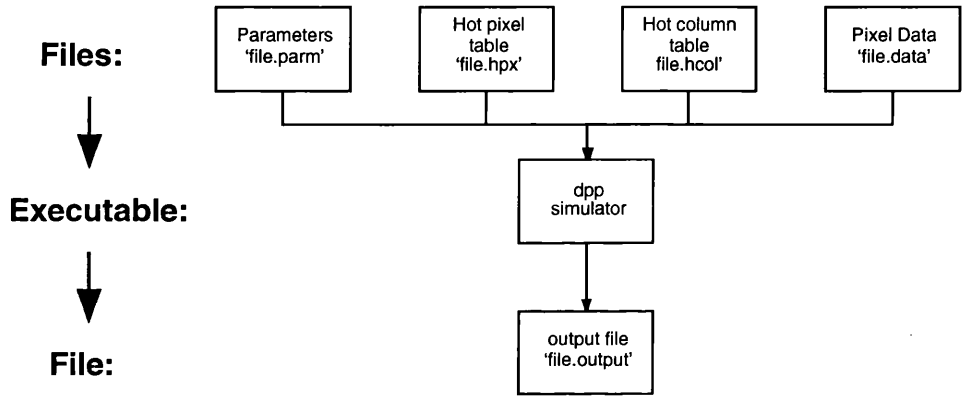


Figure 4.1: Components of the original software DPP. The executable takes all of its inputs from ASCII files, and its output is a further such file. The 'file' in the figure is a base filename supplied by the user, and the SWDPP assumes that it will find the resource files (top row) named by the declared convention. The SWDPP will also use the base filename to form part of the name for the output file.

Figure 4.1 shows a summary of the components required for a 'run' of the SWDPP. At the beginning of a run, the software DPP prompts for a filename. This is the 'base' filename, for example "test1". The software then builds the rest of the filenames that it needs by appending specific suffixes (test1.data, test1.hpx, test1.hcol, test1.parm), and looks to see if it can find them all, returning an error to standard output if it can't. Files other than the pixel data are read into memory. The pixel data file is opened, and then each energy is tested according to the

specified lower threshold value, and either read into local memory if above the threshold, or incrementing a counter if below it. With the pixel data now internal to the program, processing can continue, with the results written to an ASCII file (test1.output) in the form of three columns, x, y and energy, with the processing executed according to the science mode selected.

No tools were constructed to handle or analyse the output files automatically because the input and output data files were small and with known contents—i.e. they could be analysed by hand.

There were of course options to extend the functionality of this simulator to read genuine data files, but with limited time available for the primary work (writing the onboard code itself) this would have would have been too distracting. Furthermore, the author felt that this was the best approach to ensure that *algorithms* were being debugged by the SWDPP, and not the SWDPP code itself. Though simple, the SWDPP was in fact of extreme value to the author, as it was a useful testing ground for ideas which were often modified as a result—in the hot pixel processing software development, for example, it was during the writing of the simulator that it became obvious that ordering the entries in the hot pixel table such that the y-coordinate came first would shorten the search routine.

Not all of the code was written in C first: this was dependent on system access, but whether for checking code written elsewhere in Ada, or for coding in C as a precursor to coding in Ada, the author found great value in approaching the problems in two languages since the different structures of syntax forced the underlying purpose of each code section to be re-examined, and often seen in a different light.

4.1 The later SWDPP for 'public' use

4.1.1 Design Specification and philosophy

Throughout the development phase of the RGS it was hoped to generate a complete software version of the entire instrument and optics for calibration and verification purposes, and some thought was put into this from the MSSL side, though this thread never took off for the common enough reason that for each institution involved this had to be prioritised behind the main work of getting the instrument built. In 1999, however, after some lengthy discussion at a meeting with fellow collaborators and ESA (Erd, 1999) it was finally agreed that the ground segment software that was predigesting the data for the astronomers using the instrument (this is the RGS 'pipeline') ought to contain an event reconstruction phase based exactly on the onboard case. The opposition to this had come about since at the time there was still disagreement about the value of the author's proposed method of performing the split-event reconstruction, and so why should the pipeline copy it? The author's view, which was eventually accepted, was that since the onboard SER algorithm can reasonably claim not to introduce any ambiguities in the data, then the pipeline should echo this technique whether onboard reconstruction has taken place or not. This could be implemented in the form of a preprocess: it is necessary for data not reconstructed on board, and if events have been reconstructed on board then the preprocess will be redundant, but will make no difference. This 'first pass' preprocessing could then be optionally followed by some more comprehensive mechanism, and it was agreed that the pipeline team would implement the author's algorithm.

Partly as an input to the documentation for the pipeline people, and partly with a view to making the existing SWDPP more portable, the structure and implementation of the existing simulator was revisited. To save development time, and to inherit some pedigree from the work that had gone before, a new version of the SWDPP should use as much as possible of the existing code: but what form should the output take? Consideration of this last question was coincident with a request that the author be able to perform some independent analysis of early orbit Diagnostic data with a view to feeding back some suggestions for settings for the various thresholds. This task would require the author to be able to work with raw data taken from the queue memory, applying the various processing options, and deducing from them the optimum values for the instrument settings that affected the DPP. This suggested that most useful output would be a file of statistics, listing the numbers of pixels remaining after each of the various operations. A sample output template for a single run of this is shown in figure 4.2.

```
MSSL SW-DPP Diagnostic Data Summary
-----------------------------------
Input Filename:

From FITS header:
-----------------
Xlength:                Ylength:
CCD-ID:
Readout Nodes:

From Parameter File:
-----------------
Upper threshold C:          Upper threshold D:
Acceptance Threshold C:     Acceptance Threshold D:
Lower Threshold C:          Lower Threshold D:

From SW-DPP:
------------
Mode                  Basic   HER   SER   SES   eSES
-------------------------------------------------------
Number below lt:
Number below at:
Number above ut:
No. of Events of
type 0:
type 1:
type 2:
type 3:
type 4:
type 5:
type 6:
type 7:
No. telemetered:
```

Figure 4.2: *Template for the contents of the output file from one run of the software DPP. The user specifiable items are taken from a parameter file, which itself may be generated automatically, recursively stepping through a range of values.*

4.1.2 Code Implementation

This simulation of the DPP system is, then, a relation of the first software DPP, but not the same thing. For reasons of practicality, the best use of the already written C was sought, and the first area of concern was the input of the data. The data files from the ground segment were to be supplied in 'Flexible Image Transport System' (FITS) format (Wells *et al* 1981, Schlesinger, 1994), but after examination, there seemed little point in the author directly modifying the software DPP to be able to handle this type of file since there are many tools which can be used for this purpose, such as FITSIO (Pence, 1995), and its relative FTOOLS (Blackburn, 1994). The author duly installed FTOOLS, as this was available on all of the computers in use elsewhere in the ground segment. With FTOOLS able to open up FITS files and write them into ASCII, and with `awk` (a UNIX language utility for text manipulation) similarly available to reprocess the ASCII into the form expected by the software DPP, the 'front-end' of the existing DPP could be adopted with little modification. With FTOOLS and `awk` being readily available, some detail is given here about the process as it may prove useful elsewhere.

4.1.2.1 Header Data

The first portion of the FITS files is a header, which for the software DPP was interesting because it contained details about CCD-ID, X length and so on. This can be readily extracted into an ASCII file with this command:

```
prompt% fdump prhead=yes   filename.FITS   temp.output - -
```

An example output ASCII file is given in appendix 2. The next task is to extract the parameters of interest. There are five, and these can be found by using awk to match the patterns in the ASCII file just made. This is the awk script used for the software DPP:

```
/^CCDID   =/ {print $3} # the CCD ID
/^WINDOWX0=/ {print $2} # the X start
/^WINDOWY0=/ {print $2} # the Y start
/^WINDOWDX=/ {print $2} # the X length
/^WINDOWDY=/ {print $2} # the Y length
/^CCDNODE =/ {print $2} # the read out nodes
```

The text in the left most column represents the pattern to match, and the print statement causes awk to echo what it finds in the indicated column after it finds the given pattern. (One might have expected to find all of the numbers in the same column, they certainly look to the eye as if they are—but this is a fairly typical sort of thing that can go awry and must be checked for). These parameters are piped to a file, 'header.txt', the name of which is hard-coded into the software DPP. By this mechanism all of the parameters associated with a readout are extracted to a single, uniform, input mechanism.

4.1.2.2 Pixel Data

An extension of the technique discussed in 4.1.2.1 is used for the data, which in the RGS case is in a FITS extension. This adds an extra step from the FTOOLS set...

```
prompt% fim2lst filename+1 tmp.fits
```

... creating a new FITS file containing only the pixel data. Now, `fdump` can be used to make an ASCII text file as before...

```
prompt% fdump prhead=no   tmp.FITS   tmp.txt - -
```

... and we have a data file that looks like this:

```
          X       Y         VALUE
1         1       1           16
2         1       2           18
3         1       3           19
4         1       4           18
5         1       5           18
6         1       6           19
7         1       7           19
8         1       8           13... and so on
```

The text preceding the numbers, and the index count in the first column are not wanted, so once again an awk script can be used to strip out only what is wanted:

```
/^\/n/;
/:alpha:/;
/[0-9]/ {print $2; print $3; print $4};
```

Note that the result of these preprocessing steps is an ASCII file with the x- and y-coordinates already present, so with very few steps the supplied FITS file can be repackaged to give data in a very convenient form for manipulating in C.

4.1.3 Software DPP Object code

Overall, the deliverable software DPP makes the assumption that both FTOOLS and awk are available, and comprises a shell script to organise the pre-processing, and some object code the source of which is written in C. The object code:

- reads in only those pixels above a lower threshold, counting the rest into a statistical word, sorting the pixels into buffers for side C or D origin, according to knowledge about the read-out nodes extracted from the header;
- performs basic spectroscopy, HER, eSES, SER recording the results in more statistical variables;
- produces a formatted ASCII output file with the results, as illustrated in figure 4.2.

The particular benefit of using a shell script to run the software DPP is that there is no need to hard-code values for items such as the lower thresholds. These values are picked up from a parameter file and can themselves be modified by the shell script such that the object code part of the software DPP can be called repeatedly while incrementing threshold settings so that the aggregate of the results can be used to determine an optimum value. This technique was used by the author for some analysis of early data. An example shell script for the software DPP is presented in appendix 3, and figure 4.3 shows the overall relationship between the components of the software DPP.

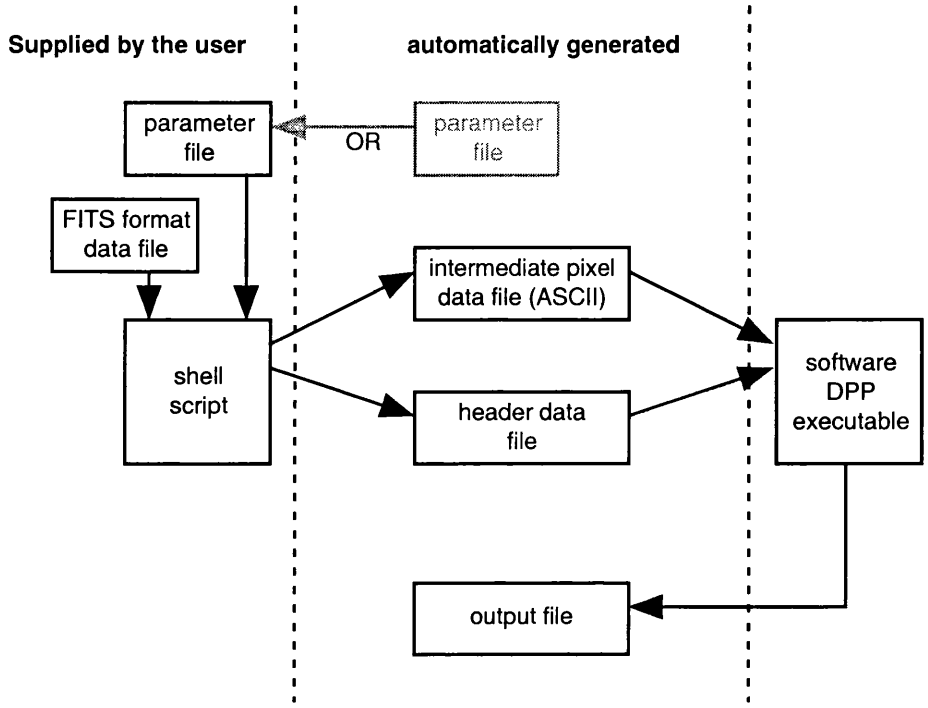


Figure 4.3: *The 'deliverable' software DPP. The components to the left of the diagram are supplied by the user and those in the centre are automatically generated files. The 'parameter file' contains the settings for the DPP to use (i.e. thresholds, readout pattern etc.) and may be supplied as a file, or may itself be automatically generated by a further script. The latter technique is useful when it is desired to process a particular data-set repeatedly while changing the DPP settings in order to find an optimum value for some or all of the parameters. Overall orchestration is handled by the 'shell script'. The 'intermediate pixel data file' and the 'header data file' are temporary files and may be deleted automatically. When running a recursive analysis these two files need generating only the first time around the loop, thereafter the step where they are made can be missed out to save cpu time. The 'software DPP executable' is object code generated from the C source code for the SWDPP. This source code has been successfully compiled on several flavours of UNIX.*

Chapter 5: Results

Chapter five presents the results from the original work described in chapters three and four, dividing these into three broad areas:

- unit and system level testing of the onboard software on the bench and at other sites as the instrument was integrated;
- early results from the onboard software used on the spacecraft in orbit;
- experiences with the software DPP in analysing early diagnostic data from the spacecraft in orbit

5.0 Ground Testing the Onboard Software

5.0.1 The test set-up

The equipment used for testing the software comprised flight representative DPP and IC boards in a 'breadboard' enclosure, plus some spacecraft simulation units and a host workstation. These 'engineering model' (EM) DPP and IC boards were made available to the DPP software development system once the 'qualification model' (QM) boards started to become available. Though built from commercial quality parts, these boards were in every other way identical to their flight equivalents, and were modified as time went by to reflect improvements in the design of the 'real' hardware. The host workstation was a Sun SparcStation running Solaris 1.1.2, and communicating via a second (private) ethernet link to a rack system containing simulators for:

- the spacecraft interface, allowing software executing on the Sun to communicate with the IC at representative data rates, as well as higher rates for testing;
- the 'analogue electronics', e.g. current monitors, clock sequence generator etc.;
- the CCD data stream.

This last was a relatively simple device built around a field-programmable gate array, and known as the 'minimum DPP data source', or minDS for short (Rees, 1996), which is described in more detail later on.

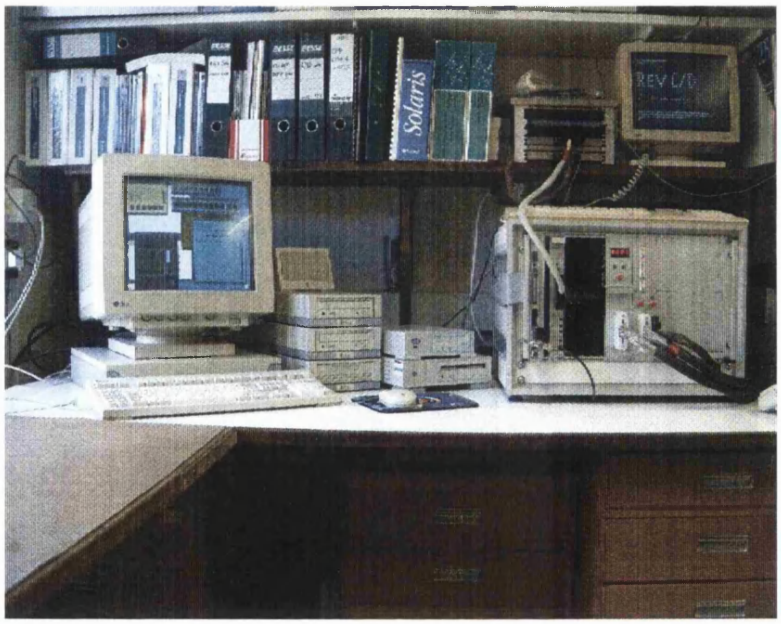


Figure 5.1: *Photograph of the development system used for the DPP software. To the left is the Sun workstation host, and to the right is the VME rack system containing, from left to right, the VME master computer, the spacecraft interface simulator, the analogue electronics simulator and the minDS. Above this rack is the enclosure containing the EM IC and DPP.*

Figures 5.1 and 5.2 show a photograph of the development system, and a schematic representation of it respectively. All of the hardware shown was developed at MSSL, along with all of the low level software. The user interface software running on the Sun was provided by The Space Research Organisation of the Netherlands (SRON). This user-interface comprises several discrete software modules (Philippus and den Herder, 1994):

- The Conductor

  This is the core unit, handling communication between the various modules, of which the spacecraft interface simulator appears as only one. This unit also provides a browser for the raw telemetry packets, and a tool for building and storing packets to command the instrument;

- The Configurator

  This unit provides a method for configuration of various instrumental parameters (not used in this work), and a convenient widget to command the instrument into the various states (i.e. 'kernel mode', 'configure mode' etc.);

- The Inspector

  This unit gives a 'quick-look' at the incoming CCD data, displaying a surface map of the image, and a summary of the various settings and statistics. The Inspector also generates FITS files for each CCD's pixel data;

- The Commander

  This unit is used for executing scripts. An EGSE control language (ECL) was devised as part of the user interface, and this was used extensively to both partly automate the DPP software testing, and to ensure that the configurations used were identical from one session to the next.

Also supplied by SRON was a CCD image Analysis tool (CIA) which could be used to look in detail at the FITS files created by the Inspector (Philippus, 1993 and 1995).

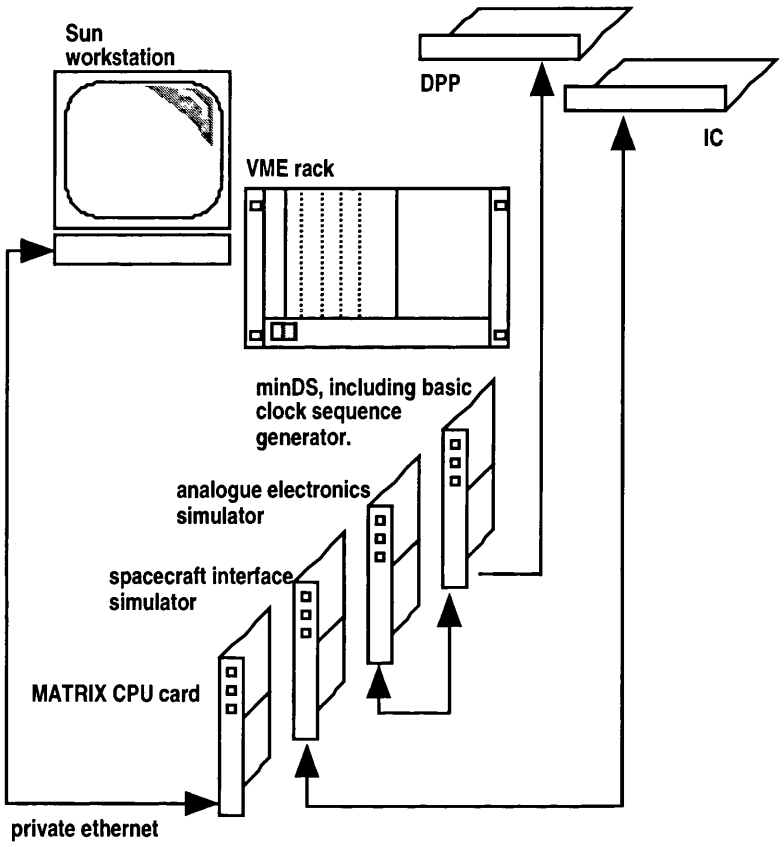


Figure 5.2: Schematic of the development system. The four boards in the foreground are connected to each other via the backplane in the VME rack, and the two boards from the onboard electronics set (IC and DPP) are connected to each other via a backplane in their enclosure.

A typical testing session would start by powering each item 'on', then using the Commander to load the software into the IC. When loaded, the IC could then be put into 'configure' mode (the most basic situation when running the software

just loaded, effectively an idle state where the IC monitors and relays housekeeping information and handles requests for programming the DPP). Next the Configurator would be used to load the DPP software, piping it through the IC software, as with the parameters for the IC and the DPP's parameter bank. All of the packets sent would be logged in the Conductor, and therefore easily recalled and edited: for example, the packet loading the part of the parameter bank with the setting for the science mode (e.g. HER, SES etc.) could be pulled up, a single digit changed and the packet sent, making the switching between RGS modes relatively convenient.

Note that not only is the DPP an 'embedded' processor, already making it relatively difficult to 'see' inside, but that it can only be communicated with via the IC. In full, then, processes on the Sun communicate with processes on the VME cpu, which communicates with the IC which communicates with the DPP. This level of remoteness was to cause some difficulties in debugging the DPP software, especially during the earliest phases while the behaviour of the system was still being learned. The system complexity was a further driver in the quest to find simple and repeatable testing and measurement techniques.

5.0.1.1 Using the minDS for testing

The minDS provides simultaneous outputs for each CCD node, in the form of a 12 bit value which can start at either 4095 and be decremented, or at 0 and be incremented. The counter is 10 bits, so it rolls over at 1023, meaning that the minDS can provide a complete row of pixels for a readout from a single node in 1x1 OCB, and when used in dual node readout the count is spread over two rows, as can be seen in figure 5.3.

```
512   513   514   515   516   ..............  1023
0     1     2     3     4     ..............  511
512   513   514   515   516   ..............  1023
0     1     2     3     4     ..............  511
512   513   514   515   516   ..............  1023
0     1     2     3     4     ..............  511
```

Figure 5.3: *Output from one node of the minDS. The table is a representation of the contents of the pixel bins in x (along) and y (up). The numbers are the energies in the pixel, and stop at 1023 owing to their being only 10 bits in the counter. Were the counter set to count down from 4095, then the lowest figure reached would be 3071. The readout pattern is 512 x 384, hence the count range is wrapped over two rows, with pixels of the same energy value occurring at the same x-position but in alternate rows.*

With judicious use of the lower and upper thresholds, ranges of pixels can be made to form certain patterns, and having a known matrix the results can be forecast for the contents of each statistical counter. For example, with the upper threshold set to reject pixels with value 1023 and higher, and the lower threshold set to remove pixels of 1021 and lower, then a simple pattern of pixels in a column will remain (figure 5.4, A). Since the input pattern is known, the output can be checked to see that: the remaining pixels have energies of 1022, and have coordinates following the pattern (510, 1), (510, 3), (510, 5) etc; the number of pixels rejected as above the upper threshold is 192 (one every other row from 384 rows); and that the number of pixels rejected below the lower threshold is (192 x 512 +

192 x 510) = 196,224 (alternate rows are either completely rejected, or rejected all but two: one kept, one above the upper threshold). Increasing the gap between the upper and lower thresholds to two and three apart will yield pixel pairs and triplets, as shown in figure 5.4, B and C.

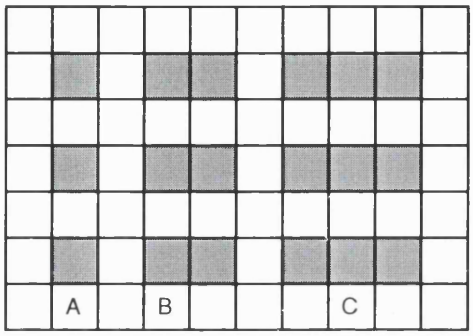


Figure 5.4: *Section from a CCD surface showing the results of balancing the settings for upper and lower thresholds (when used with the minDS). In 'A', the thresholds are one apart, meaning that there is a pixel in every other row at the same x-position (readout is in 1x1 OCB). In 'B' and 'C' the settings are two and three apart respectively, yielding the patterns shown.*

5.0.1.2 Generating more complex patterns with the minDS

Whilst allowing some testing of eSES, SER and HER modes, the patterns generated by the minDS are clearly insufficient. It had been expected that a more complete data source would be built, possibly with the ability to replay data taken with the integrated instrument (at Panter, for example), but owing to time constraints this never reached the top of the priority list and the author had to find another way to make do. Happily, a workaround could be found using the minDS. With a fixed pattern being generated, not only do we know what the data looks like, we also know where it resides in the internal data buffers. Using this knowledge, the software can be modified to include some lines adjusting the data

already taken by the front-end. In practice, this was done by making a complete clone of the software directory and modifying this to ensure that none of these experimental edits were left behind when returning to the original state. Of course any modification to the software under test is undesirable, but with no other options then the best must be made with what is available. Risk is minimised by modifying only a clone of the software and by keeping the modifications simple. In this case, at the last moment before the pixel array is handed over to a subsequent processing routine some lines are added to alter certain values, as illustrated in figure 5.5.

```
EVENT_BUFFER_C(3).Y-coord := 2;
EVENT_BUFFER_C(4).Y-coord := 4;
```

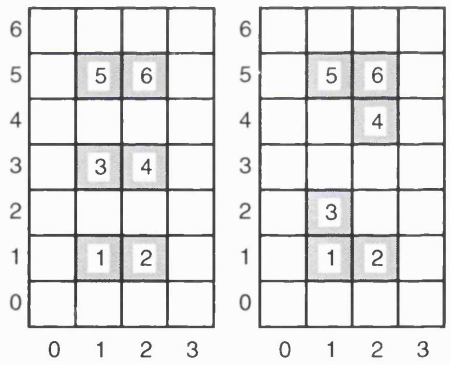


Figure 5.5: *Two sections from a CCD surface. In the frame to the left is the pattern resulting from the settings on the minDS and the thresholds in the DPP. Above the frames are some lines from the test software, showing that pixels '3' and '4' have their y-coordinates altered on-the-fly to new values, resulting in the pattern shown in the frame on the right.*

Though working out the values by hand is laborious, all of the possible split event shapes that might be encountered can be synthesised, without influencing the real-time performance of the hardware parts of the system. Note that care is needed with this technique to ensure that the pixel arrival order remains the

same as this is critical to the routines used in this software. Figures 5.6 and 5.7 show the results from such a pattern adjustment in the form of screenshots from the CIA software mentioned in 5.0.1.

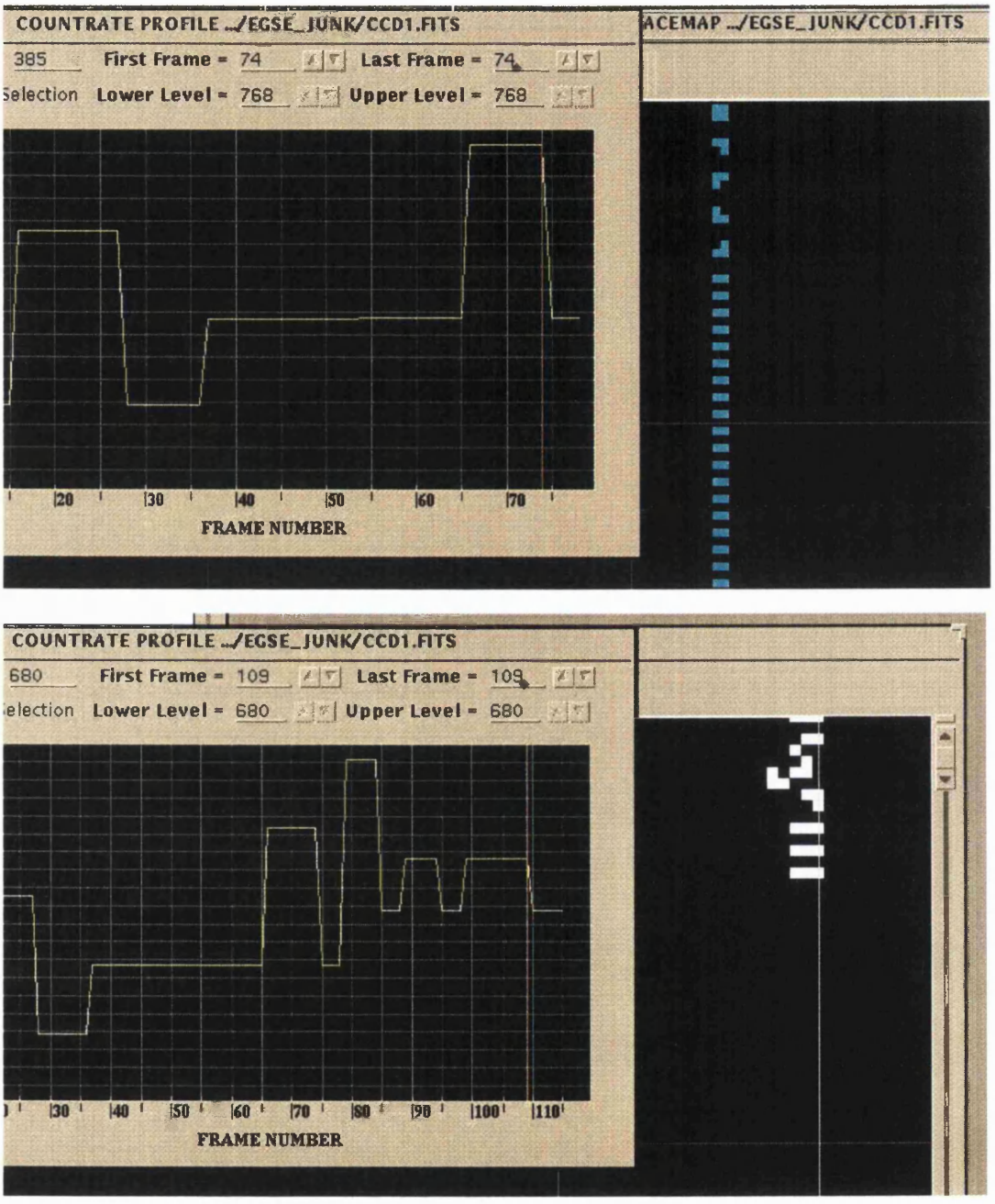


Figure 5.6: *'Before': Complex pixel patterns synthesised from the data generated by the minDS. The upper frame shows the pattern in the C side (based on a column of alternate pixel pairs), and the lower frame side D (based on a column of alternate pixel triplets). The data is taken in HER mode, in order to preserve the shapes for checking. Overlaying the surface map images is a plot of the 'countrate profile' which shows the number of pixels taken in each frame during the course of the run. Each screenshot shows the data from a single frame (number 74 for the side C image, and number 109 for the side D image). The rearmost window is a 'surface map' where the energy of the pixel is represented on a colour scale, but in these cases, with the pixel energies being only one or two levels apart the colour mapping fails to distinguish them.*

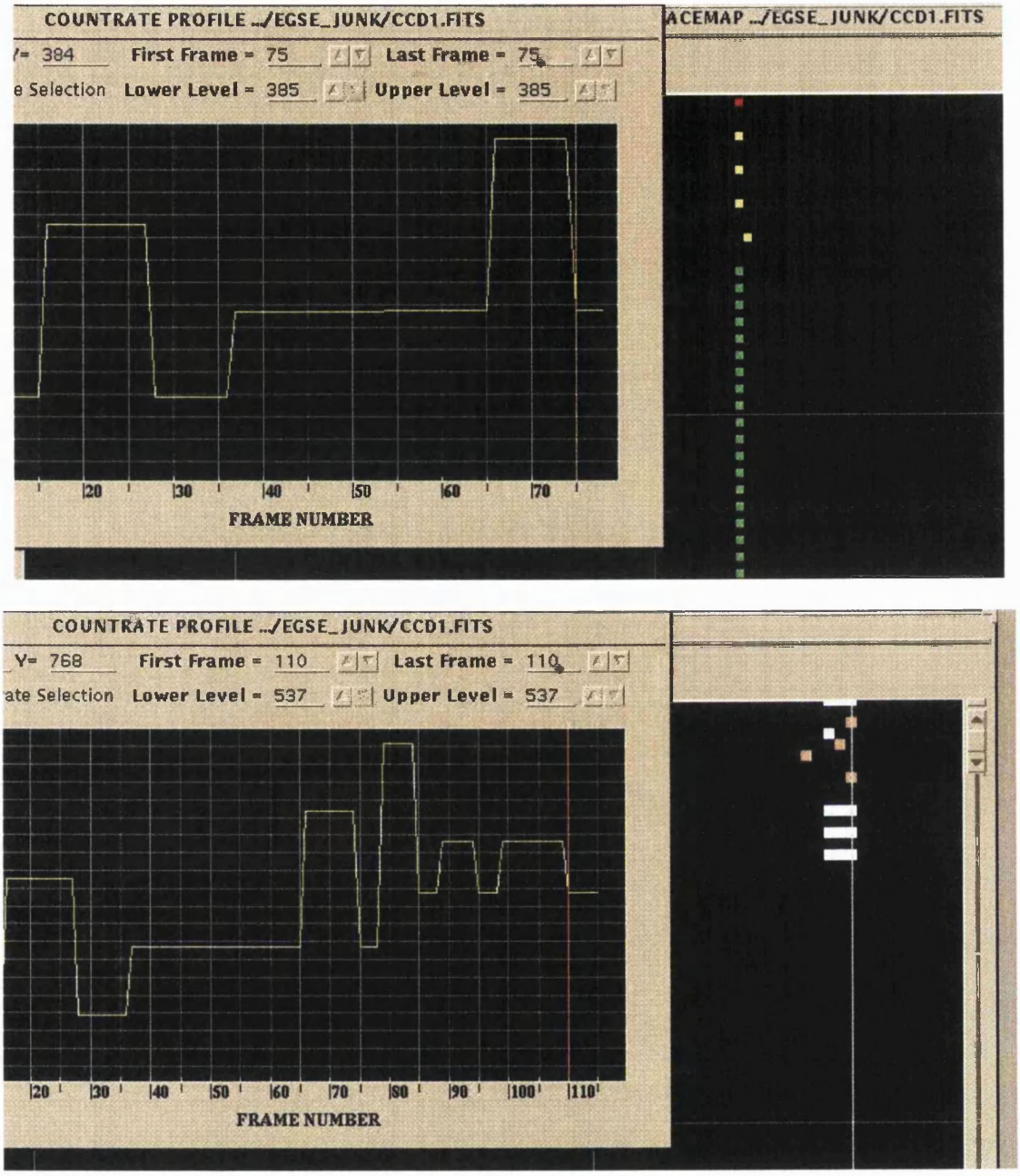


Figure 5.7: *'After': These screenshots echo those on the preceding page, but now the data is taken in SER mode. The shaped events have been replaced by single pixels with higher energies, the positions of which can be understood by noting that for each cluster the first arriving pixel is to the upper left (side C) or upper right (side D), and this is the location for the summed energy. Looking in the countrate profile windows in the foreground of each panel, an abrupt drop can be seen in the number of pixels in the frame being viewed. The 'wiper' in each window is at the location of the frame in question, 75 for side C, 110 for side D.*

5.0.1.3 Throughput Performance Testing method

Until the cross-calibration campaign at Panter in March 1999, the DPP s/w had not been exposed to any 'realistic' data, and so questions remained about the likely throughput of the system—especially when performing its most complex mode, SER with HIP enabled. Measurement of this throughput is complicated by:

- Limitations of the data fed to the input – 'real' data will vary in ways which are practically unreproducable with the available equipment at SRON or MSSL, where the best that can be achieved will be data which is representative for a single frame, but which is repetitive. This, though, can be a relatively harsh test *in extremis* since the profile of real data would of course vary from frame to frame;
- Limitations of the rest of the data path, particularly the spacecraft simulator, which could be overwhelmed by the data produced by the instrument: this is exceptional of course, since the simulator can handle an even higher data rate than the actual spacecraft, nevertheless this made it harder to explore the maximum throughput of the instrument, which is information of interest.

For the speed tests the author modified the DPP hardware by adding some extra integrated circuits to enable some address decoding to a flip-flop. The output of this flip-flop was connected to a logic analyser. Setting and clearing of the flip-flop could then be actioned by extra lines in the software. By this technique

the time taken to execute sections of code could be measured, and a picture of the overall performance estimated.

These measurements are useful because they provide a mechanism by which forecasts can be made, but need to be treated with caution because:

- not every permutation has been measured;
- the forecasts are necessarily a little simplistic, e.g. the permutations of influence of the FIFO buffering are not explored in depth;
- the act of setting/resetting the flip-flop itself consumes cpu time, though this is negligible with respect to the DPP processing speed.

Note that timings for set-up at the beginning of a frame, and writing the statistics at the end are not included as they are small, and in any case are further reduced by the latency between frames, which can be very large when there is a telemetry bottleneck.

5.0.2 Results from Basic Mode and HTR

With a known pattern of data being generated by the minDS (5.0.1), the output from these modes could be checked quite easily by browsing the raw telemetry. Figure 5.8 shows a screen shot from an EGSE user-interface display where the contents of a packet of data sent from the IC can be viewed. The telemetry packets could also be stored and viewed in a text editor.

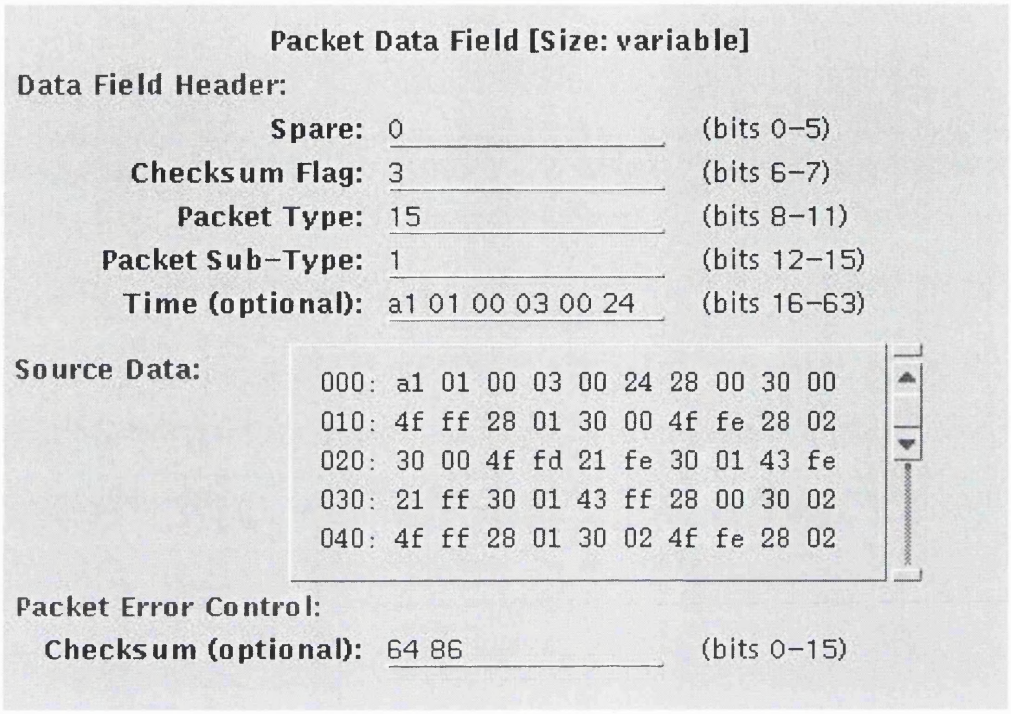


Figure 5.8: *Screenshot from the telemetry browser in the user-interface's 'Conductor' unit. Each hexadecimal digit in the source data field represents four bits, so there are four digits per word. x, y and energy have most significant nibbles of '2', '3', and '4' respectively, and after the first three words which relate to something else, the pixels can be seen following the pattern: [2800, 3000, 4fff], [2801, 3000, 4ffe] and so on.*

Also tested were the permutations of data overload, with the lower threshold being reduced such that pixels were lost as the telemetry output exceeded the available bandwidth, and also with the spacecraft simulator output data rate reduced, causing the instrument controller to have to pause the readout. The pur-

pose of these experiments was to demonstrate the ability of the software to stand the abuse without crashing, or behaving in an unexpected way.

Using the logic analyser as described in 5.0.1.3, basic mode was measured as needing 118µs to process each pixel, implying 8475 pixels could be processed per second with the hot items processing disabled:

OCB Mode	R/O Time per frame	Pixels per Frame
1x1	3.94s	33,389
3x3	0.57s	4,830

Table 5.1: *Summary of the throughput capability of the MSSL DPP in basic mode, with hot items processing disabled.*

Enabling the hot items processing yields three possible outcomes for a particular pixel, which can be:

(a) kept (slowest: all hot column segment and pixel testing, but no matches, so all software steps are executed);

(b) rejected as part of a hot segment (fastest because it short circuits hot pixel testing);

(c) rejected as a hot pixel (has been tested and kept as part of a hot segment).

Table 5.2 shows the time taken to process each of these types of pixel, with the number of pixels processed per second assuming that the population of pixels is entirely composed of the type in question.

Outcome	Time per pixel	# per second	Designator	Proportion
a	194μs	5,154	N	p max
b	122μs	8,196	M	q max
c	158μs	6,329	K	r max

Table 5.2: *Time taken by the DPP to apply the different levels of processing to each pixel when HIP is selected. Table key: 'a' is the worst case: hot column segment and hot pixel testing is performed, but the pixel is kept anyway; 'b' is when the pixel is discarded after hot column segment testing; 'c' is when the pixel is kept after the 'b' case, but is then rejected after the hot pixel testing; N, M, K and p, q, r are arbitrarily chosen labels, with* $p + q + r = 1$.

How can this information be used to forecast the throughput? If 'V' is the instantaneous throughput of the system in terms of pixels per second, then its value will vary according to the relative proportions of each class of pixel:

$$V = N(1 - r - q) + M(1 - p - r) + K(1 - p - q)$$

Note that simply enabling hot items processing has increased the per-pixel time consumption to 194μs per pixel from 118μs, depressing the maximum possible throughput as shown in table 5.3.

OCB Mode	R/O Time	Pixels per Frame	% change
1x1	3.94s	20,309	–39
3x3	0.57s	2,938	–39

Table 5.3: Depression in throughput with HIP selection. These figures represent the worst-case—all pixels are given all tests, and all are kept—in practice, selecting the HIP function implies the presence of known hot items to be removed, lifting the throughput by some amount.

High Time Resolution mode was tested using the same methods, and the throughput measured, with the time taken to process a pixel reducing as expected since only the x coordinate is extracted. In this case, hot items processing is limited to hot segment processing (see table 5.4).

	Time per pixel	Number of pixels per second	No. of pixels per frame (1x1)	No. of pixels per frame (3x3)
HIP Disabled	84µs	11,904	46,901	6,785
HIP Enabled	135µs	7,407	29,183	4,222

Table 5.4: Summary of throughput of the MSSL DPP in HTR mode

5.0.2.1 Results from HER, SER and eSES

Functional testing of these modes proceeded by constructing synthesised data sets and comparing the results with the forecast. This was found to be an effective method, though it required careful record keeping to ensure that all of the results were consistent.

Speed testing these modes required breaking the measurements of the software modules up in order to consider the times as far as the pixel buffer (the same for HER, SER and eSES, results shown in table 5.5) and then the time taken to read the pixels out again according to the different processing options.

Processing type	Time taken
HIP off	90.4µs
HIP keep	174µs
HIP discard in hot segment	126µs
HIP discard as hot pixel	140µs

Table 5.5: Per-pixel processing times for the input section of the HER, SER and eSES modules. 'Time taken' is the amount of time taken to process a pixel from the input FIFO to the internal software buffer.

Output timings from HER mode are easy to describe, there being only two possible outcomes—pixels are 'connected' and therefore kept, or they are unconnected, and are kept only if they are above the programmed acceptance threshold.

Processing type	Time taken
Pixel is connected	92μs
Pixel thresholded and kept	97μs
Pixel thresholded and discarded	70μs

Table 5.6: *Per-pixel processing times for the output section of the HER module. 'Time taken' is the amount of time taken to process a pixel from the internal software buffer to the output FIFO.*

Output timings from SER mode are harder to quantify since there is such a variety of paths through the routines. Figure 5.9 illustrates this in the form of a time-flow diagram.

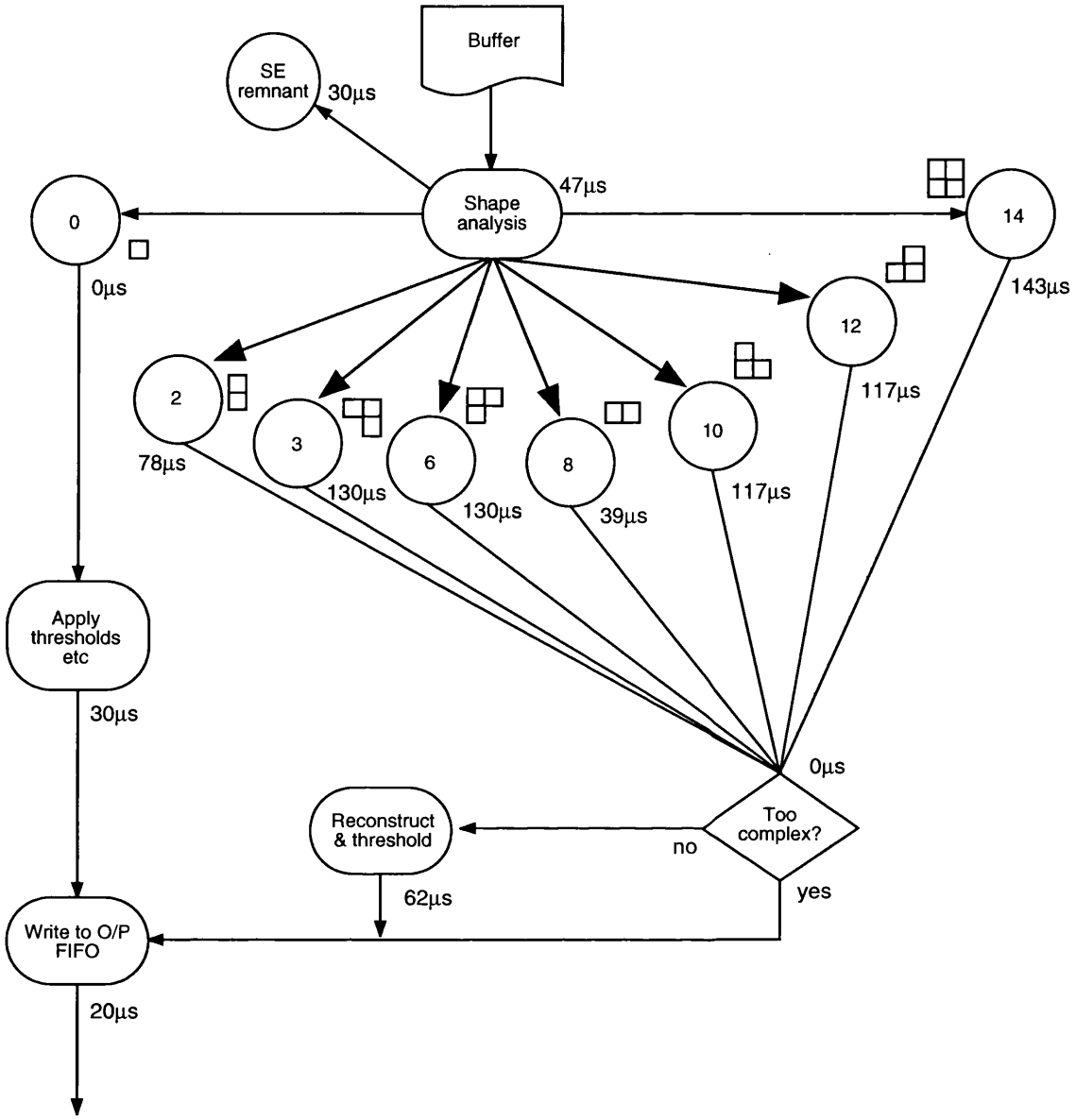


Figure 5.9: *Time flow for the SER module. All of these pixels pass the 'shape analysis' section, but their ongoing cpu time consumption is dependent on the outcome of that stage. The different paths are indicated, and a picture of the throughput can be built up by estimating the relative proportions of each type of event (see text for details).*

The total time taken to process each pixel can be found by summing the appropriate paths.

For split events the components are:

1 x 174μs (HIP kept pixel from part 1)

1 x 47μs (inner shell analysis)

1 x ? μs (outer shell analysis time, differs with shape)

1 x 62μs (reconstruction)

1 x 20μs (writing out)

and for single events:

1 x 174μs (HIP kept pixel from part 1)

1 x 47μs (inner shell analysis)

1 x 30μs (thresholding)

1 x 20μs (writing out)

This apparently makes the forecasting of throughput very difficult, but this is helped by a balancing effect: the more complex events take more time to reconstruct, but they yield more pixels which can be deleted as 'remnants', so:

$$\text{Average time per pixel} = \frac{1 \times \text{Total Time} + (n \times \text{SE remnants})}{n + 1}$$

Where 'SE remnant' time is 30μs (see figure 5.9). A summary of these results is given in table 5.7.

Possible route	Inner shell analysis	Outer shell analysis	Number of pixels in the event	Total time for seed pixel	Average time per pixel	Number of pixels per second
SE remnant	—	30μs	—	204μs	—	—
Single	47μs	—	1	271μs	271μs	3,690
Type 2	47μs	78μs	2	381μs	293μs	3,413
Type 3	47μs	130μs	3	433μs	281μs	3,558
Type 6	47μs	143μs	3	446μs	285μs	3,509
Type 8	47μs	39μs	2	342μs	273μs	3,663
Type 10	47μs	117μs	3	420μs	276μs	3,623
Type 12	47μs	117μs	3	420μs	276μs	3,623
Type 14	47μs	143μs	4	446μs	353μs	2,833
Average number of pixels per second						3,489

Table 5.7: *Summary of the event processing times in SER mode. Note that though significantly different routes can be taken through the algorithm, the net results in the 'number of pixels per second column' vary by only a small amount.*

Considering the results summarised in table 5.7, it can be seen that the most extreme time difference is between the expected cases of a single pixel and a block of four, but even in this case the difference is only 30%. The two dominant event types (single and vertically split pair) differ by only 8%. Note that the discussion so far has been only on the single and simple events. SER and eSES modes perform in the same way as far as these events are concerned, but the modes differ in their treatment of the remaining population, though this has only a small effect on the outcome. In SER, where all complex pixels are retained, after the analysis of the outer shell components a complex pixel event component will

not be reconstructed, but will be written to the output FIFO, saving 62μs every line. In eSES, all of the inner- and outer-shell analyses are performed, but complex pixels are discarded, saving 50μs against the kept case. A summary of the forecast throughput by mode is shown in figure 5.10, relative to the typical maximum input to the IC.

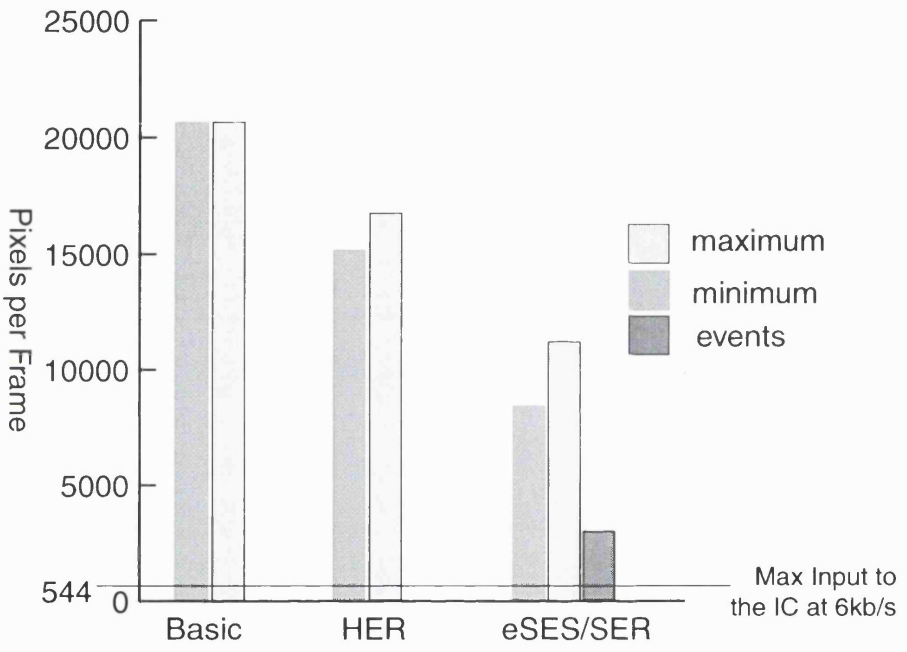


Figure 5.10: *Overall throughput forecast for the MSSL DPP by mode. Several assumptions are made in producing this graph: 1) Readout is in 1x1 OCB, 2) The maximum input to the IC is inferred from the telemetry rate, assuming that the IC is in 'packed data' mode (the default, whereby there are 5 bytes per pixel) and allowing 10% overhead for packetisation (Al Janabi, 1999); 3) A total of 1170 hot items are assumed, as suggested by Branduardi-Raymont, 1996.*

It can be seen from figure 5.10 that there is a comfortable margin between the number of pixels that the software can successfully process, and the maximum number which the telemetry can accommodate, such that though more exact numbers could not be calculated without knowing the actual profile of the data, some confidence could be held that the processing speed of the DPP would not be an issue that would limit the performance of the RGS instrument.

5.0.3 Ground Testing External to MSSL

Other testing opportunities for the onboard software were at SRON, and at the Max Planck 'Panter' Facility. Since the EGSE user interface at both SRON and Panter was the same as that in use at MSSL a very similar test facility was available, this time with 'real' electronics replacing at least some of the simulators.

When at SRON, the CCDs were only occasionally cooled to their operating temperature, but another method was also available to produce useful data. Incorporated into the analogue electronics is a 'test-pulse generator' (TPG) which can be used to insert data into the processing stream apparently as if they came from the actual CCDs. This too is a limited facility, but it gave an alternative method of generating patterns for testing the software. Apart from occasional integration visits, most of the testing was carried out in parallel (i.e. at MSSL and SRON simultaneously), and the author only has email and verbal results for these out of MSSL tests—mostly relating to the (long) debugging phase. Happily most of the problems along the way related to questions of understanding how the DPP should be operated, particularly with respect to the loading and construction of the various tables. Being a complicated unit to operate, it was useful

to have the opportunity of spreading the understanding of it, and to enhance the comprehendability of the supporting documentation.

The independent testing of the software was also invaluable in finding errors, obviously a different test set-up may throw up different problems—but just having a new mind on the job can help greatly to spot those insidious errors to which the author of a piece of work simply becomes blind. An example of the latter came with the discovery at SRON that the SER routine was blind to events of shape '10' (one of the rotations of a group of three pixels in an 'L' shape, see table 3.3). This was not a bug in a part of code—there was no code to process them at all! Early in the coding process the author had generated a table to work to for all of the searching routines, and this shape was erroneously missed off. Since all of the subsequent testing continued to use this table as a prototype, the error remained invisible to the author, even though once seen it becomes all too obvious. This might have been found by another programmer examining the code, but this sort of approach will always be limited by available manpower and time, and hence cost. A valuable and pragmatic approach to this must be to section the task, and action peer reviews of at least key items. This is hard to quantify, but the software design stage will generate certain diagrams and matrices and tables, and the writer of these must develop a sense of those which are of particular importance, and have them reviewed by a new mind.

Another error found at SRON could not be reproduced at MSSL, and ultimately needed a trip to the SRON site to untangle what was occurring. The instrument was being operated in a very non-representative way, with the 'telemetry' link from the IC to the EGSE being throttled such that the IC was frequently required to pause the reading out of the CCDs. This occasionally meant

that pixels were lost at the DPP's front-end, and occasionally that the DPP would be idle, but both of these conditions had been extensively explored at MSSL, and were not a concern. At SRON, however, after some time—sometimes minutes, sometimes hours—the DPP software would appear to crash. This is the worst sort of problem to work with in such a system, and the problem was ultimately found by the author operating the system at SRON with the DPP circuit board plugged in via an extender card (i.e. sticking out of the enclosure) and waiting for the 'crash' to occur. Once finally in the hung state, a logic analyser was used to investigate the state of the hardware. It was quickly established that the software was not 'wedged', but was in fact executing something, but what? The address bus was then examined to look for an address which was being accessed, and after some false starts, one was found which was not a common value. Knowing the value of an address read in this loop, whatever it was, gave something to work with. The contents of this address could be found from the machine code file, and tallying this with an intermediately produced assembly language version of the software, it was possible for the author to deduce what the cpu was doing. It was in fact working in the 'heartbeat' loop (see 3.0.1). Comparing the Ada source code with the knowledge of the system state when the error occurred, a bug was found in the placement of when the routine interrogated the status register to see if the output FIFO was full: under certain conditions it would be possible for the software to continue to think that this FIFO was full, even after it had been emptied. Whether the software could be considered to be 'crashed' at this time is a moot point, nevertheless it needed to be fixed, and was undoubtedly an error on the part of the author—all too obvious in the inevitable *post facto* way that is the nature of this type of endeavour.

Testing of the DPP software was carried out at SRON for all of the functional modes and conditions, but the software testing at Panter was more patchy. It should be remembered that the main purposes of the tests at Panter were to measure the performance of the mirrors and gratings, and to get calibration data on them. The test set up was more rigorously organised, and though some out-of-hours system access was granted for DPP software testing, the author had to work within the framework of existing scripts and procedures in order to guarantee that 'real' measurements—running from about 0800–2400 each day—would never be interrupted, delayed or compromised. Nevertheless, this apparatus provided a third arrangement of testing kit and use was made of this: it was useful to see at least more confirmation of the software executing correctly, even if no *in extremis* tests could be conducted.

One notable change in the accommodation of the software adoption occurred at the testing campaign in April 1999 however, when it became clear that the DPP software might be useful in alleviating a hardware related problem. This problem had first started to appear in the QM testing phase of the analogue electronics where a faint herring-bone pattern of noise could be observed in the CCD images. It was ultimately concluded that this was crosstalk due to ripple on the CCD bias voltages, which, though meeting the original specification, led to a greater effect when summed over many frames. By the time this was clear, however, and all of the other potential sources of noise (EMC etc) had been eliminated, the FM test programme was under way. Close examination of the data revealed a tendency for the noise to appear in isolated pixels, meaning that the HER mode with its greater finesse at applying noise thresholds could potentially make a lot of difference to this, without having to break the hardware schedule.

Accordingly, overnight on the 8th of April 1999 the author was able to demonstrate that the HER mode could run without crashing for all the time its was used and explored by the author, and that the version of the code including this mode appeared to harmlessly switch back to basic mode on demand (of course it was designed this way, but it still needed to be formally demonstrated). During official runs on the 9th of April, using acceptance thresholds suggested by the author, the noise pattern was significantly reduced with no apparent degradation in the scientific data compared with previous runs. Figure 5.11 shows before-and-after flat-field surface maps illustrating the marked difference found.

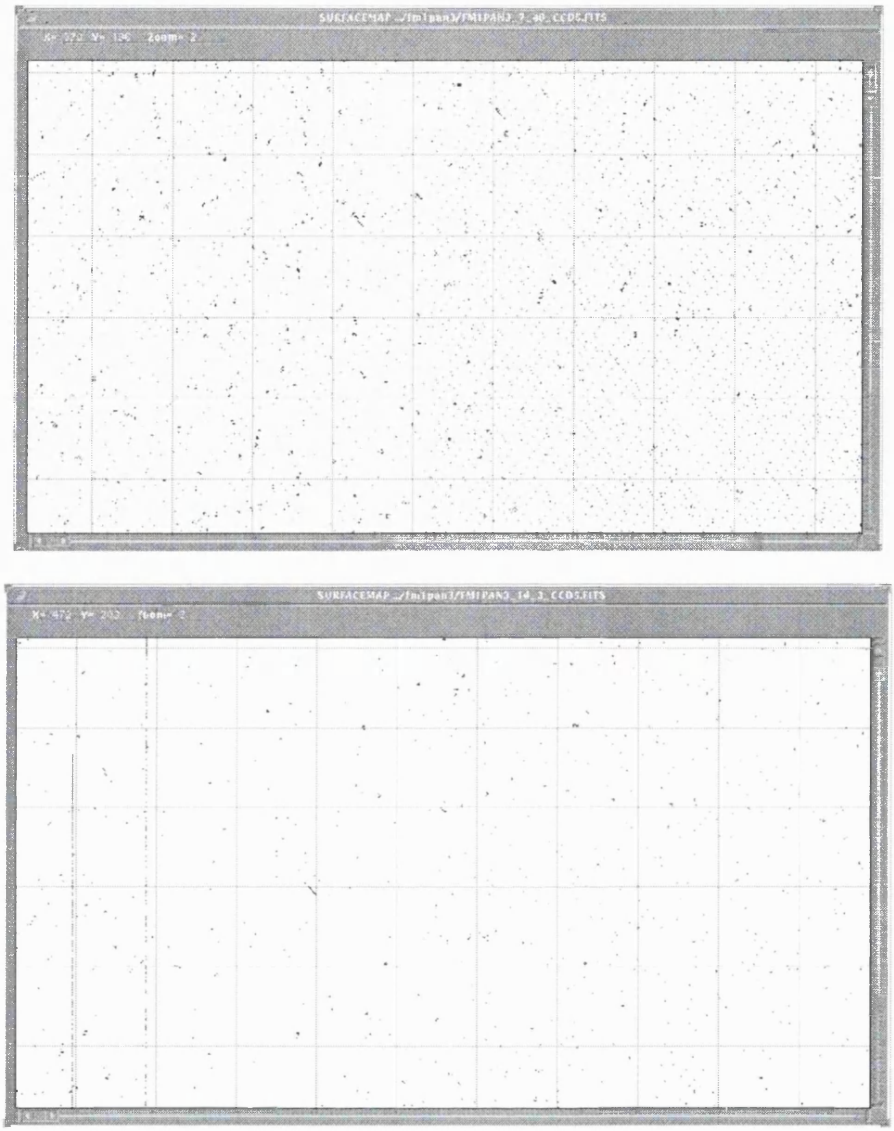


Figure 5.11: *Before and after views of 'flat-field' data taken in basic spectroscopy (upper panel) and HER mode (lower panel). The images are 'surface maps' of CCD5, with x and y axes along and up the page, and energies mapped to colours. Note the pronounced herring bone pattern visible in the upper panel, especially in the right hand side. All of the pixels are at very low energies, and the energy colour map has been telescoped and inverted to show them more clearly. The later run in HER was without the hot column tables loaded (not under the author's control), and unremoved hot items can clearly be seen to the left of the centre of the lower panel image.*

The decision was then made by the manager of the testing programme that the Panter tests would proceed using HER as the default mode. It was also later decided that HER would also become the default operational mode for flight, and that no further investigation of the hardware source of the crosstalk would be pursued. Work is still progressing on characterising the impact of this source of noise on the scientific data (e.g. Tamura, 2000).

Panter testing of the other modes was limited to out-of-hours work, where only a go or no-go result was possible—the X-rays available then were coming from the in-flight calibration source, and were hence not representative of cosmic source data. It was, however, useful to have all of the modes exercised by data of this sort, especially for long durations (hours), as they could have shown some real-time issue that would need to be explained.

5.1 Early Results from Flight

5.1.1 Spectroscopy

XMM-Newton was successfully launched on December 10th 1999. The early orbit commissioning phase employed a much earlier version of the DPP software than the version 19 which was by then available. This was for the logical reason that since this earlier release (v16) had been in use for some time by ESA, they could have the maximum confidence that any problems found with the instrument would not be due to relatively untried software. The next oldest version available was a small upgrade on version 16 and it had been negotiated with ESA that at the start of the calibration and performance verification stage this would be uploaded in substitution. The assumption at this time remained that the default operational mode would be 3x3 OCB with HER mode selected in the

DPP. However, it soon became clear that the general particle background was significantly higher than the forecast (Peterson, 2000), and that the telemetry allocation was being saturated by this unwanted data. The consequences of this are shown in figure 5.12. The two panels show the surfaces of the nine CCDs in the bench: the upper panel displays x, y and number of counts per pixel mapped to colour; and the lower panel x and PHA. In this figure it is clear that the spectrum of the faint X-ray source (EXO-0748-67, a low-mass binary) is completely swamped by the high background level.

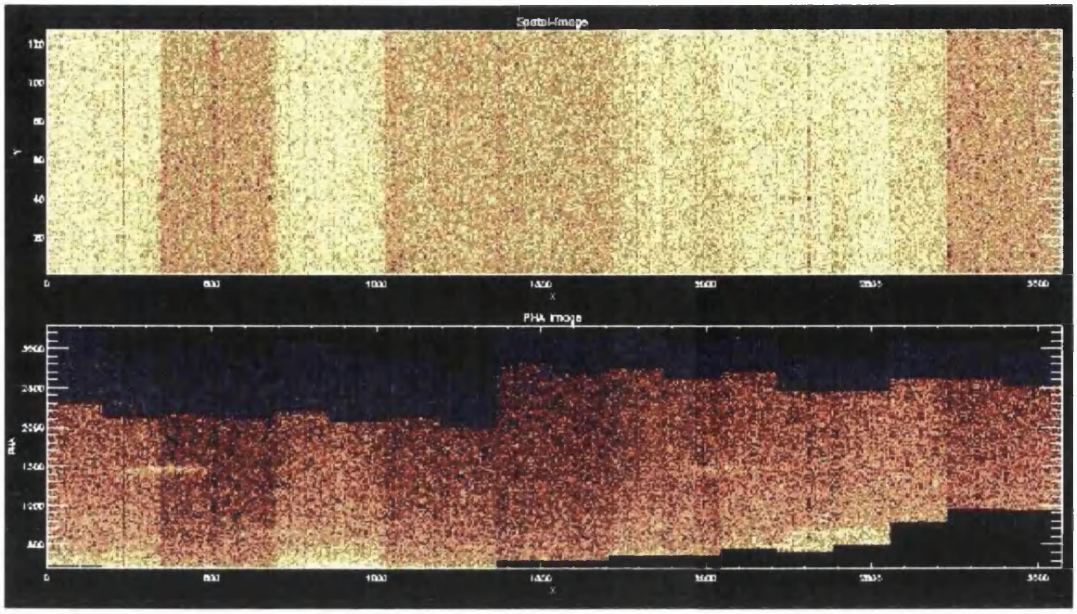


Figure 5.12: *Surface map of the CCD bench taken in the 38th orbit of XMM-Newton. The upper panel shows the 9 CCD surfaces side-by-side, with x across the page, and y up. The number of counts per pixel is mapped to colour. The lower panel shows the same data, but now the vertical axis is the PHA, and the colour map reflects the number of pixels occurring with a given energy. The data was taken with the DPP s/w in HER mode, and has been further processed on the ground by the RGS pipeline, which corrects for some instrumental effects, and performs split event reconstruction. In the lower panel the different settings of the gains per half CCD show quite clearly, giving the appearance that the image is formed from 18 vertical strips. The abrupt truncations at the tops and bottoms of the strips occur because of the selected upper and lower thresholds applied on-board, the fainter blue spots being split events reconstructed on the ground. In order to meet the telemetry allocation, the lower thresholds have been raised, with the result that it is not possible to discern the source from the background. The top panel should show a line running from left to right, being the dispersion direction of the gratings with the lowest energies at the left. The lower panel should show plots of this spectrum with the different spectral orders separated in the y axis. In fact no detail can be seen, but for some hints of the non-dispersed in-flight calibration source (the four short horizontal lines).*

Though not yet delivered, the author had several further iterations of the on-board s/w to offer; and when asked for his opinion, the author was able to advocate the use of the eSES mode in version 19 of the software in order to substantially reduce the background. This version was uploaded on the 27th of March in substitution for the proposed upgrade for version 16.

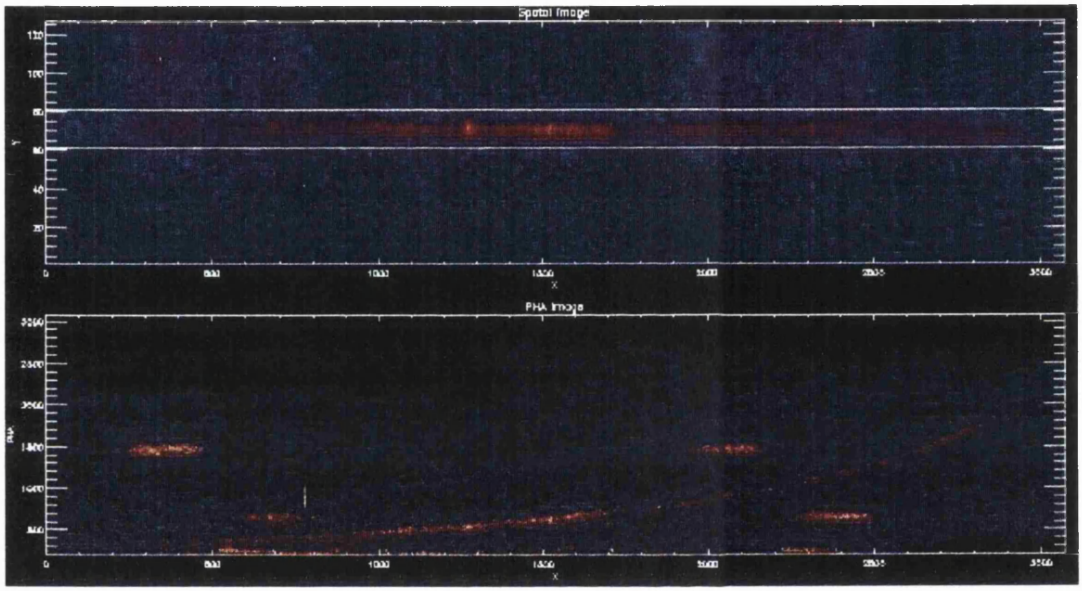


Figure 5.13: *Surface map of the CCD bench taken in the 55th orbit of XMM-Newton. The panels represent the same data as before, but now there are things to see. The key is that since eSES mode is so successful at reducing the background (by a a factor of 10 compared with HER mo de as used in figure 5.12) the lower thresholds could be also lowered, enabling a much improved ratio of wanted to unwanted data. With more detail available a spectrum can be plotted, and though a faint source, first and second orders from EXO 0748-67 can now be seen in the lower panel. The images in 5.12 and 5.13 have been selected to be as close as possible to being a 'before and after pair', being the same target, the same RGS (of the two) and similar exposures. No measurement is presented of the difference between the two, they are included only to provide a pictorial representation of the difference in the background present in the data received in the ground segment.*

Early tests on this data at SRON suggested that the reduction in background by using eSES ranges from a factor of ten to a factor of fifteen. Accordingly, version 19 of the software was retained onboard with eSES the default type of spectroscopy, from orbit number 49. A first clue to the likely improvement to the signal to noise quality of the data from the DPP between the two observations comes with a glance at the changes in the threshold settings used for both RGSs which was made possible by eSES, as shown in table 5.8.

	EXO 0768–67, HER			EXO 0768–67, eSES		
	LT	AT	UT	LT	AT	UT
1C	250	275	2600	175	200	2600
1D	275	300	2600	200	225	2600
2C	250	275	2600	175	220	2600
2D	275	300	2600	200	245	2600
3C	250	275	2600	175	240	3100
3D	325	350	2600	200	265	3100
4C	300	325	2600	175	270	3100
4D	325	350	2600	200	305	3100
5C	400	425	3100	175	300	3100
5D	425	450	3100	200	325	3100
6C	450	500	3100	175	350	3100
6D	475	525	3100	200	375	3100
7C	500	600	3100	175	375	3600
7D	525	625	3100	210	325	3600
8C	500	700	3100	175	300	3600
8D	825	1025	3100	200	500	3600
9C	800	1200	3100	175	600	3600
9D	825	1225	3100	200	625	3600

Table 5.8: *RGS 1 lower, acceptance and upper threshold settings used for each CCD for observations of EXO 0748–67 with HER mode or eSES mode selected. Note the large reductions for the LT in particular, made possible by eSES's ability to remove unwanted data by more precise methods.*

The figures shown in table 5.8 came directly from the 'observation data files' (ODF) supplied to the experimenters. These ODFs include FITS files for the spectroscopic data from each of the CCDs used in the observation (usually nine of course); an auxilliary file listing the observational meta-data (including instrumental settings such as the thresholds used); and as many diagnostic data FITS files as were taken during the observation.

To further quantify the effect of the DPP software on the quality of the science data from the CCDs, some further software was written to break apart these ODF data-sets allowing some numerical analysis of the data supplied. An immediately useful discovery was that the ground segment software was misinterpreting the meaning of some of the statistical data supplied by the DPP, and that this would need to be corrected: not a direct effect on the science itself, but important to the people monitoring and plotting the various diagnostic trends.

The individual FITS files from the ODFs were unpacked and analysed using further software written by the author, with the results plotted using IDL. Firstly, the auxilliary file from which all of the DPP reported statistics could be extracted: these being

- numbers of pixels exceeding the lower threshold side C/D
- numbers of pixels below the acceptance threshold side C/D
- numbers of pixels exceeding the upper threshold side C/D

These data could then be compared with the pixel data extracted from the spectroscopic files for each CCD, and some accounting carried out to relate the outcomes of the two processing modes, including the distributions of the event shapes, the results for the first hundred frames of which are shown in table 5.9.

CCD	Number of events by Shape (eSES only)							
								
1	1773	0	24	23	1	0	0	0
2	576	1	40	22	0	0	3	0
3	379	0	28	17	1	0	0	0
4	245	0	38	25	1	0	0	0
5	283	0	27	19	0	1	1	1
6	405	1	61	26	1	1	0	3
7	377	1	64	27	0	2	0	0
8	190	0	32	17	0	0	0	1
9	98	1	27	18	2	2	1	0

Table 5.9: *Numbers of events by shape for the same 100 frames considered in figures 5.14 to 5.22. Notice that—consistent with the expectation—single events dominate the account, with pairs of pixels the next most common. Vertically split pairs are rendered more likely owing to the wider channel stops between adjacent pixels in the same row.*

CCD	Total (HER)	Total (eSES)	Events/s (HER)	Events/s (eSES)	% Change
1	3437	1821	6.36	3.37	−47
2	3083	642	5.7	1.18	−79
3	3508	425	6.49	0.79	−88
4	2686	309	4.97	0.57	−88
5	2349	332	4.35	0.61	−86
6	3398	498	6.29	0.92	−85
7	3732	471	6.9	0.87	−87
8	3506	240	6.49	0.44	−93
9	2753	149	5.1	0.29	−95

Table 5.10: *Comparison of numbers of pixels telemetered by the DPP software between HER and eSES modes when pointing at the same object. The numbers refer to the same 100 frames as before, each frame having an integration time of 5.4 seconds.*

Options for directly comparing the actions of eSES on the raw data are limited by the small number of queue memory data frames that are available—four each for CCDs 1–3, three each for the others from the EXO 0748-67 observation in orbit 67. Plotting the queue data after removing those pixels below the lower thresholds used in the companion eSES frames reveals no particular shapes from the 'illegal' set which are more common than any other. Figure 5.14 shows a screen shot of a section of a CCD surface showing the raw pixels minus only those below the (eSES) threshold.

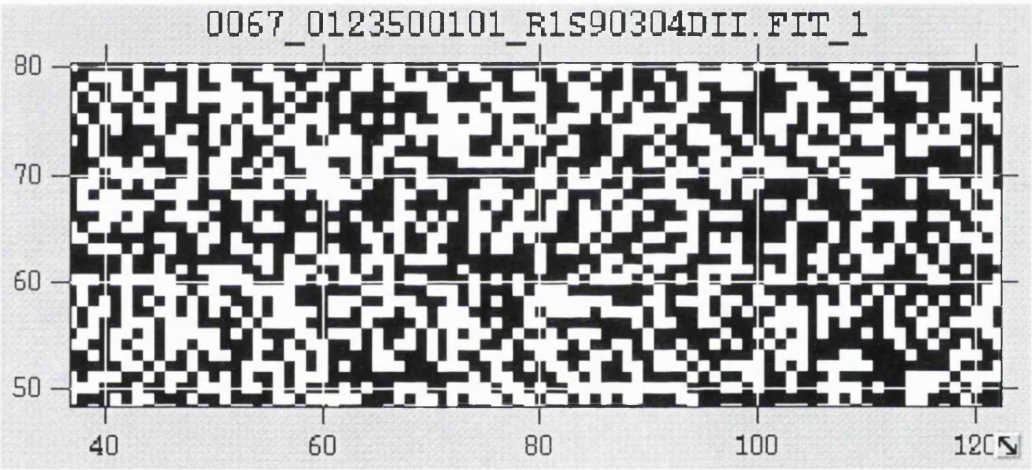


Figure 5.14: Section of CCD surface showing queue data after application of a lower threshold. The picture is of a region approximately 80 by 30 pixels from the middle of the CCD.This is from a single frame from within the same set for CCD 4 considered in table 5.9 and elsewhere. The 'colour map' has been telescoped to black and white and the region 'zoomed' only to enhance clarity. Empty squares are black. The raw data has been reduced by removing only those pixels below the lower thresholds as used in the on-board eSES processing which was being conducted at the same time (175/200 for C/D). There are queue memory frames for all nine of the CCDs, each presenting a similar picture.

Looking at figure 5.14 it can be seen that there are indeed many clusters of pixels which extend beyond the 2x2 region, though there are no particular predis-

positions for particular shapes. When looking at figure 5.14 it is helpful to recall that diagonal connections are not counted.

An interesting effect is gained by plotting the raw image to the screen and then sliding the colour map up and down such that the lower threshold is varied in real-time. The result observed is that the CCD surface abruptly switches from spartanly populated to swamped. The threshold level at which this visual effect occurs is quite high relative to the levels set for eSES mode, which is the expected result: i.e. the eSES processing is able to reduce the data by selecting out legal events from a field that looks to the eye as if it were already overloaded.

One hundred frames proves about the ideal number for producing legible plots at the scale that they are printed in this thesis, but there is no special reason for choosing the first hundred from either observation: in both cases the profile of the data is similar throughout, with no outstanding features which would make choosing a subset misleading. Table 5.10 highlights the change in the data rate to the ground segment. Considering the difference between figures 5.12 and 5.13 where a 'spectrum' becomes visible where it wasn't before, then the result is strongly suggestive of better use of the telemetry bandwidth: i.e. the signal to noise performance is improved, so the majority of on-board rejected pixels must be from the unwanted background.

Figures 5.15 to 5.23 present the results of this investigation graphically. Each figure has six plots, in two columns of three. All of the data are from EXO 0748-67, not an ideal choice being a somewhat variable source, but the options were constrained by availability of 'before and after' data, eSES having been adopted quite early in the mission. Two orbits are considered, 38 (DPP software in HER mode) and 67 (DPP software in eSES mode), with all data from RGS 1, RGS 2 by now having deactivated CCD 4.

The top two plots show numbers of pixels:

- above the lower threshold;
- below the acceptance threshold;
- above the upper threshold.

Immediately obvious in each case is that the number of pixels above the lower threshold (i.e. those actually entering the software processing stage) is significantly higher in eSES mode, consistent with the new thresholds as listed in table 5.8. Similarly, the numbers of pixels below the acceptance threshold in the HER plots is generally zero or very small, as is to be expected with the high setting for the lower threshold (always exceeding the *acceptance* thresholds that could be employed in eSES, see table 5.8). The shape of the plot of 'number of pixels above the lower threshold' is, in the HER case, dominated by the number of pixels exceeding the upper threshold. The number of pixels above the upper threshold in the eSES plot often appears much reduced, but this is because the statistic measure here includes only single and 2x2 reconstructable events: there are other pixels above the upper threshold, but they become classified amongst the pixels excluded on shape grounds. This is an important result since it demonstrates that single pixel upper thresholding as applied in HER mode is removing

pixels which are also part of an 'illegal' shape, and must therefore be leaving behind data which is ambiguous.

The middle two plots compare the 'number of pixels telemetered' (found by counting up the pixels listed in the appropriate FITS files) and the total found by tallying the threshold statistics. In the case of the HER plots a gap between the two plots is exactly accounted for by pixels removed by hot pixel or column processing. A graph where the two plots are superimposed indicates that there were no hot pixels, as was the case for CCD 4 at the time of orbit 38 (figure 5.18). In the eSES case, the gap is mostly due to the mis-shaped events which are rejected.

The last two plots are histograms of the pixel energies telemetered for each CCD. These were made by extracting the pulse height (PH) data for the first one thousand frames since one hundred proved too few to show much detail. The energies are then deduced according to the instructions in the appropriate calibration file for the orbits in which they were recorded, and the results plotted using IDL. Some of the features visible in the eSES plots have been labelled, and they have been identified by comparing an image of the surface map of the CCD for selected ranges in pulse-height space. So, for example, where there is an in-flight calibration source it can be seen easily in an image as it illuminates a particular corner of the CCD, whereas pixels associated with the source will tend to appear in a strip where the spectrum is imaged. The in-flight calibration sources are identified on the plots as 'F' and 'Al' and appear at approximately channels 800 and 2300 respectively (already indicating the need for further PH scaling, appropriate to each CCD, as the Al peak should occur at about 1500). First order and second order pixels (if evident) are marked (I) and (II) respectively.

With a variable object such as EXO 0748-67 it is unwise to make too many comparisons of course, but there are a couple of assertions that can be reasonably made.

Firstly, despite the fact that more pixels are telemetered to the ground segment in the case of the HER mode data, it was not possible to extract a spectrum: this suggests that the number of pixels in the data set is of secondary importance to their quality, and the view of the resulting energy histograms supports this, as in the case of the eSES data it is possible to see features in the 'spectrum' after even only one thousand frames. Secondly, the number of pixels entering the DPP software is larger in the case of eSES, demonstrating that allowing the software to select events from the pixel stream is a more effective tool than simply raising the thresholds since the smaller number of pixels telemetered in the case of eSES contains a higher proportion of useful data.

The reduced lower and acceptance thresholds permitted by the use of eSES are of particular relevance to the 'spectral redistribution' which is necessary in the ground segment. The photoelectric effect which is primarily responsible for the absorption of the X-ray produces a photoelectron of about the same energy as the incident photon. It is this photoelectron which then interacts with the lattice atoms, in some cases boosting a valence band electron to the conduction band, creating an electron-hole pair per 3.65eV of the X-ray energy: and it is electrons from these pairs that can be trapped in the CCD charge wells. The conversion of the X-ray energy to electrons is a statistical process with an uncertainty determined by the Fano factor (F), and the variance $(\Delta E/E)^2$ is not $1/n$ as it would be for the Poisson case, but F/n, with $F = 0.12$ for silicon. A 200 eV X-ray, then, produces on average $200/3.65 \approx 55$ electrons, with a standard deviation of

$\sqrt{(55 \times 0.12)} \approx 2.6$ electrons. This corresponds to a full width at half maximum (FWHM = 2.354σ) of the pulse height distribution in the CCD of $2.354 \times 2.6 \approx 6$ electrons, not including noise. Along with the uncertainty introduced in the energy measurement, there is the possibility that the X-ray might be split such that some portion of its energy finishes under the lower threshold. If this happens then some of its energy is 'lost' and its PH will appear lower than the true value. So, all incoming X-ray energies are redistributed to slightly different values with a certain probability distribution (in essence, a single monochromatic line is spread into roughly a Gaussian). The ground segment processing uses a 'response matrix' for the instrument, containing such spectral redistribution (generally in tabular form), together with (or multiplied by) the instrument effective area. It is clear then, that it is advantageous in terms of CCD energy resolution to retain as much data as possible from the lower PH range, and reducing the data rate by simply increasing the settings for the lower and acceptance thresholds is not effective in science terms.

How can this improvement in signal-to-noise be quantified? This is made difficult by the paucity of data available from the commissioning phase which has a sufficiently complementary observation from the later eSES observations. Of the two EXO 0748-67 observations compared, orbits 38 and 67, more scientific analysis has been performed on the latter (Cottam *et al*, 2001) than the former, though some work was done by the author: enough to confirm at least that the spacecraft attitude and roll-angle are the same, meaning that the optical background is unchanged between the two pointings.

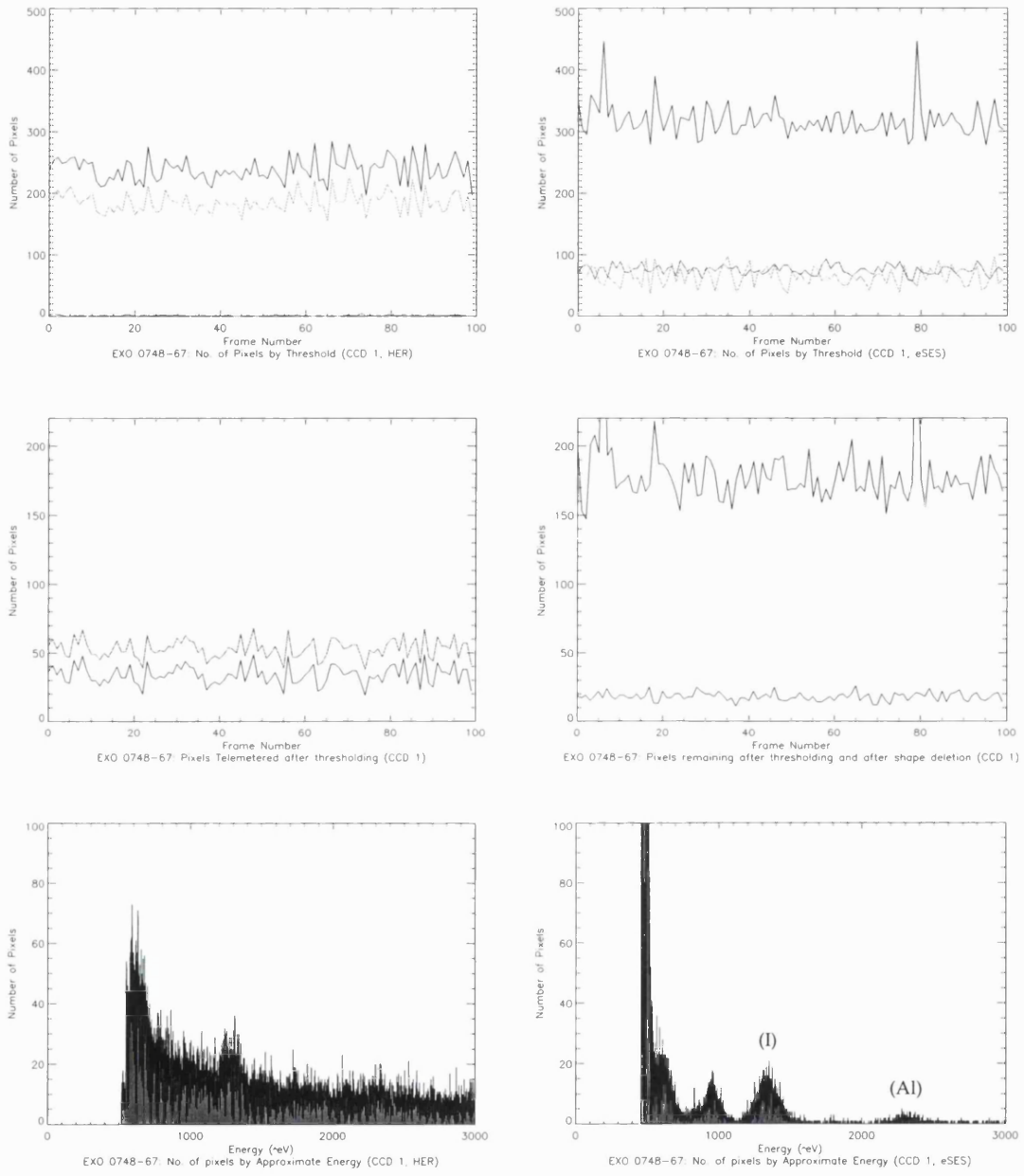


Figure 5.15: *Set of plots comparing HER (left column) and eSES (right column) processed data from the same object, EXO 0748-67, for RGS1, CCD 1. The upper pair show the numbers of pixels above upper threshold (light grey); acceptance threshold (mid grey, barely visible in the HER plots) and above lower threshold (black), as recorded in the statistics produced by the DPP. The middle pair show the number telemetered to the ground (lower line) and the number remaining after applying the thresholds. The gap between shows the number of pixels rejected on other grounds (can only be Hot Items in the case of HER). The bottom row shows histograms of the pixel energies. In both cases the cut off at the lower threshold is artificially raised as a by-product of an offset added in the processing on the ground. CCD 1 sees the aluminium IFC, and this is indicated on the eSES*

energy histogram along with first order pixels from the source. The peak below first order is caused by some unremoved warm pixels.

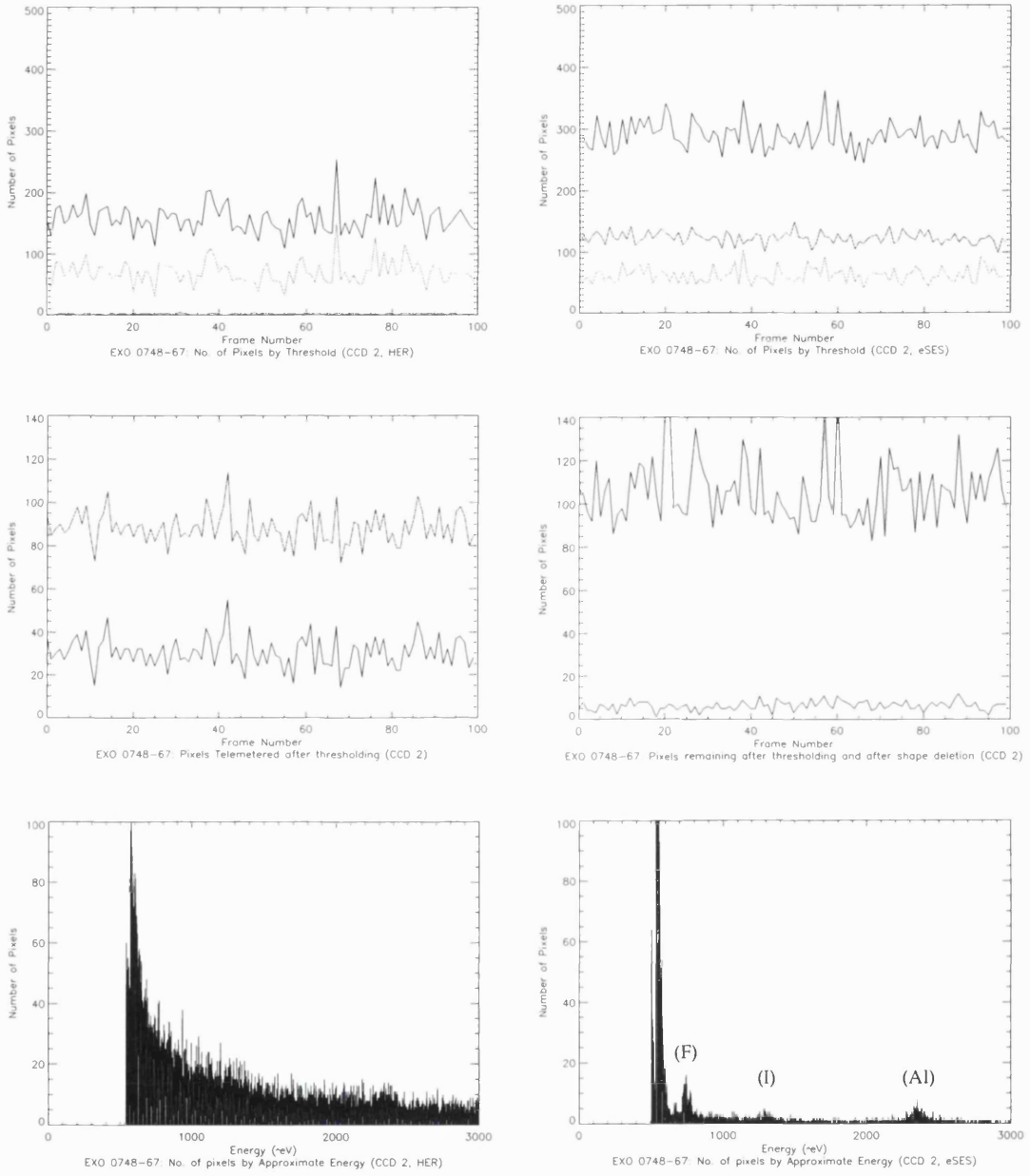


Figure 5.16: *Set of plots comparing HER (left column) and eSES (right column) processed data from the same object, EXO 0748-67, for RGS1, CCD 2. A description of the format of the plots can be found in the caption for figure 5.15. This CCD sees both aluminium and fluorine IFCs and these are indicated on the eSES energy histogram along with the position of first order pixels from the source.*

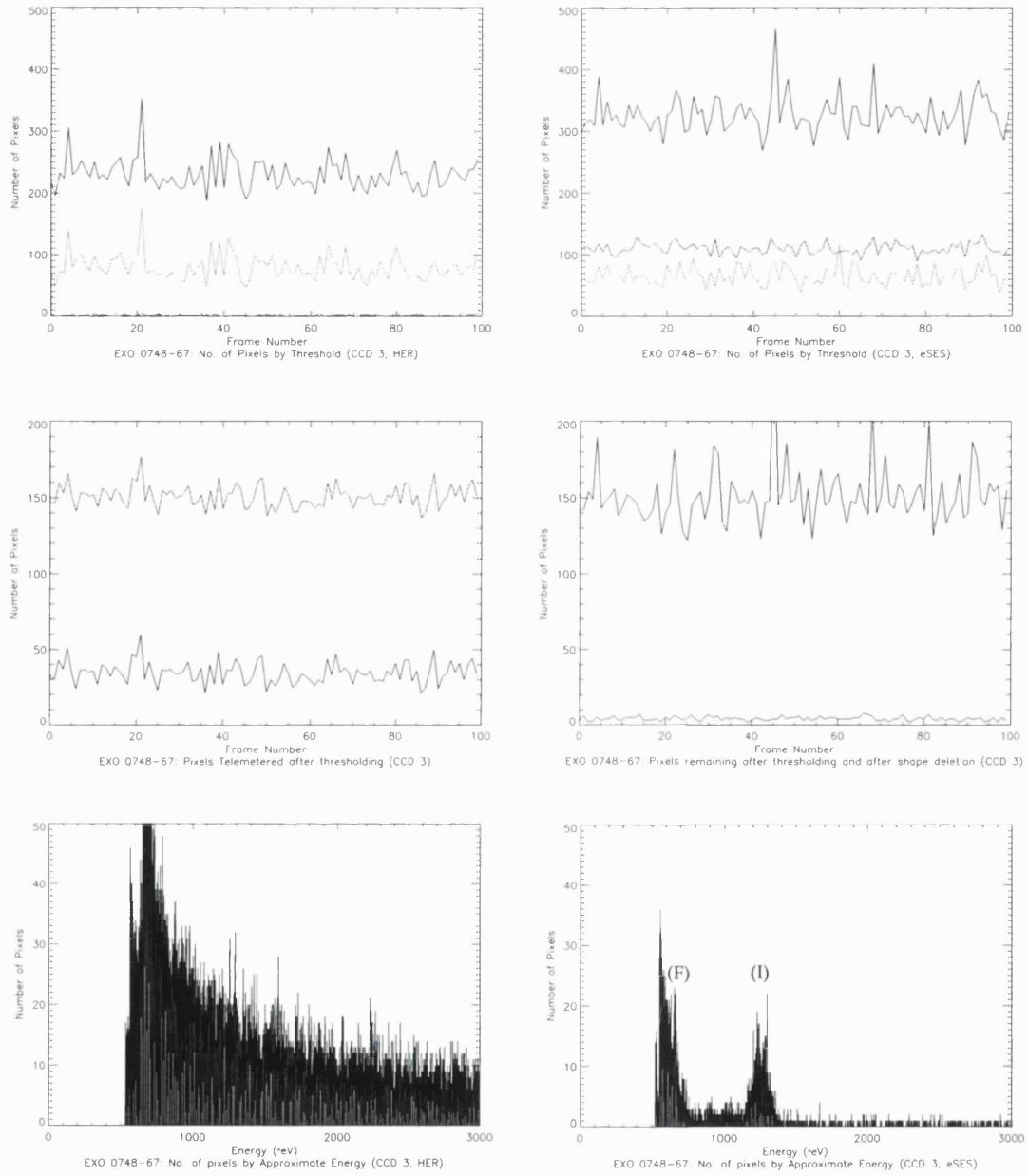


Figure 5.17: *Set of plots comparing HER (left column) and eSES (right column) processed data from the same object, EXO 0748-67, for RGS1, CCD 3. A description of the format of the plots can be found in the caption for figure 5.15. At the time of these observations there was a particularly large number of pixels rejected as 'hot' in CCD 3, and this shows especially well in the HER plot. CCD 3 sees the fluorine IFC, and this can be seen on the eSES energy histogram, along with first order pixels from the source.*

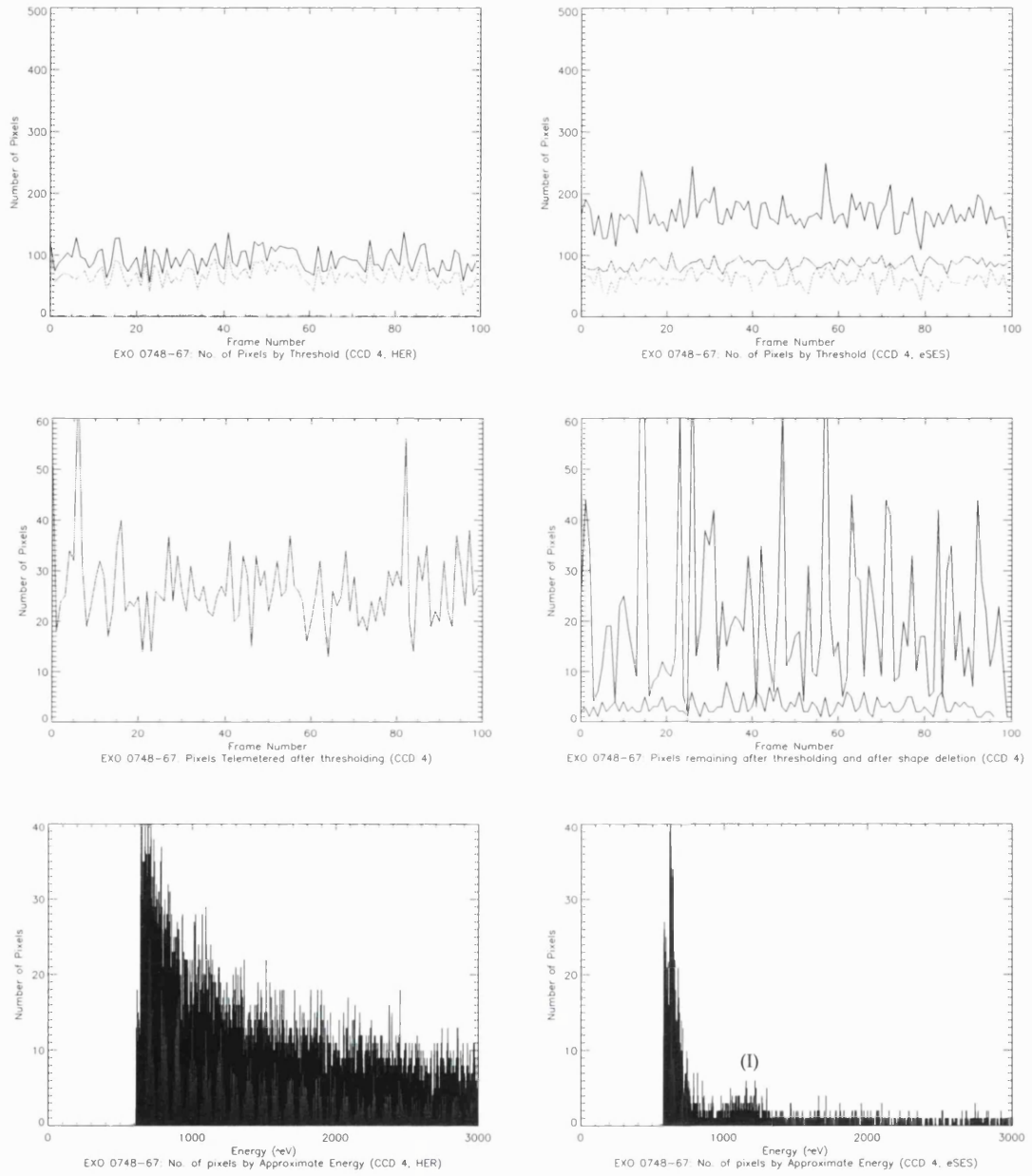


Figure 5.18: *Set of plots comparing HER (left column) and eSES (right column) processed data from the same object, EXO 0748-67, for RGS1, CCD 4. A description of the format of the plots can be found in the caption for figure 5.15. There were no hot items at all in CCD 4 at the time of these observations, and this can be seen clearly in the superposition of the lines in the middle plot for the HER mode data. First order pixels from the source can bee seen in the eSES energy histogram at about channel 1200.*

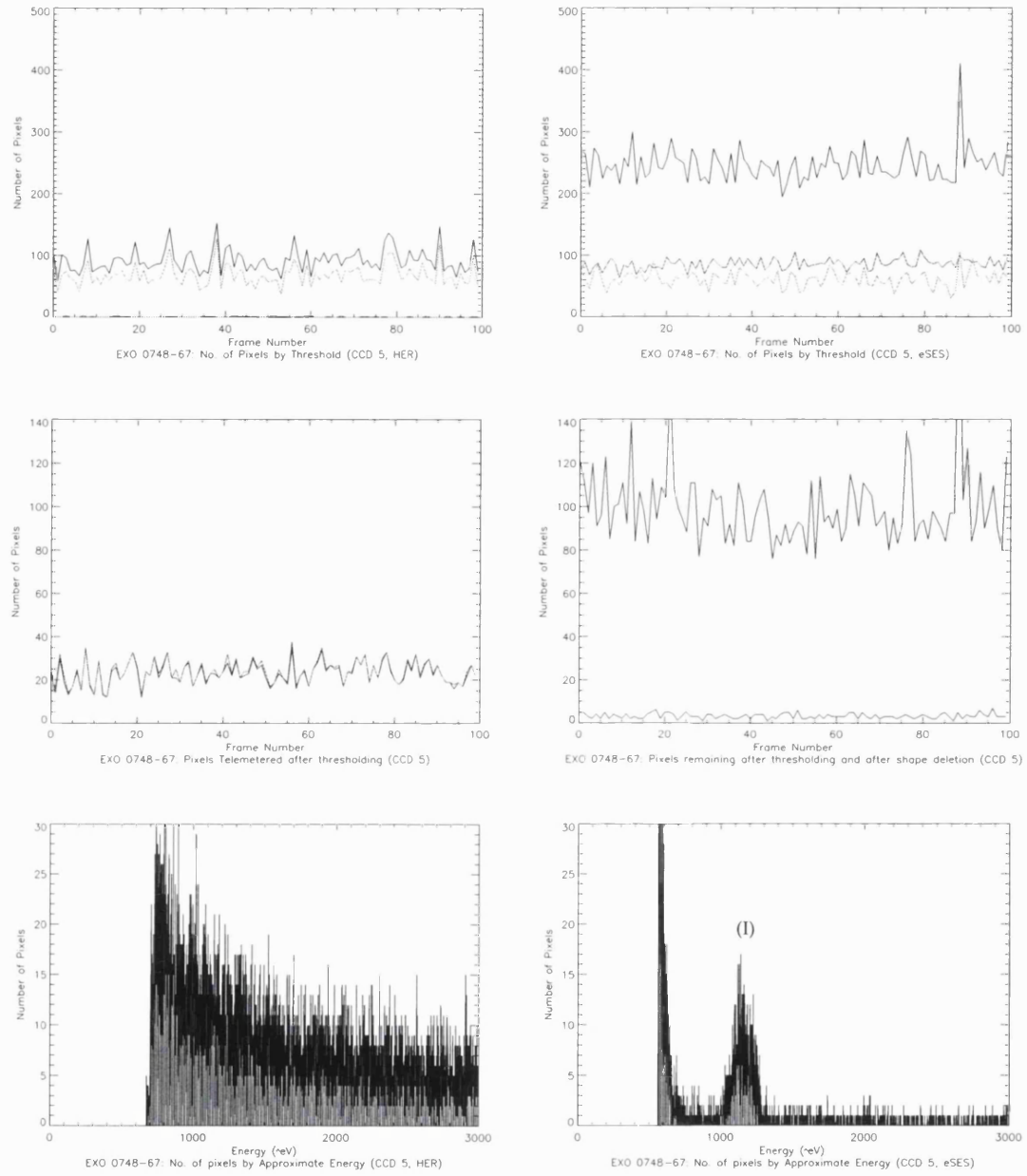


Figure 5.19: *Set of plots comparing HER (left column) and eSES (right column) processed data from the same object, EXO 0748-67, for RGS1, CCD 5. A description of the format of the plots can be found in the caption for figure 5.15. Note the presence of flickering pixels as the plot lines in the 'number of pixels telemetered' graph for HER mode occasionally converge.*

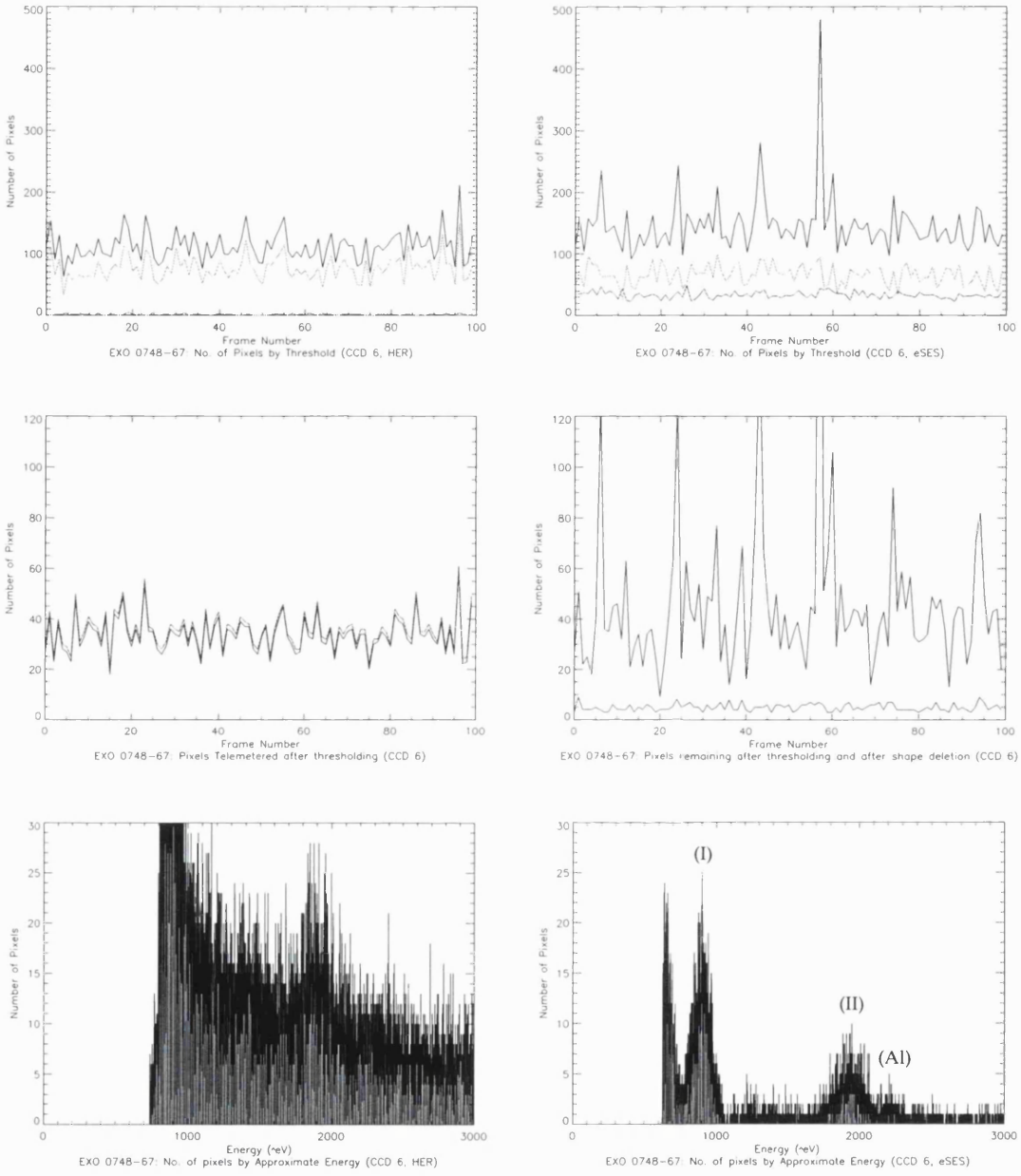


Figure 5.20: *Set of plots comparing HER (left column) and eSES (right column) processed data from the same object, EXO 0748-67, for RGS1, CCD 6. A description of the format of the plots can be found in the caption for figure 5.15. This CCD sees the aluminium IFC as marked on the eSES energy histogram, but also quite clear in this case are first and second order pixels from the source.*

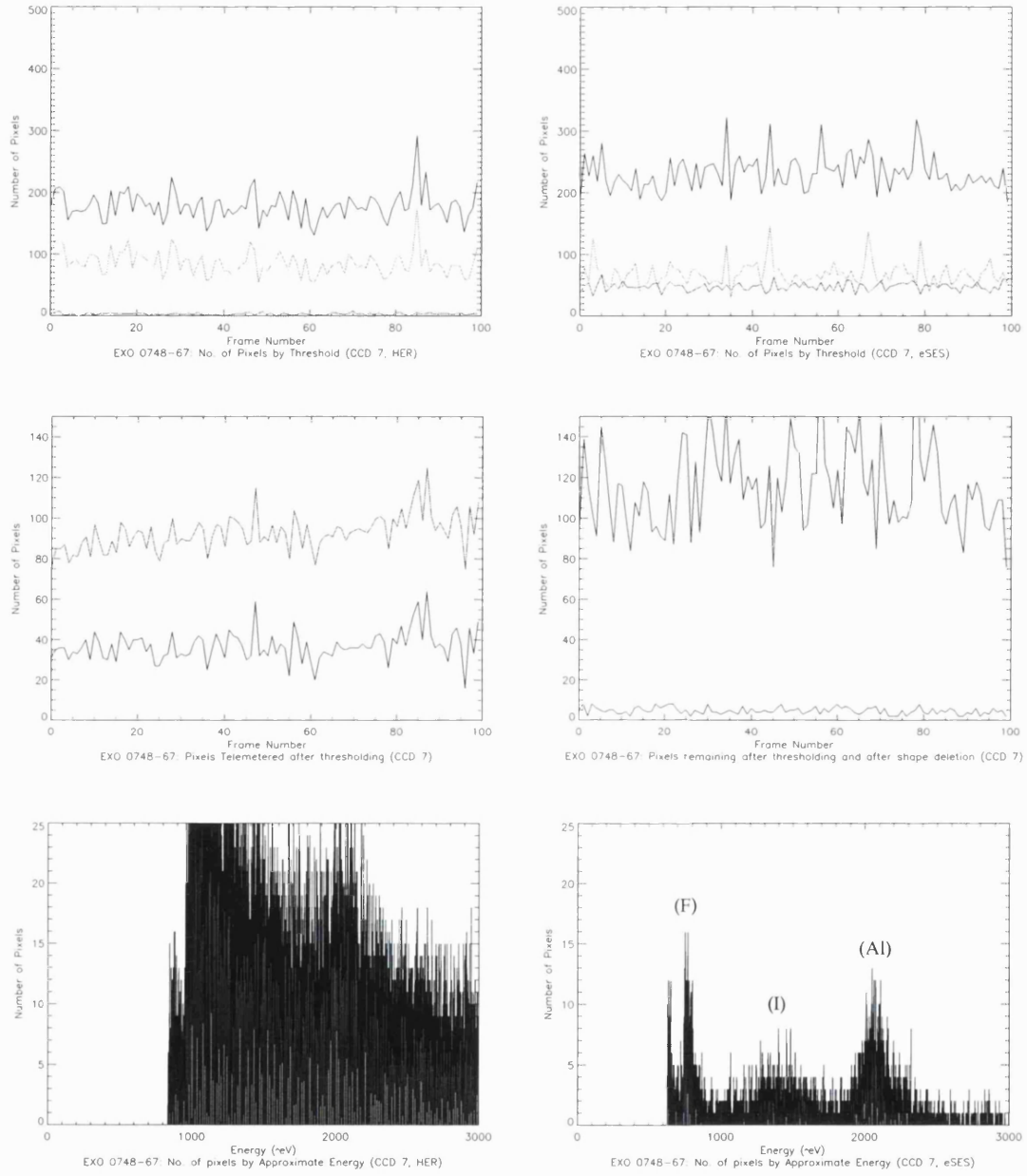


Figure 5.21: *Set of plots comparing HER (left column) and eSES (right column) processed data from the same object, EXO 0748-67, for RGS1, CCD 7. A description of the format of the plots can be found in the caption for figure 5.15. This CCD sees both aluminium and fluorine IFCs, and these are marked on the eSES energy histogram, along with first order pixels from the source.*

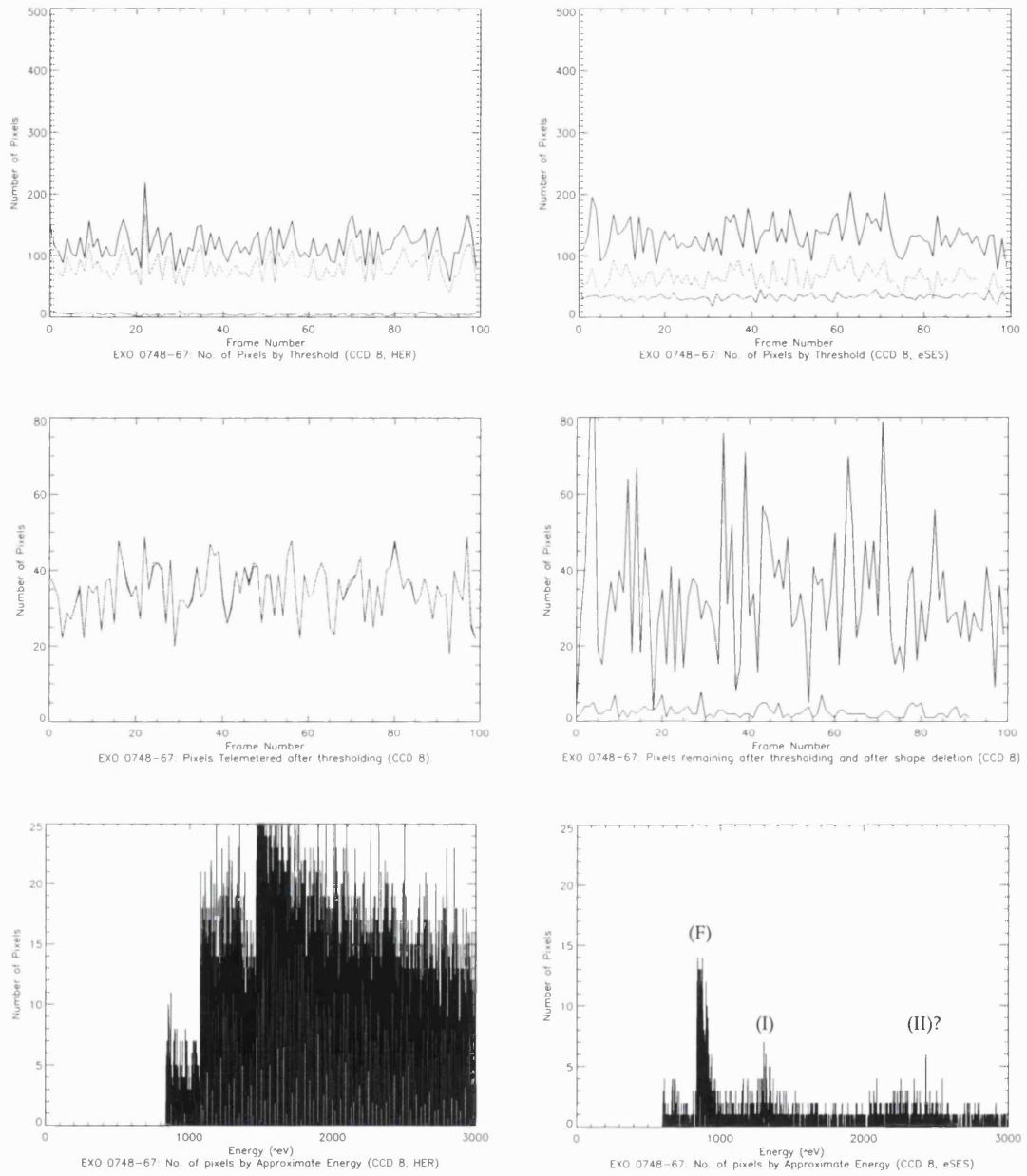


Figure 5.22: *Set of plots comparing HER (left column) and eSES (right column) processed data from the same object, EXO 0748-67, for RGS1, CCD 8. A description of the format of the plots can be found in the caption for figure 5.15. This CCD sees the fluorine IFC and this is marked on the eSES energy histogram along with first order pixels from the source.*

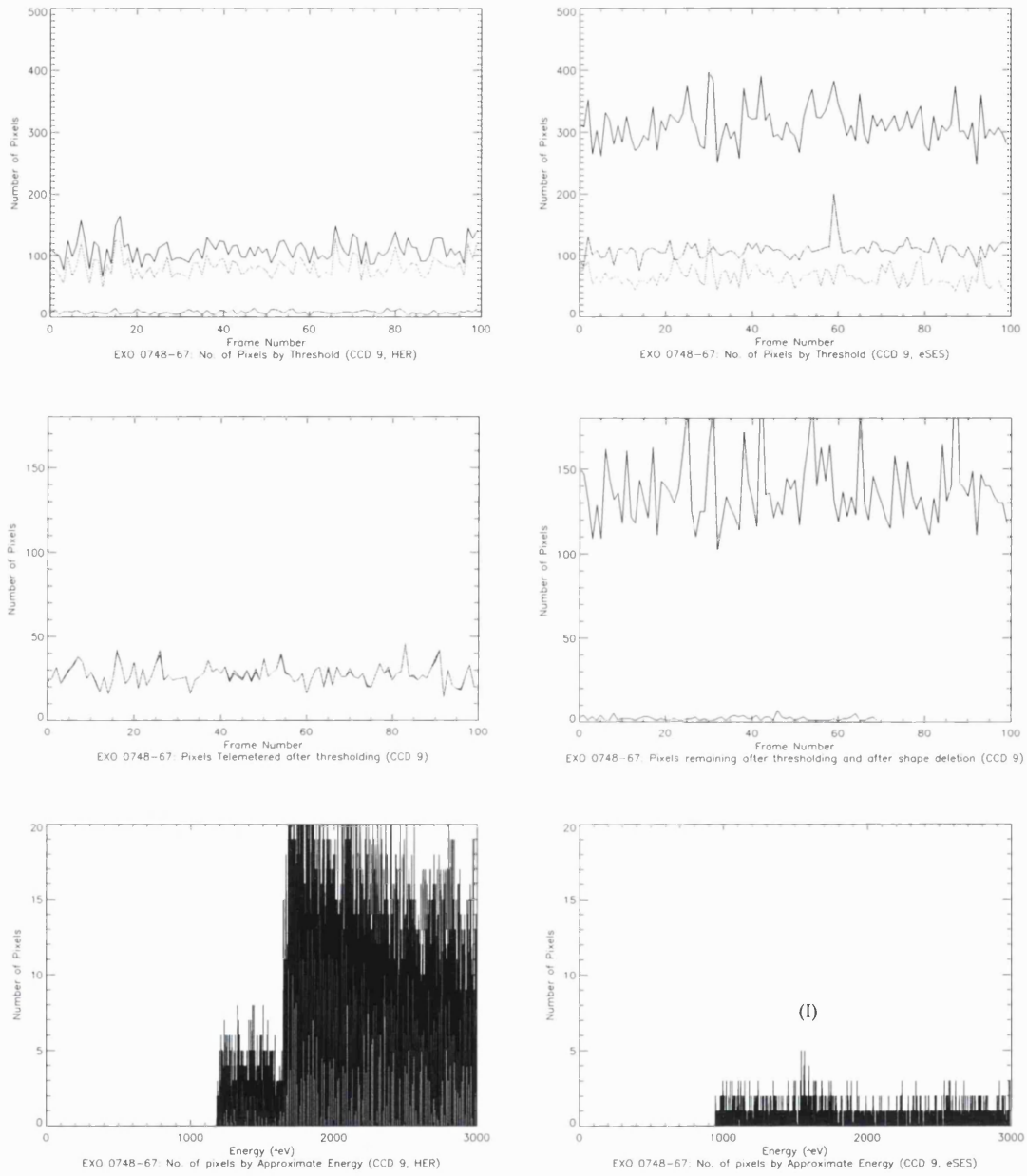


Figure 5.23: *Set of plots comparing HER (left column) and eSES (right column) processed data from the same object, EXO 0748-67, for RGS1, CCD 9. A description of the format of the plots can be found in the caption for figure 5.15. The impression given in the eSES plot of numbers of pixels telemetered to the ground segment is correct: there are periods of time for this (and other) CCDs where there are no pixels telemetered. This CCD has the highest energy pixels from the source, and these can be seen emerging on the eSES energy histogram.*

Separating the total flux from the nine CCDs by using the spatial filtering appropriate to the implied wavelengths, figures can be deduced for the numbers of pixels from source and background, and these are summarised in table 5.11.

Revolution Number	38	67
Mode	HER	eSES
Total counts per frame	7.85	0.61
source counts per frame	0.08	0.30
total counts	102731	71221
total source counts	974	23328
signal-to-noise (source counts / √total counts)	3	87

Table 5.11: Quantifying the improvement in signal-to-noise between HER and eSES modes of the DPP software. Numbers used for source and total counts courtesy Dr. C. P. de Vries at SRON.

The result shown in table 5.11 can be used as an indication of the improvement in signal quality, and quantifies the overall impression of the value of the eSES mode:

- signal-to-noise performance is dramatically increased;
- quality of spectral redistribution is enhanced through the application of reduced lower and acceptance thresholds;
- no ambiguous remnants of too-high energy events are left behind.

5.1.2 High Time Resolution

A first run with an RGS in HTR mode was conducted during orbit number 207 with an observation of Her-X1, a binary system containing an X-ray pulsar. The mechanism for processing HTR data in the pipeline being not complete, some ersatz routines were brewed, and early analysis conducted at SRON. While

some optimisations are noted for the handling of the data by the ground segment and by the IC, the overall conclusion is that the instrument as a whole is working correctly in this mode (den Herder, 2001).

Though minimal in its extent, this first 'quick look' at HTR yields a gratifyingly positive result insofar as the DPP software continues to prove effective.

5.2 Results from the software DPP

In the early phases of the mission, along with the change in onboard software functionality, a rethink was to be applied to all the different thresholds used as a part of this, in January 2000 the author used the software DPP to analyse some diagnostic data files from the twenty-fourth orbit of the spacecraft, and made some independent proposals for the levels of these thresholds. Accordingly, a script was set up to run the software DPP repeatedly over each CCD's raw data set, applying lower threshold values starting at 200, and incrementing in steps of 10. The acceptance thresholds were provisionally set at lower threshold + 20, and such hot items information as was available was used to remove hot pixels and hot column segments. Aside from CCD 4, each CCD yielded a clear result, with an elbow in the plot of the data rate, making it straightforward to estimate the optimum threshold values for each of the cases. Figure 5.24 shows a graph of the results of a run on CCD 1. The x- and y-axes are 'value set for the lower threshold' and 'number of pixels telemetered' respectively. There are three lines plotted, the uppermost is the result from basic spectroscopy, the middle the result for HER and the lowest line the result for eSES.

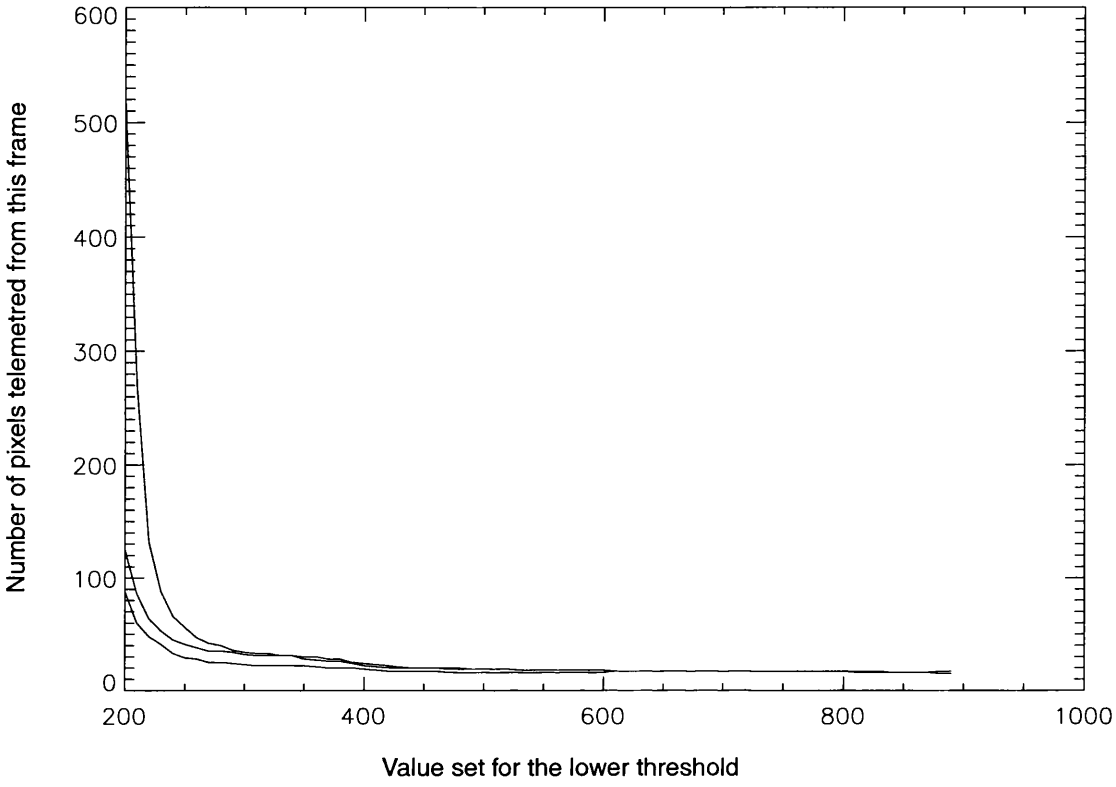


Figure 5.24: *Summary plot from the output of the SWDPP for CCD 1. The input was a queue memory data file from the 24th orbit of the spacecraft (target was HR 1099). The SWDPP has been used to calculate the number of pixels telemetered in each of basic, HER and eSES spectroscopic modes, according to lower- and acceptance-thresholds incremented by the software across a pre-determined range. The graph shows the value of the lower threshold increasing along the x-axis, with the 'number of pixels telemetered' up the y-axis. The highest of the three curves is the result from basic mode, and the lowest is that from eSES. Optimum threshold values could be found by looking in detail around the 'elbows' clearly seen in the plots.*

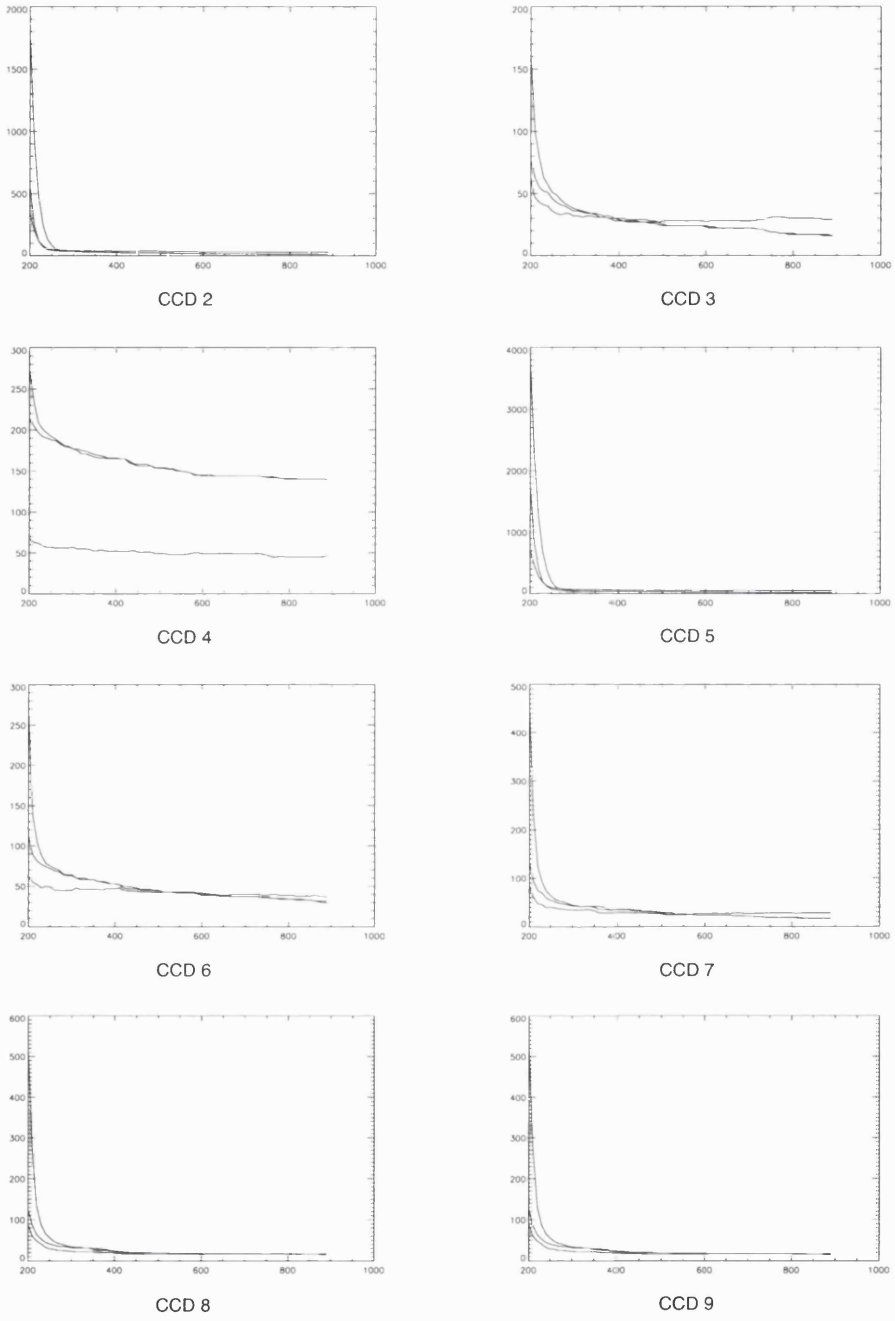


Figure 5.25: *Plots for CCDs 2–9, equivalent to the plot for CCD 1 shown in figure 5.14. Note that the plots for all of the CCDs, except for CCD 4, follow the same general pattern. The larger offset exhibited by CCDs 3 and 6 is accounted for by the presence of unremoved hot items. The uncharacteristic result seen for CCD 4 is shown all the more clearly in figure 5.26.*

Figure 5.25 shows the same graphs for CCDs 2–9. It can be seen that the general shape is echoed by each, and the offsets in altitudes of the plots are accounted for by variations in the gain settings in the analogue electronics chains associated with each CCD, and with unremoved hot items. It can be seen at a glance that there is something odd about the output for CCD 4, and this is shown at its clearest in figure 5.26, which is a summary graph, overlaying all of the individual CCD plots. With no clear turning region in the data, no suggestions could be offered for the thresholds.

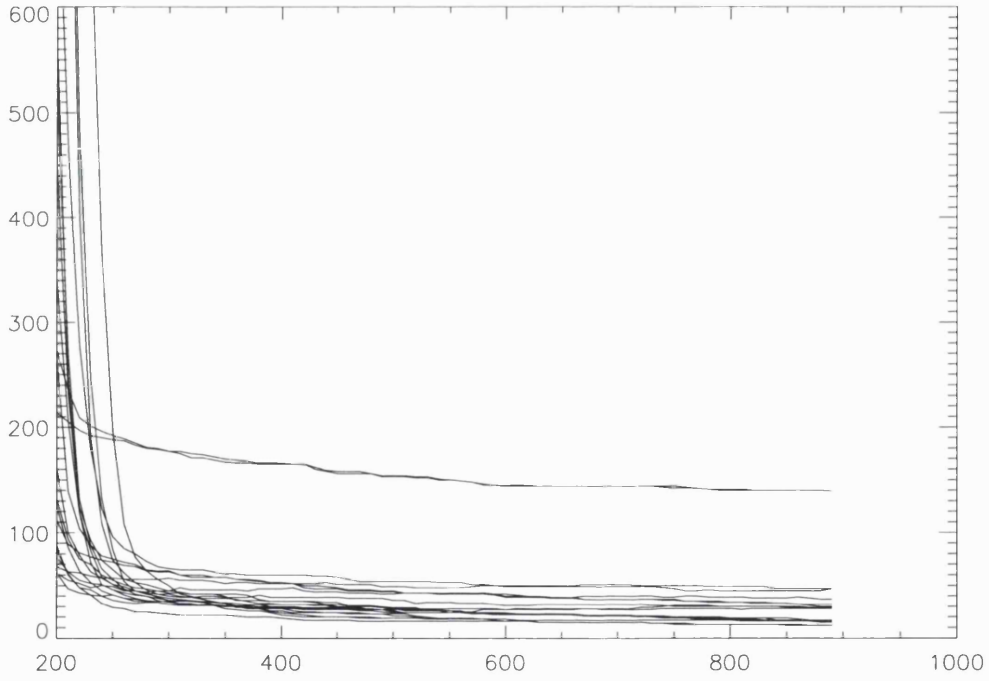


Figure 5.26: *In this summary graph the results from all of the CCDs are overlaid, clearly showing the deviation in the results for CCD 4. The other CCDs having some elbow in their plots, it was possible to propose thresholds for them. The unexpected results from CCD 4 suggest some relation to the hardware fault in that CCD signal chain which developed shortly afterwards.*

The analysis results were sent by email on the Sunday, and on the Monday the news broke that there was a hardware problem with CCD 4, and that the instrument was being held in 'standby' mode (i.e. with the digital electronics active, but the analogue electronics powered off). After some detective work by the designers of the analogue electronics, a specific part was identified, which had it failed would have produced the results seen in the housekeeping monitors of the currents and voltages associated with CCD 4. The part in question was well known, used frequently in space applications and tolerant to radiation doses much higher than that to which it had been exposed by that time; the cause of the failure remains unknown to this day.

Of course the extra activity provoked by this occurrence rather reduced the interest in the work done with the software DPP, though the figures produced by the author were in accordance with those generated elsewhere. The fact that the possible effect of a hardware problem could be identified in the instrument simulation software, suggests some third party verification of its quality—and indeed of the potential value of such standalone simulators.

Chapter 6: Looking Ahead & Conclusion

Though the existing onboard data preprocessing software can be considered successful, insofar as it meets the objectives currently set, there is of course always room for improvement, and some suggestions are made in this chapter. Some of the possible changes are owing to knowledge gained in the design process, and some as a result of the early flight experience. Whether any changes will be required—or even countenanced—is difficult to predict: certainly there will need to be compelling reasons to modify a system that is working, but as the performance of the instrument is likely to degrade with time and accumulating radiation damage, it is wise to consider developments of the software to provide further options.

6.0 Where to find the resources

Two classes of resources are called for: cpu time and available memory. Considering the headroom available according to the results presented in chapter 5, then there is further cpu time available—memory, however, is more problematic. More memory needs to be made available before new code can be added, but happily this is possible on consideration of the current usage of the modes provided. There may be argument for going further than this, but for the time being it is clear that at least the code for 'basic' mode spectroscopy is redundant (recall that HER is now the declared 'minimum impact' mode owing to the crosstalk noise). Removal of this code is simplified by the architecture of the code as a

whole: being the contents of a 'case' statement, the lines of Ada providing this mode can simply be deleted with a low probability of harming the implied reliability of the overall structure. Since there is software intervention (i.e. the Ada compiler) between source code and object code one cannot say for certain what overall effects a change in the source code may make, but with the looping and flow structure of the overall program unchanged, this is the most benign form of large-scale change that could be envisioned. Aiding the proposal to delete basic mode is the fact that it can be easily synthesised by HER: recalling that HER applies an acceptance threshold to isolated pixels and deletes those below it, were this threshold set to zero, then the behaviour of HER mode will exactly echo that of basic mode. No other changes are required by operators or analysers: simply requesting basic mode is enough since the onboard software can detect its selection, and replace the science type with that of HER, and reset the acceptance thresholds to zero on the fly, without needing to make any changes to the contents of the parameter bank.

6.1 Improvements to Hot Items Processing

6.1.1 Complete Pointer Housekeeping for Hot Pixels

The full treatment for the hot pixel table pointer housekeeping has already been described (3.1.4.2), but whether there will be any value in implementing this is difficult to forecast, being rather dependent on the way in which the hot pixels in the flight CCDs mutate. The basic assumption is that the number of hot pixels will increase, but this is not a problem for the existing software, which will only fall short if the number of undeleted pixels becomes greater than the number of

'spare' pixels typically available in any given frame. For completeness some measurements of the performance of the full solution are presented.

In the current onboard code, the arriving hot pixel is used to increment the hot pixel table pointer. If the pixel doesn't arrive for some reason, the pointer will not get advanced, and the following hot pixels on the same row will not be rejected. This compromise was chosen because there is an overhead of processing even when a pixel does turn up. In the fullest treatment of this situation there will be an additional unit of time consumed for every pixel, plus some other time consumed in actually resynchronising the pointer.

Extension of the existing hot pixel treatment continues to look unnecessary at the time of writing this thesis: indeed there is a proposal to not use the hot pixel rejection at all but only the segmented hot column processing since the hot pixels are currently very few in number (Gabriel, 2001). One of the possibilities for improving the hot column processing, described in 6.1.2, will benefit from this, however, and will also free up some cpu horsepower for the extension. Table 6.1 summarises the impact of applying the 'full' pointer housekeeping strategy to the pixel processing times.

Test	**Overhead for Every Pixel**	**Each Resync. (maximum)**	**Comment**
Check row only	25μs	55μs	Max is 1 row up
Check row & column	102μs	1040/840μs	Max is 1 row up, 1022 or 510 pixels along

Table 6.1: *Per-pixel processing times for the full hot pixel table pointer housekeeping strategy. The 'overhead' column compares the impact on each pixel, whether the preceding hot pixel on the same row has turned up or not, with the full solution increasing this load by a factor of four; further time is consumed when a designated hot pixel doesn't arrive.*

Comparing with times already measured for reading the input and processing pixels into the internal buffer, the throughput is again depressed, as shown in table 6.2.

Solution	Time per Pixel	Number of Pixels per second	Delta
Existing	174μs	5747	
Full	251μs	3984	–31%

Table 6.2: Depression in pixel throughput by addition of extra pointer housekeeping. This depression is a minimum: non-arriving hot pixels will add further time to the processing according to the number of tests necessary.

6.1.2 More Intelligent Hot Column Treatment

The onboard code depends on the operators at the ground segment observing the need to add or remove hot columns to or from the onboard tables. If there is some delay in the removal of a hot column from the table, then the risk is that for some portion of an observation fewer wanted pixels will be telemetered to the ground segment: undesirable, but not harmful. If the converse can occur, i.e. a hot column appears during an observation, then there is potential for data to be lost since unremoved hot items can saturate the telemetry allocation. This may be more of an issue later in the mission as the CCDs become progressively more damaged by the radiation dose received.

It is interesting to note that the experience so far (approximately one year from launch) is that hot columns can appear and disappear quite rapidly, indeed after as few as thirty orbits the entire hot column map can be completely differ-

ent. The mechanism for this rapid variability in the hot column placements is not yet fully understood: the expectation was that the disruptions in the semiconductor crystal lattice that yield unwanted charge emitters would be slow to anneal owing to the low temperature of the CCDs, but this seems not to be the case.

Two schemes are discussed whereby the onboard software can be improved in this respect.

6.1.2.1 Enhanced Hot Column Table Management

The most obvious area where the existing technique could be improved lies in the 'ownership' of the hot column tables. If the onboard software were given the ability to add hot column entries on its own account, then a defence is provided against the loss of useful scientific data owing to the presence of an excess of bad pixels of this type.

Such a new routine would require two further entries in the parameter bank for each CCD:

- maximum number of pixels in a column side C
- maximum number of pixels in a column side D

and a new data tag from the DPP to the IC:

- new hot column tagged

The concept works by causing the onboard software to now keep, in effect, a histogram of the number of pixels it finds in each column. A single memory array is needed for this, large enough to hold a word per column of one CCD. The user specified value 'maximum number of pixels in a column side C/D' is used as an indicator for the onboard software to decide whether a column has become hot. If this value is exceeded, then the code will set all of the bits in the appropriate col-

umn in the actual hot column table such that the column is rejected next time, and the CCD ID and the x-position of the new hot column is relayed to the ground segment. The operator of the instrument can then decide whether this new hot column should be kept, or perhaps refined to be just a part of a column, and the 'master' copy of the hot column table can be amended (or not) accordingly. Sorting the columns into segments onboard is too cpu intensive and risks affecting the throughput again: the philosophy here is for speed and simplicity in order to give a safety-net to the existing observation, which might otherwise get lost or abandoned. Figure 6.1 illustrates the placement of the main extra functions by modifying the flowchart from figure 3.6.

What about apprising the ground segment of the newly selected column or columns? This is not shown, and is an implementation detail that will need to be agreed with the authors of both the ground segment software and instrument controller software. The obvious locations are either in an extension of the flowchart shown in figure 6.1—i.e. report each column as it is discovered—or in the statistics handling routine where the other 'meta' data is included. Alternatively, the onboard hot-column table could be downloaded periodically to compare against the version originally uploaded, though this would not by itself indicate *when* the changes to the table were implemented and this might be important for calculations concerning the exposure time or efficiency.

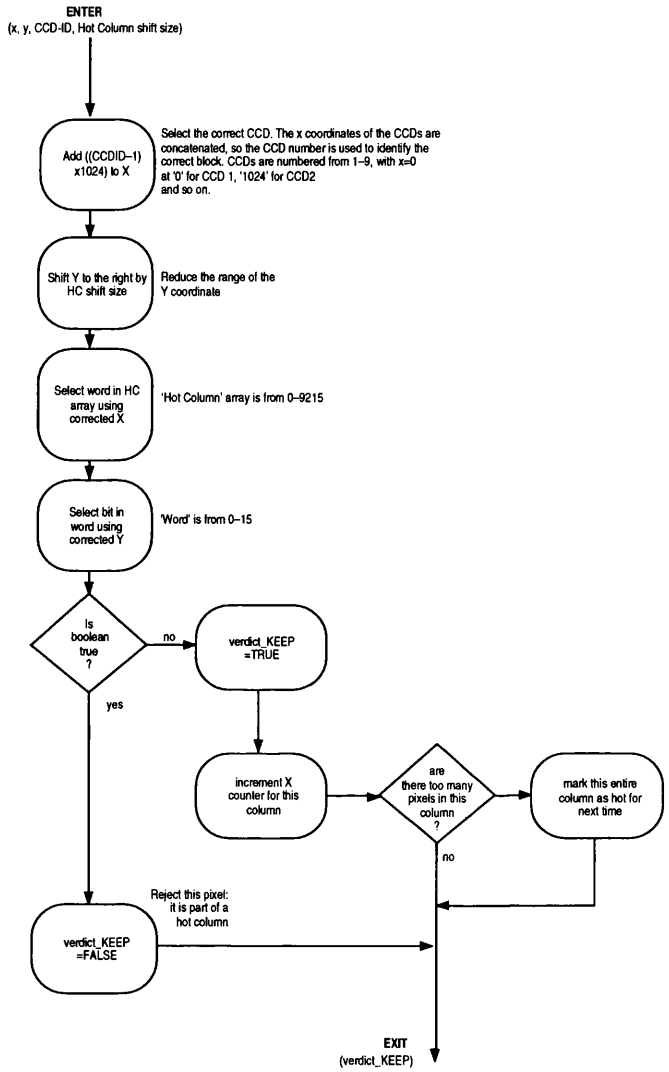


Figure 6.1: *Hot Column segment test flowchart, modified from figure 3.6. Before returning to the calling program the x-coordinate histogram is updated, and the value tested to see if the user specified ceiling has been met. If so, then that column is marked at 'hot'. The location of these extra steps is important: we know that this routine will be called frequently if there is a new hot column, and placing it at this point in the flow means that these extra functions will not be called again for any component of the new column.*

6.1.2.2 Deleting the Onboard Hot Column Processing

This 'dramatic' option is made possible by dependence on the eSES mode's ability to reject connected events which extend beyond two pixels. The existing treatment of individual hot pixels would need to be retained as single pixels are nothing exceptional to eSES, but removing the hot column handling would free up significant amounts of memory, and cpu horsepower. Though it sounds like a simplification of the code, this idea actually represents bigger changes to the onboard software than those proposed in 6.1.1.1 since the existing task switching model allocates the cpu time according to its need—simply switching off the hot column rejection will not allow more time for eSES since the program pointer doesn't switch to that context until the readout of the CCD is complete, or the relevant internal buffer is full. With few hot columns present, the existing disposition of the processing will be adequate, but to make the best of this option the scheduling of the task switching will need to be rethought.

The attraction of allowing eSES to remove the hot columns is that this mechanism is naturally dynamic: if a hot column segment anneals then it is retained automatically, and as soon as a segment is made hot then it is rejected. For data quality some work will need to be carried out to assess the possibility of hot column segments extending over only two pixels as these will be misleading, especially if their sum when reconstructed fails to exclude them on the grounds of energy selection. The author will be looking into this, possibly arranging for some data to be taken by the instrument with hot items processing turned entirely off. This will give a useful comparison, and is feasible while the number of hot columns is low (currently not more than one per CCD half).

6.2 Improvements to HER processing

The current HER code applies only a simple upper thresholding strategy on a single pixel basis. With the high incidence of cosmic rays in the RGS data this is shown to be a somewhat imperfect technique since it is prone to leave their high energy 'splashes' either untouched, or, even worse, left ambiguous by having only some components deleted. In this situation eSES and SER modes give a better treatment since they make some effort to collect the components of an event together—albeit over a small region. To improve the quality of the treatment given in HER, and to return some flexibility to the instrument when dealing with fainter sources, or for occasions when a higher telemetry allocation may be available, the author suggests the introduction of 'speculative reconstruction'.

The purpose of this is to apply the same search criteria and rules for analysis of shapes and connections as for eSES and SER, but with the difference that the reconstruction is only used to improve the identification of events whose total is above the upper threshold: no permanent reconstruction takes place. Figure 6.2 shows a flowchart of such a mechanism, closely based on the flowchart for SER mode seen in figure 3.26. The mode can be constructed this way since the underlying logic is unchanged: in SER mode any pixel which is a part of an event extending beyond the permitted shapes (figure 3.22) is left untouched and this remains true in HER. The difference is that now a legal event is summed into a temporary variable, and if the total is above the upper threshold or below the acceptance threshold then all components are deleted, otherwise they are all telemetered as-is. The acceptance threshold test is included since its exclusion

suggests an unnecessary limitation on how the thresholds might be set, though if the intended $3\sigma/5\sigma$ settings are employed for lower and acceptance thresholds then it will be impossible for the reconstructed pixel to be below the acceptance threshold.

Limited tests with this technique when applied to the same flight data set used with the software DPP in chapter 5 suggest a data reduction between the existing HER and the 'improved' version (let's call it eHER) of around 4%. This is a small difference in terms of consumption of telemetry bandwidth, but represents a potentially important improvement in the data quality by reducing the chance of a very high energy event leaving fragments behind which could be confused with real X-ray components; as has been demonstrated to happen in chapter 5. The difference, then, between eHER and SER modes is that with no onboard reconstruction taking place the distribution of the energy around the components in the split event is not lost. eHER will improve the performance of HER mode at rejecting pixels above the upper threshold, but will still not match the performance of eSES, and since it will accordingly probably still require the lower thresholds to be set higher than for eSES, this mode may only be practical for use with bright sources such as Capella.

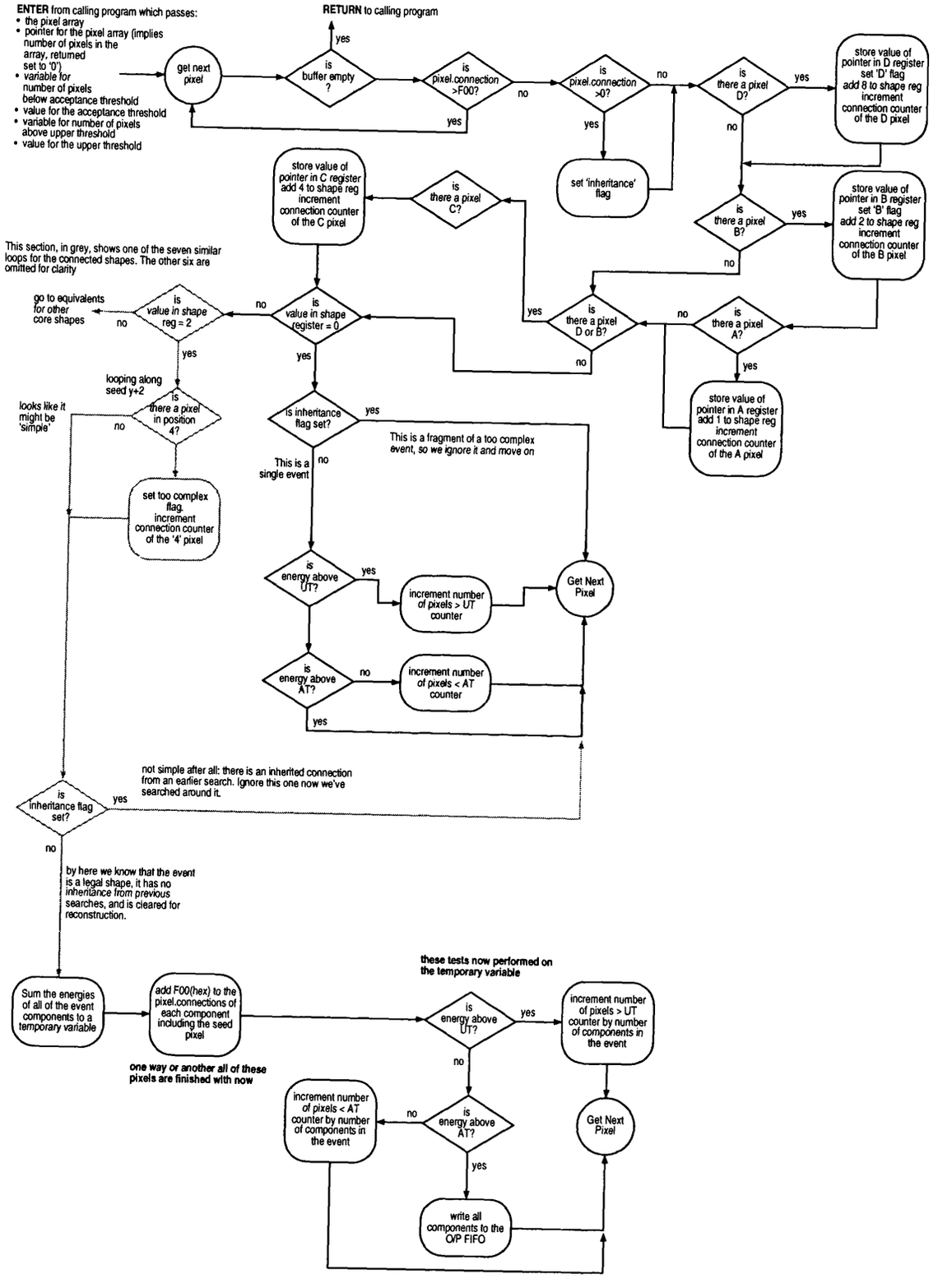


Figure 6.2: Flowchart for 'eHER'

6.3 Cosmic Ray Investigation

The default operational mode of the RGS is operation in eSES, with queue memory data trickled to the ground segment as bandwidth permits. These full frame unthresholded read outs may prove useful as a tool for investigating cosmic rays: over time the library of queue data frames is accumulating and as each is accompanied with detailed information about spacecraft attitude and time (hence position in the orbit) these may provide a serendipitous source for at least some statistical information about flux and direction. The queue data frames would need to be analysed to look for certain signatures, but this is relatively straightforward work, and could use the same sort of approach as the software DPP.

6.4 Summary

XMM-Newton was successfully launched from Kourou in French Guiana in December 1999, on the Ariane flight 504. This observatory is the largest, most expensive scientific spacecraft ever launched by a European consortium, it is a cornerstone mission in ESA's space science programme, and in its first year of operations has already produced data of unique quality (Astronomy & Astrophysics, volume 365, number 1 January 1, 2001; Astronomy & Geophysics February 2001, Volume 42, Issue 1, p7). Data from the RGS, with its unprecedented energy resolution, is being met with great enthusiasm by astronomers using this instrument.

The onboard software developed by the author for preprocessing the CCD data is a key element in the success of the RGS: it has executed faultlessly throughout the mission so far, and has met or exceeded all of its requirements, proving itself to be:

- reliable—the software has never crashed;
- effective—no data has been lost through lack of throughput capacity, no telemetry bandwidth has been wasted by excessive data removal;
- adaptable—the software was easily tailored to cope with the higher than forecast background.

The original work described in this thesis took place largely over the five year period prior to launch and the year following this. Some analysis of the early flight data was conducted, with the results described in Chapter 5, and in Chapter 6 some further proposals are made for possible enhancements to the product delivered.

The kernel of the work of this thesis is the creation of novel algorithms which will permit optimal data reduction with no loss of scientific information where constraints of computing resources might be significant, such as in a satellite mission (Chapter 1). As CCDs become more common, and are developed with increasing resolutions and areas, the implied quantity of data rises also, with the result that even a ground based system may benefit from enhanced execution speeds owing to the volume of data which needs to be processed.

The algorithms presented in Chapter 3 were developed during a period where building the spacecraft hardware was the high priority for most of the collaborating scientists; having demonstrated quite early in the programme that the author was capable of providing effective solutions to the challenging data pro-

cessing problems, they were pleased to be able to give him a large degree of autonomy during this period. The split event reconstruction technique is only relevant in cases when the incoming photon energy is high and the total amount of energy in an event needs to be recorded, making it inappropriate for CCDs used at optical wavelengths; however, the underlying algorithm is easily transported to use in other cases where split events can occur. The novel technique of using segmented columns to achieve the removal of regions from the CCD read-out window is appropriate to any case where such data reduction is necessary, and is easily transportable using, as it does, an elegant and fast method.

Also presented, in Chapter 4, is the description of an instrument simulator in software, the SWDPP. The value of this was proven to the author at least in its utility for writing and testing algorithms when the main development system was unavailable. The usefulness of the SWDPP was taken beyond this point by the conversion of the software DPP into a generic tool that can be used elsewhere in the RGS consortium. Writing the software DPP into a form similar to other popular astronomical utilities was logical and aided its adoption and use by at least one other institute in RGS consortium.

The value of such simulators was demonstrated by the ability of this unit to detect problems in the source data caused by a failure mode in the hardware itself, as shown in Chapter 5. Such tools may be of use again, and may provide an alternative method for tracking the performance of an instrument over time by independently analysing the raw data as the mission progresses.

6.5 Conclusion

In the case of the CCDs used in the environment of the RGS, much of the background in the interesting X-ray band arises from charged particles and reducing the data by simply applying high and low energy level thresholds results in CCD data with a poor signal-to-noise ratio, of the order of 1:50. Reducing the CCD data to fit the available telemetry bandwidth by selecting a narrower energy window is unhelpful since:

- the nature of the X-ray energy absorption mechanism and the possibility of X-ray events becoming split, make the setting of the low level threshold critical to both the separation of lower energy events from the background, and the spectral resolution of the CCD;
- the possibility of charged particles contaminating more than one pixel with energy deposit fractions which are within the energy range of the source's X-rays means that (a) the upper level threshold cannot be set low enough to exclude them completely and (b) incomplete removal of a high energy charged particle's energy deposits will lead to further ambiguities.

It is observed, however, that genuine X-rays tend to give rise to charge deposits confined to a single pixel, only occasionally extending to adjacent pixels within a 2x2 region—and then mostly to pairs of pixels. Since most background pixels, because they are associated with charged particles, occur as parts of larger pixel groups or clusters, removing these cluster-associated pixels—using the techniques described in this thesis—can therefore significantly reduce the number of background pixels, without significantly affecting the X-ray signal: in the

case of the RGS instrument on XMM-Newton improving the signal-to-noise ratio by at least an order of magnitude, and reducing the number of pixels to be transmitted to the ground segment by approximately the same ratio.

Appendix 1:

Factors Affecting Communications Bandwidth

The normal operating mode of the RGS instrument on XMM-Newton utilises a telemetry allocation of 6Kbits/second. This compares to the 2000Kbits/second that can be transferred from a floppy disk—already considered rather a slow medium on a desktop computer of today, but not atypical of a spacecraft subsystem's allocation. The main factors which decide the communications bandwidth are the physical factors of distance of the spacecraft from the groundstation (and the influence that this has on the transmitter power required), and the operational factors affecting the profile of the rate of acquisition of data and the amount of time spent in contact with a groundstation.

Physical Influences

A striking feature of the design of the link between a spacecraft and its groundstation is the effect of the distance involved. Were it radiated isotropically, then P watts of power from a transmitting system diminishes to a flux, F, according to:

$$F = \frac{P}{4\pi R^2} Wm^{-2}$$

where R is the radius of the sphere. This means that even at distances modest in spacecraft terms the flux of power at the receiving antenna can be vanishingly small: typically less than $10^{-10}\,W/m^2$ from a geosynchronous spacecraft at a

height of 36,000 km (Pratt et al, 1986). Of course the power is not radiated isotropically, but will be focussed by an antenna system. The design and complexity of the antenna will, however, also be limited by mass constraints, and its footprint limited by the pointing accuracy attainable by the spacecraft. Options for increasing the power radiated from the antenna by increasing the transmitter amplifier gain will be limited by the available electrical power from the spacecraft, and by thermal design constraints. A complete exposition of the difficulties of spacecraft design is not appropriate here, but it ought, at least, to be seen as inevitable that in an environment where volume, mass and power are in short supply, and heat is difficult to dissipate, and all these aspects feed back to each other, the maximum radiated power from the spacecraft will be limited, even before folding the concomitant implication on cost (Fortescue *et al*, 1992). It is also true that at certain frequencies the flux at the footprint of a spacecraft antenna may be constrained by pre-existing terrestrial use of the same frequencies.

There are fewer technical constraints on the design of the receiving antenna of course, but limitations of cost and practicality will still obtrude and there will be an ultimate limit to the sensitivity of the groundstation to the signals from the spacecraft.

The overall product of this physical design of the spacecraft-to-groundstation system will be a 'link budget', the scale of which feeds directly back to the error rate that link will be able to accommodate. This 'bit error rate' (BER) is determined by the quality (the 'carrier-to-noise ratio') of the link with the spacecraft, and according to the data handling techniques used there will finally be a maximum bit-bandwidth which can be supported. Improvements to the bit errors in the transmissions can, and are, made by the use of encoding techniques,

but these inevitably require extra data to be included—in short, for an error to be detected then the set of 'legal' values that can be received must be smaller than the potential total set that can be transmitted (Cruise *et al*, 1998).

Operational Influences

Are limitations to the communications bandwidth available to spacecraft necessarily problematic? An instinctive response is to assume 'yes', since a higher resolution measurement from some instrument will inevitably require more information to describe it, nevertheless, there are other parameters to consider to give an idea of the differences wrought by data profile and choice of orbit.

The choice of orbit, along with influencing the link budget, will have an implication on the groundstation contact: low Earth orbiting spacecraft will have the advantage of shorter signal path length for transmission, but will have shorter visibility by each groundstation. A spacecraft in a highly elliptical orbit will on the other hand have the opportunity for very long periods of contact with a particular groundstation, though with high altitude apogees the path loss becomes a significant issue.

The 'data profile' reflects the nature of the change in data rate from an instrument over a unit period—say one orbit. Obviously if the data is not generated at a constant rate, then the only concern is that the net rate during the transmission to the groundstation contact period cannot be exceeded. This may mean queueing the data on board in some form of storage.

Appendix 2:

Sample FITS Header

This is a sample header from a FITS file of diagnostic data taken soon after launch of XMM-Newton. This text was extracted using 'fdump' from the FTOOLS suite to produce an ASCII text file.

```
SIMPLE  =                    T / file does conform to FITS standard
BITPIX  =                    8 / number of bits per data pixel
NAXIS   =                    0 / number of data axes
EXTEND  =                    T / FITS dataset may contain extensions
COMMENT    FITS (Flexible Image Transport System) format defined in
Astronomy and
COMMENT   Astrophysics Supplement Series v44/p363, v44/p371, v73/p359,
v73/p365.
COMMENT   Contact the NASA Science Office of Standards and Technology
for the
COMMENT  FITS Definition document #100 and other FITS information.
ORIGIN  = 'XMM-SOC '           / Processing site
DATE    = '2000-01-25T15:04:06' / File creation date
FILENAME= 'RRRR_9999990022_R2U00908DII.FIT' / Filename
CREATOR = 'XSCS-PMS'           / FITS generated code
CATEGORY= 'XMMODF  '           / The file is part of an XMM ODF
END
XTENSION= 'IMAGE   '           / IMAGE extension
BITPIX  =                   16 / number of bits per data pixel
```

(continued)

```
NAXIS   =                    2 / number of data axes
NAXIS1  =                  342 / length of data axis 1
NAXIS2  =                  128 / length of data axis 2
PCOUNT  =                    0 / required keyword; must = 0
GCOUNT  =                    1 / required keyword; must = 1
EXTNAME = 'R2DII1  '           / Extension name
EXTVER  =                    1 / Extension version
TELESCOP= 'XMM     '           / XMM mission
INSTRUME= 'RGS2    '           / RGS Instrument
DATATYPE= 'DIAG.IM '           / Type of Data
OBS_ID  = '9999990022'         / Observation Identifier
EXP_ID  = '9999990022009'      / Exposure Identifier
TMEXP_ID=                  108 / Exposure Identifier in exposure packet
CCDID   =                    8 / CCD ID
CCDOCB  =                    3 / OCB(Frame-Diag)/SID-science
WINDOWX0=                    0 / CCD-n X-start
WINDOWY0=                    0 / CCD-n Y-start
WINDOWDX=                  342 / CCD-n X Length
WINDOWDY=                  128 / CCD-n Y Length
FTCOARSE=              3996560 / Coarse time
FTFINE  =                42808 / Fine time
BLOCKNO = 'X       '           / Block number
PIXELNO =                43775 / No. of pixels to be acquired
CSGID   =                    6 / CSG-ID
CCDFLUSH=                    0 / No. of CCD flushes
CCDNODE =                    3 / CCD Readouts
DATE-OBS= '2000-01-25T15:04:01' / Exposure Start Time
DATE-END= '2000-01-25T15:05:33' / Exposure End Time
END
```

Appendix 3:

Example Shell Script

This is an example shell script used to generate files for, and manage the execution of the software DPP described in chapter four. The script itself is an executable, and is called by typing:

```
prompt%  swdpp filename.FITS
```

where 'filename.FITS' is the file to be analysed.

```
#!/bin/csh -f
# shell script to operate the MSSL software DPP.
# Requires FITSfile for pixel data and parameter
# file for DPP

set logfile = $1
#logfile is the filename passed by the user

echo ""
echo ""
echo "--------- MSSL XMM RGS: SWDPP  Version 1 ---------"
echo ""

echo "Examining header from " $logfile

fdump prhead=yes $logfile header.tmp.out - -

awk -f header.awk < header.tmp.out > header.txt
#strip out the unwanted bits

echo "Pre-Processing " $logfile

fim2lst $logfile+1 tmp.fits
#pull out the pixel data from the extension

fdump prhead=no tmp.fits tmp.out - -
#turn the pixel data into ASCII
```

(continued)

```
awk -f pre.awk < tmp.out > tmp2.txt
#strip out the unwanted bits


echo "                    ... done"
echo ""

echo "Analysing ... " $logfile
echo ""

swdpp.o
#call the s/w DPP once using the parameters in parm.txt

echo "                    ... done"

rm tmp.fit tmp.out tmp2.txt header.tmp.out header.txt
#clean up


echo ""
echo "---- Execution complete: output in analysis.txt ----"
echo ""
echo ""
```

Appendix 4:

Project Structure

Four institutions were directly involved with the production of the RGS:

- SRON (The Space Research Organisation of the Netherlands, PI A. C. Brinkman), is the PI institute and is responsible for the overall management of the project, and for the production of the RFC and RAE (http://saturn.sron.nl/divisions/hea/xmm)

- UCB (The University of Columbia, New York, Co-I S. Kahn) was responsible for the development of the reflection grating arrays (RGA) (http://xmm.astro.columbia.edu)

- MSSL (The Mullard Space Science Laboratory, University College London, Co-I L. Culhane) was responsible for the development of the digital electronics, the power supply and the onboard software (http://www.mssl.ucl.ac.uk/www_astro/rgs/rgs.html)

- PSI (The Paul Scherrer Institut in Switzerland, Co-I A. Zehnder) was responsible for the thermal; and mechanical design of the detector assembly, and the design and production of the radiator (http://www1.psi.ch/www_lap_hn/ASTR_XMM.HTML)

Glossary of Abbreviations, Acronyms & Terms

Actel	A brand of one-shot, field programmable gate array
Active area	The area of the CCD that is generating data that reaches the DPP
AD	Applicable Document
AESIM	Analogue Electronics Simulator
AIT	Assembly, Integration and Testing
AIV	Assembly, Integration and Verification
APID	Application Identification
Baffle	A photon-proof partition used to limit stray light
BB	Bread Board
BOL	Beginning of Life
Byte	Eight bits
CCD	Charge Coupled Device
CCS	Central Checkout System (aka Overall CheckOut Equipment)
CDMS	Control Data Management System
CIA	CCD Image Analysis
CoG	Centre of Gravity
Configure Mode	IC is running s/w uploaded from the ground segment and is monitoring instrumental parameters. DPP is held in 'reset' and available for loading of parameters and software

Configuration	For each instrument, and/or the Observatory, a fixed set of commandable conditions which are set up in order to collect science data. Because filters or window sizes may be changed within one mode by varying user definable parameters, any mode may have a number of possible configurations.
CSG	Clock Sequence Generator
CTMS	Central Time Management System
CTU	Central Terminal Unit
CVCM	Collected Volatile Condensable Material
DBI	Digital Bus Interface
DBU	Data Bus Unit
DPP	Data Pre-Processor
DH	Data Handling
DMA	Direct Memory Access—a technique whereby circuits external to the cpu can read and write memory without needing cpu intervention
EGSE	Electrical Ground Support Equipment
EGSE-UI	EGSE User Interface
EGSE-II	EGSE Instrument Interface
eHER	enhanced High Event Rate (spectroscopy)
EID	Experiment Interface Document
EM	Engineering Model
EMC	Electro Magnetic Compatibility

EOBB	Electro-Optical Breadboard
EOL	End of Life
EPIC	European (X-ray) Photon Imaging Camera
EQM	Engineering Qualification Model
ESA	European Space Agency
ESD	ElectroStatic Discharge
eSES	enhanced Single Event Selection
ESOC	European Space Operations Centre
ESTEC	European Space research and Technology Centre
FIFO	First In First Out
FITS	Flexible Image Transport System
Flux Map	Pictorial representation of the contents of a CCD frame or frames with the number of occurrences of each pixel mapped to colour
FM	Flight Model
FMECA	Failure Modes, Effects and Criticality Analysis
FOP	Flight Operations Plan
FOV	Field of View
FPGA	Field Programmable Gate Array
FPP	Focal Plane Platform
FPPR	Focal Plan Platform Reference cube
FPT	Full Performance Test
Frame	'A data set generated by a detector during a quantised period', in the case of the RGS instrument this is a single CCD read out

FS	Flight Spare model
GSE	Ground Support Equipment
HER	High Event Rate
HK	Housekeeping
HTR	High Time Resolution
IC	Instrument Controller
ICB	Instrument Controller Bus
ICD	Interface Control Drawing
IFC	In-Flight Calibration (Source)
inner shell	the pixels immediately abutting a seed pixel: i.e. the pixels in a 2x2 region
i/p	input
IPR	Instrument Positional Reference
IS	Instrument Station
ISR	Interrupt Service Routine
KAL	Keep Alive Line
Kbit	The capital 'K' signifies 1024 rather than 1000
Kernel Mode	IC is running s/w from its onboard ROM and monitoring basic HK parameters. DPP is in 'reset'.
LET	Linear Energy Transfer
LPT	Limited Performance Test
LTMS	Local Time Management System
MA	Mirror Assembly
MCC	Mission Control Centre
MGSE	Mechanical Ground Support Equipment

minDS	Minimum DPP data source, a simple data generator
MIP	Million(s) of instructions per second. A common measure of the capability of a microprocessor.
Mode	The instrument set-up which can be called by telecommand and with which is associated some user specified parameters. A unique data format may be associated with each mode. A specific set-up within the ground segment is required to process/display instrument data from each mode.
MoI	Moment of Inertia
MOS	Metal Oxide Semi-conductor
MLI	Multi-Layer Insulation
MM	Mirror Module
MSSL	Mullard Space Science Laboratory
NB	Narrow Band
Nibble	Four bits
OBDH	On-Board Data Handling
Observation	The set of exposures requested by an observer to fulfill the science investigations into a source, described in the proposal. This may comprise more than one instrument/observatory configuration.
OCB	On Chip Binning
ODE	OM Digital Electronic Unit
OGSE	Optical Ground Support Equipment
OM	Optical Monitor

o/p	output
outer shell	the pixels around the edge of a 2x2 square. Must be empty for the 2x2 event to be reconstructable by v16a
PA	Product Assurance
PCB	Printed Circuit Board
PCU	Power Conditioning Unit
PDU	Power Distribution Unit
PI	Principal Investigator
Pixel	A picture element. In the context of RGS a pixel is specifically one cell of a CCD substrate
PLM	Payload Module
Pointing	A period of time during which the reference axis is held to a specific commanded location in the celestial sphere.
PSPC	Position Sensitive Proportional Counter
QM	Qualification Model
RAE	RGS Analogue Electronic
RBI	Remote Bus Interface
RDE	RGS Digital Electronics
RF	Radio Frequency
RFC	RGS Focal Camera
RGA	RGS Grating Array
RGS	Reflection Grating Spectrometer
Rowland circle	The circle, tangent to which spectral lines are in focus when its radius of curvature is equal to the effective radius of curvature of the gratings used

S/C	Spacecraft
Scenario	The complete suite of set up considerations for an experiment, e.g. CCD read-out mode, DPP thresholds etc.
SCSI	Small Computer System Interface
SCSIM	Spacecraft Simulator
SER	Split Event Reconstruction
SES	Single Event Selection
SEU	Single Event Upset
SI	Système International
SMCC	Service Module Central Cylinder
SOC	Science Operation Centre
Split Event	an incoming photon the deposition of whose resulting charge impinges on the CCD across a pixel boundary
SRON	Space Research Organization, Netherlands
STM	Structural Thermal Model
Surface Map	Pictorial representation of the contents of a CCD frame or frames with the energy of the pixels mapped to colour
TAG	Technical Advisory Group
TBC	To Be Confirmed
TBD	To Be Defined
TC	TeleCommand
TCT	Telescope Central Tube
TM	Telemetry
TML	Total Mass Loss

UCS	Unit Coordinate System
UTC	Universal Time Coordinate
VME	An industry standard backplane specification used for cpu cards and peripheral modules (Versa Module Eurocard)
WFC	Wide Field Camera (as used on ROSAT)
wiper	general user-interface term for a (usually) vertical cursor which can be moved ('wiped') over a region of data displayed in one window, with the effect of this movement displayed in another window
XMM	X-ray Multi-mirror Mission

Bibliography

Wherever possible, references are constrained to those which are likely to be directly available to the reader, but in the case of 'internal' documents used as an authority for some specification which had to be followed, most are available on-line or on request from their originating institute (refer to the contact details in appendix 4). Following the list of items referenced in the thesis is a list of text books found useful during the course of the work.

Al-Janabi K. F. (1999) "The RGS Instrument Controller Software Detailed Design Document" RGS-MSSL-SW-021

Beynon J.D.E., Lamb D. R. (1980) "CCDs and their Applications" McGraw-Hill. ISBN 0-07-084522-0

Blackburn J. K.(1995) "FTOOLS: A FITS Data Processing & Analysis Software Package"ASP Conference Series, Vol 77, 1995

Bond P. (2001) "A Year of XMM-Newton "Astronomy & Geophysics, February 2001, Volume 42 Issue 1, p7

Bootsma, T.M.V., Aarts, H.J.M., Berg, M.L. van den,Brinkman, A.C., Boggende, A.J.F. den, Herder, J.W. den, Jong, L. de, Korte, P.A.J. de, Olsthoorn, S.M. Zwet, E.J. van. and Owens, A.(1996) "Back-illuminated CCDs developped for the reflection grating spectrometer onboard of XMM", in Proc. SPIE vol. 2808 p.p. 481-491, August 1996.

Bootsma T. M. V., van Zwet E. J., Brinkman A. C., den Herder J. W., de Jong L., de Korte P., Olsthoorn S. M. (2000) "Synchrotron Calibration and Response Modelling of Back Illuminated XMM-RGS CCDs "Nuclear Instruments and Methods in Physics Research A 439 (2000) 575–5811

Branduardi-Raymont G. (1996) "RGS Instrument Modes and Telemetry Rates" RGS-MSSL-SE-003

Brinkman, A.C., Aarts, H.J.M., den Boggende, A.J.F. , Bootsma, T.M.V., Dubbeldam, L., Herder, J.W. den, Kaastra, J.S., Korte, P.A.J. de, Leeuwen, B.J., Mewe, R., Zwet, E.J. van, Decker, T.A., Hailey, C.J., Kahn, S.M., Paerels, F.B.S., Pratuch, S.M., Rasmussen, A., Branduardi-Raymont, G., Guttridge, P., Bixler, J.V., Thomsen, K., Zehnder. A. Erd C. (1996) "Reflection Grating Spectrometer on board XMM" Proc. SPIE vol. 2808-42, p. 463, August 1996

Burns A., Wellings A. (1995) "Concurrency in Ada"Cambridge University Press. ISBN 0-521-41471-7

Chambure D. de, R.Laine, K.van Katwijk, J.van Casteren,P. Glaude (1997) "Producing the X-Ray Mirrors for ESA's XMM Spacecraft"ESA Bulletin, issue 89, February 1997 p68

Chun H-J., Bowles J. A., Branduardi-Raymont G., Gowen R. A. (1996a) "On-Board Event Processing Algorithms for a CCD-Based Spaceborne X-ray Spectrometer" Nuclear Instruments and Methods in Physics Research A 376 (1996) 254–2625

Chun H-J.(1996b) "Simulation and Real-Time Processing Techniques for Space Instrumentation" PhD, London

Cottam, J., Boggende, A.J. den, Branduardi-Raymont, G., Brinkman A.C., Erd, C., Guedel, M., Herder, J.W. den, Jansen, F., Kaastra, J., Kahn, S.M., Mewe, R., Paerels, F., Rasmussen, A., Sakelliou, I., Sako, M., Vries, C.P. de "High Resolution Spectroscopy of EXO 0748-67 with the RGS on XMM" (2000), AAS 196, 3418C, 2000

Cruise A. M., Bowles J. A., Goodall C. V., Patrick T. J. (1998) "Principles of Space Instrument Design" Cambridge University Press. ISBN 0-521-45164-7

Danner R. (1993) "Teilcheninduzierter Hintergrund bei Röntgensatelliten", MPE Report 242, January 1993 (Dipl. TU München (3652)

de Vries C. P. (1999a) "Finding Bad Pixels and Columns "RGS-SRON-CAL-ME-99/007

de Vries C. P. (1999b) "Event Reconstruction" RGS-SRON-CAL-ME-99/004

de Vries C. P. (1999c) "Event Sizes for Diagnostic Tools" RGS-SRON-CAL-ME-99/005

de Korte P.A.J.(2000) "Cryogenic imaging spectrometers for X-ray astronomy" Nuclear Instruments and Methods in Physics Research A, in press

de Korte P. A. J., Hoevers H.F.C., Briujn M. P., Bento A. C., Mels W. A., Bleeker J. A. M., Holland A. D., Turner M. J. T. (1999) "The X-ray Evolving Universe Spectroscopy Mission (XEUS)—The Narrow Field Imaging High Resolution Spectrometer 2 (1–10keV)" Proceedings SPIE, Vol. 3766, pp.152-159, 1999

den Boggende A. J. F. (1998) "Hot Pixel Analysis of RFC/FM1 & RFC/FM2" RGS-SRON-CAL-RP98

den Herder J.W.A., van den Berg M., de Korte P. A. J. (1994) "DPP Front-End Requirements & Throughput"RGS-ROL-SE-045

den Herder J.W.A. (2000a) "RGS: Comissioning Results" RGS-SRON-OPR-RP-99/001

den Herder J. W. A., Brinkman A. C., Kahn S. M., Branduardi-Raymont G., Thomsen K., Aarts H., Audard M., Bixler J. V., den Boggende A. J. F., Cottam J., Decker T., Dubbeldam L., Erd C., Goulooze H., Gudel M., Guttridge P., Hailey C. J., Al-Janabi K. J., Kaastra J. S., de Korte P. A. J.,van Leeuwven B. J., Mauche C., McCalden A. J., Mewe R., Naber A., Paerels F. B., Peterson J. R., Rasmussen A. P., Rees K., Sakelliou I., Sako M., Spodek J., Stern M., Tamura T., Tandy J., de Vries C. P., **Welch S. J.**, Zehnder A. (2001) "The Reflection Grating Spectrometer on board XMM-Newton"Astronomy and Astrophysics, Volume 365, No.1 January I, (2001)

den Herder J. W. A. (2001) "Her-X1: HTR Mode Analysis" RGS-SRON-RP-CAL-01/002

Dougherty D., Robbins A. (1997) "sed & awk" 2nd editionO'Reilly. ISBN 1-56592-225-5

Eggel H. (editor) (1995) "XMM Experiment Interface Document, Part A" ESA RS-PX-0016

Erd C. (1999) "RGS Pipeline Meeting #2" ESA ESTEC Meeting Minutes

ESA (1991) "ESA Software Engineering Standards" ESA PSS-05-0, Issue 2

ESTEC (1996) "XMM Experiment Interface Document, Part B" ESA RS-P00191

Ferrando P., Arnaud M., Bouere A., Cara C., Lortholary M., Pigot C., Sauvageot J. L., Schmitt D. (1999) "X-ray Events Recognition and Characterisation in the EPIC/MOS Cameras Onboard XMM"Astron. Nachr. 320

Fraser G. W. (1989) "X-Ray Detectors in Astronomy" Cambridge University Press. ISBN 0-521-32663-X

Gabriel C. (2001) "RGS Diagnostic Trend Analysis Report, February 19th" XMM_SAX_VILSPA/2000_121/TN

Guedel, M., Mewe, R.(1998), "X-ray Spectroscopy Diagnostics with XMM and AXAF: Prospects and Challenges" , in: Proc. 10th Cambridge Workshop on cool stars,stellar systems and the sun, eds. R.A. Donahue and J.A.Bookbinder, ASPConf. Series 154, 1051, 1998.

Hatton L. (1997) "Software Failures: Follies and Fallacies"IEE Review, 03/97, p49-52

Howell S. B. (2000) "Handbook of CCD Astronomy"Cambridge University Press. ISBN 0-521-64834-3

Howell S. B. (Editor) (1992) "Astronomical CCD Observing and Reduction Techniques"Astronomical Society of the Pacific Conference Series, Volume 23 ISBN 0-937707-42-4

James C. H., **Welch S. J.**"Split Event Processing in the MSSL DPP"RGS-MSSL-DPPSW-TN26

Jansen F., Lumb D., Altieri B., Clavel J., Ehle M., Erd C., Gabriel C., Guianazzi M., Gondoin P., Much R., Munoz R., Santos M., Schartel N., Texier D., Vacanti G. (2001) "XMM-Newton Observatory, The Spacecraft and Operations"Astronomy & Astrophysics Volume 365, Number 1, January 2001

Kahn S. M., Cottam J., Decker T.A. (1996) "Reflection Grating Arrays for the Reflection Grating Spectrometer on XMM" SPIE Proceedings vol. 2808, p450–462

Mewe, R., Kaastra, J.S., Bleeker (1996) J.A.M. "XMM: Spectroscopic capabilities",UV and X-ray spectroscopy of Astrophysical and Laboratory plasmas', eds.K. Yamashita, T. Watanabe, Universal Academy Press, Tokyo, Japan, p. 229,1996.

Owens A., McCarthy K. J. (1995) "Energy Deposition in X-Ray CCDs and Charged Particle Discrimination" Nuclear Instruments and Methods in Physics Research A 366 (1995) 148–1541

Pence W. D. (1995) "FITSIO Users Guide: A Subroutine Interface to FITS Format files" Goddard Space Flight Centre/HEASARC, code 668

Peterson J. (2000) "Characterisation of the Anomalous RGS Background" RGS-COL-CAL-000071

Philippus E. (1993) "CCD Image Analysis" Object-Orientated Programming Course Project, Post Hoger Technisch Onderwijs, Amsterdam

Philippus E. (1995) "CCD Image Analysis Requirements and User Manual" RGS-ROL-GSE-016

Pratt T., Bostian C. W. (1986) "Satellite Communications" John Wiley. ISBN 0-471-85966-6

Rees K. J. (1997) "The RGS DPP Hardware Design Specification" RGS-MSSL-DPP-001

Schlesinger B. M. (1994) "A User Guide for the Flexible Image Transport System (FITS)" NASA/Science Office of Standards and Technology, v3.1

Sha L. & Goodenough J. B. (1990) "Real-Time Scheduling Theory and Ada" IEEE Computer, April 1990, pp53-626

Shaw A.C. (1989) "Reasoning about time in higher-level language software" IEEE Transactions on Software Engineering, 1989, 15(7), pp875-89

Stankovic J. A. and Ramamritham K. (1988) "Tutorial: Hard Real-Time Systems" Washington: IEEE Computer Society Press

Stankovic J. A., Humphrey M. (1999) "Predictable Threads for Dynamic, Hard Real-Time Environments"IEEE Transactions on Parallel and distributed systems, Vol 10 No. 3. March 1999

Stankovic J. A. (1988) "Misconceptions about real-time computing" IEEE Computer, October 1988, pp1-186

Tamura T. (2000) "Fixed pattern Noise in the Flight Data of RGS"RGS-SRON-CAL-RP2000/0016

Turner M. J. L., Abbey A., Arnaud M., Balasini M., Barbera M., Belsole E., Bennie P. J., Bernard J. P., Bignami G. F., Boer M., Briel U., Butler I., Cara C., Chabaud C., Cole R., Collura A., Conte M., Cros A., Denby M., Dhez P., Di Coco G., Dowson J., Ferrando P., Ghizzardi S., Gianotti F., Goodall C., V., Gretton L., Griffiths R. G., Hainaut O., Hochedez J. F., Holland A. D., Jourdain E., Kendziorra E., Lagostina A., Laine R., La Palombara N., Lortholary M., Lumb D., Marty P., Molendi S., Pigot C., Poindron E., Pounds K. A., Reeves J. N., Reppin C., Rothenflug R., Salvetat P., Sauvageot J. L., Schmitt D., Sembay S., Short A. D. T., Spragg J., Stephen J., Struder L., Tiengo A., Trifoglio M., Trumper A., Vercellone S., Vigroux L., Villa G., Ward M. J., Whitehead S., Zonca E.(2001) "The European Photon Imaging Camera on XMM-Newton: The MOS Cameras" Astronomy and Astrophysics, Volume 365, No.1 January I, 2001

Theuwissen A.J. P. (1995) "Solid-State Imaging with Charge-Coupled Devices" Kluwer Academic Publishers. ISBN 0-7923-3456-6

van den Berg M.L., den Boggende A. J. F.,Bootsma, T. M. V., den Herder J.W.A., Jansen F. A., de Korte, P.A.J.,van Zwet E.J., Eaton T., Ginige R. (1996) "Back Illuminated CCDs Made by Gas Immersion laser Doping" Nuclear Instruments and Methods in Physics Research A 377 (1996) 312–3191

Welch S. J. (1995) "The Software DPP" RGS-MSSL-GSE-004

Welch S. J. (1996a) "XMM-RGS Effective Area: Impact of the RGS Digital Electronics" RGS-MSSL-SE-019

Welch S. J. (1997) "The RGS DPP Onboard Software Specification" RGS-MSSL-SW-009

Welch S. J. (2000a) "The DPP S/W Version 19.0 Interim Test Report" RGS-MSSL-DPPSW-TN5

Welch S. J. (2000b) "Novel Techniques for the Efficient Reduction of Data Generated by Charge-Coupled Device Detectors" Review of Scientific Instruments, Volume 71, Number 12, December 2000

Wells D. C., Griesen E.W., Harten R.H. (1981) "FITS: A Flexible Image Transport System" Astronomy and Astrophysics Supplement Series 44, 363-3706

Additional Text Books

Amsbury W. (1995) "Data Structures from Arrays to Priority Queues"Wadsworth. ISBN 0-534-04590-0

Axford T. (1989) "Concurrent Programming: Fundamental Techniques for Real-Time and Parallel S/W Design" Wiley. ISBN 0-471-92154-8

Barnes J.G.P. (1993) "Programming in Ada, plus Language Reference Manual" (3rd Edition) Addison-Wesley. ISBN 0-201-56539-0

Cameron D., Rosenblatt W., Raymond E. (1996) "Learning GNU Emacs", 2nd editionO'Reilly. ISBN 1-56592-152-6

CCDS (1997) "Lossless Data Compression"Consultative Committee for Space Data Systems 120.0-G-1

Coffin S. (1991) "UNIX SVR4: The Complete Reference" McGraw-Hill. ISBN 0-07-881653-X

Corliss W. R.(1967) "Scientific Satellites" NASA SP-133, Library of Congress Card Catalogue Number 67-60008

Dubois P. (1995) "Using csh & tcsh" O'Reilly. ISBN 1-56592-132-1

Eccles M. J., Sim M. E., Tritton K. P. (1983) "Low-Light Level Detectors in Astronomy"Cambridge University Press. ISBN 0-521-24088-3

EEV (1987) "CCD Imaging III" EEV 6

Griesen E. W. (1981) "An Extension of FITS for Groups of Small arrays of Data" Astronomy and Astrophysics Supplement Series 44, 371-374

Jacobson, I. (1992) "Object-Oriented Software Engineering"Addison-Wesley. ISBN 0-201-54435-0

Kernighan B. W., Ritchie D. M. (1988) "The C Programming Language" 2nd EditionPrentice-Hall. ISBN 0-13-110362-8

Kopetz H. (1979) "Software Reliability" The Macmillan Press Ltd. ISBN 0-333-23373-5

Longair M. S. (1992) "High Energy Astrophysics, Volume 1", second edition. Cambridge University Press. ISBN 0-521-38773-6

Lynch T. J. (1985) "Data Compression Techniques and Applications" Van Nostrand. ISBN 0-534-03418-7

Lumb D. H., Berthiaume G. D., Burrows D. N., Garmire G. P., Nousek J. A. (1991) "Charge Coupled Devices (CCDs) in X-ray Astronomy" Experimental Astronomy, 2, p179-201

Paull M. C. (1988) "Algorithm Design" John Wiley. ISBN 0-471-81688-4

Skansholm J. (1995) "Ada from the Beginning", 2nd editionAddison-Wesley. ISBN 0-201-62448-6

The Software Productivity Consortium (1992) "Ada Quality and Style: Guidelines for Professional Programmers" SPC-91061-CMC

Acknowledgements

I am very grateful for the extensive help received from my supervisor Dr. Graziella Branduardi-Raymont, whose diligent guidance helped to shape not only the work as it progressed, but this thesis also. Grateful thanks are also offered to Mr. K. Norman and Dr. R. Gowen who made up the rest of the Ph.D. panel: they were kind enough to offer plenty of assistance with proof-reading and advice on the contents of this thesis and I was happy to take advantage of them for this.

Further thanks for miscellaneous proof-reading and advice go to Dr. A. Khanifar; Dr. I Sakelliou; Dr. D. M. Walton and Dr. K Al-Janabi (author of the XMM-RGS instrument controller software), Mr. R Ottley and Dr. C. P. de Vries at SRON.

Finally, I am especially grateful to Mr. Phil Guttridge, my line-manager at the Mullard Space Science Laboratory, for his support and encouragement in this endeavour—and his toleration of my 'working to rule' while getting the thesis write-up completed.

Colophon

Some details are given about the production of the software source code and the mechanical production of this thesis.

All of the source code was edited using the native text editor available in the OpenWindows desktop running under Solaris 1. During the course of the project, I discovered 'emacs', which I downloaded and installed along with its Ada language extensions, and this certainly seemed to have promise as a replacement for line editors such as 'vi', but learning this product seemed too likely to be a distraction, though it did yield two very helpful aids. Firstly, it provided a mechanism for matching opening and closing statements—very useful for tracing errors in loop structures (i.e. one could highlight, say, an 'if' and have emacs find the corresponding 'end if'). Secondly, when working with the very long text files engendered by extensive use of in-line coding, it was very easy to lose track of indentation conventions—especially with extensive use of cut-and-paste (most occasions where there were identical sections of code for side_C or side_D origin pixels I worked on side_C only, and when compiling and executing ok, would copy this to a temporary file and let the machine search and replace all of the '_C' suffixes with '_D' suffixes. This was an effective means of ensuring that both versions of the code behaved in the same way, but tended to randomise the indentation structure). To avoid spending too much time on this largely aesthetic issue, periodically the source files could be loaded into emacs and the 'indent lines in

region' command called, having selected the entire body of the text. This would then sort out the formatting according to the standard Ada conventions.

A further product which appeared and was experimented with was 'GRASP', and this showed promise as a useful graphical text editor with extra widgets for the programmer, but this was also rejected for practical reasons owing to time constraints.

Portions of this thesis have been underway for several years, and a number of different applications employed. The line diagrams were mostly produced in ClarisDraw, and the flowcharts in ClarisImpact, both very useful programs now sadly moribund. The bulk of the text was originally written using WordPerfect, but during the collation of the material it became clear that this also abandoned product would not be able to cope with the size and complexity that was being accrued. This didn't become clear until the last few months before submission was due—rather late to be learning a new product; but the bullet was bitten and other options were investigated. Meanwhile individual chapters were composed in WordPerfect and released one-at-a-time to the panel for reading. After some research, Adobe FrameMaker emerged as the clear product of choice, and a copy was duly procured to start learning while the individual chapters were out for proof reading.

Every graphic entity was individually saved into its own file as encapsulated PostScript, including the screenshots which had to be taken with the Solaris 'snapshot' tool, then opened in Xview to be saved as a '.jpg' format, and finally opened and edited in Adobe Illustrator for saving into PostScript. IDL was used to generate the graphs, and each frame is again its own '.eps' file. !p.multi was not used to compose the various collections of graphs, it being more flexible to

leave this composition to FrameMaker or Illustrator as these were the two work-horses available on my computer at home. The body text is set throughout in Palatino (I might have chosen a more unusual face, but I was very keen to use all the correct typographic components, and the basic Palatino font set being so widely included in operating systems and printers meant that I only needed to purchase the extras for the old-style figures). Diagram labels are in Helvetica-Narrow, which survives printing at very small sizes.