Reliability through redundant parallelism for micro-satellite computing

IAN VINCE MCLOUGHLIN
School of Computer Engineering
Nanyang Technological University
Block N4, Nanyang Avenue
Singapore 639798
and
TIMO ROLF BRETSCHNEIDER
EADS Innovation Works Singapore
European Aeronautic Defence and Space Company
41 Science Park Road #01-30
The Gemini, Science Park II
Singapore 117610

Spacecraft typically employ rare and expensive radiation tolerant, radiation hardened or at least military qualified parts for computational and other mission critical sub-systems. Reasons include reliability in the harsh environment of space, and systems compatibility or heritage with previous missions. The overriding reliability concern leads most satellite computing systems to be rather conservative in design, avoiding novel or commercial-off-the-shelf components. This paper describes an alternative approach: an FPGA-arbitrated parallel architecture that allows unqualified commercial devices to be incorporated into a computational device with aggregate reliability figures similar to those of traditional space-qualified alternatives. Apart from the obvious cost benefits in moving to commercial-off-the-shelf devices, these are attractive in situations where lower power consumption and/or higher processing performance are required. The latter argument is particularly of major importance at a time when the gap between required and available processing capability in satellites is widening. An analysis compares the proposed architecture to typical alternatives, maintaining risk of failure to within required levels, and discusses key applications for the parallel architecture.

Categories and Subject Descriptors: B.8 [**Hardware**]: Performance and Reliability - *Reliability, Testing, and Fault-Tolerance*; C.3 [**Computer Systems Organization**]: Special-purpose and application-based Systems - *Real-time and embedded systems*; Performance of systems – *Fault-tolerance*; C.1 [**Computer Systems Organization**]: Processor Architectures – *Parallel Architectures*
General Terms: Design, Reliability, Performance
Additional Key Words and Phrases: Fault tolerance, parallel processing, COTS components, on-board computer, micro-satellite, hardware redundancy, FPGA

1. INTRODUCTION

A survey of contemporary micro- and mini-satellites reveals that the majority incorporate specialised radiation qualified computational components for control and data handling subsystems [Kramer 2002]. This also holds true for medium-sized and large-scale missions such as ENVISAT. Two attributes of radiation qualification conspire to ensure that such qualified technology lags several generations behind current commercial technology: Firstly, the correlation between radiation-induced error and semiconductor feature size and secondly, the high cost of performing complex and destructive qualification tests. A number of initiatives, for instance the ERC32, have attempted to address this generational lag, however, satellites powered by early derivatives of the x86 processor are still common. While these provide sufficient computing resource for some missions, they cannot be expected to keep pace with the likely increasing complexity of most of the forthcoming missions. The only practical solution for many projects will be the utilisation of commercial-off-the-shelf (COTS) components. Note that in this paper the COTS concept is handled purely from a technical point-of-view. Issues relating to the procurement and adoption of such devices were addressed in [Bretschneider 2008].

The case for judicious use of COTS components has of course been widely published for military systems and for satellites [Black and Fletcher 2005; Abbott et al. 2001; Elias 2000; Lovellette et al. 2002], but is not normally applied to mission-critical subsystems without extensive backup. The general justification of a COTS approach revolves around cost factors, however the generational gap alluded to above strengthens the arguments for use of COTS devices wherever factors such as power consumption, integration density, or processing capability are mission relevant considerations.

Fault tolerant design approaches [Chau et al. 1999; Perschy 2000; Cardarilli et al. 2003; Cardarilli et al. 2005] have shown the ability to improve the overall reliability of a system through design, and are complementary to the question of component choice. One particularly simple, although effective, fault tolerant design technique is that of reliability through redundancy. This is already an important design approach for space systems, where single-point (and even many multiple-point) failures are discovered and mitigated through redesign or redundancy at an early stage in the satellite design process. Orbital conditions are considered harsh to electronics: failures occur in orbit due to single event upsets (SEU) induced by cosmic ray disruption of semiconductor gate charge [Wertz and Larson 1999] and other related factors. A certain density of instantaneous run-time and (semi-) permanent errors must be compensated for by design. Ideally, systems experiencing failures should degrade gracefully, rather than catastrophically, under such conditions.

This paper describes a parallel architecture computer that consists of a collection of low-power medium-performance embedded COTS processors supported by an FPGA-based interconnection framework and middleware to provide a parallel system. The design exhibits high computational performance (for a satellite computer), relatively high reliability, achieves graceful degradation and allies these attributes with low power and cost. This computer was originally designed to operate as the Parallel Processing Unit (PPU) for Singapore's first remote sensing satellite, X-Sat [Bretschneider et al. 2005].

The remainder of this paper is organised as follows: Section 2 presents the motivation for the incorporation of COTS components in the design of spacecraft and illustrates application areas, which make use of the newly provided computational resources and processing opportunities that may be rare in traditional satellite computers. Based on these, Section 3 derives system requirements and sets the development with an actual micro-satellite within a defined context. A literature survey analyses related work in Section 4 and is, together with the system requirements, the foundation for the design description in Section 5. In addition, software related aspects are covered in this section. The following Section 6 examines the proposed architecture with respect to the previously listed applications. Section 7 discusses parallel computations using such architecture before Section 8 concludes the paper.

2. MOTIVATION

The demand for powerful processing systems in space is driven by three major developments. Firstly, novel instruments, e.g. for hyperspectral data acquisition [Pearlman et al. 2003] and polarimetric synthetic aperture radar (POLSAR) imaging [Breit et al. 2007], provide a data volume rarely encountered before, and which cannot be economically downloaded to ground receiving stations unless severe restrictions in terms of capture duration, resolution and swath width are introduced. Secondly, deep-space explorations, like the successful Beagle-2 [DiGregorio 2003] and recently landed Mars Reconnaissance Orbiter [Graf et al. 2002] missions, require a high degree of autonomy to ensure instant response of the systems to the surroundings without depending on time-lagged communication with a ground control station. Thirdly, the increase in mission complexity as a response to the competitiveness of the market, especially in the area of remote sensing, demands sophisticated algorithms with massive requirements for computing power.

These issues call for increased computation which can be provided by selecting COTS devices. To compensate for any reduction in reliability caused by using COTS components rather than radiation hardened (or tolerant) space-rated parts, the technique of reliability through redundancy is commonly used [Cardarilli et al. 2003]. If this is applied to COTS processing units (PU), e.g. CPUs, DSPs, FPGAs, several may need to be provided in the design. These are either configured as hot redundant, where all are operational simultaneously, or cold redundant, where a replacement is powered up on the failure of a previously operational device. The exact operational choice is a design issue which involves questions of outage probability, outage latency, power consumption and PU lifetime in orbit. Design conservatism may well mandate a deliberate over-provision of PUs. Assuming a completely operational hot redundant system, a high aggregate processing capacity is theoretically available at launch, but during the lifetime of a mission, as components gradually fail, aggregate processing capacity will stepwise reduce. Reliability matching ensures that the probability of the computing resources remaining operable by the end of the mission duration will match the target mission success probability.

Given an enhanced computing ability made up from spare computing elements, we classify applications which could use this capacity before we discuss the exact methods of providing the capacity later. In this case, three in-orbit data processing classes are identified by their relationship to potential hardware solutions:

1. Data compression: If downlink bandwidth is a limited and operationally expensive resource, then on-board data compression becomes advantageous. Depending on the type of data, both loss-less and lossy compression can be performed in orbit on either a real-time or non-real-time basis.

2. Data selectivity: Most missions operate in batch mode using time-tagged commands (in effect a scripting language) to control future satellite operations. Local conditions at the operational location are therefore not always adequately predictable at programming time. Data selectivity describes applying some form of value judgement on captured data in an autonomous or semi-autonomous fashion. For instance, extensive cloud cover renders optical imagery worthless. An on-board analysis of data as it is captured could help to discard inadequate images prior to storage and downlink, thus increasing the proportion of stored images which are useful. In a limited duration mission, with limited downlink bandwidth, increasing the usefulness of resulting images by discarding less useful ones in orbit would increase the overall value of the mission.

3. Data acquisition autonomy: Most earth observation missions do not operate their imaging payloads continuously, partially due to the bottleneck between capture bit rate and downlink bandwidth. The introduction of high performance on-board processing would enable a number of new mission modes with more frequent scene acquisitions. An example is a “standing watch” function, where image data is acquired and analysed in real-time. When interesting features are detected, e.g. forest fires or landscape changes, the corresponding data is stored or even further enhanced through autonomous operation of additional idle instruments. Other data that shows no features of interest may be discarded. Autonomous operation involves the mode or behaviour of the satellite being influenced by the analysis result of captured data prior to that data being downlinked.

All three depicted application classes have a focus on image processing motivated by the large data volume which cannot be processed straightforwardly by radiation-hardened solutions. Non-optical sensing methods suffer from similar problems, for example POLSAR data [Breit et al. 2007] which is acquired at over 1 Gb/s.

3. SYSTEM REQUIREMENTS

The actual requirements of an on-board payload data processor include physical spacecraft limitations of volume, mass, power consumption, heat dissipation and so on, as well as overall mission objectives. This section considers requirements for a typical micro-satellite which would perform image processing. In this instance, the X-Sat satellite, similar to many university micro-satellites, is discussed as an example. Only relevant aspects, i.e. those related to the processing and handling of the acquired images, are discussed in detail.

3.1 Satellite Environment

X-Sat is a micro-satellite designed for launch into a low earth sun-synchronous orbit at a nominal altitude of 685 km [Bretschneider et al. 2005]. At this altitude, it is expected to be exposed to low levels of ionising cosmic radiation and, with a mission lifetime of three years, a total radiation dose of up to 10 krad (100 Gray). The satellite is a 600 mm x 600 mm x 850 mm cuboid with mass 110 kg. Power is stored in dual Li-Ion battery strings charged by twin deployable solar panels. Design pressure was predominantly toward power consumption, volume and mass reduction whilst increasing reliability in the face of hostile environmental conditions. Cost is a secondary, but still far from unimportant, factor.

The mission objective is primarily to capture multispectral images in the visible and near-infrared at 10 m spatial resolution over South East Asia for downlink, ideally during the same orbit. Imaging will normally be commanded by uploading a sequence of future time tagged commands as the satellite passes over a ground station. The satellite queues these and acts on them at the times encoded in each command string. Such commands, which are executed by a simple cold redundant on-board computer (OBC), can specify operations such as attitude control (orientation manoeuvring), data acquisition, and downlinking of images. In the standard configuration, image data is stored in an on-board Solid State Recorder (SSR, formerly published as the RAM-Disk), and downloaded via high-speed X-band radio modem.

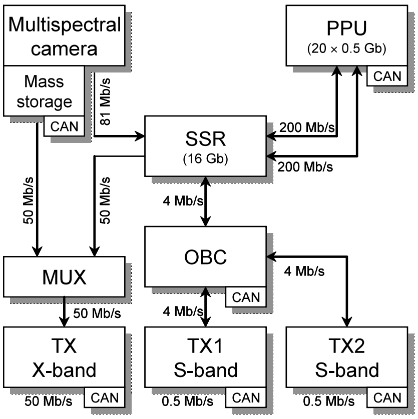


Figure 1: Data flow within X-Sat related to image data acquisition and processing

The imagery-related data flow within X-Sat is represented in Figure 1. The camera provides image data via two interfaces. The first allows image data previously stored in the camera's internal memory to be transferred directly to the X-band radio modem for downlink, allowing for acquisition of 14 standard scenes (each representing a 50 km x 50 km tile) in a 'store-and-forward' mode (in this configuration imaging and transmission cannot be conducted simultaneously due to the mismatch between acquisition and downlink rates of 81 Mb/s and 50 Mb/s). The second interface allows captured data to be transferred to the SSR during imaging for storage. From there, images can be accessed by a parallel processing unit (PPU) for processing – either streamed in real-time, or processed in units consisting of smaller image tiles. Processed data may be downlinked to the ground either via the X-band radio modem at 50Mb/s, or by one of two S-band transmitters at a much slower rate of up to 1 Mb/s. Most of the components shown in Figure 1 are controlled via cold redundant controller area network (CAN) connections, which themselves are dual redundant.

The PPU is required to provide sufficient processing power for basic compression needs constrained to a maximum power consumption of 22 W, volume of 4000 cm^3, and mass of 1 kg (incl. packaging). Due to its basic flexibility, any processing capability beyond the minimum is not wasted, but considered a bonus. The PPU is to remain functional at the end of a 3-year lifetime, exposed to space radiation, with a probability of 0.9 or greater, to match the reliability of the space-graded OBC. Connectivity includes the dual 80 kb/s CAN links already mentioned, plus two 200 Mb/s low voltage differential signalling (LVDS) connections with the SSR, all links being bidirectional and redundant.

3.2 Operational and Programmability Requirements

Two crucial aspects defining the requirements for a space-borne data processing platform are the application specific acceptable latency, and processing power. For the PPU, real-time is defined as the successful extraction of selectable features from a scene before the imaged area is out of transmission sight. Hence, the available processing time can be derived based on the nominal orbit and is approximately 300 s for nadir imaging. However, if the camera is configured to take very long image strips and not individual scenes, then a constant data stream of 81 Mb/s has to be processed continuously. In this scenario, the 300 s requirement defines the acceptable latency with respect to a particular observation point rather than the available processing time.

The computational processing power for a flexible data processing computer is difficult to accurately determine, but must be fixed and agreed before design solutions can be compared. Based upon an ensemble of likely processing algorithms including a modified JPEG2000 compression scheme [Trenschel et al. 2003] and the "standing watch" function, a figure of 2500 MIPS processing speed was adopted for the requirement in X-Sat.

The success of a deployed on-board computation platform also depends on the ease of use – primarily the programmability. While highly specialised hardware solutions may achieve the highest processing performance, they tend to require more attention in terms of system development as well. On the ground, many remote sensing applications are based on stand-alone PCs or compute clusters [Goller and Leberl 2000; Aloisio and Cafaro 2003]. Developers of such algorithms are familiar with these architectures, and have adapted their algorithms to them already. Thus, an inherent advantage in terms of porting algorithms from the ground to space can be gained by adopting an architecture with a high degree of similarity to those ground-based systems.

4. RELATED DEVELOPMENTS

Numerous developments of high performance data processing units for spacecraft in low earth orbits have been published previously. This section provides an overview of selected systems.

4.1 Radiation-Hardened Processing Components

Radiation-hardened processors are used for spacecraft computing systems to achieve high reliability under orbital radiation conditions, where radiation-hardness is defined by the withstanding of at least 10^6 rad of total radiation dose, 10^9 rad of prompt dose, and a maximum of 10^{-10} error/(bit×day) single event upsets [Baumann 2005]. Several space-worthy processors are available across the performance range. For contemporary 32-bit spacecraft computers, these include those based on the RH32, RAD750 (a radiation hardened version of the PPC750), GVSC1750A, R3000, RAD6000 and ERC32 (a radiation hardened SPARC-architecture processor). These achieve computation performances in the range of 20–300 MIPS [Miller et al. 2001].

Apart from general-purpose CPUs, a number of space-worthy DSPs exist [Persyn et al. 2001; Sampson et al. 2002]. For static data processing tasks, the utilisation of FPGAs may be preferable since they can significantly outperform DSPs in terms of integer computation speed [Baghaei et al. 2004].

4.2 Selected Commercial-off-the-Shelf Processing Solutions On-Board Satellites

A study by the US Air Force Research Laboratory on applications in the first half of this decade indicated computational power in the range of Giga instructions per second would be needed while at the same time demanding an efficiency of at least 500 MIPS/W [Nedeau et al. 1998]. A specific investigation into on-board image data processing, specifically for cloud removal and atmospheric correction, came to the conclusion that at least 300 MIPS were required, which narrows the available choice of above mentioned processors down to the RAD750.

In order to widen component choice, a constant push for the utilisation of non-radiation-hardened COTS components is observable. One successful example for the application of COTS processors is in the BIRD satellite [Brieß et al. 2005], which employed two hot- and two cold-redundant PPC750 devices for its mission critical on-board computer. DSPs are also used for particular applications such as SAR data processing or image compression.

As mentioned previously, FPGAs are applicable to static data processing tasks, and can outperform DSPs in terms of power, speed and required volume (if all auxiliary components are considered). Such solutions were launched on the Australian FedSat and the South Korean KitSat-3. The main disadvantage is the increased algorithm development effort since hardware-related programming is required in order to benefit from the available processing speed.

Finally, some designs involve processors ranging from highly specialised systems like the neural network processor NI1000 [Halle et al. 2001] as part of BIRD’s thematic on-board data processing experiment, to the 32-bit IDT R-3081 [Lovellette et al. 2003]. Other hybrid systems consist of combined FPGA and DSP solutions like the GEZGIN on-board BILSAT-1 [Ismailoglu et al. 2002], and these form an alternative design strategy which combines different implementation philosophies.

Although it is perfectly feasible to build a satellite computer using either FPGAs, DSPs, or both, a critical argument in terms of the X-Sat mission requirements was in the ease of programmability and software development. It is no exaggeration to suggest that as computer complexity increases, software development and reliability issues may overtake hardware development and reliability issues as the most mission critical activities in a new development.

In particular, the ease of porting a ground-based algorithm running on a parallel processing cluster, to satellite, is considerably eased by the availability of a compatible space-borne parallel system that utilises the same operating system (OS), and compilation tools.

5. HARDWARE DESIGN

This section describes the PPU architecture which uses a novel interconnected set of COTS processors to achieve a relatively high processing performance at low power, low cost and small volume. We will later see that the PPU design, although based on processors without radiation hardening, can match the reliability figures of a fully radiation-hardened design.

5.1 Component Choice

Straightforward programmability is an advantageous feature in shortening development timescales, and lowering the probability of software failure due to coding error. In the PPU, this is achieved through designing an on-board system that resembles common ground-based architectures as closely as possible, including utilising the same OS, development and debugging environments.

On-board processing comes at one-off costs primarily in additional lift-off mass, design cost, component cost, as well as an ongoing electrical power cost. The latter can be assessed by the computational efficiency of the entire solution using a measure such as MIPS/W. With a processing requirement of 2500 MIPS, a solution based on a single radiation-hardened processor is not achievable with the current choice of components. Therefore, a parallel approach is necessary. Assuming the most powerful of the described space qualified processors previously mentioned are used, nine RAD750 would be required to maintain such a peak performance. However, with a mean power consumption of 10.2 W at full clock rate per processor unit [Burcin 2002], or even the 5 W consumption of the raw processor core, such a solution is impossible within the 22 W power constraint. Similarly, the calculation can be repeated for the other radiation-hardened processors resulting in equivalent scenarios. Hence, COTS components with their generally lower power requirement must be utilised. The main competitors in the field of possible processors with space heritage are the PowerPC PPC750 and the Intel SA1110 (StrongARM). The first achieves 134 MIPS/W [Lammers 1998] and the latter 575 MIPS/W [Hitachi 2000]. In a multi-processor arrangement, both could meet the set target of 2500 MIPS while consuming less than 22 W, the SA1110 noticeably outperforming the PPC750 on power consumption. In terms of chip volume also, the SA1110 has a 256 ball miniBGA package whilst the PPC750 has a 360 ball CBGA, which is larger and, hence, more difficult to accommodate (note that the commonly quoted 7.6 mm x 8.8 mm PPC750 dimensions [Swift et al. 2001], are for the silicon die only, and do not include the packaging). Relating PCB area with the provided computational resources the PPC750 yields only 48 MIPS/cm^2, compared to 69.2 MIPS/cm^2 for the SA1110.

Similarly, the integration density of the individual processors has to be considered since the likelihood for radiation related upsets roughly increases in line with the manufacturing silicon feature size and density of the IC. In the absence of detailed radiation tolerance figures, feature size provides an estimate of the survivability of individual processors. The PPC750 feature size is 0.29 μm with more than 6.3 million transistors [Swift et al. 2001], and the SA1110 was manufactured in a 0.35 μm process with only 2.5 million transistors. Thus, the SA1110 may well survive slightly better than the PPC750 due to its older manufacturing process, although only detailed radiation testing would establish this definitively.

Note that the criteria addressed are predominantly integer processing abilities with respect to power consumption and area, since all of the image processing algorithms to be applied in the described mission are fixed-point. If floating-point capabilities were to be considered, such a comparison may not appear quite so one-sided. The case for power efficiency is also dependent upon the exact processing being performed, and especially whether data to be processed can reside in on-chip memory or must be off-chip (figures quoted assumed that all such issues were equivalent in the two devices).

Given the outlined requirements in terms of power consumption, volume and mass, the SA1110 was selected. The major problem introduced by this decision is, of course, the fact that the SA1110 is no longer manufactured. However, satellite computers are not mass-market items, and so we have found it relatively easy to stockpile sufficient processors for forthcoming missions (the devices are still easily available online).

5.2 Architecture and Interfaces

The PPU's interconnection backbone consists of two one-time-programmable interconnected FPGAs (Actel AX1000, 100 MHz) utilising anti-fuse technology. These enable real-time streaming image processing capability, but foremost provide the network topology connecting the 20 processing nodes (PN). Each PN consists of one Intel SA1110 (206 MHz) processor and 64 MB of Samsung SDRAM (Samsung K42561632D).

Originally the authors had envisaged a design utilising four COTS SRAM-based Xilinx FPGAs [McLoughlin et al. 2003] to service the PNs. However, since reconfigurability of the interconnect logic is not required in the mission requirements, the Xilinx FPGAs were replaced with Actel devices. It was also unnecessary to use COTS FPGAs for reasons of higher density logic or clock speed.

The selection of the anti-fuse FPGAs was also due to their central functionality within the design, where a loss would compromise up to ten functional PNs. Thus, it was decided to sacrifice re-programmability for reliability, since anti-fuse technology is more tolerant of radiation than most other alternative FPGA architectures [Wang 2003]. Flash-based variants of the final devices were loaded to the boards for prototyping.

The move to higher reliability FPGAs no longer mandated the originally intended four-way replication for fault-tolerance reasons, but could not reduce to a single device for reasons of pin count (each PN must connect to the FPGA using dedicated parallel buses – shared buses would run the risk of a faulty PN disabling the bus for all other connected PNs). Thus provision was made for two FPGAs, each in a 676-ball package.

Given the architecture designed by the authors, shown as a block diagram in Figure 2, and viewing the FPGAs as a network, the PPU resembles a Beowulf cluster [McLoughlin et al. 2005].

Every PN implements an individual 17-bit parallel data bus to the hosting FPGA, and contains dedicated power switching circuitry to isolate it from power faults occurring in other PNs. The 17-bit data buses operate at a selectable speed to provide an interrupt-driven direct memory access (DMA) channel into the central routing network. Both raw data and commands share the parallel link as 16-bit wide entities differentiated by the state of the 17th bit. The 17-bit communications scheme, which appears to be unique to this development, allows a PN to differentiate between incoming data and commands without requiring a read to any status register, thus doubling the efficiency of message transfers from FPGA to PN over a pure 16-bit addressed scheme [McLoughlin et al. 2005]. Efficiency is achieved by wiring the 17th bit to the most significant bit (MSB) position of the 32-bit processor bus, rather than to the more obvious bit 17. In the ARM processor architecture, this has the effect of making the transfer either signed or unsigned depending on the type of message (i.e. the MSB, sign bit, is either 0 or 1). Thus, a simple differentiation in the handling of received words can be made with very low instruction overhead, due the ARM instruction set characteristic of having predicated instructions based on status flags, in this case, the negative flag.

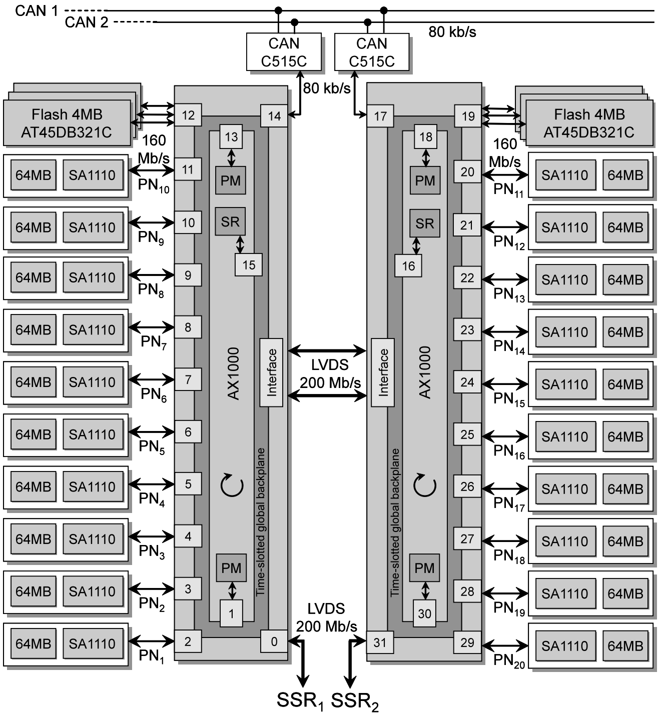


Figure 2: Schematic overview of the PPU

PN to FPGA write efficiency is neither improved nor degraded since these can be distinguished though write addresses by the PN to denote different message types. PNs act as interface masters, but have the responsibility to be sensitive to command messages provided by the FPGA. This means that interrupt-triggered transfers are used, with minimal buffering within the FPGA based on experienced worst-case latency. Regular heartbeat signals are command messages from the FPGA, eliciting a response from the CPU, which provide mutual liveness indicators. When absent from a CPU, the hosting FPGA immediately power cycles the PN. Hardware over-current monitors perform the same function much quicker in response to high-current conditions, such as caused by incipient single-event burnout (SEB) instances [Wertz and Larson 1999]. Following power up, PNs are booted from program images residing in the centralised 6-way redundant flash memory, arbitrated by the FPGAs. The flash contains basic OS, and interfacing code, along with common applications. PNs can voluntarily reboot, or be power cycled by the FPGAs in which case, new code is loaded on demand. Since any PN can be commanded to receive any untyped stream of data from the SSR, other PNs, OBC or elsewhere, this system allows additional programs to be uploaded and then executed by PNs. This mechanism provides the capability to upload new programs from the ground during a mission, and for software payloads [McLoughlin 2001] which would allow multitasking programs supplied by third parties to occupy individual PNs in an analogous way to the hosting of physical satellite payloads by third parties.

6. SOFTWARE DESIGN

VxWorks, Linux and several other embedded operating systems (e.g. μCOS, ECOS, μITRON, SCOS, Salvo, DOS) were potential choices by virtue of having space heritage. Of these, Linux is arguably the least real-time oriented choice, but the most feature-rich. Other have a far lower resource profile while VxWorks has the better space heritage, but it is not open source – something that is an extremely important attribute in ensuring a reliable and fault-free system through review and examination of the underlying code.

Adapting the Linux OS to a multiprocessor system is trivial since it is already commonly used this way on the ground – even in '5 nines' reliability systems – and built-in OS support is readily available for the handling of higher-level protocols, including message passing. These advantages lead to more rapid deployment and compact code, which in some way compensates for the larger footprint. Although the space-heritage of VxWorks cannot be matched, there has been significant and growing space heritage for Linux. NASA has used it for rendezvous and docking purposes [Ortega 1999] and Surrey Satellite Technology, one of the leading designers of small satellites, tested it successfully on the UoSat-12 spacecraft. PPU software is written in a subset of ANSI standard C adopted from the Motor Industry Research Association (MISRA), designed to maximise software reliability [Hatton 2004]. Embedded Linux has proven to be reliable and developer-friendly, and most importantly is readily modifiable to a mission's needs through its open source nature. Approximately two man-months of effort were required for porting a standard ARM-Linux 2.6 kernel to the PPU hardware.

PN software is divided into four modules: Firstly, low level self-test, boot, and programming software operate on each PN in an identical fashion. These implement monitoring functions and software error detection and correction (EDAC) on data and program memory areas. A second level of software encompasses the drivers that handle communications with the FPGA. These drivers respond to control events from the FPGA (such as heartbeat signals), and manage buffers and queues that are used by the third software module which maps a given parallel processing topology to the PPU infrastructure. Finally, the fourth module is the application code, which is selected from a library of possible code images held in local flash memory, within the SSR, or uploaded directly from the ground.

Each node runs its own instance of a customised Linux kernel under a loosely asynchronous scheduling paradigm. The boot initialisation of the system registers, memory and other peripherals for each PN is performed by a simple tightly coded two-stage custom boot-loader. The first stage code is served dynamically by the FPGA to each PN at boot time, sufficient to load and execute the second stage which is stored in two independent blocks of triple-redundant flash memory, connected to the FPGAs. The boot-loader is ultimately responsible for verifying the status of the SDRAM and loading the Linux kernel as well as initial *ramdisk* images. Integrity is assured through pre-computed stored checksums.

In order to keep the cost and complexity of the PPU low whilst providing reliable protection against SEUs due to harsh space radiation conditions, a software EDAC (error detection and correction) [Shirvani et al. 2000] scheme was implemented, rather than a hardware solution. Applications can request certain sections of their memory to be EDAC protected by the OS [Ramesh et al. 2004], primarily read-only data areas or program code, which are not expected to change during normal program execution. The OS uses a discrete block of memory to compute and store Hamming checksums for protected memory blocks. This is used during read accesses to check for and correct any errors detected. Scrubbing of the entire EDAC protected region is scheduled periodically, at the expense of additional overhead.

Flying a configurable computer that resembles earth-bound cluster systems in space can lead to improvements in software quality assurance once the hardware, OS and process as a whole are assured, and can improve ease of testing. Firstly, the same compiler and library functions source code are used for both the flight software and the software developed and tested on a PC on the ground. This brings the advantage of similarity to extensively tested code developed on the PC and the code flown on the satellite. Moreover, the approach has the associated benefit of allowing far more extensive test cases to be constructed. Moving the bulk of software testing to an environment with easier developer access and upon which many useful tools are located (such as *lint*, *gprof*, *electric fence*, *doxygen* etc.) can improve the software in reliability terms. Secondly, the loose asynchronous real-time nature of the PPU is unlike that of a tightly coupled hard real-time signal processing computer. Whilst this may have disadvantages in terms of overall computational latency, it is highly advantageous in removing the complication of hard real-time interactions in a system built of inherently failure-prone nodes. Failures can thus be handled at a software level rather than hardware level.

6.1 Control Software

The OBC holds a file allocation table for storage areas in the SSR, and is responsible for tasking both this and the PPU as standalone modules. These are peripheral systems from both hardware and control perspectives. Messages, defined as command or data, are dispatched by the OBC via CAN to the PPU and are addressed to certain PPU bus nodes, for instance PNs, processing modules or status registers. Job specifications, sent to individual PNs to command them to commence processing, provide a handle to a processing application which is either already in the PN (having been loaded with the OS from flash memory), streamed from the SSR, or transferred over CAN as a file from the OBC. The job specification can also indicate the degree of parallelism required, depending upon the application being ordered, tasking the selected PN to dispatch sub-jobs to any other idle PN (which becomes a slave node in the process). The information on PN status is kept inside the FPGAs. All slave nodes report back to their master node which communicates with the OBC. The main objective of this administrative concept is to minimise the involvement of the OBC, which regards the PPU as a slave-driven computational resource. All data has to be provided at the beginning of the execution since no interaction with the OBC is intended other than return of the final result. This reflects the classical parallelisation concept of UNIX using the *fork* command. A master PN is responsible for error handling, restarting crashed nodes, ensuring internal communication, redistributing jobs away from unreliable nodes, load balancing etc., performed on a per-application basis. The OBC only becomes involved if the master PN itself crashes, the SSR crashes, or if communication links and backups die.

6.2 Communications Architecture

The PPU enables not just arbitrary programs to be executed on the PNs but also caters for the reorganisation of the two FPGAs. In the default FPGA configuration, the various network entities in the PPU architecture are interconnected via the FPGAs, providing a communication network called a timeslotted global backplane (TGB). This is a message-passing bus specifically designed for general purpose inter-network-entity and broadcast communication, as shown in Figure 2. This provides flexible network services to connect, for instance, a designated number of physical PNs in a topology that appears to users logically as a mesh. Both FPGAs are symmetrical with the difference that during the initialisation of the PPU a self-test selects one of the FPGAs to be the master, which then holds the system status register (SR). The SR describes the state of the different PNs in terms of available functionality and processing status as well as the status information regarding crucial links and internal modules. A corrupt SR can quickly and efficiently be rebuilt through another PN or the OBC broadcasting a heartbeat message to the TGB node of each PN to determine its current status.

The design of the TGB system was motivated strongly by a desire to operate a very simple, and thus reliable, system with minimum FPGA overhead. As with the PNs, the system was designed to tolerate failures, although individual messages or in-progress jobs would often be lost in the process. In the default TGB arrangement, messages are conveyed to the various entities as shown in Figure 2, wiring all the addressable nodes in one FPGA together. The TGB then loops through the elements in the other FPGAs before returning (in a link or FPGA failure situation it reverts to single FPGA mode). If a PN is turned off or has failed, the internal TGB node for that element continues to operate.

Each node on the one-way loop has a unique address identifier from 0 to 31. Thus, there are 16 nodes in each FPGA and 32 in both FPGAs. The TGB links consist of a data bus and a frame synchronisation signal. TGB messages are 32 bits long, with fields indicating message type, sender address, destination field, broadcast bit and address parity. The most significant bit of a message is always set, to distinguish empty slots, which are always at least 32 bits long. The message type identifier differentiates between control and data messages, both of which have a 16-bit data payload. Parity bits protect the address fields, payload and control fields separately.

The destination field contains a big-endian count of the number of nodes from source to destination that decrements each time it traverses a node. Since messages are passed serially, this involves a bit-serial decrement-by-one, which is a very efficient operation performed with single clock cycle overhead. Using this mechanism, a message is found to have reached its destination when the address field becomes zero, in which case it is then deleted from onward transmission. With 32 logical addresses encoded in five bits and a single MSB to indicate validity, this mechanism requires a seven cycle latency per node. Broadcast messages are received by all nodes, and the address field now becomes a time-to-live, decremented by each handler until it expires and is deleted. This prevents broadcast messages from being circulated continually, and requires no list of broadcast messages to be maintained.

With the seven cycle delay at each node handler, the latency of TGB messages completing a long loop can be relatively large, but the message throughput is extremely fast, matching the requirements for processing blocks of image data streamed from an image capture device.

One special part of the system is the provision of processing modules (PM) for on-the-fly data manipulation. Any manipulation is specified in the message header, and reflected in the conveyed message content used to configure the PM. Such message processing is performed entirely within the FPGA, and includes decimating the spatial resolution and radiometric calibration. These tasks are easily performed in real-time during data transmission and bring a great reduction in computational workload for the PNs. In total, each PM can be configured with one of eight processing tasks, setup to apply either to every data message which passes, or only selected messages (by source and/or destination address). After processing, the PM actively modifies the message header to reflect the performed processing.

TGB messages received erroneously are simply deleted, as are messages experiencing bit errors in their address fields. To the higher layer protocols, the TGB is a best-effort communications medium. The system has been tested to work well for a typical packet error rate of 10^{-9}.

7. ON-BOARD PROCESSING

As described in Section 2, downlink of data during a mission is constrained, firstly by the bandwidth of the downlink, and secondly to times when the satellite is within range of a ground station. This scenario is common to many satellites.

Given a downlink bandwidth B in bits per second, an orbital repeat cycle of R seconds, and a time within range of a ground station of T seconds, set by link budget equations, then it is possible to downlink

$$D_O=BT \tag{1}$$

bits per orbit. On the other hand, if on-board sensors can capture data at a rate of C bits per second, then the proportion of captured data which can be downlinked is

$$P_{CD}=D_O/CR=BT/CR. \tag{2}$$

For X-Sat with a 50 Mb/s downlink and an average time-within-range of the ground station of approximately nine minutes per orbit, Do=26.37 Gb/orbit. By comparison, even the relatively low resolution X-Sat 3-band multi-spectral camera payload, with a 5000 pixel swath width, sampling at 8-bit resolution per band every 1.4 ms generates almost half a terabit per 98 minute orbit. Ignoring other factors, the proportion of data which can be downlinked is P_{CD}=1.7e-4, meaning that a potential 5800 images are lost for every one received – an alarming factor for a system that costs several million dollars to construct and launch.

Of course, the reality is that optical imaging rarely occurs in darkness and more importantly, the power-hungry camera will be limited by a typically tight satellite power budget. However, there is still a compelling case for placing computational hardware in orbit where it can access this data, process and interpret it, selecting which items are to be downloaded.

7.1 Increase of Data Value

A cloud detection algorithm rejects non-useful images (or regions), thereby increasing the average value of images that are downlinked. In addition, within a typical mission profile for imaging over a particular region, the required SSR size in bytes needed to ensure a given number of useful images is reduced. If 90% of the images captured over Singapore are obscured by clouds (any resident would attest that this figure is reasonable), and assuming that a cloud detection algorithm is only 50% effective, i.e. 0% false positive and 50% false negative detection of clouds, then the required SSR size needed per useful image is reduced by a factor of 1.8. By reducing the SSR size from 4 GB to 2.2 GB, volume, mass and power consumption are also reduced. The cost of doing this is evidently the additional requirements in terms of volume, mass and power consumption of a device needed to execute the algorithms (i.e. a PPU). In order to analyse this trade-off further, experimental measurements were made using a standard space-qualified OBC employing the SPARCv7-architecture ERC32 processor, the operational PPU engineering model, and a prototype SSR design comprising COTS SDRAM controlled by FPGAs. The results are summarised in Table I. Power measurements include required associated hardware at their rated operating points.

Table I: Comparison of power and volume for computational hardware alternatives.

		Size [GB]	MIPS	Power [W]		Volume [cm^3]	
				Individual	Σ	Individual	Σ
Standard architecture	SSR	4.0	NA	22	26	5120	5844
	OBC	NA[1]	20	4		724	
Proposed architecture	SSR	2.2	NA	12	18	2844	4636
	PPU	NA[1]	4000	6		1792	

[1] Not applicable, i.e. available memory is reserved for local processing

In Table I, the standard architecture comprises a fairly simple OBC with a larger SSR. As can be seen the total power consumption for operating this system is around 26 W. The PPU architecture makes use of a far more intelligent computer, consuming more power overall, but requiring a smaller SSR size by using compression and selection. It can be seen that increasing the processing ability results in an overall power reduction. It should be noted that in reality the OBC and SSR would operate continuously, whereas the PPU would not need to operate at full power continuously. In particular, it only needs to operate at full speed immediately after an image is captured in order to process it, and thereafter at far lower power required for mission control operations.

The SSR size is reduced by using the PPU to reject unimportant images. The same argument does not apply to the OBC since its computational resources are insufficient for image processing. For the X-Sat mission, the SSR's storage capacity of 2 GB has been chosen with such calculations in mind, however it should be noted that the X-Sat also retains an OBC.

7.2 Autonomous Analysis and Data Validation

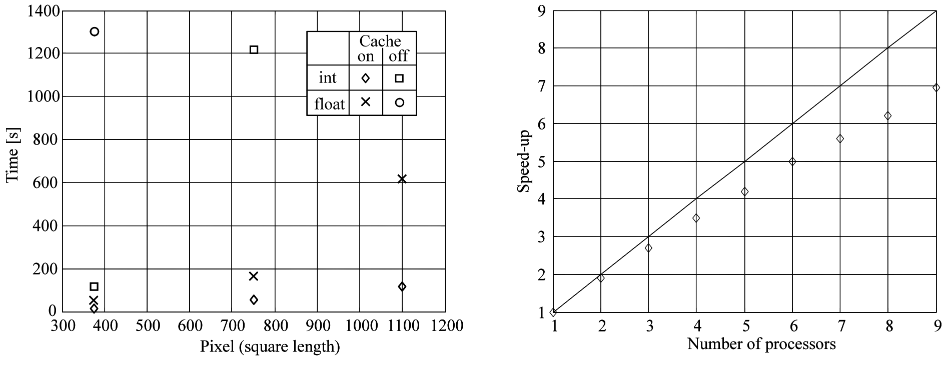
Two major application areas were identified for the X-Sat in terms of on-board image processing, namely content-based data compression and increasing the value of mission data through selection. One of the required processing steps for both application areas is unsupervised segmentation and classification [Liu and Bretschneider 2003]. In the following, several different implementations of this, and hardware settings were investigated in order to characterise the PPU's performance. Results are shown in Figure 3(a) using differently sized test images of 375 x 375, 750 x 750 and 1100 x1100 pixels. All test images were obtained from the same original satellite scene and re-sampled to the required size to allow the comparison of the results. Note that no measurements are provided in Figure 3(a) for configurations that exceeded the tolerable latency.

Since the SA1110 has no integrated floating-point unit, all floating-point operations have to be emulated in software, which has a significant impact on the overall performance. However, a version of the algorithm was coded with integer-only operations for comparison. Note that this had no impact on the result accuracy for this application.

Results were obtained with SA1110 instruction cache off and on, respectively. The utilisation of the cache increases the risk that radiation-induced bit errors cause errors in program instruction words, resulting in complete program crashes, or worse, in untraceable soft errors. Many satellite computers are set to operate without cache, however at the expected error rate of X-Sat it is preferred to operate with cache: it is more efficient to simply repeat the occasional erroneous calculation than it is to slow down all calculations.

Finally, Figure 3(b) depicts the results for the parallel integer-based implementation and shows an almost convergent-free characteristic between the number of utilised PNs and the obtained speed-up for up to eight nodes. In each case, only a very small amount of inter-PN communication was required.

Recalling the main objective of the PPU, i.e. the reduction of data for the transmission by compression or information extraction, the approach taken has proven to achieve this. In particular for integer-based computation a high data throughput is enabled that fulfils the purpose of easing the downlink bottleneck, and meets the mission objective of same-orbit processing.



(a)

(b)

Figure 3: Performance of the unsupervised segmentation and classification algorithm on the PPU: (a) processing time in seconds for different images with respect to the individual data types and chosen processor configuration, (b) speed-up with respect to the number of utilised PNs

7.3 Reliability

The second main PPU requirement was to achieve reliability through redundancy. Each COTS PN is susceptible to the damaging effects of radiation in orbit, and has a not negligible probability of ceasing to function during the satellite's lifetime. However, the probability of all PNs doing so is much lower. In fact, the calculated reliability of the overall PPU due to radiation effects can be shown to match that of an all radiation-hardened solution.

Aggregate reliability information may be difficult to determine accurately for a complete system, but for comparative purposes between COTS and non-COTS components, the reliability difference rather than absolute reliability can be calculated in terms of the radiation tolerance of the devices – ignoring mechanical and thermal issues for the present. In particular, latch-up performance and total dose figures are used [Lammers 1998]. The total dose (TD) is a cumulative measure in rads that relates to the ageing of the silicon substrate in integrated circuits. The semiconductor becomes "softer" with increased exposure and eventually leakage currents become so large that gate switching is compromised [Wertz and Larson 1999]. Latch-up (LU) performance measured in MeV-cm^2/mg defines the maximum instantaneous amount of energy of incident ionising radiation that can be withstood before latch-up results.

For the PPU design, reliability is computed in terms of the major active semiconductor devices that it comprises and is compared to a straightforward FPGA-only solution similar to that proposed in [Dawood et al. 2002], differing only in that we here assume use of a radiation-tolerant FPGA. For simplicity, components common to both designs (such as power supply components, connectors and passives) were not included in the analysis: in both the PPU and the FPGA-only solution these are each replicated per-block, and thus have an equivalent influence on the overall reliability. The design of the PPU is such that it should remain operational through swapping out PNs that exhibit faults, until only one PN and one associated FPGA survive. The published radiation test figures of the critical components are as shown in Table II.

Table II: Radiation performance in terms of total dose (TD) and radiation flux before latch-up (LU) of the reliability-critical components used in the PPU.

Device	Part number	TD [krads]	LU [MeV-cm^2/mg]	Reference
FPGA	RTAX1000S300	300	104	[Actel 2007]
SDRAM	K4S561632D-TC75	27	82	[ESA 2003]
CPU	Intel FADES1110	20	80	[O'Bryan et al. 1999]

It should be noted that the figures provided in Table II do not necessarily imply performance guarantees, and in all cases were obtained from rather small sample sizes. In the analysis that follows, we will therefore impose an arbitrary safety de-rating d_r on the manufacturers' figures to reflect the lack of confidence in the generality of the published figures.

In order to calculate an overall reliability figure it is first necessary to determine the individual probabilities of component survival due to radiation effects over mission lifetime. In particular, the upper bounds of 10 krads total dose and 30 MeV-cm^2/mg of radiation flux are used as guidelines hereafter. These figures are inputs to the design process, and therefore not derived, but such figures for a range of orbital altitudes are available from a number of sources, not least from Chapter 8 of [Wertz and Larson 1999]. In the present analysis, we make the simplifying, and safe, assumption that exceeding the latch-up and total dose thresholds will result in definite catastrophic failure of the affected parts.

Moreover, it is necessary to identify conditional structural probabilities relating to reliability that are imposed through design, for example, a working CPU requires two healthy SDRAM devices, and therefore the probability of a working PN is determined by the product of the individual probabilities of the three working component devices at end of mission (EOM).

The mission environmental figures cannot be considered as exact values that will be experienced, but as the central mean of a probability density function. In fact, space radiation distribution tends to follow a log-normal distribution [Tylka et al. 1997] with standard deviation between 1.2 and 1.5.

For a log-normal distribution, the probability density function is given by

$$f(x) = \frac{1}{x^{\sigma}\sqrt{2\pi}} \exp\left(-\frac{[\ln(s) - \mu]^2}{2\sigma^2} \right) \tag{3}$$

where μ and σ are the mean and standard deviation of the logarithm of the variable respectively. We apply this to model the actual radiation experienced in orbit, set so that the cumulative distribution function of the model is such at there is a d probability that rated conditions will occur, and with mean equal to $\exp\{\mu+\sigma^2/2\}$ where we choose a standard deviation of 1.2 in this case. Using the cumulative distribution function of the log-normal distribution,

$$CDF = \frac{1}{2} + \frac{1}{2} erf\left[\frac{\ln(s) - \mu}{\sigma\sqrt{2}}\right] \tag{4}$$

we can now assess the probability that rated survivability conditions will be exceeded for each of the three main active devices. This allows the calculation of the probability of survival due to TD or LU of each component independently.

For example, the probability of survival of the CPU due to TD, given by the probability that the experienced TD will exceed the rated survival dose of the CPU (20 krads in this case) is calculated as follows:

$$P(TD < 20) = \frac{1}{2} + \frac{1}{2} erf\left[\frac{\ln(20) - \ln(3.97)}{\sqrt{1.2}\sqrt{2}}\right] = 0.93 \tag{5}$$

The other figures are calculated similarly, and listed in Table III. The table also calculates the conditional probability that either of the two failure conditions occurs, assuming they are independent variables either of which alone may lead to device failure, derated as explained above so that the probability of device survival is *P(TD<device rating)* x *P(PU<device rating)* x d_r,

Table III: Reliability against total dose and radiation flux
before latch-up for the main reliability-critical components used in the PPU.

Device	P(TD<device rating)	P(LU<device rating)	Conditional probability
FPGA	1.000	0.976	0.781
SDRAM	0.960	0.961	0.738
CPU	0.930	0.959	0.713

Next, we need to include the design-imposed conditional probabilities of overall failure implied by the system structure. We will use the notation Ps_{DEVICE} to denote the survival probability of a named device.

Each PN will survive only if both the CPU and both SDRAM device survive, thus:

$$Ps_{eachPN} = Ps_{CPU} \text{ x } Ps_{SDRAM} \text{ x } Ps_{SDRAM} = 0.388 \tag{6}$$

Next, we calculate the ability of each half of the PPU to survive independently, because each half of the PPU is tied to a separate FPGA. This is determined by having at least one of the ten PNs alive at EOM:

$$Ps_{any\text{-}PN} = 1.0 - (1.0 - 0.388)^{10} = 0.993 \tag{7}$$

However, a working PN must be supported by the FPGA that it is connected to being operational. The probability of this at EOM is:

$$Ps_{anyPN\text{-}anyFPGA} = Ps_{FPGA} \text{ x } Ps_{any\text{-}PN} = 0.776 \tag{8}$$

However, there are two halves, either of which may remain operational. Thus, the probability of having a relatively slow, crippled, but operational PPU at EOM is:

$$Ps_{PPU} = 1.0 - (1.0 - 0.776)^2 = 0.950 \tag{9}$$

Based on radiation data alone, it is clear that the survival probability of the PPU after its 3-year mission lifetime is 0.950, albeit in a much degraded form. Generalising the above computation, Figure 4 plots the probability of PPU survival based upon the number of PNs that must be operational in either half of the PPU at EOM, disallowing the inconvenient situation of sufficient PNs remaining, but being spread over two halves of the PPU.

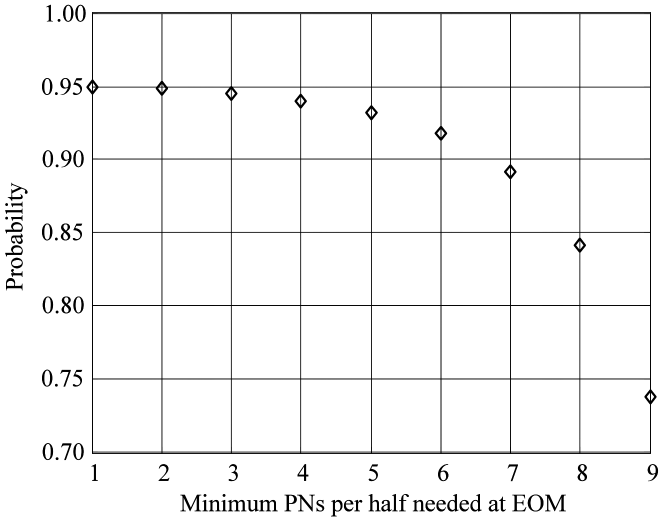


Figure 4: Probability of maintaining the given minimum processing capability in one half of the PPU until end of mission.

Contrast these PPU reliability figures with an alternative solution: a dual-redundant OBC constructed using two radiation-tolerant FPGAs similar to the system in [Dawood et al. 2002]. In this case no external CPUs or memory are used:

$$\mathrm{Ps}_{\mathrm{1of2\text{-}FPGAs}} = 1.0 - (1.0 - \mathrm{Ps}_{\mathrm{FPGA}})^2 = 0.952. \qquad (10)$$

Although both Equations (9) and (10) ignore several factors, they show that aggregate PPU reliability is similar to the reliability of an FPGA-based computer in terms of radiation survivability.

Above all, the calculations illustrate the concept of achieving reliability through redundancy.

Truly hardened solutions for space, especially deep space, would utilise higher voltage power supplies, hardened silicon (perhaps on a sapphire substrate), shielding, triple or greater internal redundancy, larger silicon feature sizes and different physical packages – especially ones proven to withstand extreme launch vibrations better than a CBGA. Empirical evidence also suggests that clock speed de-rating can improve the survivability of semiconductor devices in orbit. Typically, a 50% frequency de-rating is used.

7.4 Operating Modes

The survivability analysis of Section 7.3 leaves unanswered the major issue of reliability against temporary disruption such as SEU. The PPU approach of turning off, or power cycling faulty processors implies that any fault disrupts normal operation even though the hardware itself might survive.

Although the PPU does resemble a ground-based Beowulf system as mentioned previously, and even employs the same OS (Linux), the prevalence of runtime errors in orbit is a major difference in this comparison.

In fact several approaches are possible to tackle this issue. The simplest is probably to ignore run-time errors, and this may well be the method of choice when high-speed data is being streamed through the system with little overhead for error mitigation. Otherwise when data is important, the flexibility of the system allows several PNs to be set aside as an *n*-way majority voter, illustrated in Figure 5.

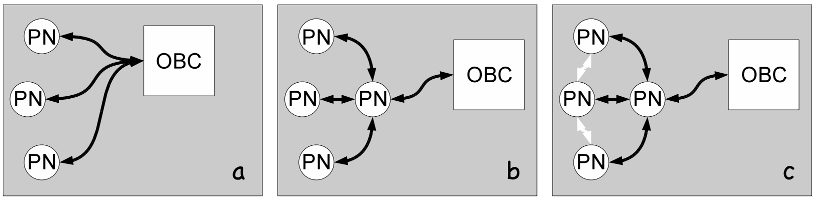


Figure 5: Alternative OBC-PPU PN interaction strategies showing (a) a direct three-way majority voter , (b) a PN-arbitrated three-way majority voter and (c) a PPU arbitrated cluster computation.

In fact, if 20 processors are operating at any one time, and total processing requirement falls short of aggregate capacity, it is a simple matter to re-deploy some of the spare capacity towards redundant calculations. The actual configuration and degree of majority could even be adjusted on an intelligent basis, under either ground control or through local autonomy based on current operating conditions, although this has not been implemented in the system to date. Several well-known software techniques sacrifice processing power for increased reliability [McLoughlin 2001], with the *n*-way majority voter being simplest. In the scenario shown in Figure 5(a), an important calculation required by the OBC is given to three separate PNs to perform. Each then passes its result back to the OBC. If the results differ, this is due to error, and thus the majority answer is accepted as the correct one. Although a three-way majority voter is pictured, five and even seven-way systems could equally have been used. There is also no need for three PNs to operate in parallel – if latency is not an issue, a single PN could repeat the same calculation *n* times. One issue with this process is the impact on the OBC, which needs to reissue the calculation request, and track its progress. This motivates the structure of Figure 5(b) in which the OBC passes off not only the calculation but its arbitration and majority voting to a master PN. Although this is a sensible solution, it does run the risk of an SEU error in the master PN itself from invalidating a correct result. Thus, it is advisable only in cases where the proportion of time consumed by the calculation itself is significantly greater than the time within which the result is being handled by the master PN.

Finally, the structure of Figure 5(c) allows the OBC to hand off a calculation to a master PN which then instantiates other PNs similarly to Figure 5(b), but with the difference that those PNs are aware of each other. In fact this is a full multiprocessing solution arbitrated by a single master PN.

So while Beowulf-style algorithms can be ported as-is to the PPU, it is preferred to explicitly analyse the interconnection of those algorithms, and determine the action to be taken on error, i.e. the application of majority voting strategies to critical calculations, or parts of calculations. This is a manual process, however it does not need to impact the algorithm implementations per-se when there is a stateless handing off of calculations over MPI or similar, just the tracking and handling of the underlying transport for the message passing interface.

8. CONCLUSIONS

COTS components have long been considered as candidates for cost-sensitive space applications, but have made little progress in mission critical areas such as on-board control and mission computation. This paper discusses a COTS-based control computer – the Parallel Processing Unit (PPU) – that not only demonstrates the usual benefits of smaller size, lower current and greater processing capability, but also does so in such a way that maintains system reliability. The technique of reliability through redundancy is employed to improve survivability prospects in space, and this FPGA-interconnected architecture naturally lends itself to a parallel computational structure. Since many of the ground-based satellite image processing applications currently execute on Beowulf-style or similar parallel clusters, the act of porting these to orbit is eased by the similar nature (and OS) of the PPU.

Placing an over-specified and flexible computer inside a micro-satellite will also inspire a more open-ended use of software processing in orbit. Excellent reasons were given, in terms of bandwidth mismatch between sensor data availability and downlink capability, which would encourage greater use of software applications in space.

In this paper, the system was introduced, described and analysed in terms of performance, cost, and reliability. The benefits of the PPU are clear, both for enhancing the value of current low complexity missions, but also in terms of utilising inbuilt processing capabilities to increase the flexibility and inherent intelligence of future missions. Whilst such a system is not currently well suited to the rigours of deep space radiation exposure, it is to be expected that adoption of parallel redundant systems for low-earth orbit space missions will become more popular, and gradually such techniques are likely to spread outward.

REFERENCES

ABBOTT, L.W., COX, G., AND NGUYEN, H.A. 2001. Cost effective critical space systems design approach. *IEEE Aerospace and Electronic Systems Magazine 16(4)*, 7-21.

ACTEL CORPORATION 2007. RTAX-S/SL rad-tolerant FPGAs. Online document [www.actel.com/documents/RTAXS_DS.pdf].

ALOISIO, G., AND CAFARO, M. 2003. A dynamic Earth observation system. *Parallel Computing 29(10)*, 1357-1362.

BAGHAIE, M.A., KUO, S.H., AND MCLOUGHLIN, I. 2004. FPGA implementation of space-time block coding system. In *Proceedings of the IEEE Circuits and Systems Conference on Frontiers of Mobile and Wireless Communications 2*, 591-594.

BAUMANN, R. 2005. Soft errors in advanced computer systems. *IEEE Design & Test of Computers 22(3)*, 258-266.

BLACK, R., AND FLETCHER, M. 2005. Next generation space avionics: layered system implementation. *IEEE Aerospace and Electronic Systems Magazine 20(12)*, 9-14.

BRETSCHNEIDER, T., TAN, S.H., GOH, C.H., ARICHANDRAN, K., KOH, W.E., AND GILL, E., 2005. X-Sat mission progress. *Small Satellites for Earth Observation, Special issue from the International Academy of Astronautics*, 145-152.

BRETSCHNEIDER, T. 2008. Towards commercial-off-the-shelf components in air- and spacecraft: From push to pull. In *Proceedings of the New Challenges in Aerospace Technology and Maintenance Conference*, CD-ROM.

BREIT, H., FRITZ, T., SCHATTLER, B., BORNER, E., LACHAISE, M., NIEDERMEIER, A., EINEDER, M., AND BALSS, U. 2007. TerraSAR-X payload data processing - First experiences. In *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium*. 3936-3936.

BRIEß, K., BARWALD, W., GILL, E., KAYAL, H., MONTENBRUCK, O., MONTENEGRO, S., HALLE, W. SKRBEK, W., STUDEMUND, H., TERZIBASCHIAN, T., AND VENUS, H. 2005. Technology demonstration by the BIRD-mission. *Acta Astronautica 56(1-2)*, 57-63.

BURCIN, A. 2002. RAD750. In *Proceedings of the Microelectronics Reliability and Qualification Workshop*, [www.aero.org/conferences/mrqw/2002-papers/A_Burcin.pdf].

CARDARILLI, G.C., LEANDRI, A., MARINUCCI, P., OTTAVI, M., PONTARELLI, S., RE, M., AND SALSANO, A. 2003. Design of a fault tolerant solid state mass memory. *IEEE Transactions on Reliability 52(4)*, 476-491.

CARDARILLI, G.C., OTTAVI, M., PONTARELLI, S., RE, M., AND SALSANO, A. 2005. Fault tolerant solid state mass memory for space applications. *IEEE Transactions on Aerospace and Electronic Systems 41(4)*, 1353-1372.

CHAU, S.N., ALKALAI, L., TAI, A.T., AND BURT, J.B. 1999. Design of a fault-tolerant COTS-based bus architecture. *IEEE Transactions on Reliability 48(4)*, 351-359.

DAWOOD, A.S., VISSER, S.J., AND WILLIAMS, J.A. 2002. Reconfigurable FPGAs for real time image processing in space. In *Proceedings of the IEEE International Conference on Digital Signal Processing 2*, 845-848.

DIGREGORIO, B.E. 2003. Mars: Dead or alive? *IEEE Spectrum 40(5)*, 36-41.

ELIAS, M. 2000. Development of a cow cost, fault tolerant, and highly reliable command and data handling computer. In *Proceedings of the IEEE Aerospace Conference 5*, 251-261.

ESA 2003. European Space Agency – Tables of radiation tested components. Online document [http://www.esa.int].

GOLLER, A., AND LEBERL, F. 2000. Radar image processing with clusters of computers. In *Proceedings of the IEEE Aerospace Conference 3*, 281-285.

GRAF, J., ZUREK, R., JONES, R., EISEN, H., JOHNSTON, M.D., JAI, B., AND MATEER, B. 2002. An overview of the Mars Reconnaissance Orbiter mission. In *Proceedings of the IEEE Aerospace Conference 1*, 171-180.

HALLE, W., VENUS, H., AND SKRBEK, W. 2001. Thematic data processing on board the satellite BIRD. In *Proceedings of the SPIE 4540*, 412-419.

HATTON, L. 2004. Safer language subsets: an overview and a case history, MISRA C. *Information and Software Technology 46(7)*, 465-472.

HITACHI 2000. SuperH Family Brief – SuperH RISC Engine. Online document [www.icdevice.co.kr/korea/product_info/file/SH_FB.pdf].

ISMAILOGLU, N., BENDERLI, O., KORKMAZ, I., DURNA, M., KOLÇAK, T., AND TEKMEN, Y.Ç. 2002. A real time image processing subsystem: GEZGIN. In *Proceedings of the AIAA/USU Conference on Small Satellites*, CD-ROM.

KRAMER, H. 2002. *Observation of the Earth and its Environment – Survey of Missions and Sensors*, 4th Edition, Springer Verlag.

LAMMERS, D. 1998. Latest PowerPC 750 runs faster on less power. EE Times Online [http://www.eetimes.com/news/98/1019news/latest.html].

LIU, B., AND BRETSCHNEIDER, T. 2003. D-ISMC: A distributed unsupervised classification algorithm for optical satellite imagery. In *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium 6*, 3413-3415.

LOVELLETTE, M.N., WOOD, K.S., WOOD, D.L., BEALL, J.H., SHIRVANI, P.P., OH, N., AND MCCLUSKEY, E.J. 2002. Strategies for fault-tolerant, space-based computing: Lessons learned from the ARGOS testbed. In *Proceedings of the IEEE Aerospace Conference*, 2109-2119.

LOVELLETTE, M.N., CAMPBELL, A., CLARK, K., WOOD, K.S., WOOD, D.L., ROSS, A., BEALL, J.H., AND CLIFFORD, G. 2003. Implications of the different classes of exceptions experienced during the COTS processor test flight on the ARGOS satellite. In *Proceedings of the IEEE Aerospace Conference 5*, 2481-2491.

MCLOUGHLIN, I.V. 2001. Design, testing and verification of a micro-satellite on board data processing unit using commercial grade processors. In *Proceedings of the International Conference on Information, Communications and Signal Processing*, CD-ROM.

MCLOUGHLIN, I.V, GUPTA, V., SANDHU G.S., LIM S., BRETSCHNEIDER, T. 2003. Fault tolerance through redundant COTS components for satellite processing applications. In *Proceedings of the International Conference on Information, Communications and Signal Processing*, CD-ROM.

MCLOUGHLIN, I., BRETSCHNEIDER, T., AND RAMESH, B. 2005. First Beowulf cluster in space. *Linux Journal 137*, 34-38.

MILLER, J., FLATLEY, T., STAKEM, P., AND VECENTE, G. 2001. On-board cloud contamination detection with atmospheric correction. In *Proceedings of the Earth Science Technology Conference*, CD-ROM.

NEDEAU, J., KING, D., LANZA, D., HUNT, K., AND BYINGTON, L. 1998. 32-bit radiation-hardened computers for space. In *IEEE Proceedings of the Aerospace Conference 2*, 241-253.

O'BRYAN, M.V., LABEL, K.A., REED, R.A., HOWARD, J.W., BARTH, J.L., SEIDLECK, C.M., MARSHALL, P.W., MARSHALL, C.J., KIM, H.S., HAWKINS, D.K., CARTS, M.A., AND FORSLUND, K.E. 1999. Recent radiation damage and single event effect results for microelectronics. In *Proceedings of the Radiation Effects Data Workshop*, 1-14.

ORTEGA, G. 1999. Linux for the International Space Station program. *Linux Journal 59*.

PEARLMAN, J.S., BARRY, P.S., SEGAL, C.C., SHEPANSKI, J., BEISO, D., AND CARMAN, S.L. 2003. Hyperion, a space-based imaging spectrometer. *IEEE Transactions on Geoscience and Remote Sensing 41(6)*, 1160-1173.

PERSCHY, J.A. 2000. Space systems general-purpose processor. *IEEE Aerospace and Electronics Systems Magazine 15(11)*, 15-19.

PERSYN, S.C., MCCLELLAND, M., EPPERLY, M., AND WALLS, B. 2001. Evolution of digital signal processing based spacecraft computing solutions. In *Proceedings of the IEEE Digital Avionics Systems Conference 2*, 1-10.

SAMPSON, S., DUGGAN, P., BURNELL, R., MCENDREE, S., TAUSCH, J., SLEETER, D., ALEXANDER, D., KOGA, R., YU, P., AND CRAIN, S. 2002. Foreign comparative test of space qualified digital signal processors. In *Proceedings of the IEEE Aerospace Conference 5*, 2355-2364.

SHIRVANI, P.P., SAXENA, N., AND MCCLUSKEY, E.J. 2000. Software-implemented EDAC protection against SEUs. *IEEE Transactions on Reliability 49(3)*, 273-284.

RAMESH, B., BRETSCHNEIDER, T., AND MCLOUGHLIN, I. 2004. Embedded Linux platform for a fault tolerant space based parallel computer. In *Proceedings of the Real-Time Linux Workshop*, 39-46.

SWIFT, G.M., FANNANESH, F.F., GUERTIN, S.M., IROM, F., AND MILLWARD, D.G. 2001. Single-event upset in the PowerPC750 microprocessor. *IEEE Transactions on Nuclear Science 48(6)*, 1822-1827.

TRENSCHEL, T., BRETSCHNEIDER, T., AND LEEDHAM, C.G. 2003. Using JPEG2000 on-board mini-satellites for image-driven compression. In *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium 3*, 2033-2035.

TYLKA, A.J., DIETRICH, W.F., AND BOBERG, P.R. 1997. Probability distributions of high-energy solar-heavy-ion fluxes from IMP-8: 1973-1996. *IEEE Transactions on Nuclear Science 44*, 2140-2149.

WANG, J.J. 2003. Radiation effects in FPGAs. In *Proceedings of the Workshop on the Electronics for LHC Experiments*, CD-ROM.

WERTZ, J., AND LARSON, W. 1999. *Space Mission Analysis and Design*, 3rd Edition, Space Technology Library, Kluwer Academic Press.