WARSAW UNIVERSITY
OF TECHNOLOGY

Faculty of Electronics and Information Technology

Ph.D. THESIS

Rafał Graczyk, M.Sc.

Dependable control and measurement systems modeling for unmanned spacecraft

Supervisor
Professor Krzysztof T. Poźniak, Ph.D., D.Sc.

Warsaw, 2016

Thesis title:

“Dependable control and measurement systems modeling for unmanned spacecraft”

Abstract

Reliability and processing performance modeling is an important issue for aerospace and space equipment designers. From system level perspective, one has to choose from multitude of possible architectures, redundancy levels, component combinations in a way to meet desired properties and dependability and finally fit within required cost and time budgets. Modeling of such systems is getting harder as its levels of complexity grow together with demand for more functional and flexible, yet more available systems that govern more and more crucial parts of our civilization's infrastructure (aerospace transport systems, telecommunications, exploration probes). In this thesis promising method of modeling complex systems using Petri networks is introduced in context of qualitative and quantitative dependability analysis. This method, although with some limitation and drawbacks offer still convenient visual formal method of describing system behavior on different levels (functional, timing, random events) and offers straight correspondence to underlying mathematical engine, suitable for simulations and engineering support.

Proposed modeling methodology being focal point of this thesis is then shown in use, for evaluation of several variants of mission – functionality – technology trade-off. This is common part of every space endeavor and every additional of supporting decision making process is invaluable.

Real life example of analysis performed using proposed methodology is shown on case of Coronagraph Control Box equipment designed and built for European Space Agency PROBA3 mission. PROBA3 and accompanying equipment is designed to be technology sandbox allowing some experimental solution to be flown in space and collect operational heritage.

Tytuł rozprawy:

„Modelowanie wiarygodnych systemów kontrolno-pomiarowych dla bezzałogowych pojazdów kosmicznych”

Streszczenie

Modelowanie niezawodności i zdolności do dostarczenia wymaganej mocy obliczeniowej, jest ważnym zagadnieniem dla projektantów aparatury lotniczej i kosmicznej. Z punktu widzenia poziomu systemu, należy jedną z wielu możliwych kombinacji architektury, poziomów redundancji, zestawu komponentów, w taki sposób, aby spełnić wymagania odnośnie funkcjonalności, niezawodność i ostatecznie zmieścić się w planowanym budżecie kosztów i czasu. Modelowanie takich systemów jest coraz trudniejsze, wraz z tym jak ich poziom skomplikowania rośnie wraz z popytem na bardziej funkcjonalne i elastyczne, jeszcze bardziej dostępne systemy, które rządzą coraz bardziej istotnymi elementami infrastruktury technicznej naszej cywilizacji. W poniższej rozprawie przedstawiona jest obiecującą metodą modelowania złożonych systemów, w szczególności opartych o konfigurowalne układy logiczne, z wykorzystaniem sieci Petriego, do stosowania w kontekście analizy i oceny jakościowej i ilościowej ich wiarygodności. Metoda ta, choć z pewnymi ograniczeniami i niedogodnościami oferuje wygodne, wizualne i formalne sposoby opisywania zachowania systemu na różnych poziomach (funkcjonalnym, czasowym, zdarzeń losowych) i oferuje wygodne wsparcie aparatu matematycznego, idealnego do prowadzenia symulacji i wsparcia technicznego procesu podejmowania decyzji.

Proponowana metodyka modelowania będąca centralnym punktem prezentowanej rozprawy jest pokazane w użyciu, do oceny kilku wariantów kompromisów w układzie orbita - funkcjonalność – wykorzystana technologia. Jest to stały element projektowania każdej misji kosmicznej i każda dodatkowa sformalizowana i rygorystyczna ocena pozwala lepiej przeprowadzić porównanie rozważanych wariantów.

Praktyczny przypadek użycia proponowanej metodologii jest zaprezentowany na przykładzie analizy rozwiązań dla budowy sterownika koronografu (Coronagraph Control Box) projektowanego i budowanego dla misji PROBA3 prowadzonej przez Europejską Agencję Kosmiczną. PROBA3 jest misją badawczą która pozwala wprowadzić w życie nowe rozwiązania eksperymentalne do lotu w przestrzeni kosmicznej i tym samym zbierać doświadczenia dziedzictwo działalności operacyjnej wybranych i zaimplementowanych rozwiązań.

Table of Contents

1 Preface

1.1 Context of the dissertation

Building complex system, the one that involves multidisciplinary background, always poses a technological, organizational and methodical challenge. Electronic equipment for aerial or space applications, among military, medical or digital entertainment devices are good examples of such complex systems, in great extent, shaped by their operational environment and user requirements.

Space equipment of this kind, avionics, has to face one of the most demanding operational conditions and environmental stresses, multitude of known and unknown unknowns in design process, prohibitively high cost of components and qualified processes. All in all, bearing in mind all the difficulties in front of engineering teams the earlier the prototyping process starts the better for the teams, the stakeholders, the customers and mission and equipment itself.

PROBA-x is the series of European Space Agency technological satellite missions. Each of these missions explores new possibilities to build advanced equipment and to gain necessary space heritage for technologies used on-board. PROBA-3 is the mission currently in development, focusing on sophisticated space metrology for multi-satellite formation flying concept exploitation. In case of PROBA-3 goal of precise formation flying (in formation of only two satellites) has a side effect utilized by scientific community in form of payload. PROBA-3 payload is a Coronagraph Instrument, ASPIICS, intended to photograph Sun's corona in order to understand it's dynamics at vast range of distances from Sun's surface. Centrum Badań Kosmicznych Polskiej Akademii Nauk (CBK PAN, Space Research Center of Polish Academy of Sciences) is responsible for PROBA-3 ASPIICS Coronagraph Control Box (CCB), in other words, main instrument controller providing all the power and data services to run the ASPIICS. CCB is being designed and (soon to be) built under close cooperation with ASPIICS prime integrator, Centre Spatial de Liège from Liège, Belgium and main customer European Space Agency.

CBK PAN is also a leader of Polish companies consortium (Astri Polska, Creotech Instruments, N7 Mobile), cooperating in joint effort of design, manufacturing, assembly, integration and testing. Each of consortium members has specific tasks assigned:

- CBK PAN
    - CCB and its subassemblies design technical and quality assurance management
    - CCB general systems engineering and modeling

    - CCB Digital Processing Unit design and tests
    - housing design and manufacturing
    - integration and tests
- Astri Polska
    - CCB Power Conversion Unit design
    - CCB Power Conversion Unit manufacturing and tests
- Creotech Instruments
    - CCB Ancillary Electronics Unit design
    - CCB Ancillary Electronics Unit manufacturing and tests
    - CCB Digital Processing Unit manufacturing
    - Electrical Ground Support Equipment design and manufacturing
- N7 Mobile
    - Boot Software design, coding and test
    - Application Software, design, coding and test

Author of this dissertation is Coronagraph Control Box Project Manager and System Engineer employed in CBK PAN.

1.2 Structure of the dissertation

The dissertation is structured in following way:

- chapter 1 provides the reader a context in which presented dissertation is written and guides through its contents
- chapter 2 provides background information, in particular:
    - avionic architectures, their evolution, trends and state-of-the-art solutions
    - operational environment of space equipment, especially scientific instrumentation
- chapter 3 provides background information on modeling techniques present in avionics design
- chapter 4 provides background on ASPIICS instrument on PROBA-3 technological mission
- chapter 5 reveals methodology for space avionics modeling, including thesis and goals of this dissertation , as well as instructs the reader on how proposed methodology is going to be implement in course of CCB DPU system analysis

- chapter 6 provides model construction, simulation and CCB DPU system analysis from processing performance perspective
- chapter 7 provides model construction, simulation and CCB DPU system analysis from reliability perspective
- chapter 8 summarizes obtained results, provides a critical discussion of present work and suggests interesting paths of further development of proposed modeling techniques
- chapter 9 provides bibliographic list of references to papers, books and document supporting presented dissertation
- appendices extending information present in main body of dissertation
    - ASPIICS instrument details
    - CCB / DPU components radiation susceptibility calibration
    - information on Petri Net tools used for simulations

2 Introduction

The goal of the introductory chapter is to outline and briefly discuss all the main topics, notions and concepts that are used later on in dissertation. The intention is not to elaborate the subjects, but rather to indicate their importance and relevance context.

Reader is introduced to:

- Systems, systems engineering and necessity to tackle the system complexity
- Avionics, especially in space context and related constraints and requirements
- Avionics operational environment and the consequences of electronics irradiation

2.1 Systems

Unfortunately, there is no, commonly accepted, as exhausting and fully correct, definition of what System is. Merriam-Webster dictionary [1], considered as a common source of definitions, states that system is a regularly interacting or interdependent group of items forming a unified whole.

A system could be defined a set of objects with relationships between the objects and between their attributes, as defined by [2]. Similarly, more thorough explanation is proposed by [3], where a complete system is any complex of equipment, human beings, and interrelating logic designed to perform a given task, regardless of how complex the task may be. Logically, very large or complicated systems are broken into subsystems, to be fitted together like blocks to form the entire or total system.

Interestingly, above definitions, state that system is a bounded entity, while, for example, [4] proposes unbounded definition: "…a collection of things working together to produce something greater…". A system has the further property that it is unbounded — each system is inherently a part of a still larger system:

1. A system is a complex set of dissimilar elements or parts so connected or related as to form an organic whole.
2. The whole is greater in some sense than the sum of the parts, that is the system has properties beyond those of the parts. Indeed, the purpose of building systems is to gain those properties

Nevertheless, all authors agree that key characteristic of a system, defining it uniquely, is its architecture. System architecture is set of all of the most important, pervasive, higher-level, strategic decisions, inventions, engineering trade-offs, assumptions, and their associated

rationales concerning how the system meets its allocated and derived product and process requirements [5].

Finally, for the sake of clarity, system definition used in this dissertation is following: system is any entity within prescribed boundaries that performs work on an input in order to generate an output, that consist of number of lower level entities interlinked and interdepended within the system. System can be a constituent of higher level entity (i.e.: other system).

Therefore, in the light of proposed definition, system exhibits following properties:

- it has an architecture (set of lower level entities linked together in specific way)
- it is bounded (by interface)
- it has a behavior (allocated functionality)

2.2 Systems engineering

Systems engineering as distinct discipline, thorough the years has developed well described and understood process, that is involved in design of every complex system. Process consist of three phases which covers whole product life cycle, starting from initial idea, finishing at unit obsolescence.

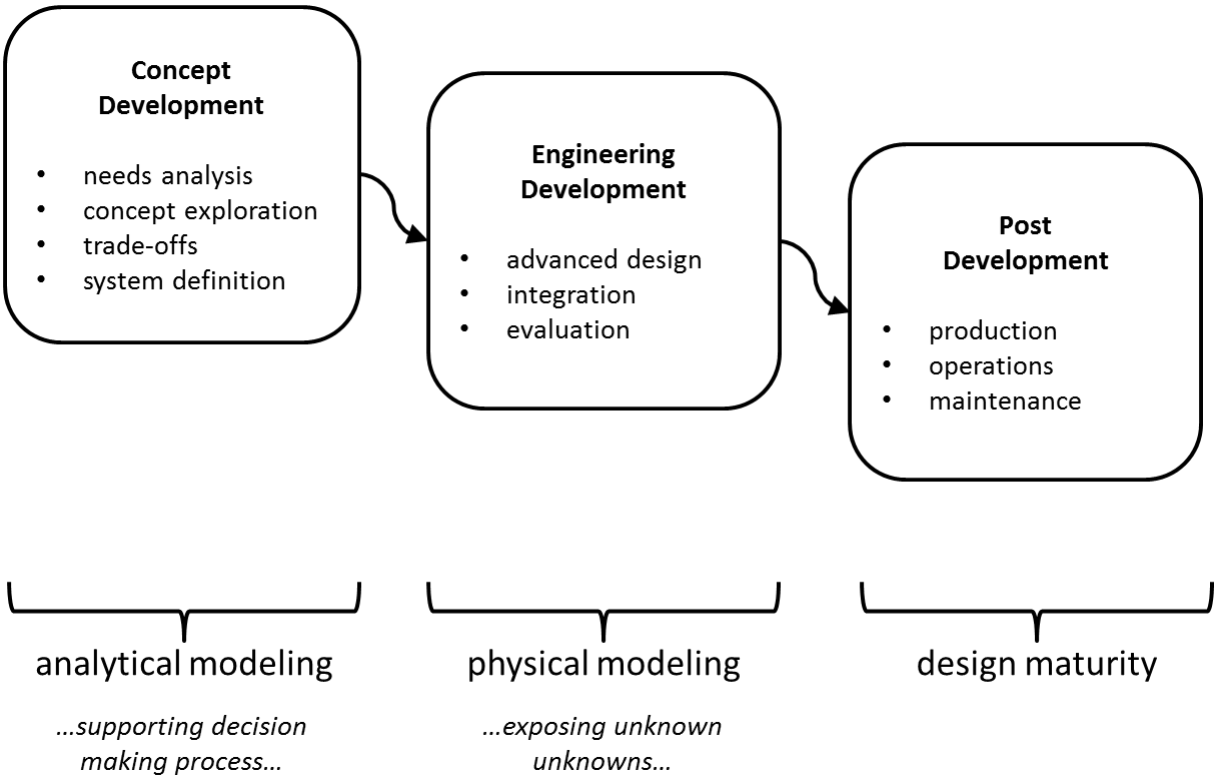

Figure 1 Systems engineering as a way to manage product lifecycle [6]

The flagship example of complex system is avionics.

The main three phases of system engineering are concept development, engineering development and post development. Concept development phase is devoted to needs analysis, concept exploration and concept definition. Engineering development phase is devoted to advanced design development, engineering solutions design and their integration and evaluation. Finally, post development phase is devoted to production, system operation, support, maintenance, system components obsolescence management, disposal and reprocessing if necessary. System concept development phase is crucial to provide high quality and high robustness answer to questions regarding economical and technical feasibility of system to be developed. In particular, analysis of what is the best way to satisfy requirements, has to be performed. As a part of analysis, consequences of potential choices, have to be carefully investigated and evaluated against each other. Ultimately, concept development phase verifies validity of system need and feasibility of its construction, explores system concepts space and selects most attractive and promising solutions, refines them in form of set of requirements and preliminary system definition [6].

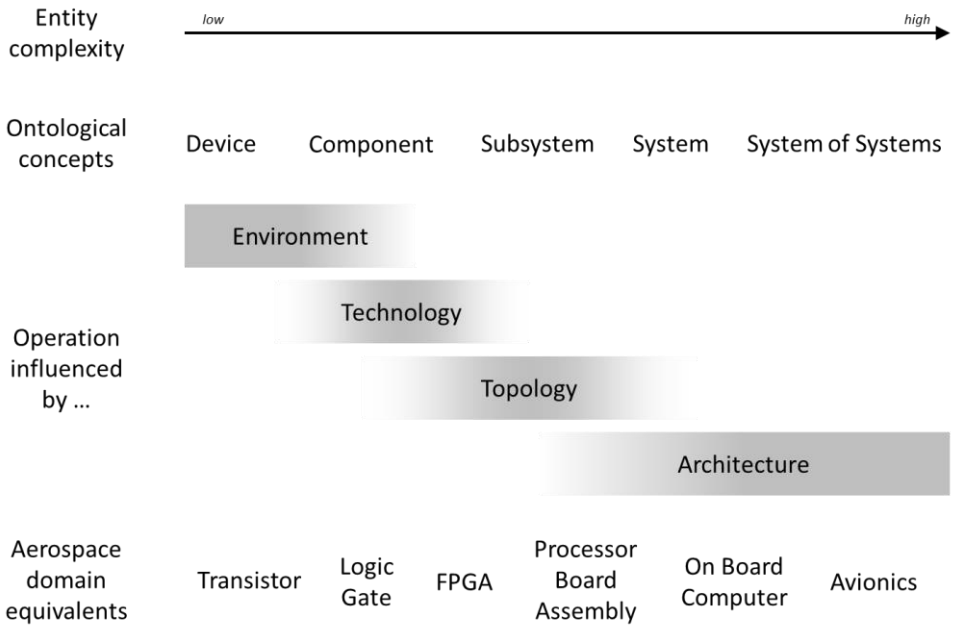

Figure 2 System complexity levels

It is worth to notice that systems engineering discipline operates at several levels of complexity simultaneously. In the same time, features and solutions are selected and tweaked on various scales. Consequences of each decision, even very low level, affects whole and must be understood globally. On

Figure 2 categories are arranged in rising complexity. For each category, a dominating source of influence is shown. Low complexity, basic entities are heavily affected by environment (radiation levels, temperature of operation) and technology (bi-polar or CMOS gates, Silicon-On-Insulator) in which they are manufactured, while higher complexity entities are under stronger influence of more abstracts concepts of topology (for integrated circuits it is a way of linking functional blocks on the same die, while for printed circuit board it is a way of linking components on the same copper - dielectric substrate). For readers convenience, also a typical examples of each entity category from aerospace domain are proposed.

System concept development phase is crucial for mission, project or product success. It is due to simple fact that all mistakes made in this phase, will propagate through system life cycle, being more expensive and harder to mitigate with every step taken deeper into the detailed system development. That is why system concept development phase, especially in the light of multi-level complexity it has deal with, shall be supported by additional tools, of which prototyping bears the most significant impact on improving the robustness (here understood as resilience to engineering mistakes and erroneous assumptions) of design.

Prototyping, most likely, deals with following three main problem areas (a. k. a. why to prototype?). First, it helps to refine initial requirements, that is, it allows system engineers to correctly understand what client wants by proposing functional (to a given extent) equivalent of final product. Second, it allows experimentation, that is, validating if selected strategy to overcome engineering obstacles is correct or is it a dead end. Third area where prototyping is necessary is process of closing technology gap between implementation requirements and implementation itself, in other words, it is feasibility check and possibility to optimize.

Prototyping expresses itself in two ways of tackling system complexity and engineering challenges. Namely, these are throw-away approach and incremental-evolutionary approach. Throw-away prototyping, simpler, one-shot method, is used to evaluate accuracy of system specification, at least in areas of increased risk or uncertainty, by using simplified approach or simulations. Incremental-evolutionary approach is more complex. Its basis is to maintain prototype through system development life-cycle and update it continuously and accordingly as system specification is refined and design is more detailed in course of product development. If this approach is deployed in form of (semi-)automated process then could be very beneficial for coping with uncertainties and performing conscious trade-offs and keeping reasonable design margins. When incremental-evolutionary prototyping approach is tightly coupled with system development then additional burden of updating the prototype has

to be taken into account. Keeping the prototype out of date may result in catastrophic consequences [7]. One of most appealing ways of prototyping is software based models creation and their simulation.

2.3 Avionics

Satellite and launcher systems are made of many elements which cover whole range of features: commanding, communication, data processing and transmission sensing, tracking, attitude and altitude (navigation) control, thermal control and contingency actions (faults detection and isolation). Avionics is the glue that connects all pieces together and acting force making it operate as a whole, executing desired onboard functions.

Term avionics dates back to 50s of XX century and has been created by merging word aviation and electronics. It describes equipment and software used to manage and control aerial vessels and spacecraft. Avionics consist of many functional blocks of various functionalities, properties and applications. Most common examples of avionic systems components are:

1. on board computers
2. on board data processing systems
3. data storage
4. remote terminal units
5. avionic buses
6. attitude determination and control systems

Term avionics, nowadays, cover hardware but also software executed on hardware avionic platform.

2.3.1 Avionic architectures

Avionic architecture is the general approach to designing and building aerospace embedded systems in terms of how hardware and software components are logical and functionally connected and managed together in order to fulfill mission objectives.

Review of technology development roadmaps prepared by ESA quickly concludes that there is a common industrial agreement on consolidating on two major classes of avionic architectures. First class is a standard platform creation that aims at achieving reduced non-recurrent engineering costs (in terms of, both, hardware and software procurement). In other words, increase in functionality and performance has to be evolutionary and contained within

standard line replace units (interchangeable between vendors) with standard physical and logical interfaces. This kind of avionics class covers earth observation and monitoring, telecommunications and other services performed by satellites built in industrial volumes [8].

Second class of avionics, emerging from European roadmaps, is for dedicated for experimental missions, either or space exploration, exploitation or technology development. Here, in turn, costs are traded off for extremely high performance and specific designs, dictated by science or technology goals, different on case by case basis [9].

First, "industrial" class of avionic architectures takes advantage of heavy industrialization of designs and "one size fits all" philosophy. Emanation of this approach is the famous SAVOIR (Space AVionics Open Interface aRchitecture) initiative [10].

SAVOIR is on-board data systems reference architecture (depicted on Figure 3) and outlines:

- definitions of function
- performance needs
- security needs
- fault detection, isolation and recovery

then maps and allocates abovementioned needs and capabilities to hardware and software units and outlines:

- connection and operation of avionics units
- connection and operation of payload units
- on-board time distribution and synchronization
- interfaces satellite and ground segments
- proposes coherent and unified systems testing and validation

While SAVOIR compliance is not (yet) obligatory for units providers, it ensures interoperability of software and hardware items of different origin. Such harmonization is very beneficial for cost and schedule management at customer or integrator tier, at the expense of innovation.

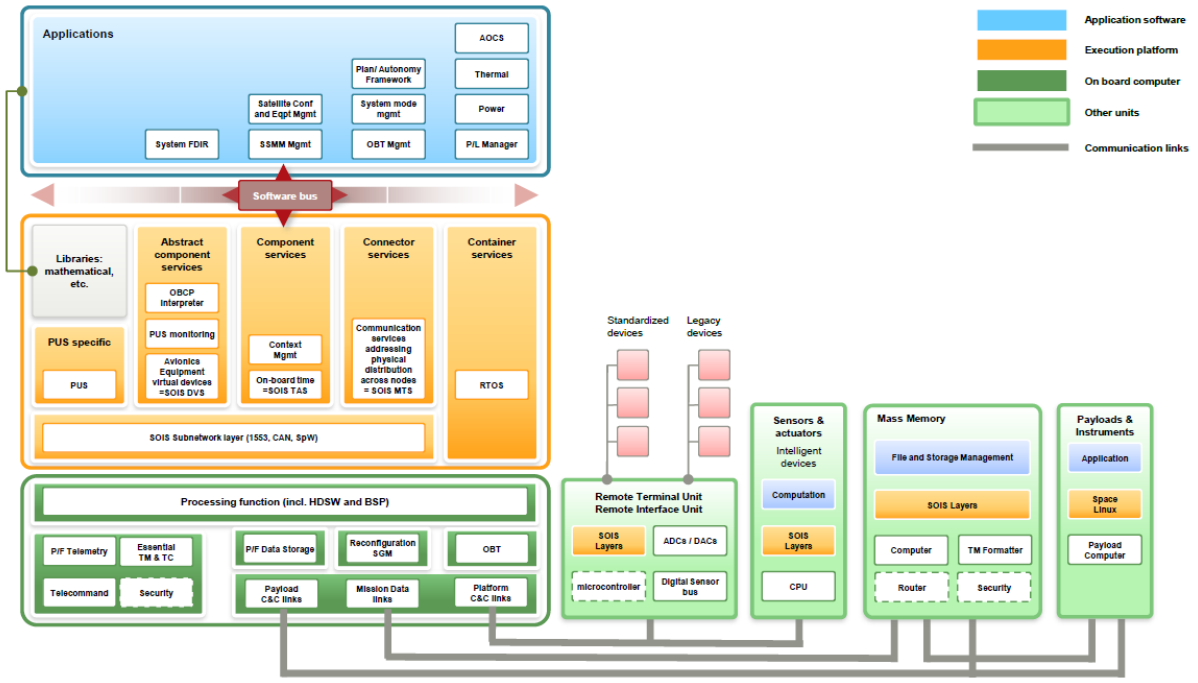

Figure 3 SAVOIR avionic architecture [10]

Seconds, "scientific" class of avionics architectures takes advantage of new (relatively, in space technology context) microelectronic components base and employs advanced on board autonomy, complex algorithms (formation flying, interferometry) and dynamic reconfiguration on satellite and subsystem levels to deliver outstanding processing performance and unmatched functionality sets (including capability to change the functionality upon entering new mission phase or encountering unexpected conditions).

Scientific class avionics architectures are a little bit harder to characterize as the difference between current state-of-the-art systems and future developments is much larger than in case of industrial class avionics. The latter exhibited evolutionary process of technology harmonization which ordered and structured existing building block into more generic and manageable specification, commonly accepted and, soon to be, implemented. In case of scientific class architecture the change is more disruptive as space technology community is experiencing shift from static, pipelined architectural model (Figure 4) to more flexible and scalable multi units aggregated furnished with high speed serial interconnect, reconfigurable systems (Figure 5).

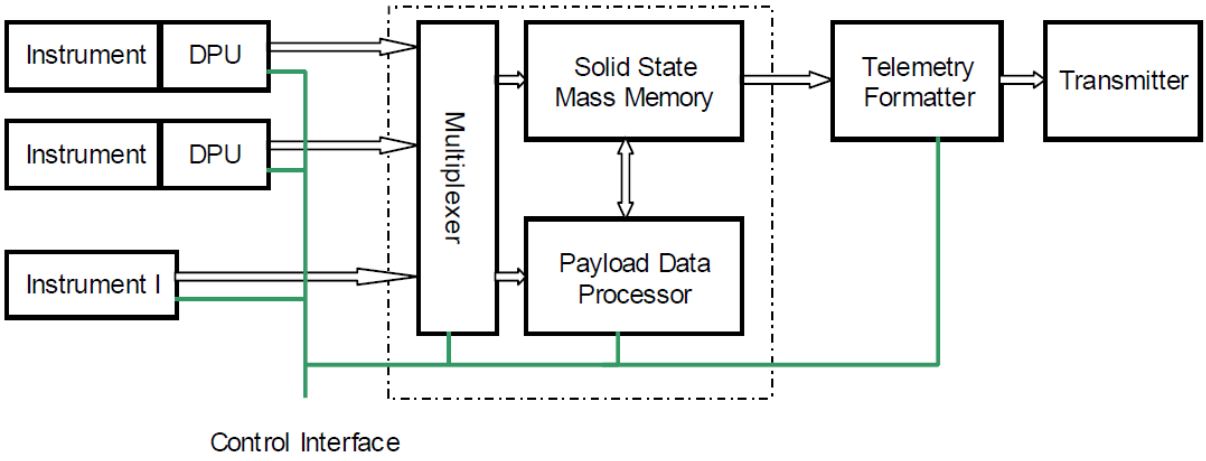

Figure 4 Multi-pipeline architecture scientific class avionics [11]

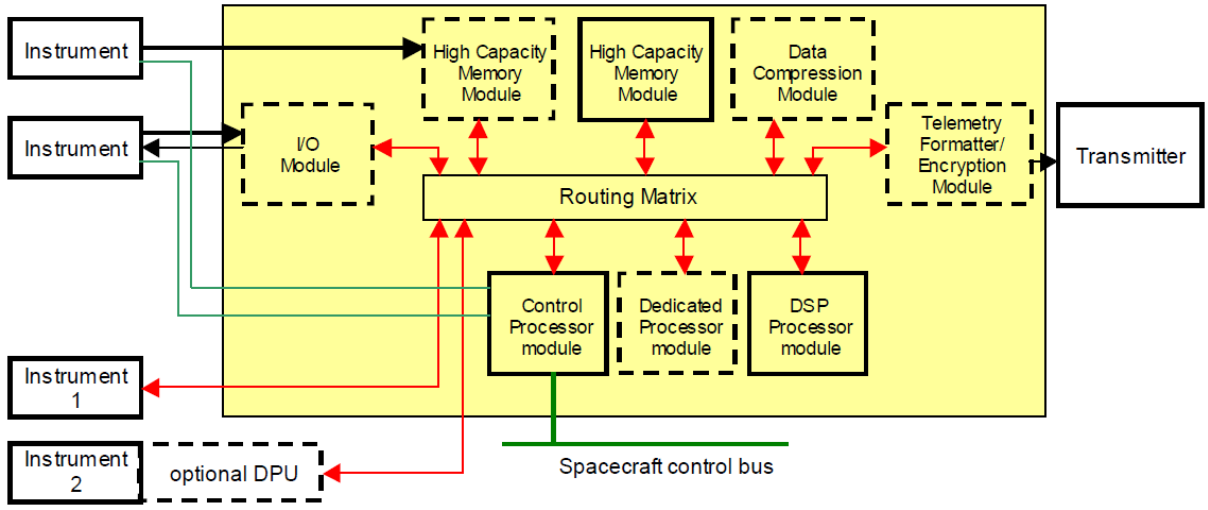

Figure 5 Reconfigurable architecture scientific class avionics [11]

It is worth mentioning that these two classes of avionics could be mixed in one satellite system, where industrial approach is used for satellite bus manufacturing while scientific class avionics is a foundation of payload. These two approaches could be mixed in various proportion, depending mission budget, schedule and, ultimately, objectives.

2.3.2 On-board computers

On-board computer, heart of "industrial" class avionics, also known as Command and Data Handling or On Board Data Handling is an embedded system responsible for satellite mission realization (execution) by collecting sensor data, by managing actuators and executing control application (on-board software).

On board computer controls avionic communication bus, which in turns connects remote terminal units , radio transponders and all other satellite systems which are equipped with

processing power that enables their network connectivity (i.e.: GPS signal receivers, cryptographic modules, telemetry units in various other subsystems). Key functional elements of modern On-Board Computers are listed in table below (Table 1):

Functional block	State-of-the-art implementation
CPU + peripherals	Europe: Mainly LEON family processor designs (UT699, UT700, GR712RC): SPARC V8 is available as an IP core and as a hardware device (Cobham, Atmel). Equipped with FPU, cache memory, a lot of peripheral controllers including CAN controllers and the Space Wire, as well as the capability to operate multi-core systems (AMP and SMP), delivers computing power of more than 100 MIPS.
	Other: USA: PowerPC 603 & 650, MIPS R3000, RAD750 Japan: Hitachi SuperH China: ARM processors in FPGAs
operations memory	Typically fast, volatile and large memory: SRAM or SDRAM. This memory is used to run the operating system and perform all necessary control algorithms. This memory also includes all the variables and configuration parameters that are used by OBC applications. Due to the crucial importance for the safety and continuity of satellite operations, the memory is protected against ionizing radiation induced errors by error correction codes and checksums (various EDAC mechanisms and protection techniques). The process of encoding and decoding memory is automatic and "transparent" for seamless, higher level, hardware-software integration. Rough, top features, per chip [12]: • SRAM: 32Gb, 12 ns access time, • SDRAM: 4 Gb, 133 MHz clock • SDRAM DDR2: 8 Gb, 333 MHz clock
"boot" and "local mass" memory	non-volatile memory, EEPROM, Flash This kind of memory includes a loader software for fetching operating system and control applications from boot memory itself or any other local storage. Additionally self-tests and minimal communication capability is also available with boot software for patching and contingency operations. Boot and local mass memory are now seldom implemented in separate subsystems but, for failure tolerance reasons, are often in different physical chips or modules on the same board. Rough, top features, per chip [12]: • EEPROM: 8 Mb, 150 ns access time • NOR Flash: 256 Mb, 90 ns access time • NAND Flash: 64 Gb, 25 ns access time

"safeguard" memory	fast, volatile RAM with battery backup Includes satellite configuration vector: information about the health of each of the modules, information about which modules are switched on in current operational configuration, information about the phase of the mission and the key variables in the control software.

Table 1 On Board Computer building blocks overview [8], [13]

Solutions presented in Table 1 presents current state-of-the-are implemented, therefore verified and validated options. There are many R&D activities ongoing that include AVR and ARM radiation tolerant versions of popular microcontroller families [14] or LEON4 processor in quad-core configuration [15]. Advances in volatile and non-volatile storage are progressing by wider acceptance of screening and repackaging of standard, commercial chips into hybrid, multi-chip modules qualified according to regular space-borne standards [12].

2.3.3 Payload computers

Payload computer, heart of "scientific" class avionic system is often called Instrument Control Unit (ICU) which fully describes it role in satellite system. Payload computers, ICUs, are there to control the measurement instrument and to perform kinds of processing of payload data (preprocessing, discrimination, aggregation and rearranging, compression, encapsulation). Very often, ICUs offer intermediate of final storage capabilities [11]. Interestingly, the division between industrial and scientific avionic classes is less sharp for microsatellites where stringent constraints on size, weight and power leads to situation where both avionic types are integrated into one hardware platform (i.e. integrated modular avionics, IMA) where distinct functionalities exist in form of separate chunks of software executed in isolated slots managed by hypervisor kernel [16]–[19]. Nevertheless, payload computers, are tailored to fulfill very specific mission needs which, in turn, can be satisfied by vast number of technologies or concepts. Table 2 summarizes current state-of-the-art implementation variants:

Processing block	State-of-the-art implementation
generic processor based (GPP)	CPU utilization e.g.: PowerPC 7448 (computing power to 3GMIPS) or 750FX. Good for the implementation of sequential algorithms with the support of operating systems. High flexibility but significant bottleneck at input / output interfaces. Similarly, LEON4 quad-core GR740 is very efficient for sequential algorithms mixed with regular housekeeping tasks – unfortunately there are no space-qualified operating systems (RTEMS, VxWORKS, QNX) that are capable of running on several

	cores.
signal processor based (DSP)	Radiation tolerant processor from Texas Instruments SMV320C6727B-SP, with 2400 MIPS of computing power / 1800 MFLOPS (maximum).
multi-core based	PACT XPP-III - reconfigurable pipelined processor, ideally suited for parallel processing of wide data streams. Large input / output bandwidth, it is best suited to "preprocessing" and aggregation (packetizing) data [20].
co-processor based	A processor (i.e. LEON2 or 3) with the i.e.: Fast Fourier Transform Coprocessor (FFTC) which is optimized for the processing of one or two dimensional Fourier transform, and offloads GPP with processing intensive tasks. Solutions are often dedicated to a range of applications such as GNSS receivers (like AGGA-4 [21]) or integrated spacecraft-controller-on-chip (like SCOC3 [22]).
reconfigurable logic (FPGA) based	Currently all state-of the-are FPGAs are available to a limited extent due to US export regulations: reconfigurable Xilinx Virtex 4 QV, Virtex 5 QV [23] as well as Microsemi anti-fuse RTAX or Flash based RTG4 [24].

Table 2 Payload Computer processing blocks overview [11], [13], [25]

2.3.4 Communication busses

Currently implemented or developed space avionics buses are listed below (Table 3):

Technology	Avionics class	Comment
RS-422, RS485	“industrial”	100 kbps, balanced, differential, multi-point, accepted by ESA
SpaceWire	“industrial” / “scientific”	2-200 Mbps, balanced, differential LVDS Pairs, full-duplex, point-to-point, accepted by ESA
MIL-STD-1553	“industrial”	1 Mbps, balanced, redundant, half-duplex, multi-point, accepted by ESA
CAN bus	“industrial”	1 Mbps, balanced, high noise environment compatible, multi-point, one failure tolerant, accepted by ESA
I2C bus	“industrial”	3.4 Mbps, unbalanced, two-wire, multi-point, accepted by ESA

(Gigabit) Ethernet	“industrial” / “scientific”	100 / 1000 Mbps, balanced, real-time overlays, still in R&D
Fibre Channel	“scientific”	1 Gbps, differential, point-to-point, still in R&D
Rapid IO	“scientific”	10 Gbps, balanced, differential, star/mesh topology, backplane or cable, still in R&D
InfiniBand	“scientific”	2.5 Gbps, balanced, differential point-to-points, still in R&D

Table 3 On-Board Avionic Bus technologies overview [8], [13]

The trend is clear: “industrial” class avionic busses trade communication robustness and responsiveness over throughput, and on the other end, “scientific” class avionic busses are designed to offer high throughput and automated error management over deterministic (real-time system compatibility) behavior and multi-point (i.e.: for broadcast or heartbeat signaling) access.

2.3.5 Resulting avionics requirements

As it has been mentioned before, avionics of different classes are typically mixed in a satellite systems in a way that either emphasizes cost-effectiveness and time-to-delivery factors, failure resilience, robustness or highest possible performance. Some typical characteristics and requirements of both described avionics classes are summed up in table below (Table 4):

requirement or characteristic	**avionics class**	
	“industrial”	**“scientific”**
mission type	telecommunication, Earth observation, navigation	astronomy, exploration
control	TC/ TM bandwidth varies, typically low	periods of increased TC traffic, typically high telemetry for autonomous operations monitoring
autonomy	low, limited to automated plan executions and survival actions	high, with long fully autonomous phases and no ground contacts
fault management	from simple, managed at constellation level to advanced mechanism for ensuring high availability	complex and extensive, driven by autonomy, often reconfigurable depending on mission phase (i.e.: high availability on cruise phase, high reliability on orbital maneuvering or descent phase)

implementation examples		
performance		
data processing	~ 100 MIPS	~ 10 GFLOPS
data storage	~ Mb to Tb	~ Gb
data transmission	~ 100 Mbps to Gbps	~ 10 Mbps
dependability		
reliability	average geostationary telecommunication satellite statistics: 97.7% after 6 years in orbit 92.2% after 12 years in orbit	94% after 6 years in orbit 91% after 12 years in orbit
security	high security concerns	low security concerns

Table 4 Avionics classes characterization [9], [26], [27]

Indeed, as for the moment of writing, most satellite buses (in high-end market) have an "industrial" class satellite platform and "scientific" class satellite payload, like it is expressed in example diagram below (Figure 6).

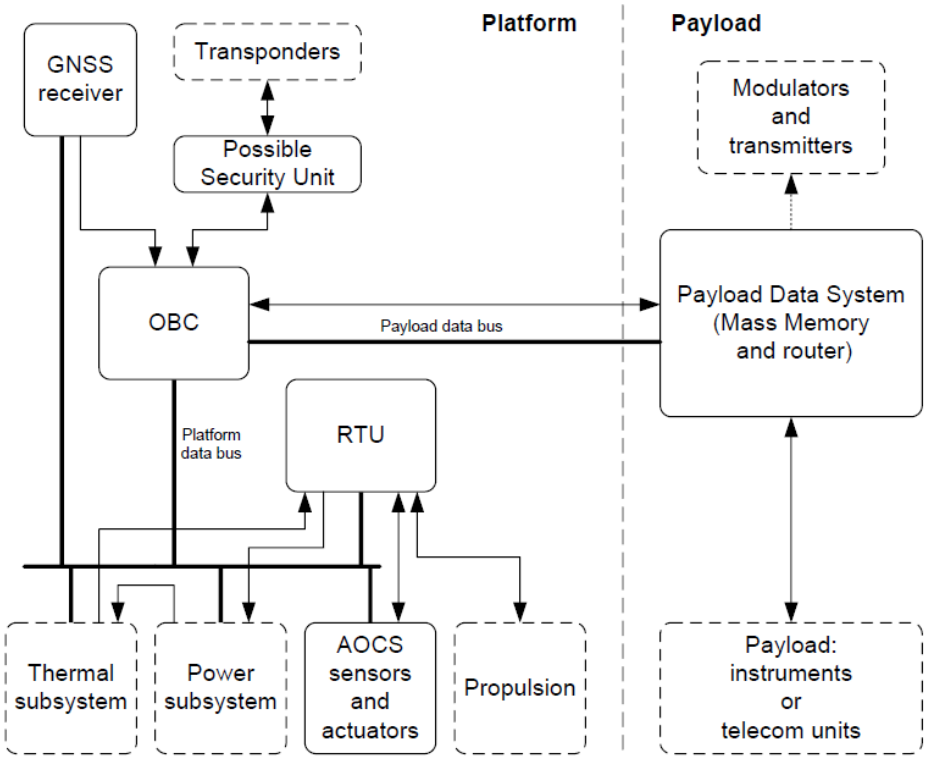

Figure 6 Typical modern hi-end satellite avionics architecture [10]

In fact, designers of space-borne electronics, control and data-processing systems in particular, have to face a decision whether the mission objectives will be best accomplished by system that is either:

- highly reliable, or,
- highly available, or,

- of high performance [9].

Decision has to be taken after careful and detailed trade-off analysis and with full visibility of system, sub-system and component level consequences, not to mention other, classic, project indicators, like:

- cost
- schedule
- functional user requirements
- physical user requirements (Size, Weight and Power)

2.4 Operational environment of space equipment

Following subchapter sums up main environmental influence vectors on space avionics. The environmental effects outline by any means does not exhaust all environmental aspects (like gravitational, geomagnetic, electromagnetic) that could be considered, but underlines those which are important and elaborates on radiation which contributes most to the electronic component faults and upshot of avionics systems failures.

2.4.1 Spacecraft failures

Operational environment of space equipment is very different from laboratory conditions or even harsh operations of military grade devices. An airborne appliance comes closest to space grade avionics but still stress levels and environmental influence are at significantly lower levels for aircrafts [28], [29].

As a side note, to advise the reader, challenges of launch and space are not the greatest that space equipment will face. Avionics and other space grade devices, if properly designed, are able to successfully cope with much higher levels of environmental influence that they will actually encounter during their operational life. In fact, often only brief consideration of ground handling, maintenance and test activities results in risky compromises necessary to conduct envisioned grounds operations. Creator’s hands often pose more serious threat to satellite hardware than space environment [30].

Nevertheless, several reports analyzing sources of spacecraft malfunctions and failures have been published. Several common conclusion, underpinning importance of careful avionics design and components base selection, with environmental conditions in mind. Failure reviews and analyzes show that [31]:

- electronics failures are the single largest type of failures. 45% of all failures in spacecraft are related to electronics. The second highest category are mechanical failures.
- attitude and orbital control systems have the highest failures (32%) followed by power systems (27%).
- 40% of the failures caused catastrophic loss of mission.
- 16% of the failures were due to the space environment itself (solar storms, debris, etc.)
- most of the failures (48%) occurred within the first year of the mission

Recapitulating, major factor contributing to spacecraft failures were electronics (avionics) design flaws or drops in manufacturing quality (indicated by early occurrence after mission start)

Other interesting findings, in [32] are:

- The effects of improving electronics parts quality from the 1970's to the 1990's are substantial. Failures due to parts quality issues went from 26% of the total failures down to 11% of the total failures.
- The total number of incidents related to failures dropped enormously from ~2,000 for pre-1977 spacecraft to less than 200 for 1990's design spacecraft.

The improvement in satellite mission success rate is attributed to introduction of meticulous component and assembly quality control procedures as well as more advanced and sophisticated design methodologies that are able to face the design challenges imposed by environment (by simulation and computer modeling).

2.4.2 Environmental effects review

Operational and non-operational environment of space equipment, avionics is the biggest factor that makes it different from terrestrial counterparts. Spaceborne systems must withstand the satellite or probe launch, then it must last in orbit for years (in rare cases, even decades). Space environment is definitely one of the harshest not only for human beings or any know life forms, but also for man-made electronics and mechanics. Moreover, the operational environment differs from mission to mission, often significantly. To make situation even worse, for the design or system engineering teams, the environment severity can also change depending on subsystem positioning inside the spacecraft or year of launch, which have direct correspondence to Sun activity and resulting radiation mix affecting the equipment [33]. Key environmental factors are explained in following section and subchapters.

Vacuum

Vacuum (or partial vacuum) disables convection which is one of major heat transfer vectors and increases risk of arcing even at moderate voltage gradients due to Paschen breakdown effect (easy ionization of low density neutral gases).

Heat transfer still happens through conduction and radiation but is much less effective, resulting in high thermal resistance to ambient. In turn electronic components, when powered up, heat up fast, causing large thermal gradient. Thermal gradients, cause fatal mechanical stresses, both, internal and external to component.

Finally, underestimated and not properly sink or radiated heat dissipation within component may lead to its burn out and failures.

Mechanical stresses

Mechanical stresses are related to launch process which lasts several minutes. Stress manifests itself in axial load of accelerating vehicle, lateral loads from vehicle steering and turbulences, mechanical vibration from engine, acoustic loads of breaking sound barrier or reflected from launch pad and and finally shocks from stages shutdown, stages firing, fairing jettisoning and pyrotechnic separations.

Mechanical stresses in avionics results in severed wires, fractured PCBs (including broken tracks) and cracked solder joints in component leads.

Meteoroid and debris

Every space vehicle is subject to hypervelocity (> 3km/s) impacts caused by man-made debris and meteoroids of natural origin. Damage caused by collisions varies and depends on trajectory, size and mass of impactors. Final impact results also depend on internal structure of spacecraft and secondary debris ejection.

Meteoroids, which may puncture spacecraft structure and equipment form streams whose occurrence can be predicted as their orbits and flux densities are known. There are also meteoroids which are not correlated to any stream, so called sporadics, whose flux is constant [28], [29], [33]–[37]

Spacecraft charging

Spacecraft is subject to interaction with low energy (< 50keV) particles already in higher parts of atmosphere (~70km). Plasmatic environment reaches far beyond ionosphere (~2000 km)

into magnetosphere boundaries (magnetosheath, magnetotail) up to solar wind and down into interplanetary and interstellar mediums.

Particles are collected on surfaces of spacecraft and lead to creation of strong electrical fields which may result in electrostatic discharge. There are two types of ESD related anomalies that affects avionics systems. First is surface charging with high differential potentials where arc can couple into spacecraft harness and affect avionics. Second is internal charging caused by penetrating electrons resulting in discharge arcs in proximity to fragile components (typically not very well protected against ESD).

Spacecraft charging used to cause most environmentally related anomalies, and in the same time, spacecraft charging is accounted for most of fatal anomalies leading to loss of craft. Nowadays, charging phenomena is much better understood also the design principles and proper ground testing methodologies has been established to reduce this environmental factor influence [34], [35].

2.4.3 Ionizing radiation and radiation effects on microelectronics

Radiation in space environment has wide spectrum of origins and can be considered as a mix of electrons, protons and heavy ions, at least from the perspective when particles affecting electronic components are taken into consideration.

In terms of near Earth radiation sources can be divided into three groups: energetic electrons and proton trapped in Earth'a magnetosphere (Van Allen belts), very high energy protons and heavy ions from interstellar origin (Galactic Cosmic Rays, GCRs) and protons, along with heavy ions, ejected from Sun Corona (Solar Flare events).

Earth's radiation environment is quite complex due to fact, that all mentioned radiation sources influence spacecraft in proportion heavily dependent on spacecraft orbit, spacecraft position in relation to Earth's magnetosphere and, generally, Sun's activity. As a rule of thumb, spacecraft in Low Earth Orbit will experience less than 5 krad(Si) per year and numerous single event effects when passing through polar regions, and, especially when passing South Atlantic Anomaly (check Figure 7 for clarification). In addition, in random moment, it will be affected GCRs (in case of LEO, low fluence, high energy) and solar flare protons (high fluence, medium to high energy) [28], [29], [36]–[38].

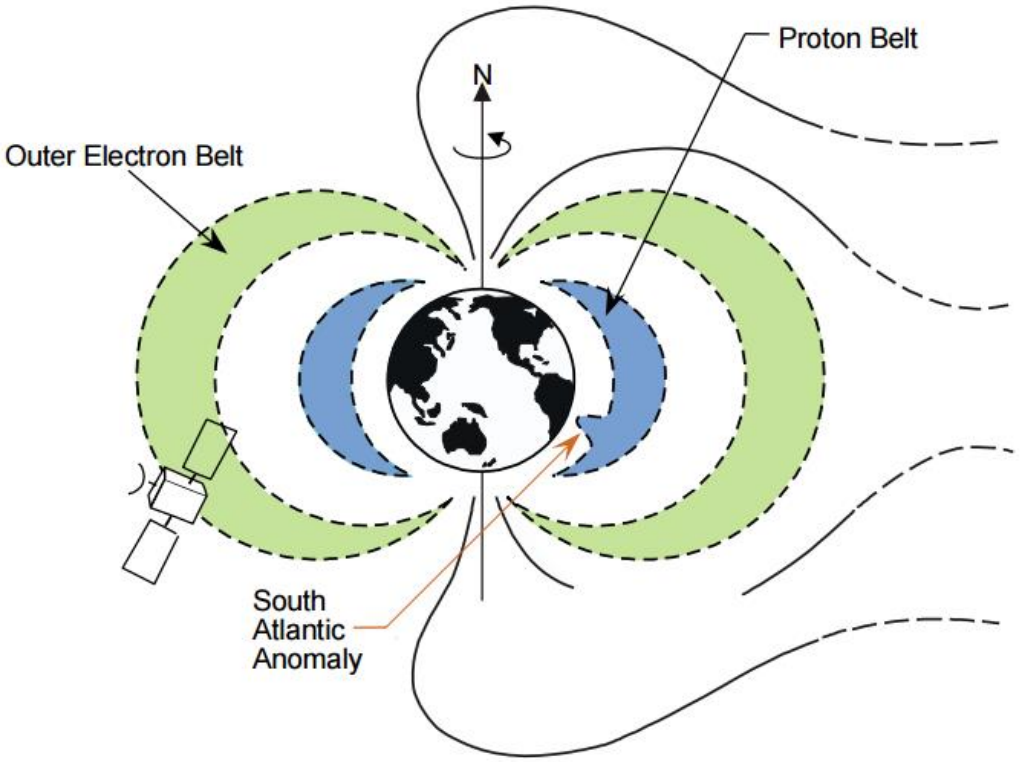

Figure 7 Earth's radiation belts and South Atlantic Anomaly [39]

From system designer viewpoint there are two major types of effects of radiation interacting with electronic components – Single Event Phenomena (SEP) and Total Dose effects. Total Dose effects are related to gradual and continuous change of semiconductor parameters due to radiation induced damage.

2.4.3.1 Total Ionizing Dose

TID, Total dose effects cover those caused by displacement damage (atoms moved from their original node positions in lattice generate additional energy levels in forbidden bandgap) and ionization effects (particle interaction generate electron-hole pairs of which some will not recombine and will drift freely in presence of electric field, forming photocurrents and trapped centers). Total Ionizing Dose (TID) is considered as a main design driver as Displacement Damage Dose (DDD) is of less concern due to natural shielding of electronics by spacecraft mechanical structure. DDD has to be carefully analyzed in longer missions and a higher energy particle fluxes as becomes significant part of total damage. It is important to note that DDD often has similar long-term degradation characteristics to TID, but one should be aware that TID tolerant device is not necessarily tolerant to DDD.

At their basics, Total Dose and Single Event Phenomena are very similar in nature, with key difference in energy involved in particle – semiconductor lattice interaction. Increased

transient dose rate can cause Single Even Phenomena using mechanisms for Total Ionizing Dose accumulation in rare events of i.e.: nuclear explosion. Hence, TID is sometimes referred as long-term effect and SEP, as short-term effect.

Total absorbed dose is referenced as absolute amount of energy deposited in unit, usually in avionics context, expressed in rads (100 ergs of energy per 1 g of specified irradiated material, where erg is 10^{-7} J).

TID, in most common and widespread, CMOS technology, brings two major consequences for device operation: changed threshold levels in MOS transistors, increased leakage currents and changed propagation times. Also input / output, low/high state levels may be modified.

There is very little that can be done to prevent TID effects except tinkering with design and equipment configuration. Shielding is one of obvious approaches, often supported with statistical analysis ensuring that device local environment will be soft enough that total radiation dose absorbed in mission life time will be many times lower than device specification verified in radiation tests.

It is worth to note that shielding is heavy (therefore adds to launch costs) and that its effectiveness is indeed limited thanks to secondary effects like Bremsstrahlung. Component shielding rather that subsystem shielding may be one of preferred choices to meet desired environmental exposure levels taking into account necessary design margins (ECSS design standards define this margin as a factor of 2 - device nominal TID / TID absorbed during mission lifetime) [28], [36], [40]–[45].

2.4.3.2 Single Event Phenomena

SEP, are high energy events that happen when charged particle travels through semiconductor lattice depositing energy along the path. Deposited energy ionizes the lattice forming electron-hole plasma following the particle trajectory. Plasma generation causes a current induction, and, as a consequence, depending on where in semiconductor this event occurs, results in transient signals generation, bit-flips, latch-ups that eventually leads to burnouts.

The sensitivity of device to SEP is expressed in Linear Energy Transfer (LET) threshold. LET is a measure of the energy deposited per unit length as an ionizing particle travels through a material. Unit of LET is MeV * cm^2 / mg. Mentioned LET threshold (LET_{th}) is a minimum LET that causes a SEP effect, assuming particle fluence of 10^6 protons or ions / cm^2.

Another measure of describing SEP is the device cross-section. Cross section is an effective device area sensitive to ionization and is characterized against varying LET. Cross section is expressed in cm^2 / device or cm^2 / bit .

Single Event Effects (SEEs) occur when a single ion strikes a material (SEP), depositing sufficient energy either through its prime strike (e.g., direct ionization via GCR, Figure 8) or by the secondary particles that occur from the strike (e.g., indirect ionization via protons, Figure 9) to cause an effect in the device. There are many types of SEE and they can be divided on soft errors (easily recoverable, non-destructive) and hard errors (permanent, that may lead to destruction) [46]–[51].

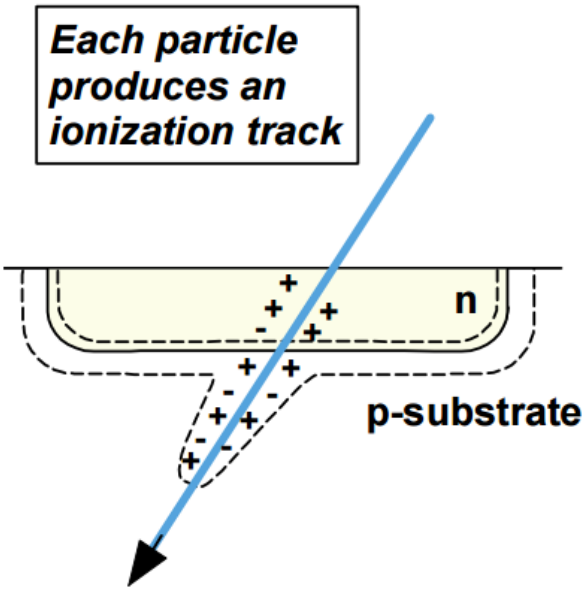

Figure 8 Direct semiconductor ionization [39]

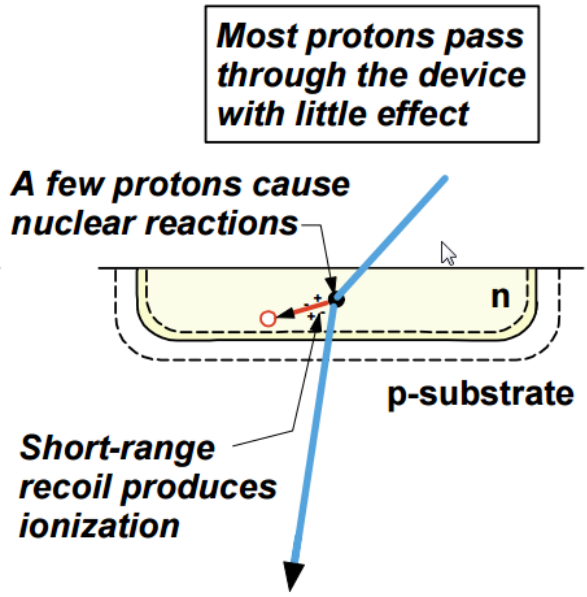

Figure 9 Indirect semiconductor ionization [39]

Most often, devices that are tested for possible utilization in space are characterized taking into account two most important susceptibility types:

- Heavy Ion Susceptibility
    - Even although spectrum is deeply cut off above 30 MeV*cm^2/mg, effective nuclei ability to generate SEE is observable till 75 MeV*cm^2/mg
- Proton Susceptibility
    - Proton upsets observable under LET_{th} < 15 MeV*cm^2/mg

Although proton LET is very low, proton test can still provide useful information about microelectronic device susceptibility to faults. Moreover, especially for Low Earth Orbits, protons make up large fraction of particles that interact with avionics and have significant contribution to overall occurring upset rate.

Soft errors occurring in modern avionics are mainly Single Event Upsets (SEUs), Single Event Transients (SETs) and Single Event Functional Interrupts (SEFIs). SETs are softest errors as are self-recoverable. In effect of SEP, generated charge, when flowing through resistance, causes a voltage spike which can be interpreted by device i.e.: as additional clock edge or interrupt or temporary fake state on input interface. Although soft in its nature, when unmitigated, SET can propagate through the system affecting the functionality. SEUs are similar to SET but charge generated in high energy particle interaction is large enough to flip the flip-flop. This change will last as long as it is not overwritten by operation on the flip-flop or by reset. SEUs are of great concern, as they affect RAMs, processor registers, finite-state machines. As a result there are plenty of errors detection and correction mechanisms that have been proven by years of in-space validation. Interestingly, as transistors are further miniaturized and lower energies are necessary to flip a bit (0.1 - 0.5 pJ) a phenomenon called Multiple Bit Upsets (MBUs) occurred [52], [53]. MBUs are simultaneous bit flips concentrated in one area, close to SEP ionization path. Difficulty introduced by MBUs is that up to date many of useful error mitigation methods used nearby bits to correct errors – like triple modular redundancy or redundancy codes. Finally, SEFI is a SEU which occurs in sensitive part of device i.e.: maintenance, test, or debug circuitry. Its peculiarity is that it cannot be corrected on the fly and whole device is rendered as malfunctioning. Only reset of device (often hard reset by power cycle) can bring it back to operation. It is distinguished from typical SEUs on purpose – probability of occurrence of SEFI is important factor contributing to overall system dependability and heavily affects system architecture.

SEFIs are important device fault source especially in complex devices like [49], [54]–[59]

- high density memories where affect
    - internal test modes
    - microprogrammed cell architecture
- flash memories where affect
    - crashes internal state controller and buffers
- Xilinx Programmable Logic Arrays where affect
    - configuration memory
    - automatic scrubbing and read-back circuitry
- Microprocessors
    - Many categories of responses
    - Detection and recovery are very difficult problems to solve

Hard SEE errors are Single Event latch-ups (SELs), Single Event Burnouts (SEBs), Single Event Gate Ruptures (SEGRs). Hard Single Event Effects occurring in one of components can cause permanent damage to a whole subsystem or system.

Single Event Latchup (SEL) is a potentially destructive condition involving parasitic thyristor (silicon-controlled-rectifier, SCR) present in CMOS, ECL and bipolar technologies, activated by strikes of highly energetic heavy ions, protons and neutrons. When activated, device draws current exceeding its specification until power is removed and thyristor closes again.

SELs are strongly temperature dependent, their initiation threshold decreases and cross-section increases with temperature rise. SELs, when mistreated, results in device excessive heating and destruction. When properly treated, SELs are dangerous, but recoverable anomalies. Typically, to cope with an issue, SEL-immune components, defined as a devices having a LET_{th} > 100 MeV*cm^2/mg, should be used. In any other case a local latch-up protection has to be employed to switch off component and its internal parasitic thyristor in event of its activation. Modern devices may have various latch-up modes resulting in various current levels which poses a difficult characterization issue and serious protection design problem.

SEB is a highly localized destructive burnout of the drain-source in power MOSFETs caused by excessive current flowing through small volume. SEGR is the destructive burnout of a gate insulator in a power MOSFET and some programmable logic devices. Only possibility to cope with SEB and SEGR hard anomalies (always destructive) is redundant device that is able to take over operation after destruction of nominal unit.

While SEBs are caused by heavy ions, protons and neutrons, SEGRs are cause only by heavy ions and their occurrence highly depends on angle of incidence and electric field in gate oxide [29], [37], [42], [45], [60].

2.4.4 Upset cross-section and fault rate calculation

As it has been mentioned before, common and convenient method of measuring vulnerability of microelectronic device to radiation and its susceptibility to exhibit a SEP is devices' cross-section, which is measured in number of given type of event per LET level resulting from radiation particle's energy.

Tests which are set-up to characterize device cross-section are performed by varying heavy mass and angle of incident which in turn varies amount of charge deposited in semiconductor sensitive volume. This process leads to characterization of SEP response of a device, and, in

particular, estimation of single event saturation value – which is ultimate measure of device sensitivity (Figure 10).

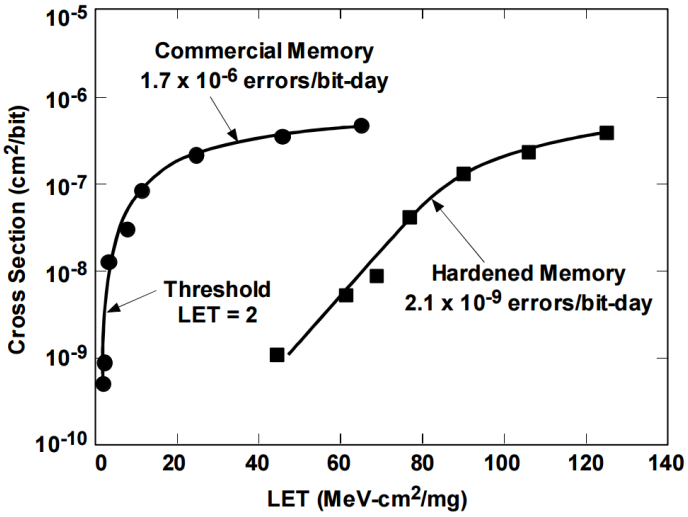

Figure 10 Example commercial and radiation-hardened device cross-sections [39].

Device cross-section, when convoluted with environmental model, provides device (or device functional block if such characterization has been separately performed) fault rate function. Fault rate function needs to be integrated over whole LET range in order to obtain fault rate value estimation for given device in given environment. In other words, the fault rate (in faults/bit-hour) is calculated by taking into account instantaneous particle flux (the environment) and the upset cross section curve (the technology and architecture) which describes the device's sensitivity to that environment (Figure 11).

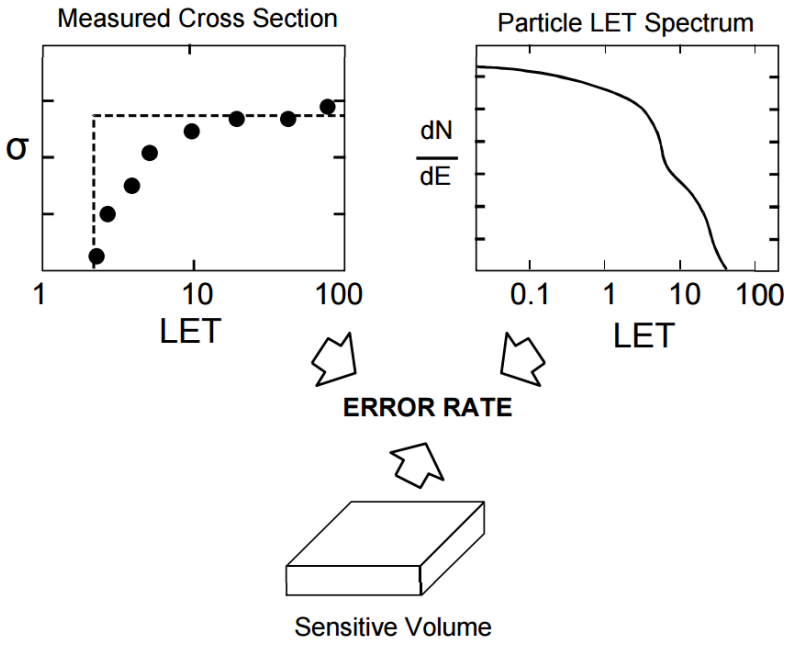

Figure 11 Error (upset) rate calculation [39]

Each SEP has its own device cross-section, therefore it is sensible to separately estimate number of events for each phenomena for which experimental data is available. For example, for SEU, the number of events could be the number of bit upsets in a circuit. For SEL, the number of events could be the number of times a circuit was triggered into a latched high-current state. The device SEU-sensitive cross section is simply calculated as the number of events divided by the particle fluence [41], [61].

3 Avionics modeling

As it has been outlined before, complex system have to be modeled in various ways in order to simplify and emphasize important or interesting system features and support design and decision making process. Different industrial domains have different approaches, for sake of clarity, following chapter deals purely with aerospace. Nevertheless, concepts presented here are generic and could be applied to i.e.: medical equipment, automotive or military domains, after some modifications.

3.1 Functionality modeling and simulation

Modeling or simulation of an avionic systems is crucial for correct high level design, equipment technical specification and finally requirements baseline which is trackable to user needs while ensuring common understanding on what is actually being built. Both modeling of the design allows for:

- idea feasibility checks
- system performance verification
- architecture trade-offs
- early prototyping
- reduction of number of hardware models

3.1.1 Analytical and mathematical models

As system always work by means of affecting the energy, mass or information flow, each model creation starts with capture of basic building blocks and attribution of mathematical function defining relation between building blocks' input and output. The first step is often called modeling of system components as transfer functions.

Natural next step is introduction of time response to modeled blocks by creating components with time response. Such components do not act instantaneously but effects of input changes are visible on outputs after some time. Additionally, outputs are often dependent not only on inputs but also on components internal states. Hence, at this level of modeling, components contain information about their history. Components with time response modeling are most often described in form of mix of algebraic and differential equations.

Moving further in avionics description after establishing components description including time dependencies leads to system balance equations. Balance equation(s) describe system dynamics in simplified manner linking storage, flow or transformation of mass, energy or

information. Balance equations are valid for fluid systems modeling (propulsion control and tank pressurization, fuel cells, life support and HVAC systems, laboratory rack systems).

Similarly, satellite attitude determination and control, modeled as rigid body system, described in full extent by set of differential equations, describing the position vector variation as axial velocities, which in turn are result of sum of forces acting along of satellite's axes.

Some electrical systems are conveniently modeled by means of flow and storage. In example, satellite power systems, especially physics behind them, are described as current flow between solar cells, battery packs, charge-discharge controller and power conversion and distribution units. This is done by formulation first order linear differential equations, with non-constant coefficients feed from operation curves of solar cells and batteries.

In this approach the ultimate goal is to describe the state space of a system by forming two equations. First one, called state equation, defines the internal state of the system to the full extent. Second one, called output equation, defines the system output as a function of the current system state and input. Both equations, ideally, thoroughly describe system behavior by leading creation of vector space containing all possible internal states of the system.

3.1.2 Dynamic and functional models

While flow or variation of physical attributes over the system is conveniently expressed in terms of differential equations, as explained in previous chapter, the modeling of computation or data flow taking into account their dynamic nature uses state charts or graphs

Finite-state machine (FSM) is a mathematical model used to represent computation in classic logic device or, when extended like in UML, to represent software execution paths and software component relations. FSM is defined in such way it can be in one state only at a time, and change of state is induced by external event. FSMs are convenient way of representing single activity over time and showing the dependency of models systems on transition triggering conditions.

Flow network is particular example of directed graph each edge has its maximum capacity and has a certain temporary flow value. Flow value cannot exceed the maximum capacity of an edge. Flow has to subordinate to preservation rule, meaning that effective network node inflow must be equal to outflow (with exception of source and sink nodes). This is simple methodology allowing for brief analysis of dynamic behavior of systems (used in later chapter for i.e. brief performance evaluation). Flow networks are especially useful for modeling

system aspects related to transportation like electric current, liquid or heat flow or data transfer. Flow networks are useful not only for analysis of system evolution flows over time but also for finding maximum flow capability of whole network.

Quite often, computing and control systems, have to be analyzed as evolving over time (in continuous domain), while system state transitions occur at discrete events, when associated conditions trigger desired reaction. Such events exhibit a competition against other triggers and each one of them, typically, has own stochastic mechanism that governs determining new system state. For each state transition, new events may be scheduled and previously scheduled events may be cancelled. **Petri Nets** (also known as PN or P/T-net) provide excellent analysis framework for this kind of modelling, especially if modelled system exhibits randomness, state-transitions, concurrency and scheduling [62].

Petri net is known as a convenient graphical method of modeling distributed and concurrent systems. A Petri net is a directed bipartite graph, in which vertices represent system's transitions (events) and places (states or conditions), while edges represent directed arc, describing state-event interaction**.** PN are especially useful in describing complex, discrete event system for purpose of functional correctness checks and performance metrics analysis.

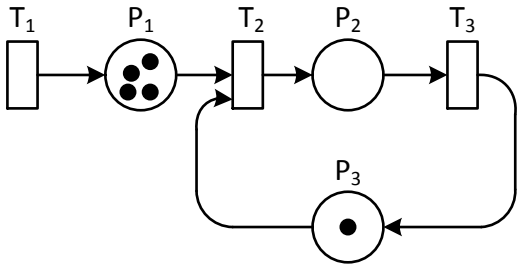

Figure 12 Petri Net for performance analysis

In case behavior of a system is non-deterministic, a time-augmented PN (Stochastic, or Generalized Stochastic) could be introduced, for modeling of statistical mechanisms, expressing underlying system behavior uncertainties.

Similarly to some industrial standards mentioned before like UML activity diagram or flow network, Petri nets have an appealing graphical notation for step by step process description that can represent execution choice, data flow and concurrency. Great advantage of PN with respect to other solution is a very well understood mathematical definition of their execution, simulation and advance mathematical tools for their analysis [63]–[68].

More details on Petri Nets are in 3.3 and Appendix C .

3.1.3 Simulators

Simulation is a natural next step after analysis. Simulation is very important part of verification process, that allows for early check that system specification may be implemented and that requirements baseline set leads to satisfying mission (operational / performance / features) objectives.

While every system design starts with general conceptualization stage involving budgeting, trade-off studies, variants analyses as well as more specific and tailored discipline or branch evaluation and design characterization (brief electrical design simulation, radio link and EM propagation and radiation analysis, thermal and structural design), it inevitably leads to first checks whether proposed solution is appropriate to solve the problem, it fits the needs and is feasible in given context.

In European (or ESA) nomenclature, following simulator distinctions could be found:

Type	Abbrev.	Description
Functional Verification Bench	FVB	Also known as "Algorithm in the Loop". First complete system (or satellite) model emphasizing key algorithm(s) to be tested within reference framework of environmental and technical capabilities.
Software Verification Facility	SVF	Also known as "Software in the Loop". Typically, a framework to execute on-board software in target functional environment. Usually enables operating system, control application, low level drivers verification. Often relies on software mock-ups of external systems.
System Testbed	STB	Also known as "Controller in the Loop". Main execution unit (On-Board Computer or Data Processing Unit) is available in hardware as an Engineering Model, which allows for first integration of controller hardware and software. Simulator provides operation environment of the controller, which typically would be avionic bus with remote terminal units as well as any other equipment under supervision and management of controller. Redundancy (if present) switching shall be evaluated at this stage.
Electrical Functional Model	EFM	Also known as "Hardware in the Loop". Extension of STB, where equipment and remote terminal units models (including avionics if not present until this stage) are change to hardware units, Engineering or Elegant Breadboard Models that allow full compatibility tests on physical and functional levels. Equipment may need its own Special Checkout Equipment (SCOE) which provides dedicated stimuli enabling full functionality spectrum and allowing for on-demand event triggering.
Spacecraft Simulator	S/C-Sim	Very detailed simulation of the satellite, containing all subsystems functional and electrical representatives,

		combined with necessary SCOEs. Typically, tied to ground station and used for operations personnel training, spacecraft operation exercises, software patches preparations as well as trusted reference for contingency activities.

Table 5 Spacecraft subsystem simulators [69], [70]

Trend in satellite simulators usage, in Europe, is clear, as shown on Figure 13 below, representing ESA contract required scope of simulator deliveries for various missions in past 15 years.

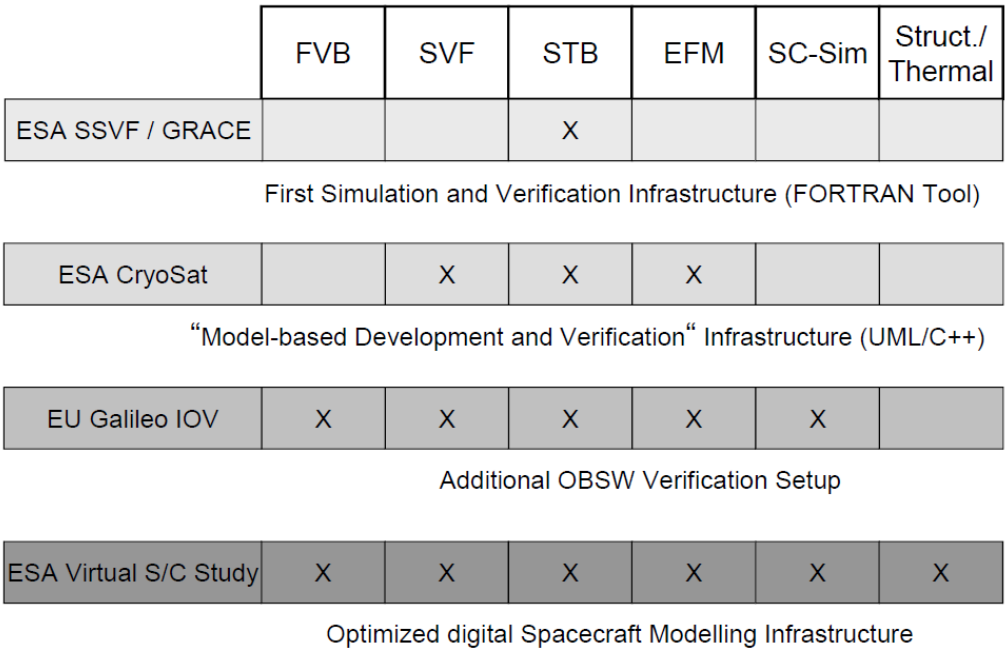

Figure 13 Expansion of simulation technologies [70]

Approximate launch dates of each of mentioned missions (top to bottom): GRACE 2002, CryoSat-1 2005, Galileo IOV 2011, ESA Virtual S/C Study 2012. Simulators gain widespread acceptance and become a de facto standard for continuous verification process along the specifications and requirements crystallization for each of systems. Although, they bring some additional costs, they enforce early subsystem / algorithm prototyping and interface freezing, they encourage design reuse. As a final result of introduction of simulators, mistakes and design errors are detected fast which in turn causes significant reduction in cost and schedule overruns [69].

3.2 Reliability prediction methods and methodologies

Reliability block diagram (RBD) is one of simplest methods of evaluating and showing how component's reliability contributes to overall reliability of a complex system as a whole.

RBD is typically a set of blocks, connected in series or in parallel. Each of the blocks represents a component of the system and its failure rate. Parallel configuration is considered as redundancy, while in series configuration each component is commonly known as single point of failure. The way blocks are connected together represents also a logical or sequential hierarchy embedded in system architecture. Moreover, if component's state (failed or operational) is represents as switch (respectively, open or closed) then system is operational as long as path exists, linking diagram logical beginning and end.

RBDs can be evaluated is faults of systems' components are statistically independent, which is often but not always, true. If statistical independence prerequisite is not met, then methodology modification, such as dynamic RBD shall be used [68], [71]–[74]

Fault Tree is systematic analysis method that allows for finding cause or set of causes that led to an event of interest (which is typically, a failure). FTA combines lower tier faults and errors into system failures, of potentially catastrophic consequences, using Boolean algebra (logic operations like sum, product). It is very useful in providing clear view on how risks spread across the system, and what is often very desirable, enables identification of common cause failure groups, considered main redundancy nullifying hazard.

FTA is often compared to **Failure Mode, Effects and Criticality Analysis** (FMECA). While FMECA is used mainly for forecasting, in bottom-up manner, how component failures influence system operation, FTA is tool for anomaly causes investigation, performed inductively and in top-down fashion .

FTA can be used to

- identify critical parts or components
- understand dependencies within the system
- optimize the design
- ensure compliance with requirements and good practices
- investigate anomalies and to create recommendations

FTA if expanded by cause probabilities, can be used to perform system level probabilistic risk assessment and to quantitatively determine critical failure likelihood.

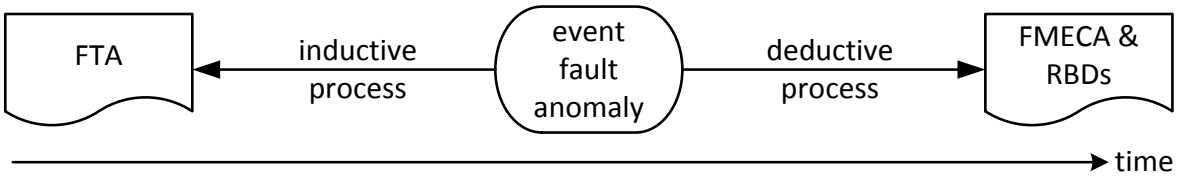

Figure 14 Inductive and deductive methodologies for anomaly analysis

There is a logic link between RBDs and FTAs. If series components connection is replaced by logic product (conjunction) and parallel components connection is replaced by logic sum (alternative) then success tree is obtained. A fault tree can be obtained by applying de Morgan's laws to success tree. FTA is described in greater detail in [75]–[78] and FMECA in [79], [80].

Markov analysis

Markov analysis uses Markov process to estimate the reliability (probability of being operational). Markov process is a stochastic process in which future depends solely on current system state. In this kind of description, system is memoryless and its history has no influence on any of its current parameters. Current system and next systems states are random variables.

A stochastic process in order to be Markovian, it has to exhibit following properties while moving from one state to the other [81]:

1. The probability of transitioning to a particular state depends only on the current state of the process.
2. The transition probabilities between states are constant over time.
3. The sum of all transition probabilities from a given state to any other state (including the current state) must be 1.

Typically, a Markov process is represent by state diagram, showing each od possible system states and all possible transitions between them. States are presented as circles with labels. State transitions are depicted as directed arcs with transition rates (probabilities between states). Self-directed state transitions (probability of staying in the same state) can be omitted for clarity. Obvious assumption, is that system can be in one state at a time. Example of Markov process is shown on Figure 15.

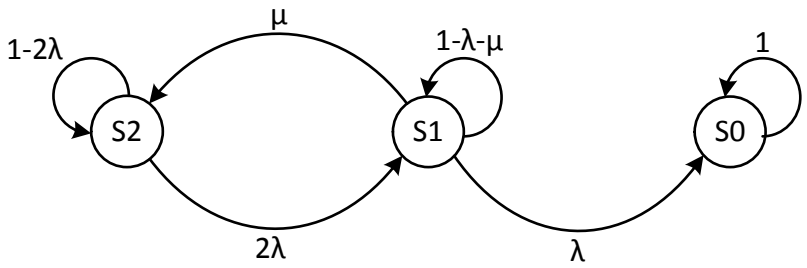

Figure 15 Markov process (chain) example

It is worth to note similarity between Markov process and Finite State Machine. In fact Markov model is a Finite State Machine, which? each state represents characteristic system feature. Such FSM, starts in initial state and transitions to other states, as it is defined by probabilities in transition matrix. Modeling of interesting system feature is done tracking of how states change over time. Moreover, depending on what is actually needed, relevant and convenient, Markov process could be discrete or continuous. Both ways of description are valid and if Markov model is simulated (rather than looking after analytical solution), both ways of description are equivalent [82]–[85].

FIDES

FIDES is a guide, created by French aerospace and military industrial entities, that allows methodologies for predicted reliability evaluation and reliability assurance process control and audit. FIDES aims to express all reliability predictions in FITs (number of failure for 109 hours) or MTBF. FIDES guidelines have two objectives that make them suitable in vast range of applications. One objective is to provide realistic and trustworthy means of evaluation of electronics components and assemblies, that takes into account not only typical operational environments, but also aggressive (airborne, chemical) and nonaggressive (storage). Second objective is equip the user with tools for construction and control of reliability

FIDES reliability methods, in general, include following factors [86], [87]:

- specifics of electrical, electronics and electromagnetic components, as separate instances or joined in PCB or subassemblies
- all physical factors affecting reliability
- life profile
- physical overstresses (duration, frequency)
- development, production, operation, maintenance processes

What differentiates FIDES from other, older methodologies, is that it is based on physics of failures and is supported by analyses of test data, feedback from operations and existing, very well known, models. Older approach was to rely on statistical analysis of operational data which estimations were sensitive to process or external conditions variations.

FIDES is directly applicable to various industry branches dealing with electronics, including: aeronautics, navy, military, power and production automation, transportation, space, telecommunication and household appliances. It is based on decades of experience and slowly but steadily supersedes other reliability estimation and control handling manuals created for each industrial niche or branch [88]–[91].

Petri nets

Very interesting feature of Petri Nets mentioned in one of previous chapters, especially in its time augmented variants (i.e.: stochastic), is that they naturally represent high level of abstraction system state. If it is considered that modelled system could be operational or failed or somewhere in between, depending on exact implementation, then tokens represent i.e. information on how many of operating units are still operating. By declaring what combination of tokens and places is interested as system still capable of delivering satisfactory fraction of its capabilities, one can evaluate system availability or reliability.

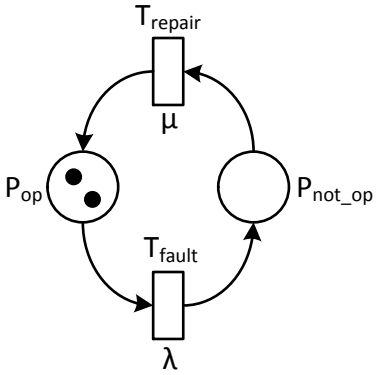

Figure 16 Petri net for reliability modeling

What is very important, flexibility and capability of Petri Nets to model very different but equally important aspects of system operation (performance and availability), makes them very interesting candidate for a tool that allows for integrated, cross-domain modeling of complex systems [64].

3.3 Petri Nets as a cross-domain modeling tool

As pointed before Petri Nets are mathematical and visual modeling tool applicable to various fields of interest. They are especially useful in describing and analyzing complex computational systems that exhibit many of properties which are included in PN design methodology like concurrency and parallelism, distribution, synchronous and asynchronous events and non-deterministic, probability based behavior. Moreover, Petri Nets, due to their graphical nature, can serve as a communication tool that facilitates conceptual exchanges. Tokens present in PNs are unbounded by any higher interpretation, can serve as indicators for concurrency, state or resource (physical or abstract) – whatever fits the needs. Indeed, also Petri Net places and transitions may have different interpretations in different applications and context [92].

Petri Net is a bipartite , directed, weighted graph. It has two vertices subsets, places and transitions. Commonly, the symbol of a place is a circle and the symbol of transition is a bar or a rectangle. The edges of the graph are called arcs. Graph edges connect vertices from each of subsets, this means that arcs are drown either from a place to a transition or from a transition to a place.

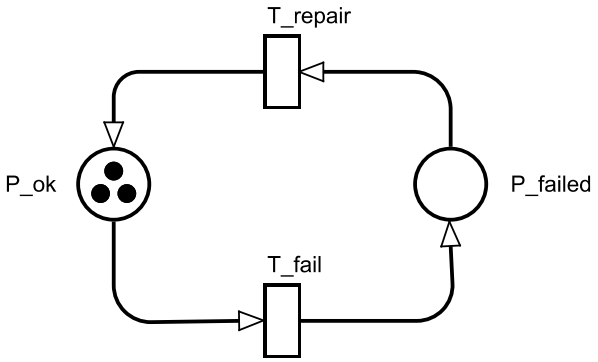

Figure 17 Simple Petri Net example

Dynamic behavior of Petri net is described by tokens and their interaction with transitions. Tokens represent a state or a resource (or lack thereof). Tokens are represented by solid dot (or number, for sake of clarity) and can reside only in places. Place can have maximum token capacity. Tokens enable the transitions, in a sense that if an arc is drawn from place to transition, it is interpreted, that, in order to fire a transition, an arc condition must be met – number of tokens in in-place(place from which the arc is drawn) must be equal or larger to arc condition (when condition label is omitted it means it is 1).

For a given transition t it is often defined that input set is set of all places (in-places) for which arcs are drawn from these places to the transition t. Similarly, output set is defined as set of all places (out-places) for which arcs are drawn from transition t to these places. Symmetric definitions can be made for input and output sets of given place p. If there is more than one in-place in transition's input set, then transition will be enabled, if and only if all input arc conditions are met.

Distribution of tokens over Petri net is called a marking – it is a function assigning to each place, number of token present in that place (as shown on Figure 17 P_ok holds 3 tokens, P_failed holds 0 tokens).

If transition is enabled, it can fire, which results in removal of tokens from places being in input set of this transition and in insertion of tokens to places being in output set of this transition. Amount of tokens removed from place or inserted to a place depends on arc label (enabling condition). There is no token preservation rule present in Petri Nets. Total numbers of tokens removed and inserted may be different (depends on network design). Additionally, transition doesn't have to have input place (then it is source transition) or output place (then it is sink transition).

PNs are useful to analyze complex system and processes. They can be simulated to show how system parameters or features change over time. Alternatively, PNs can be formally analyzed to find network attributes which may indicate some intrinsic system or process peculiarities.

In example, Reachability set is a set of all possible markings that are reachable from given initial marking. If a marking is in reachability set of other marking, it means there exists a sequence of transition firing transforming on marking into the other one. Analyzing of reachability set allows for checking whether desired (or undesired) state is reachable at all and what would be the conditions for that. The latter can be done by construction of reachability graph. Reachability graph is directed graph whose node represent markings and edges represent transitions between graph nodes – markings.

Petri Net markings analysis is helpful to find out some hidden system properties. Marking of Petri net is live if a every transition in net can be fired an infinite number of times (that is, for every reachable marking there exists a sequence of transition firing that includes any of transitions). Liveliness property of PN tells the designed about occurrence of, in example, deadlocks.

Similarly, PN boundedness is network property – if there exist a positive integer k such as for every reachable marking in reachability set of initial marking, number of tokens is in each place is lower than k (that is, at any reachable state, any of places cannot hold more than k tokens). Boundedness property is used for modelling limited resources.

There are various extensions to briefly described hereinbefore classic Petri Nets. In context of avionics modeling, especially useful are additional control capabilities and timed transitions (both deterministic and non-deterministic) which are introduced by Generalized Stochastic Petri Nets (GSPNs).

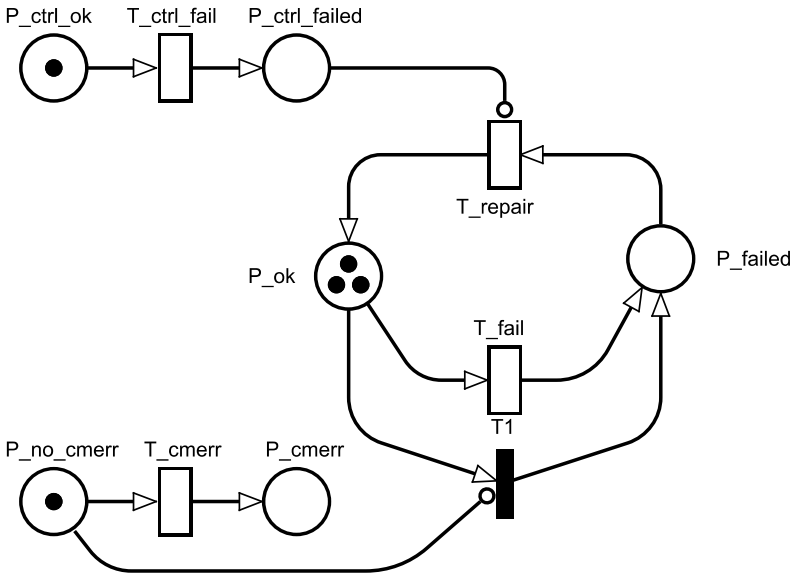

Figure 18 Complex Petri Net example (GSPN)

In example, immediate transitions are convenient way of modelling complex behavior by encoding logic and algorithms to nets or subnets. Immediate transition fires as soon as it is enabled (there is no delay). It has precedence over all other timed transitions. If there is more than one immediate transition in network, here might be precedence conflict present. There are various ways of resolving this (and other similar) conflicts in Petri Nets, like transition prioritizing or global probability switch setting pace, and sequence, of transition firing. This issue is implementation, hence, used tool, dependent. Immediate transition depicted as a narrow solid bar is depicted on Figure 18 (T1).

Another interesting feature of GSPN is inhibitor arc. Inhibitor arc is a graph edge which imposes a logical condition on transition it is connected to without capability to affect the number of tokens in place it is originating from (control place). Logic condition imposed on transition is simply that it is enable only when number of tokens in control place is lower than arbitrarily set number (inhibitor arc label). If number of tokes in control place is equal or

larger to arc's label the controlled transition is disabled. While this functionality is very interesting from simplicity standpoint it brings consequences of reducing the analyzability of mentioned model (rendering i.e. formal analysis methods useless leaving simulations as main way of observing the net activity). Inhibitor arc (arc with empty circle at the transition end) is shown of Figure 18 (connects place P_no_cm_error and transition T1).

Deterministic (delayed, timed) transitions are useful Petri Nets extension that allows to model change of state due to completion of activity (which takes defined amount of time) rather than fulfilling logic condition. In this case, transition become enabled after period associated with time necessary to perform modelled activity passes. After firing, transition "timer" is reset. This is most natural association of time to PN transitions.

Extremely useful feature of Petri Nets is their link to stochastic system description. If a timed transition firing delay is a random variable then the transition behavior is governed by probabilistic model. This aspect of PN is especially desired for modelling physical systems and taking into account uncertainties associated with, very often, incomplete view of the system or its parameters.

Stochastic PN can incorporate any probability distribution but one of commonly used is exponential probability distributions of transition firing delay. The exponential probability distribution function is the only probability distribution which is continuous and memoryless. If for Petri Net is build using random delay transitions with exponential probability distribution and all measures of interest are based solely on number of tokens in various places then reachability graph for given GSPN is an isomorphic transformation of the net into the semi-Markov process. If GSPN indeed can be reduced to embedded Markov Chain, then the associated transitions probability matrix immediately allows to yield measures of interest – which might be attractive for reliability or availability analysis.

Generalized Stochastic Petri Nets are perfect for representing dynamic systems with discrete events. PN allow to effortlessly capture even complex behavior of modeled systems and, to some extent, support automatic construction of stochastic processes and mathematical analysis of system under investigation [7], [62]–[66], [93]–[102].

In practice, PN suffer from state space explosion during reachability graph generation and typical numerical analysis limitations. Moreover, plethora of Petri Nets extensions led to lack of standardization in handling and interpretation of networks as well as multitude of software

tools, each and every of them, being far from perfect (provided for scientific purposes rather than engineering process support).

4 Coronagraph Control Box (CCB)

The Coronagraph Control Box (CCB) is the electronic controller of the ASPIICS coronagraph (Coronagraph Instrument, CI, described in greater detail in Appendix A), which is a main scientific payload of European Space Agency PROBA-3 mission. It consists of a compact housing that contains:

- Power conditioning unit (PCU) that is a power supply module that provides all the voltages required by CI along with voltage / current measurement capabilities for telemetry data generation and protection circuits.. PCU is switched on at the moment when ADPMS provides power, and supplies DPU and AEU instantaneously
- Data processing unit (DPU) is an embedded payload computer module that is capable of processing and buffering data and execute management and control algorithm. DPU is responsible for interaction with PROBA-3 on-board computer – Advanced Data and Power Management System (ADPMS)
- Ancillary electronics unit (AEU) that contains switches for ASPIICS power on and off control, as well as advanced actuation control for FWA and FDA stepper motors, COB heater system and ADC converters to gather telemetry data. Whole AEU functionality is directly controlled by DPU

System architecture as well as more detailed description of subassemblies and operation concept is expounded hereinafter.

4.1 CCB architecture

CCB consists of cold-redundant power conditioning unit (PCU) and digital processing unit (DPU) and not redundant, but physically consisting of two boards, ancillary electronics unit (Figure 19). AEU although is not required to be redundant, since there are two printed circuits boards, functionality might be split in a way they actually will be redundant.

PCUs contain hybrid and integrated DC/DC converters, power switches, overcurrent and overvoltage protection. DPUs, main processing units of CCB, contain processor (GR712RC), FPGA (RTAX2000S), boot Flash (UT8QNF8M8), SDRAM (3DSD2G16VS4364) and SRAM (3DSR4M08CS1647). AEU contains some motor control switches, signal conditioning circuits for sensors readout and amplifiers.

Most of the SEEs, both destructive and non-destructive will be localized in DPU due to high concentration of complex integrated circuits of very large scale of integration.

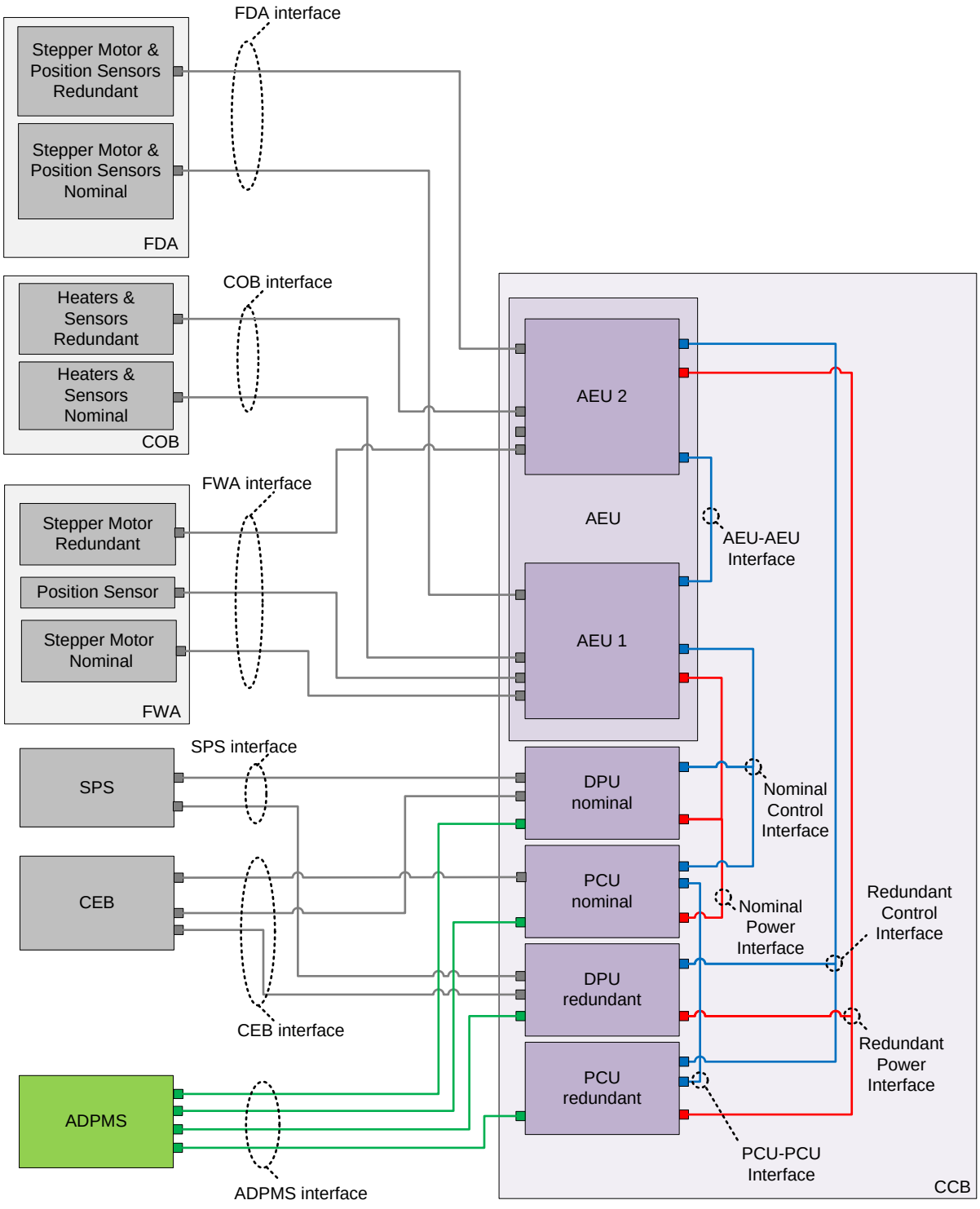

Figure 19 CCB architecture

4.2 DPU architecture

DPU, responsible for all control and scientific processing algorithms, is based on two main parts. One, is processor (GR712RC), executing control software, and FPGA (RTAX2000S), that acts as a processor's co-processors implementing all the features that are not available in processor. CPU is interfaced to Flash (for boot-image and application software storage) and SDRAM as operations memory. FPGA is equipped with external SRAM acting as a cache for scientific data packet formation. Check A.2 and A.2.5 in particular for more detailed description of CCB and DPU.

Main CPU functions:

- Scientific data acquisition from CEB to SDRAM memory
- DMA Data transfer from SDRAM to FPGA CCSDS compression engine
- Managing transfer cache data from SRAM (FPGA memory) to ADPMS
- CEB power control (3 MOSFET switch control)
- Management of communication links with CEB
- SPS power control (3 MSOFET switch control)
- Collecting health status from PCU (power) and AEU (actuators)
- Temperature control for a COB unit
- Communication with ADPMS via RS422 IF
- FWA motor control
- FDA motor control
- FLASH and SDRAM memory with EDAC

Main FPGA functions:

- Extending CPU features through SpW (AMBA)
- Provide two PWM signal for COB driver heaters
- Provide two Packet Wire A & B IF - ADPMS
- SPS data acquisition and preprocessing (averaging) from SPS
- SRAM cache with EDAC

Other important components used in DPU are volatile and non-volatile memories (SRAM, SDRAM, FLASH), interface transceivers (LVDS, RS422, CMOS), system clock oscillator and SPS power switch.

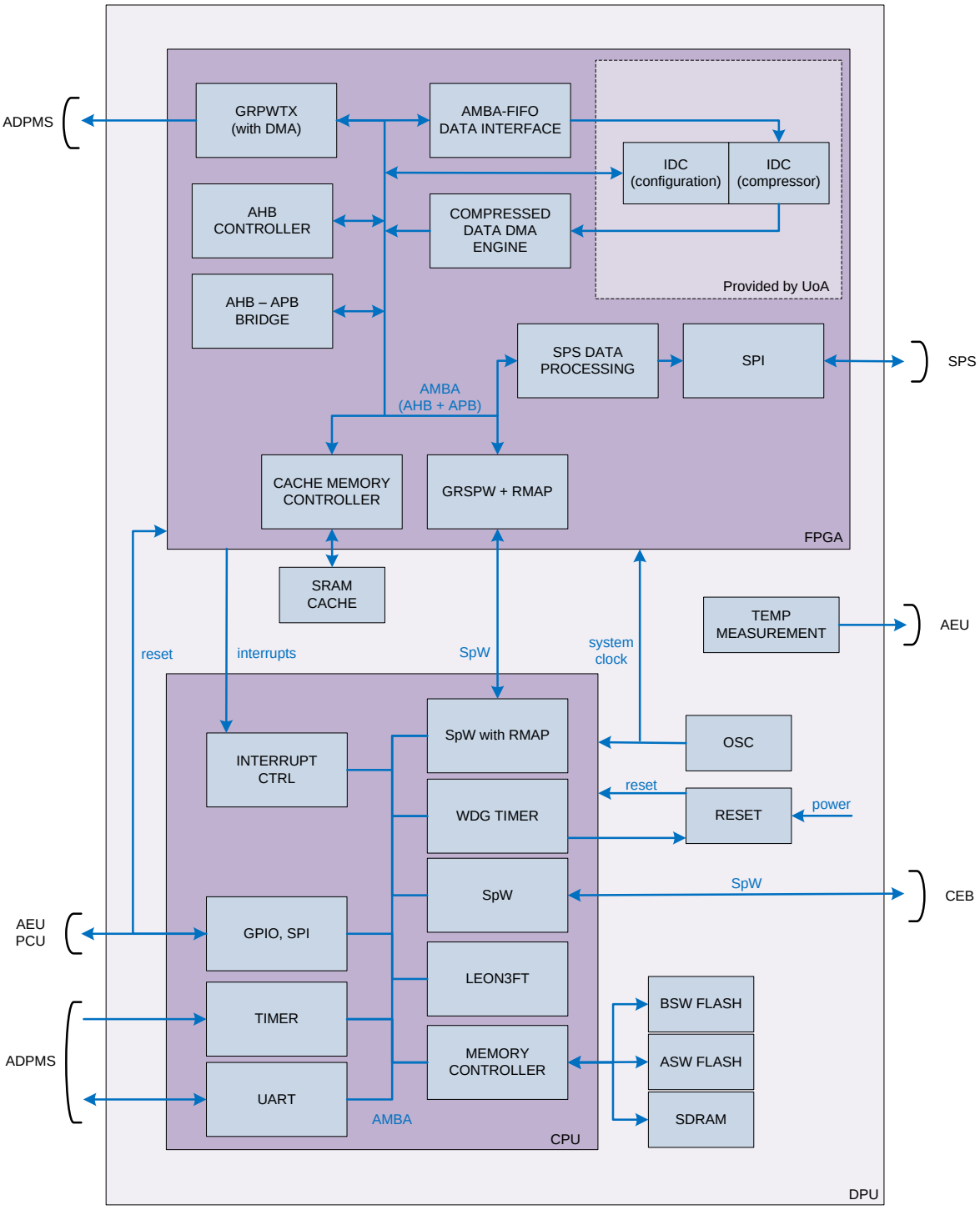

Figure 20 presents a DPU architecture. As mentioned earlier, DPU is built mainly on a CPU and FPGA. CPU and FPGA are connected together through SpaceWire data link which extends CPU functionality. All IP-cores implemented inside FPGA are accessible by CPU by means of Remote Memory Access Protocol (RMAP).

CPU itself, beside two processor cores, of which only one is enabled and used, contains vast amount of peripherals of which some are used and some are disabled through whole mission time. Table 6 summarizes peripherals used in CPU.

#	CPU functional blocks	Used for	Comment
1	LEON3-FT	processing unit	
2	FTMCTRL	access to Flash and SDRAM	
3	INTERRUPT CTRL	collecting asynchronous events	
4	AHB/APB bridge	connecting AHB and APB buses	
5	TIMERS*4 +Watchdog	watchdog timer	
6	TIMERS*4	general purpose timer	
7	GRSPW2+RMAP	SpaceWire link to FPGA	Essentially SpW0 and SpW1 are used, both RMAP capable, so both blocks have the same area.
8	GRSPW2	SpaceWire link to CEB	
9	SPI	HK & control link to SPI and AEU and PCU	
10	APBUART	ADPMS TC/TM link	
11	GRGPREG	general purpose register	

Table 6 CPU functional blocks description

FPGA acts as a co-processor that holds all the functionality that is necessary to perform DPU envisioned activities but was, neither existing in nor fitting the CPU.

#	FPGA functional blocks	Used for	Comment
1	GRSPW + RMAP	SpaceWire link to CPU	
2	AHB CONTROLLER	AHB arbiter and bus multiplexer	
3	AHB / APB BRIDGE	connecting AHB and APB buses	
4	GRPWTX	scientific data TM link to ADPMS	
5	AMBA-FIFO DATA I/F	data entry interface to IDC	custom made / design reuse
6	IDC	compression engine	UoA development
7	COMPRESSED DATA DMA ENGINE	automated data output interface from IDC	custom made / design reuse
8	CACHE MEMORY CONTROLLER	access to external SRAM memory	
9	SPI	SPS control and data link	custom made / design reuse
10	SPS DATA PROCESSING	SPS data averaging	custom made/ design reuse

Table 7 CPU functional blocks description

Relations and connection between CPU and FPGA functional blocks are depicted on

Figure 20

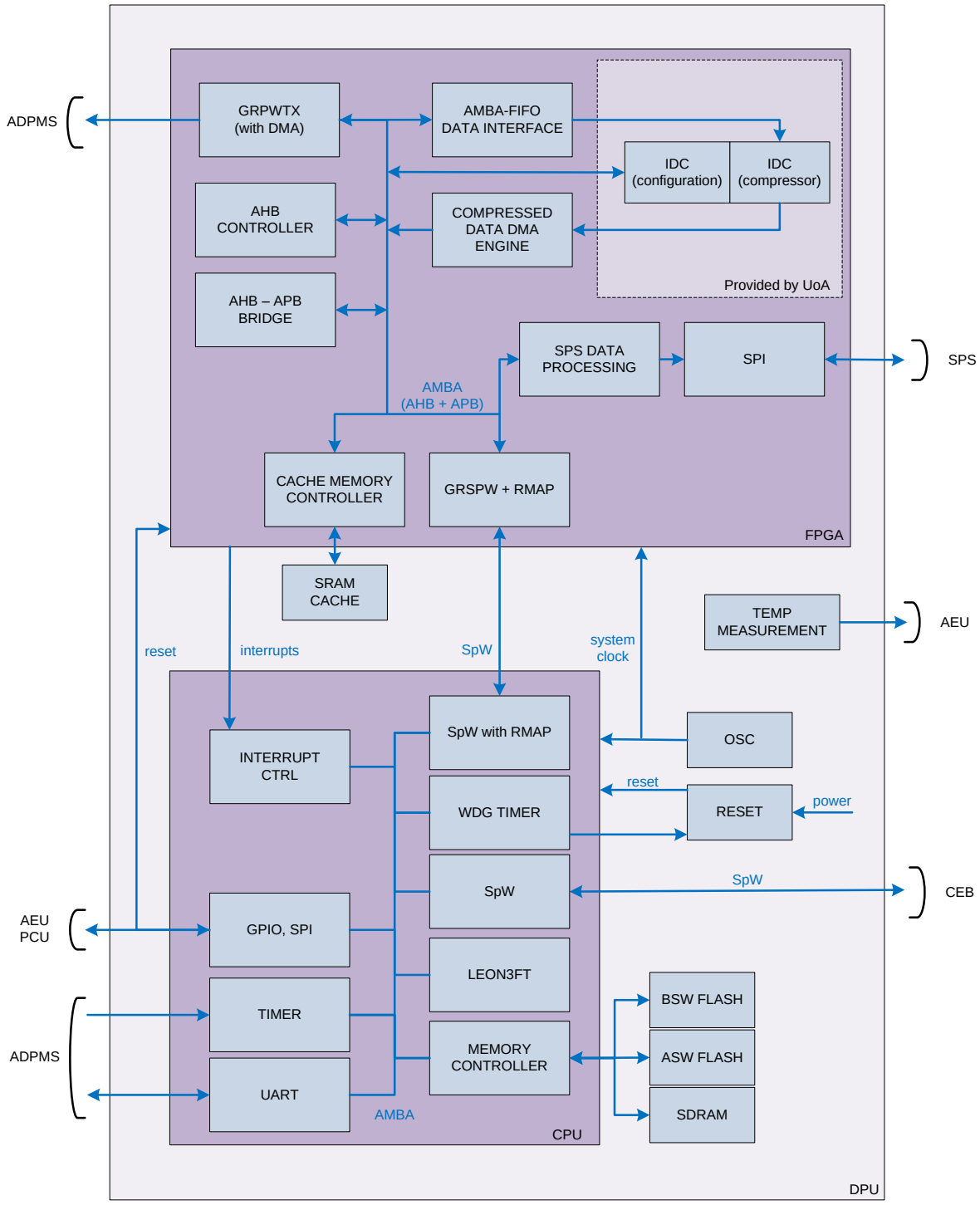

Figure 20 DPU architecture

4.3 AMBA Bus

AMBA stands for the Advanced Microcontroller Bus Architecture and is a specification that defines a generic on-chip communications bus, for use in microcontrollers and embedded system in general.

AMBA is comprised of three distinct buses:

- the Advanced High-performance Bus (AHB)
- the Advanced System Bus (ASB)
- the Advanced Peripheral Bus (APB)

The AMBA AHB is for high-performance, high clock frequency system modules. The AHB acts as the high-performance system backbone bus. AHB supports the efficient connection of processors, on-chip memories, direct memory access engines and off-chip external memory interfaces. Typically used for large and effective data transfers.

- high performance
- pipelined operation
- multiple bus masters
- burst transfers
- split transactions

AMBA APB is optimized for minimal power consumption and reduced interface complexity to support peripheral functions. Typically used for peripheral or IP-core blocks configuration

- latched address and control
- simple interface
- suitable for many peripherals
- have interfaces which are memory-mapped registers
- have no high-bandwidth interfaces
- are accessed under programmed control

AMBA ASB is not used in any of CCB DPU components.

From the perspective of CCB DPU, as mentioned before, AMBA is present in GR712RC and is implemented as a main on-chip bus inside RTAX FPGA. In both cases it is constructed around AHB controller IP-core - AHBCTRL from Gaisler library (and also AHB to APB bridge to provide means to access APB mapped registers). The AMBA AHB controller is a combined AHB arbiter, bus multiplexer and slave decoder according to the AMBA 2.0 standard (depicted in Figure 21). AMBA implemented in GR712RC and FPGA has 32-bit wide data bus.

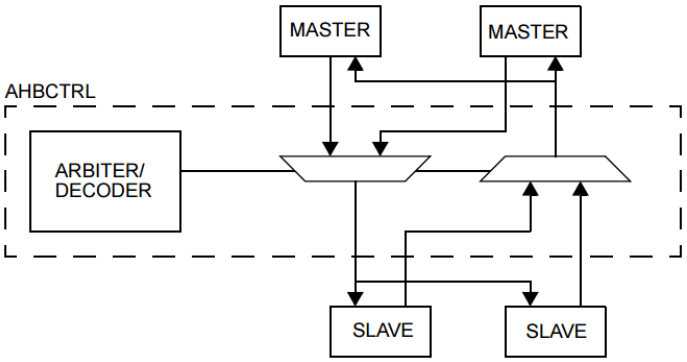

Figure 21 AMBA AHB controller functional block diagram [103]

The AHBCTRL AMBA AHB controller supports two arbitration algorithms: fixed-priority and round-robin. In round-robin mode, access to bus is rotated one step after each AHB transfer. If no master requests the bus, bus ownership remains with last user.

GR712RC AMBA uses round-robin arbitration with exception of Ethernet and MIL-1553 cores which have, fixed, highest priority. However these cores are not used in P3 CCB DPU design so their different behavior does not influence overall system operation.

RTAX implemented AMBA uses round-robin without any priority modification (TBC i.e.: SpaceWire communication controller and SPS Readout engine could have higher priority to ensure low readout latency).

AMBA bus is specified to perform burst transfers of four, eight and sixteen beats. Also single transfers and bursts of undefined length are possible. However, as bursts cannot cross 1kB address boundary, 1kB is effective limit of burst length. Similarly, for fixed-length bursts, it is important that the masters do not attempt to start and incrementing burst which would cause 1KB boundary to be crossed.

From addressing perspective, following bursts are possible:

- Incrementing bursts that access address space in linear manner and the address of each transfer in the burst is simply an increment of the previous address.
- Wrapping bursts, if the start address of the transfer is not aligned to the total number of bytes in the burst (size x beats) then the address of the transfers in the burst will wrap when the boundary is reached.

Summing up, one non-compressed tile will be transferred among functional blocks of GR712RC or FPGA using at least 8 infinite-length bursts (8 kB). Minimum 9 bursts are necessary for tile transfer when tile meta-data is taken into account. Effectiveness of tile

transfers (1kB address alignment) shall be taken into account in tile buffer memory address space design. Moreover, as AMBA bus operations are blocking and several functional and data paths are crossing, mainly on GR712RC AMBA bus, therefore special attention must be paid to proper design of accesses sequence in order to main satisfactory throughput.

4.4 Data transfer operations

CEB data stream

CEB and CCB are connected using SpaceWire interface, set up to operate at 50 Mbps of maximum communication speed. 50 Mbps is 800 tiles per second (raw, uncompressed tile is 0.0625 Mbit or 16 tiles fit 1 Mbit) P3-CSL-RS-14013 CEB-CCB Image data handling user requirements v2.2, however, limits abovementioned value to, on average, 192 tiles per second. The transfer is depicted on Figure 22.

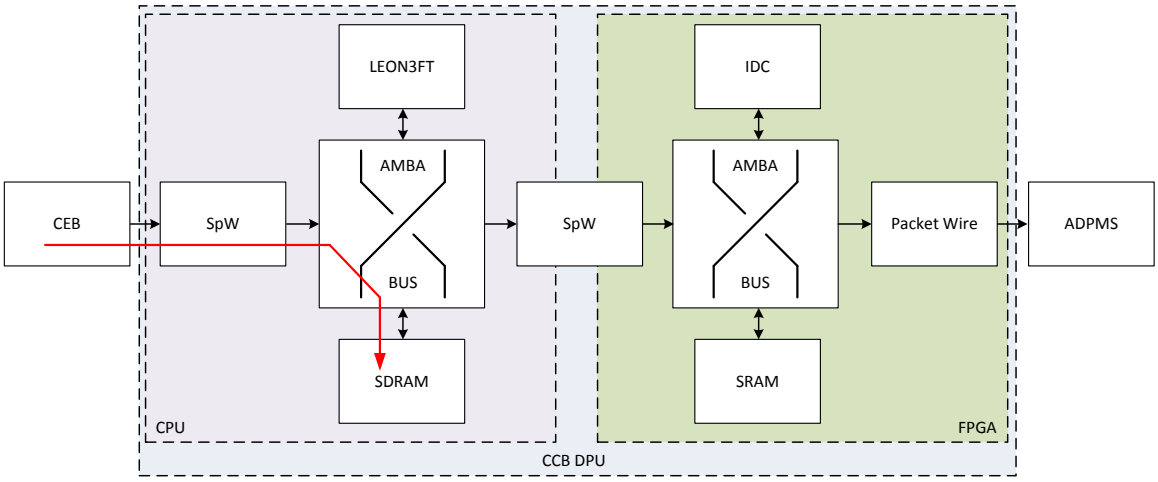

Figure 22 System data transfer: CEB - SpaceWire - SDRAM

4.4.1 GR712RC AMBA and SDRAM

AMBA bus, inside the GR712RC has a maximum throughput of 32 bit * 50 MHz = 1600 Mbps = 25600 tile/s. SDRAM, which is accessible via integrated memory controller, has throughput of 1600 Mbps, so the same as AMBA bus (memory is operated in single data rate, clocked at 50 MHz, data bus is 32 bits + 16 bits for EDAC). The transfer is depicted on Figure 23. SDRAM buffer fits about 50000 tiles (50000 * 8192B ~= 400 MB out of 512MB available in DPU).

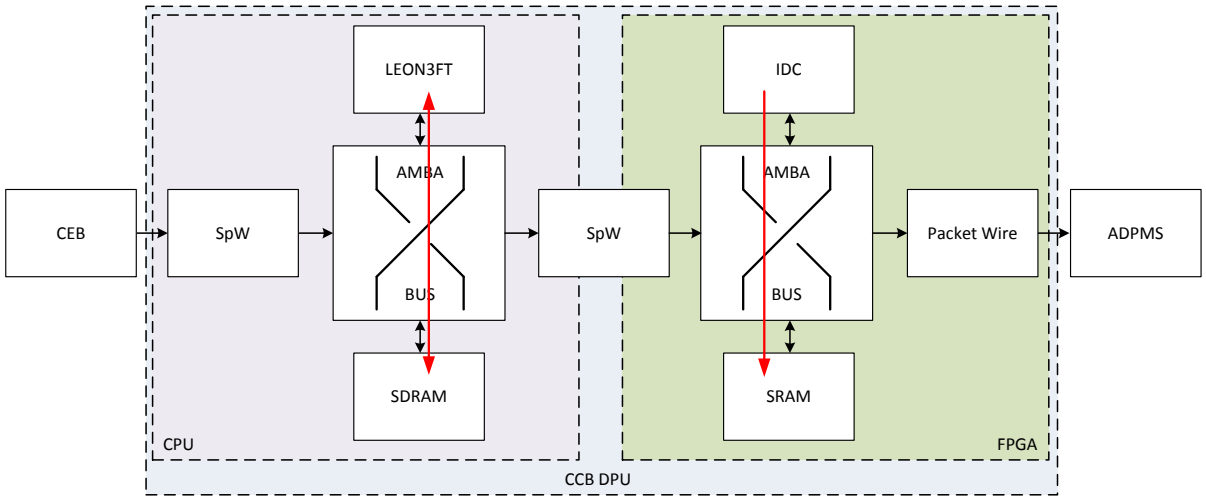

Figure 23 System data transfer: tile processing and IDC - SRAM

4.4.2 FPGA Space Wire to IDC, IDC to SRAM (caching)

SpaceWire link between CPU and FPGA is meant to operate on 50 Mbps or 800 tile/s.

AMBA inside FPGA operates at 32 bit * 25 MHz = 800 Mbps = 12800 tile/s (from SpW to IDC). Providing data to IDC to be compressed and then fetching compressed tile and putting it into SRAM for further packetizing is considered as one, indivisible operation as there are no means to store any data from the moment it is transferred from processors SDRAM, via internal DPU SpaceWire link to FPGA, feed into IDC and (semi-) automatically transferred from IDC output to right place (where future PacketWire packets are formed) in SRAM cache connected to FPGA. The transfers are depicted on Figure 23 and Figure 24.

IDC compression factor varies significantly, depending on compressor configuration. For purpose of simulations compression ratio is assumed to be 3, which is realistic, low, compression ratio.

IDC compressed output data stream is 25MHz, 8 bit wide, which results in 200 Mbps throughput = 9600 tiles/s (after compression, tile is assumed to be 0.0208 Mbit, or 48 tiles for 1Mbit). Output data stream is then transferred via FPGA AMBA (800Mbps or 38400 tiles per second) to SRAM.

SRAM cache transfers are clocked at 25 MHz, and are 8 bit wide which results in 200 Mbps (9600 compressed or 6400 non-compressed tiles per second). SRAM size is 512kB; tile is 8192B / 3 = ~3000B in total SRAM buffer is expected to fit about 150 tiles.

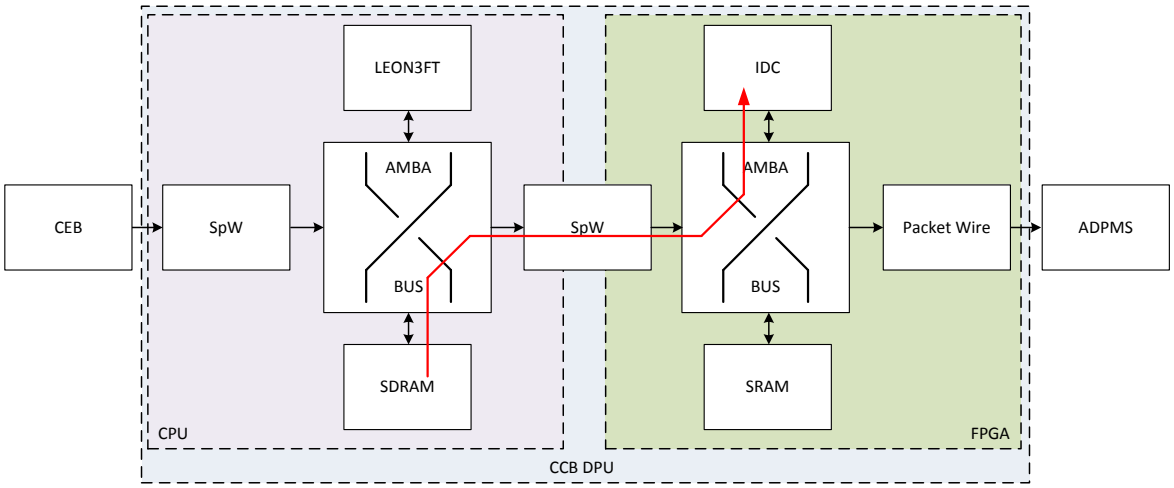

Figure 24 System data transfer: SDRAM - SpaceWire - IDC

To sum up IDC to SRAM transfers, neither SRAM nor FPGA AMBA bus is considered bottleneck, and effective operation throughput is purely defined by IDC output.

If DPU is set to operate in no compression mode, then tiles are send directly to SRAM via internal SpaceWire and AMBA. Limiting throughput of this operation is SRAM caching speed of 25 MHz of 8 –bit wide bus, leading to 200 Mbps or 3200 non-compressed tiles per seconds.

In no-compression variant, SRAM buffer can hold about 50 tiles.

4.4.3 FPGA SRAM to Packet Wire (de-caching):

As there is no Packet Wire buffer that is capable of holding whole tile, SRAM to PW and PW to ADPMS transfers cannot be considered as separate and when treated as integral, their throughput is limited by slower one, the Packet Wire transmission.

SRAM and AMBA throughput estimations are the same as above. PacketWire is a single line serial link, clocked at 50 MHz, but it takes two cycles to send one bit of data, therefore resulting throughput is only 25 Mbps. The transfer is depicted on Figure 25.

4.4.4 ADPMS data stream

PW to ADMPS transfer speed is 25 Mbps, which translates to 1200 compressed tiles per second or about 400 non-compressed tiles per second. The transfer is depicted on Figure 25.

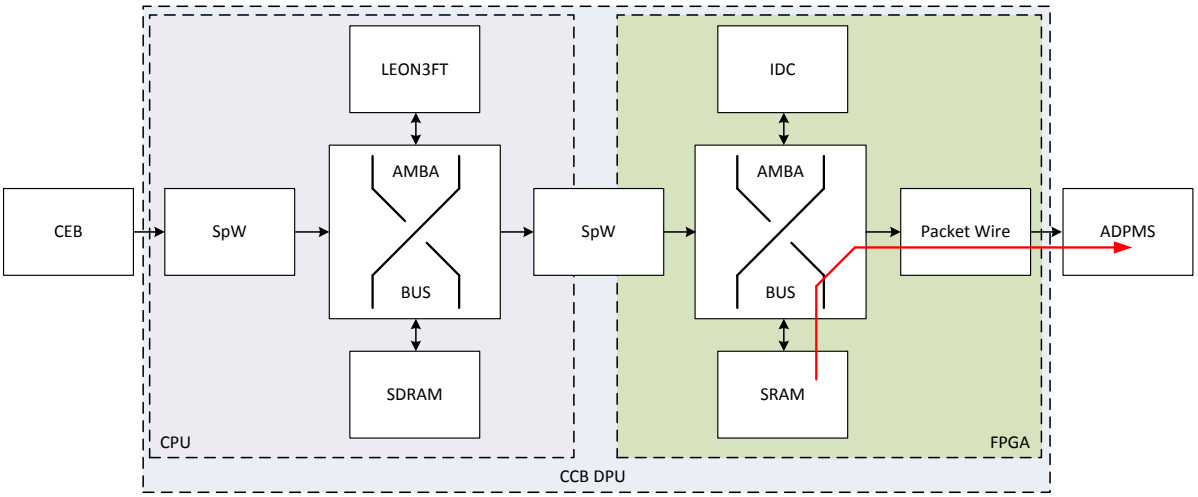

Figure 25 System data transfer: SRAM - PacketWire -ADPMS

4.5 CCB radiation environment

Radiation environment, in context components base selection and, ultimately, device reliability analysis, is best described in terms of particle spectrum (particle flux versus particle energy (LET or MeV)). Particle spectrum is heavily dependent on spacecraft orbit, namely the time spacecraft spends at each altitude (low-Earth orbit (LEO) is dominated by high energy protons trapped by magnetosphere, geostationary and higher orbits are under heavy influence of Galactic Cosmic Rays (GCRs). Also, Sun activity has to be taken into account, as it influences energy and number of energetic particles in van Allen belts (that is where trapped particles reside).

For radiation environment modeling following models are used (Table 8, stated parameters are sufficient to configure standard ESA radiation tool, the Space Environment Information System (SPENVIS) [29], [42], [69], [104]–[106]):

Radiation environment	Models and parameters
Trapped radiation (protons)	AP-8 (solar minimum, flux threshold: 1.00)
Solar particles (flares)	CREME-96 (worst week, ions H to Ni)
Solar particles (average)	ESP-PSYCHIC (total fluence, ions H to Ni, 95% confidence)
GCR	CREME-96 (worst week, ions H to Ni, solar minimum 1977)

Table 8 Radiation models

Sun's activity is relatively easy to predict as it exhibits regular periodicity. PROBA-3 mission is scheduled for launch in 2019 or later. As it can be read from Figure 26, in time of PROBA-3 in-orbit operations Sun will be in its minimum activity period. All the models used for radiation estimation have to be set accordingly.

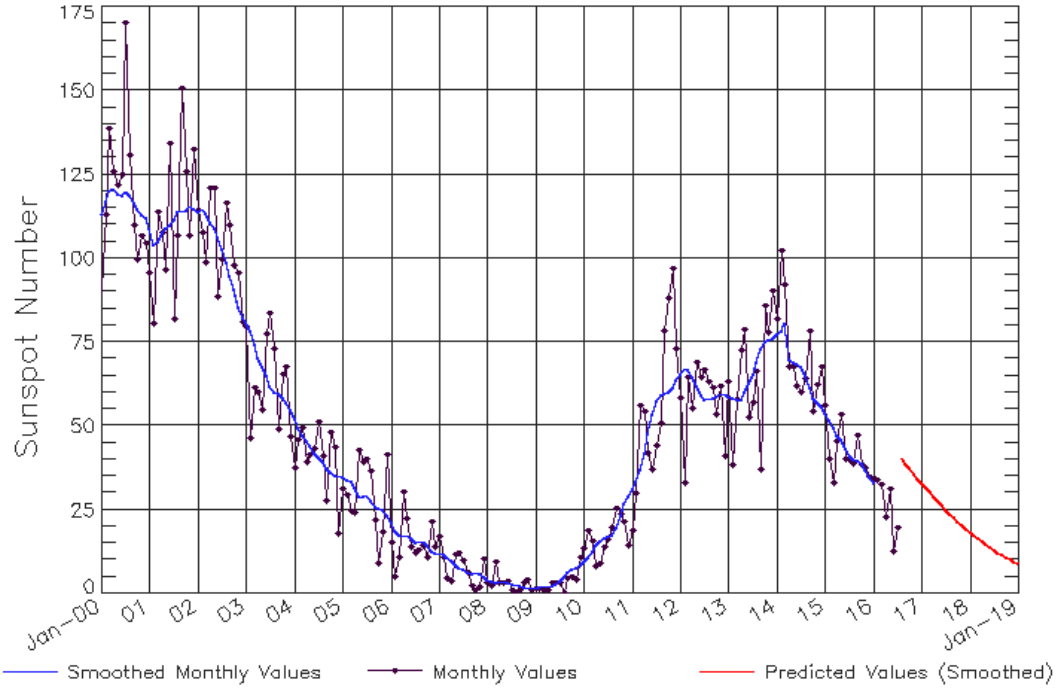

Figure 26 Sunspot number progression [107]

For purpose of presented analysis methodology two orbits are considered: geostationary orbit (GEO) for device susceptibility data calibration (see Appendix B for details) and highly elliptical target orbit (HEO) of PROBA-3 mission. Although PROBA-3 mission is scheduled to last at least two years, for results ease of use, analysis is performed in one mission segment of 1 year duration (all orbits were generated in SPENVIS). This way results are obtained per second, per day and per year, which can easily be extrapolated to 2 or more years without any significant errors – neither orbit nor sun activity will change in such way it will surpass errors or uncertainties introduced by low quality of devices susceptibility information. Orbit defining parameters used in simulations are summed up in Table 9.

Orbit	Parameter	Value
GEO	Longitude [deg]	0
HEO	Avarage perigee [km]	740
	Avarage apogee [km]	60400
	Avarage inclination [deg]	59
	Avarage argument of perigee [deg]	200
	Avarage RAAN [deg]	52

Table 9 GEO and HEO orbits parameters

4.6 CCB performance requirements

There are various requirements regarding expected CCB processing performance. Some of them deal with interfaces, while some of them deal with desired operational capabilities. Moreover, there are observation scenarios, that show examples of ASPIICS Coronagraph is intended to be used, which, in informal way, impose processing constraints on CCB as well.

Following set of requirements has been derived from main CCB Specification document [108], identifying main interface features and parameters including required design margins.

Requirement ID	Requirement Text
CCB-42210	The communication between CCB and CEB shall use the Spacewire standard at minimum 50 Mbits/s.
CCB-42212	Within an acquisition, the CCB shall not block / delay the transmission of science data by the CEB. It shall accept data at the maximum link speed.
CCB-42702	In nominal operation, the CCB shall supply the IDC with the image tiles transmitted by the CEB (after tile selection based on the quality flag), and forward the compressed data to the ADPMS through the packetwire interface
CCB-42704	It shall be possible from ground to command the by-pass of the on-board compression algorithm, so the data transferred to ground are the raw data transmitted by the CEB
CCB-61004	Margins to be applied for the processing load of the electronics shall be: - 40% at System CDR - 25% at System FAR
CCB-74100	The CCB shall provide the following electrical interface with the ADPMS : - 1+1 Switchable Power supply I/F - 1+1 RS422 UART I/F - 2+2 Packetwire I/F - 1+1 Distributed Clock I/F
CCB-74106	The 2+2 Packetwire interfaces shall be as follows : - the CCB processing unit is dual redundant, each redundant half shall have two packetwire interfaces to ADPMS.
CCB-75000	The data dump from the CCB to the spacecraft Mass Memory Module (MMM) shall be based on PacketWire interface. (…)
CCB-75004	The CCB shall allow for a data rate up to 66 Mbits per sec towards the ADPMS

Table 10 Requirements from CCB Requirements Specification [AD-01] that outline DPU processing performance.

Next set of requirements is derived from Image Data Handling User Requirement Document [109] elaborating more on how final system will be operated rather than what it should be built like.

Requirement ID	Requirement Text
URD-42330	The CEB shall output data fast enough to accommodate the worst case imposed by the fastest acquisition cadence (192 tiles/s)
URD-43500	The CCB shall manage the incoming data as they are output by the CEB (the CEB has no flow control)
URD-43510	In case of CCB memory overflow risk (the packet data outflow towards the ADPMS is slow or interrupted and the CCB buffers remaining space is less than one acquisition, or the ADPMS has notified the instrument of the impending closure of the mass memory channel) the CCB software shall abort any ongoing observation cycle and notify ADPMS, while continuing to empty its buffers towards ADPMS when possible.
URD-44200	The ADPMS shall maintain a Packet Wire input channel to the mass memory for the duration of the science part of the orbit (nominally 6 hours)

Table 11 Requirements from User Requirements Document that outline DPU processing performance.

Following requirements deal with features of Packet Wire interface (Table 12):

Requirement location	Requirement Text
chapter 3.2	The PacketWire IF operates at a frequency of 66MHz (i.e. physical transmission speed is 66 Mbps). The average data rate (over a one minute period) shall be maximum 40 Mbps.
table 3-1	PacketWire clock period is defined as min. 15 ns and max. 100 ns

Table 12 Requirements from Packet Wire Interface Control Document [110] that outline DPU processing performance.

CCB system performance is evaluated in context of amount of scientific data generated during observation period, limited time that CCB is on after finished observations as well as internal CCB data path configuration (compression on or off).

Requirement location	Requirement Text
chapter 3.2	An observation period is 6-hour long.
chapter 3.4	There are 6 hypothetical observation programs that serve as a baseline “Full Set Synoptic” observation program “Base Synoptic” observation program “Waves-2s” observation program “Waves-4” observation program “Waves-15s” observation program “CME-Watch” observation program
chapter 5	Data volume per acquisition: $DV = (n/1.5) * (q/4) * 2048 * 2048 * 12 * (1/cr)$ where DV –data volume n – number of exposures per acquisition q – number of detector quadrants covered by acquisition (for full frame q = 4)

	*cr** – compression ratio (*cr* = 3.5 (average) when IDC is on, 1 when IDC is off))
Note: *- for purpose of evaluating worst case scenario, when IDC is on, compression ratio of 3 is used.	

Table 13 Requirements from Observation Scenarios [111]that outline DPU processing performance.

Scientific data volume generated by CEB, for purpose of CCB performance evaluation is convenient to express in terms of data stream valued in number of tiles in observation periods or tiles per second (tile is defined 64 x 64 pixels; pixel coded on 2B; tiles are accompanied by some descriptive metadata), in the following manner:

$$DV_{OP} = \frac{(N * 1024)}{1.5}$$

DV_{OP} – data volume

N – number of exposures collected during observation period

$$DS_{TPS} = \frac{(N * 1024)}{32400}$$

DV_{TPS} – data volume

N – number of exposures collected during observation period

Observation Program	**Full Set Synoptic**	**Base Synoptic**	**Waves-2s**	**Waves-4s**	**Waves-15s**	**CME-Watch**
Number of programs fitting in one Observation Period	106	6	18	18	18	72
Number of Observation Cycles per program	1	4	1	1	1	2
Number of Observation Cycles per Observation Period	106	24	18	18	18	144
Number of tiles in observation period (tiles/period))	1 736 704	1 129 488	2 755 584	2 755 584	1 092 096	4 423 680

Number of tiles in observation period - no overlap (tiles/period))	1 157 803	752 992	1 837 056	1 837 056	728 064	2 949 120
tiles/second (avg)	53.60	34.86	85.05	85.05	33.71	136.53
Amount of data received in observation period [MB]	9 045.33	5 882.75	14 352.00	14 352.00	5 688.00	23 040.00
Amount of data received in observation period [GB]	8.83	5.74	14.02	14.02	5.55	22.50

Table 14 Data volume (bytes and tiles) generated in different observation scenarios. Based on [111].

There are several example observation scenarios that give an idea of how ASPIICS (the Coronagraph Instrument, CI) and CCB, is going to be operated and what would be the resulting data volume per observation period. Observation period is a set of observation programs which happen during one orbit, when CI is switched on. During that time CEB generates some volume of raw scientific data which is different for different observation programs. Table 14 sums up information available in [RD-05]. Data volumes are estimated on examples of observation periods filled with one observation program. Row "Number of tiles in observation period" in Table 14 estimates this volume in tiles per observation period, next row "Number of tiles in observation period - no overlap" takes into account fact that each exposure time acquisition overlaps other exposures, which reduces number of usable tiles by factor of 1.5. Row "tiles/second (avg)" takes previous value and divides it by 21600 s/period, based on assumption that CCB is on from the start of first observation to power switch off for exactly 6 hours. This is the time limit that is imposed by CI operational requirements. Rows expressing the volume in MBytes and GBytes are for readers information.

5 New approach to avionics modeling

This chapter focuses on:

- defining the problem
- outlining the assumptions and scope of discussion
- stating the hypothesis and supporting goals
- outlining to reader how proposed methodology is going to be implemented in further parts of dissertation

5.1 Problem definition and proposed solution

Each project (and product) in aerospace domain starts as a space of possible concepts that is explored by system engineers (and stakeholders) by use of iterative needs analysis, trade-offs, incremental detailing, re-use of existing knowledge, scenarios, data and prototypes. In other words existing approach to design and build complex equipment already relies on models. The systems engineering dependence on models is increasing and it seems it will constantly grow. Main problem that has to be dealt with is that each and every discipline and aspect of system design needs its own mathematical and engineering tools which adds additional burden on design team. There is very little incentive to build holistic models or evaluate flexible tools that enable the engineers to explore the design space in different domains, like in case of presented dissertation – reliability and processing performance. The reason for that is prosaic: system engineering team are generally paid to perform system engineering on specific projects rather than evaluate or develop new tools or methodologies to support their work.

Despite abovementioned difficulties, main motivation behind presented dissertation is to show that there are already exiting analytical tools and modeling methodologies, that have not been used previously in aerospace domain, especially for modelling space instrumentation, but when applied, have great potential to support system analysis and show clear benefits to user.

Proposed solution to outlined problem is to accept a generic model combining several levels of system complexity, its low level static behavior and attributed with high level dynamic behavior influenced by its architecture, and express it using Petri Net. As it can be noticed, solution is not bounded by any limitations with regard which aspect of system is modeled. It's a solution that is also tool agnostic, as functional blocks or components characteristics needs to be calculated or estimated in traditional way and express in simple manner of a deterministic or stochastic function with known distribution. What is key novely indicator is

the Petri Net that bounds all this information altogether, allowing to model (to simulate or to analyze) equipment on system level, disregarding the modeling goal. The idea fits reliability analysis as well as performance studies as well as other applications (i.e.: resource utilization modeling, control algorithms, scheduling of software tasks, and do on).

5.2 Hypothesis and goals

Taking into account problem defined as well as proposed solution, explained in previous section, following **hypothesis** is stated:

> It is possible to design and evaluate a dependable on-board control and data processing unit for unmanned spacecraft utilizing system model associating environmental and functional information in Petri Net.

Two partial aims are established to support the proof and demonstrate the practical applications of proposed system evaluation methodology:

Goal 1 Coronagraph Control Box performance shall be evaluated using proposed methodology by establishing and simulation of Petri Net system performance model.

Goal 2 Coronagraph Control Box reliability shall be evaluated using proposed methodology by establishing and simulation Petri Net system reliability model.

By achieving these two goals, each of them linked to different field of engineering and expertise, hypothesis shall be confirmed, also showing methodological advantages of proposed solution:

- Petri Nets are flexible tool allowing cross-domain modeling of complex systems
- Holistic models emphasizing different aspects and features of evaluated system, tackling its complexity, can be quickly created using Petri Nets

Presented dissertation is of theoretical nature, although it is closely related to real-life engineering activities and relies heavily on computer simulations.

5.3 Modeling methodology implementation concept

Model is a description of a system being prototyped. It is usually simplified version of more complex ontological structure, focused on extracting some crucial aspects, parameters, characteristics and interaction among them. Model sheds light on interdependence of

interesting or important system features, keeping the rest in shadow. Model, in order to be useful for systems engineering, has to tackle, both, structure and behavior, present on all complexity levels involved in analysis in the process. As a consequence of employing structure and behavior as vital and inseparable aspect of system under investigation, model has to consist also of two parts – static and dynamic.

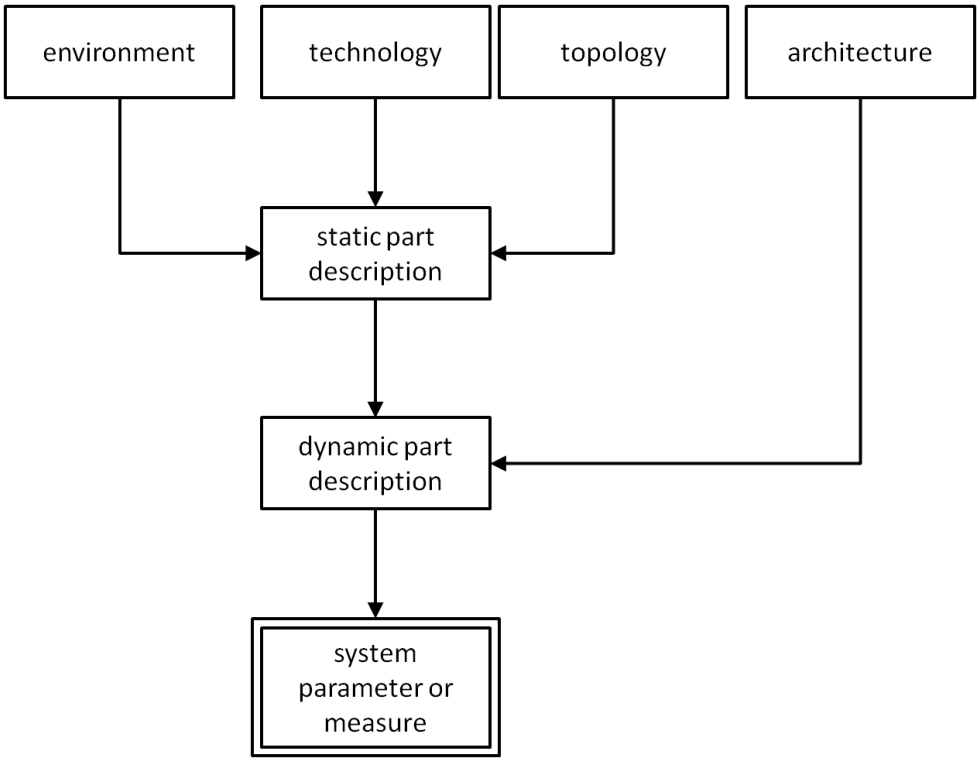

Figure 27 Generic system modeling methodology

From formal point of view, model of a system can be described as a function $\boldsymbol{h}$, estimating system functionality to a required accuracy. Model takes as an input, vector u, and delivers result of output vector y. Above statements can be expressed in following equation:

$$y = \boldsymbol{h}(u)$$

Static part ($\boldsymbol{h_s}$) of a model is focused on system internal structure, which depends only on input at the present time *t*:

$$y(t) = \boldsymbol{h_s}\big(u(t)\big)$$

Structure of a system is a main contributor to static part of a model. It is based on lower complexity constituents, so environment, technology and topology will be key influences shaping static model part Figure 27

In turn, dynamic part ($\boldsymbol{h_d}$) of a model is focused on system internal behavior, which depends on input at the present time and current system state which contains information about past:

$$y(t) = \boldsymbol{h_d}(\boldsymbol{h_s}, u(t), t)$$

Main contributor to dynamic part of system model is architecture which defines behavior of lower level entities (Figure 27).

Having in mind generic model for estimating selected system parameters measures, similar approach could be used to facilitate system engineering activities in building avionics system for spacecraft. As it has been already introduced, there are many interesting aspects of such systems that require modeling, the dependability (reliability, performance) being one of interest in context of this dissertations.

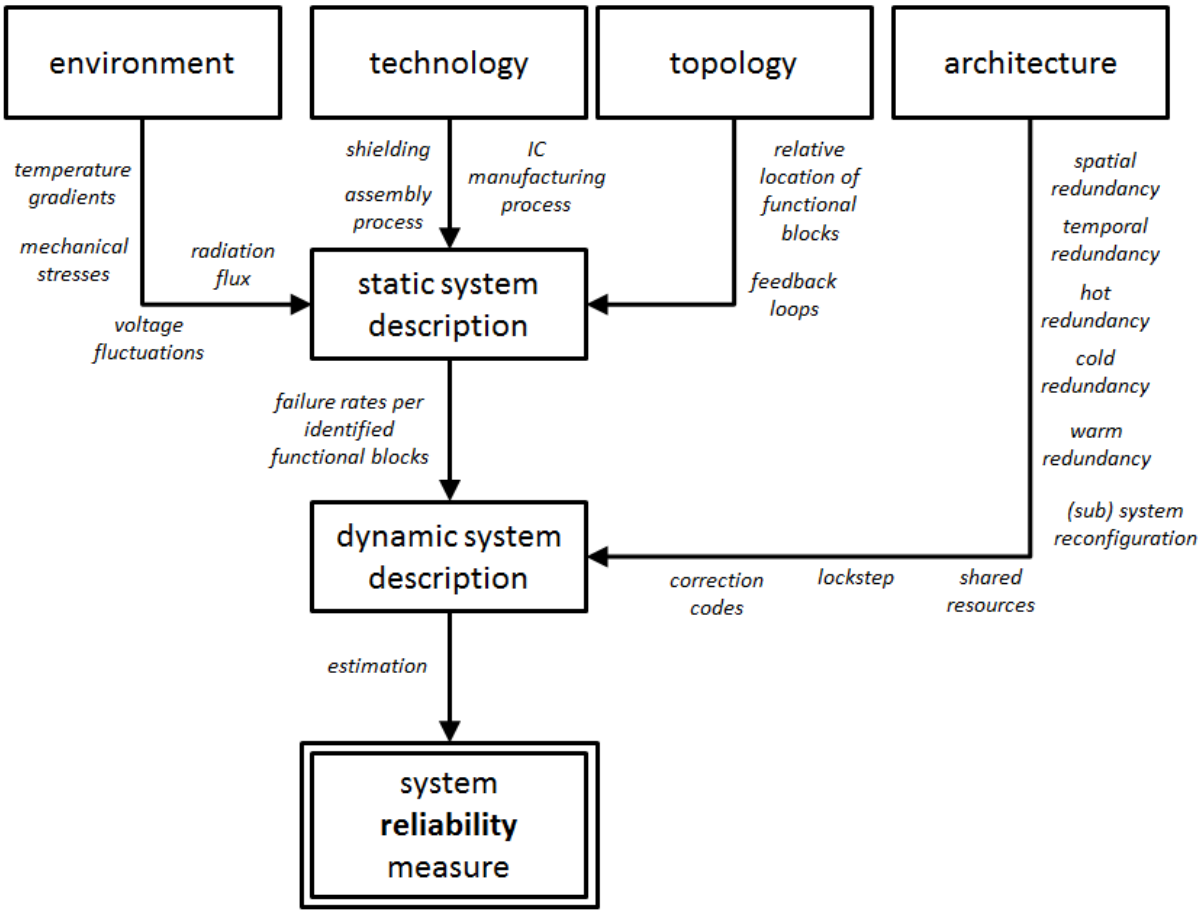

Figure 28 Generic system reliability modeling methodology

When generic system model, as proposed in this chapter hereinbefore, is being embedded in for example, reliability analysis context, each of contributors (model input) becomes a real, measurable, quantifiable parameter that is adjusted in course of system analysis. Similarly,

model output is also expected to be a measure quantitatively or qualitatively evaluating analyzed system from model perspective.

Generic system reliability model is shown on Figure 28. As it has been outlined before and perhaps it already appeals to reader's intuition, while static part of model operates on more atomic, basic elements that exhibit given characteristics, dynamic part of model operates on more complex entities and tackles functional and temporal dependencies among them. Therefore, static part of model is fed with environmental information like radiation flux and expected energies of particles interacting with semiconductor lattice, mechanical and thermal stresses affecting bonding which can be used to estimate part failure probability. In composite devices like integrated circuits also information on technology aspects is necessary to estimate influence of manufacturing process to susceptibility to destructive events like latch-ups or packaging process influence on effective shielding, reducing radiation effects. At this level also a topology of composite device plays role, as dependability influencer, as physical, relative location of functional blocks is important, i.e. blocks in close proximity can be a subject to multiple (simultaneous) bit upsets. Modeling in its static part, using whole low level, available and desired, structural information, has to result in some auxiliary measures that serve as an input for dynamic part of modeling. In turn, dynamic part of model, incorporating elementary failure / repair rates of system constituents, can evaluate how considered architecture and its internal dependencies and functional modes, impact global system reliability measure. Also, various redundancies, coding schemes, operation modes (i.e.: lockstep) and accesses to shared resources shall be described. At this stage, also detailed description of fault detection, isolation and recovery methodology shall be incorporated into model.

Dynamic part of model, and by the way, whole model, shall return a system reliability attribute measure that allows to quantitatively or qualitatively judge suitability of analyzed system architecture and components pool. What is even more profound, reliability attribute measure estimate, in the likely event of building model out of incomplete environmental, technological or even architectural information, shall allow straightforward comparison and trade-off analysis of set of architectural variants.

Very similar approach shall be undertaken for other measures, like, for example performance. Such generic system performance model is shown on Figure 29. Like in previously discussed reliability model, here, static part involves low level or external contributors. Environmental input (like system scientific data input stream, throughput and responsiveness of satellite on-

board mass memory modules) is the boundary, external factor, that is independent on design decisions. Thankfully, technology (processing capabilities of logic devices, processor cores, co-processors) and topology (caches, buffers, device-device interfaces, communication links) are, so these are the items that can be tweaked according to current needs. Then, there is the dynamic system description incorporating the high-level information on how all components and functional blocks are orchestrated altogether in various operational scenarios, software drivers, under different operation systems and applications. All in all, input streams and output streams, processing blocks, storage elements and concept of device operation providing insight on how each of these blocks influence other's operation, integrated in a model that can be simulated shall yield a simple numeric answer – whether the system in design estimated processing power meets user needs and requirements, and if yes, with what margin.

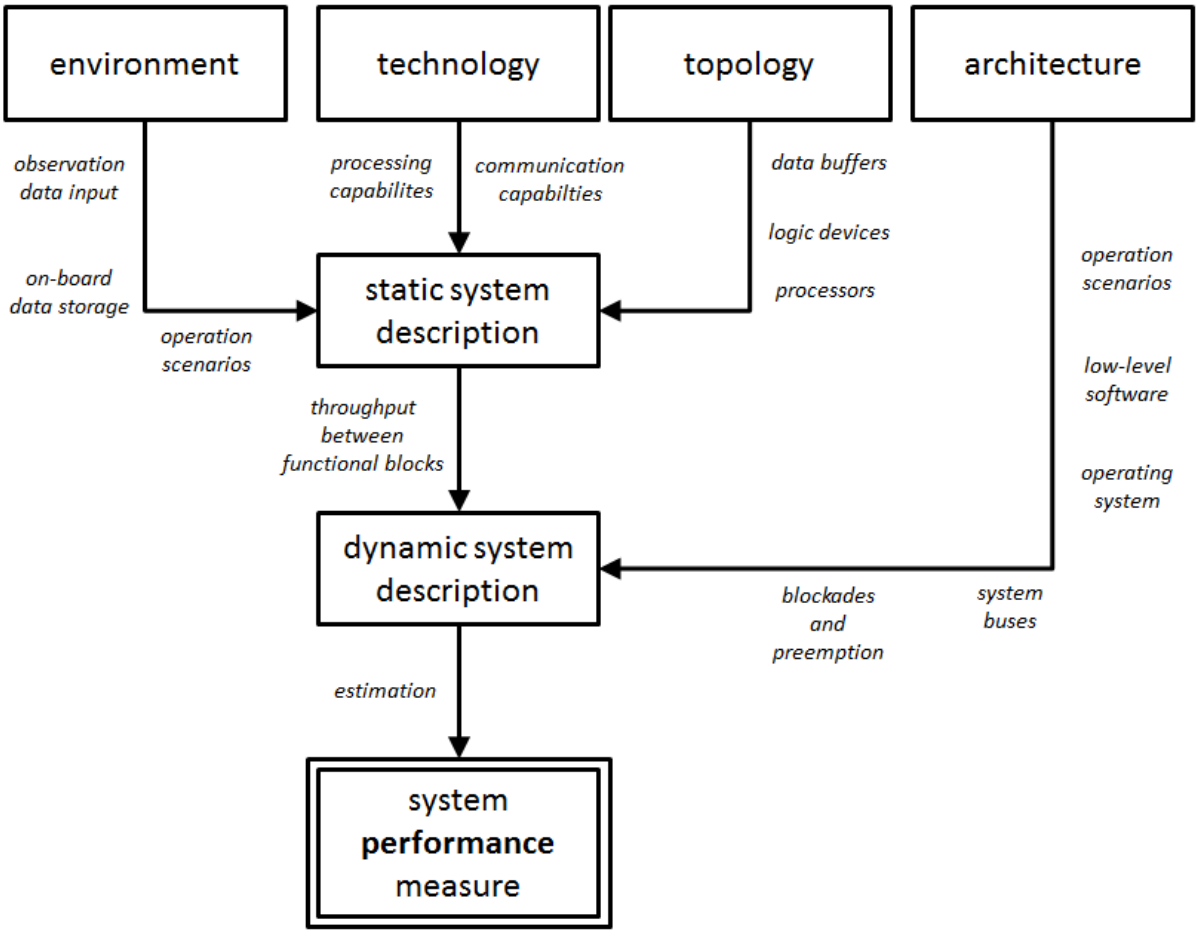

Figure 29 Generic system performance modeling methodology

This section intended to familiarize reader with abstract and generic description of proposed modeling methodology. In practice it boils down to several steps that allow to build a model

of system emphasizing this system measure which at given moment is necessary in equipment development process. Steps for proposed model creation are shown in Table 15.

Step #	Action	Comment
1	identify component's attributes	for reliability measure most preferable is cross-section curve Weibull fit parameters or at least radiation test results (LETs vs numbers of SEEs) for own approximation. For performance measure most convenient is the component internal structure, throughputs among and processing capabilities of each of functional blocks, blocking operations.
2	select blocks of interest	for reliability measure functional blocks or hardware blocks, localized within component, that are subject to SEE and whose faults affects CCB / DPU operation, for performance measure functional blocks that for control or processing chains shall be identified
3	evaluate environmental, technological and topological condition on block of interest (static part)	take into account external and static factors: orbit and mission radiation environment causes block of interest to exhibit some fault rate, similarly the manner in which scientific data is generated and received by CCB (averaged stream, bursts, bursts correlated with on-board events) influences the block of interest processing or throughput rate
4	obtain intermediate measure	calculate the fault or throughput rate for each of block of interest
6	incorporate evaluated blocks of interest into Petri nets incorporating all available architectural information	analyze the CCB system from perspective of functional chains, decide which blocks of interest form such chain, how software details will affect the behavior of block of interest
7	assign interpretation to each token, place or transition in Petri Net	ensure that tokens of different interpretation don't mix (unless colored Petri Nets are used)
8	select which places of Petri net shall be observed to obtain system measure under investigation	for example, count the number of token received in target place in unit of time to simulate the system throughput or simulate the probability (expected value of tokens) in place interpreted as operation or failure of given functional block or subassembly
9	simulate the Petri Net	export and plot the resulting statistics

Table 15 Proposed Petri Net construction and simulation process

Methodology recounted in this chapter and detailed in Table 15 is exactly followed in next chapters to evaluate reliability and performance measures of CCB and, in particular, DPU. Moreover to ensure that novel methodology yields correct or at least reasonable, plausible, results each analysis starts with use of classic system evaluation methodologies.

6 CCB DPU performance analysis

It may already appear to the reader, scientific data fed to CCB is received in GR712RC processor, stored in SDRAM memory, then fed to FPGA for compression, if necessary, and transferred by PacketWire interface to ADPMS Mass Memory Module.

On GR712RC side, process of scientific data reception, storage, (limited) evaluation and further upload is rather simple as everything is governed by LEON-3FT CPU which configures DMA transfers of its peripheral blocks, residing on common AMBA bus.

FPGA, although build around the same philosophy as GR712RC – an AMBA bus and set of IP-cores providing required functionalities, process of data processing and packetizing is somewhat more complex, as all the operations have to be remotely configured by CPU using SpaceWire RMAP protocol. Therefore, although some bus transactions inside in FPGA, like DMA transfers, happen automatically, they have to be configured by CPU beforehand. CPU remain in full control of dataflow both, in the GR712RC processor and RTAX 2000 FPGA. In addition to what has been described for scientific dataflow, FPGA reports back to CPU its internal events also related to communication and SPS operation, via 3 interrupt lines: from Packet Wire, from IDC (including IP cores for data feed in and feed out mechanisms implemented by CBK) and from SPS Engine. FPGA is also equipped with its own, external SRAM cache, for compressed scientific data buffering and data packet formatting before sending them to ADPMS.

Brief glimpse on operation stages as explained in greater detail in chapter 4 (4.4 especially) points to important finding that each operation, whether it is transfer configuration or tile transfer, involves AMBA bus. Due to AMBA bus nature, each AMBA operation blocks any other operation from being executed. This is true for both AMBA buses, the one inside GR712RC and one inside FPGA, and, all in all, significantly affects resulting system performance.

6.1 Standard performance analysis of CCB DPU

Pipeline model of DPU, is the most simplistic one. Main underlying assumption is that the tiles are transferred from one place of buffering to the other, immediately, with maximum feasible rate. No functional dependencies or blocking operations are taken into account. Results obtained this way, are theoretical maximum performance limit that is achievable in given hardware configuration and operation constraints. Real implementation and related more realistic models, analyzed in subsequent chapter, can only be worse.

In presented simplistic DPU pipeline model, input data stream is feed through "pipeline" or, in other words, data transfer channel. Data transfer channels consists of all the sub-channels through which data is transferred – various communication links and data exchange mediums, described in detail in chapter 4.4. Each of these sub-links, has its own throughput, which is utilized, in some part, for purpose of transferring tiles from CEB to ADPMS, with or without internal CCB processing. At every subsequent pipeline part, channel link utilization is estimated – sum of entries in column (green area) tells extent to which given link is utilized for tile transfer. For example, in Table 16 ,in GR712RC AHB column, it is visible that main GR712RC bus has to handle two stream, each of nominal tile flux, as tiles are first put into SDRAM and then readout from the SDRAM for further transmission.

If estimated utilization is higher than the sub-channels throughput then data stream has to be limited to allowable maximum, and residual data stream has to be buffered for further feed when main data stream dries out. If measured utilization is lower than maximum sub-link throughput, then sub-link can be utilized to transfer the data stream(s) without any obstacles.

Pipeline models of DPU performing compression in IDC (Figure 30, IDC compression-on configuration) and not compressing scientific data in IDC (Figure 31, compression-off configuration) seem to utilize physical DPU transfer capabilities only in part, leaving decent design margins. Channel utilization is obtained in following equation:

$$utilization(CL_i) = 12.58 * \sum_{transfers \in i} DS_{flux}$$

where:

12.58 – in Mbps, influx of tiles from CEB (192 tiles/s * 8192 B/tile = 12.58 Mbps)

CL_i – communication link *i* (*i* can be SpW, GR712RC AHB, and so on), expressed in [Mbps]

DS_{flux} – data stream flux (i.e.: flux of tiles transferred from SDRAM to Space Wire controller #1 through GR712RC AMBA AHB bus.

For readers convenience link utilization is also expressed in fraction of maximum link throughput. This gives an easy overview of possible bottlenecks and limitations of modeled system.

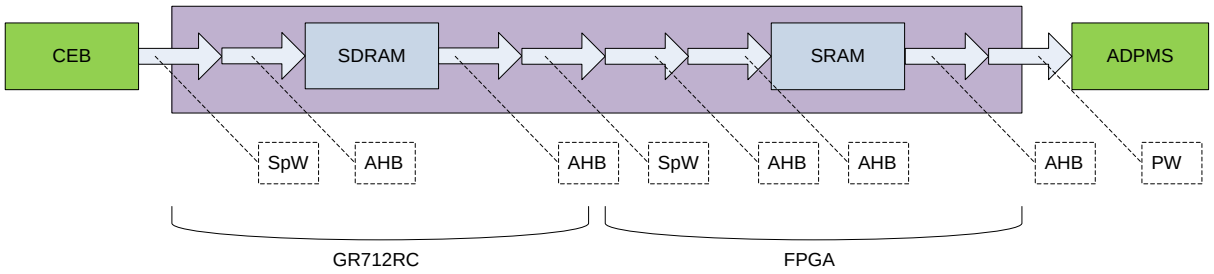

Figure 30 DPU pipeline model - compression on

			Communication link (CL_i):				
		***i*:**	**SpW**	**CPU AHB**	**SpW**	**FPGA AHB**	**PW**
		Link throughput [Mbps]:	50	1600	50	200	25
Transfer data stream flux DS_{flux}	GR712RC	CEB to SpW controller #0	1				
		SpW controller #0 to SDRAM		1			
		SDRAM to SpW controller #1		1			
		SpW controller #1 to FPGA SpW controller			1		
	FPGA	FPGA SpW controller to SRAM				0.33	
		SRAM cache to PW controller				0.33	
		PW controller to ADPMS					0.33
		Link utilization [Mbps]:	12.58	25.17	12.58	20.09	4.15
		Link utilization ratio [%]:	25.17%	1.57%	25.17%	10.44%	16.61%

Table 16 DPU pipeline dataflow model analysis – compression-on configuration

In DPU model analysis with compression on (Table 16) it can be noticed that stream of data flowing out of IDC, travelling through link of FPGA AMBA AHB to SRAM and further to PacketWire is only a third of stream entering the compressor – this is numerical way of showing an averaged effect of compression.

In case the compression is off (Table 17), data stream is not feed into IDC but directed to SRAM cache and then it follows the usual path to the ADPMS. No data stream flux is modified at any of the links present in "pipeline" in this case.

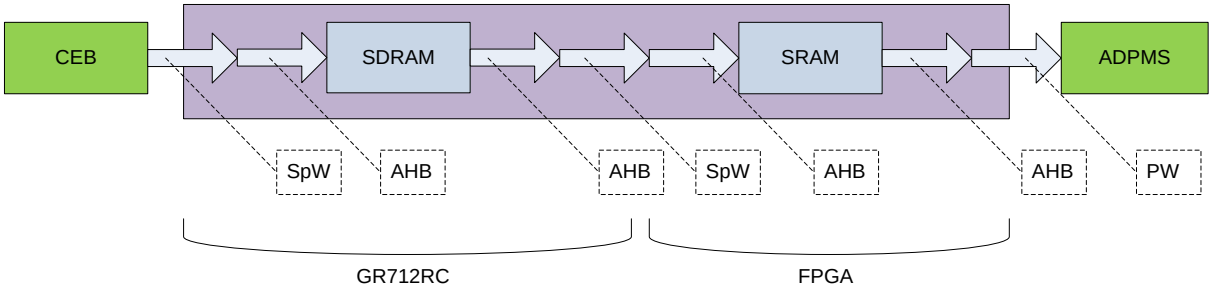

Figure 31 DPU pipeline model - compression off

			Communication link (CL_i):				
		***i*:**	**SpW**	**CPU AHB**	**SpW**	**FPGA AHB**	**PW**
		Link throughput [Mbps]:	50	1600	50	200	25
Transfer data stream flux DS_{flux}	GR712RC	CEB to SpW controller #0	1				
		SpW controller #0 to SDRAM		1			
		SDRAM to SpW controller #1		1			
		SpW controller #1 to FPGA SpW controller			1		
	FPGA	FPGA SpW controller to SRAM				1	
		SRAM cache to PW controller				1	
		PW controller to ADPMS					1
		Link utilization [Mbps]:	12.58	25.17	12.58	25.17	12.58
		Link utilization ratio [%]:	25.17%	1.57%	25.17%	12.58%	50.33%

Table 17 DPU pipeline dataflow model analysis – compression-off configuration

6.2 Dynamic performance analysis of CCB DPU

In the proposed dynamic model, pipeline model is treated as a baseline and it is modified by dynamic operations, like bus blocking by the processor, processor control over FPGA data flow and buffer size limits.

6.2.1 Model check

On Figure 32 is presented a Petri net model of DPU, which is intended to closely resemble an idealized pipeline model from previous chapter (6.1). Model consist of several places and transitions between them and is, in fact, conceptually, very similar to what is presented on Figure 30 or Figure 31. Main, difference is that pipeline model operated in terms of throughput, and conveniently allowed to compare fluxes of data. Here, in Petri nets realm, it is much more convenient to operate in terms of tiles – simply represented as tokens stored in places representing sources, sinks and buffers (like P_SDRAM and P_SRAM, P_CEB,

P_ADPMS) or tiles in transfer (like P_SPW[0..2], P_IDC, P_PW). Transition, when fired, takes a token from originating place and puts it into target place(s). Transition, namely its firing rate, which beside topology, is a second important factor that creates link between model and reality. Firing ratio has to be set as such, that resulting tile throughput matches the real transfer capabilities of physical system.

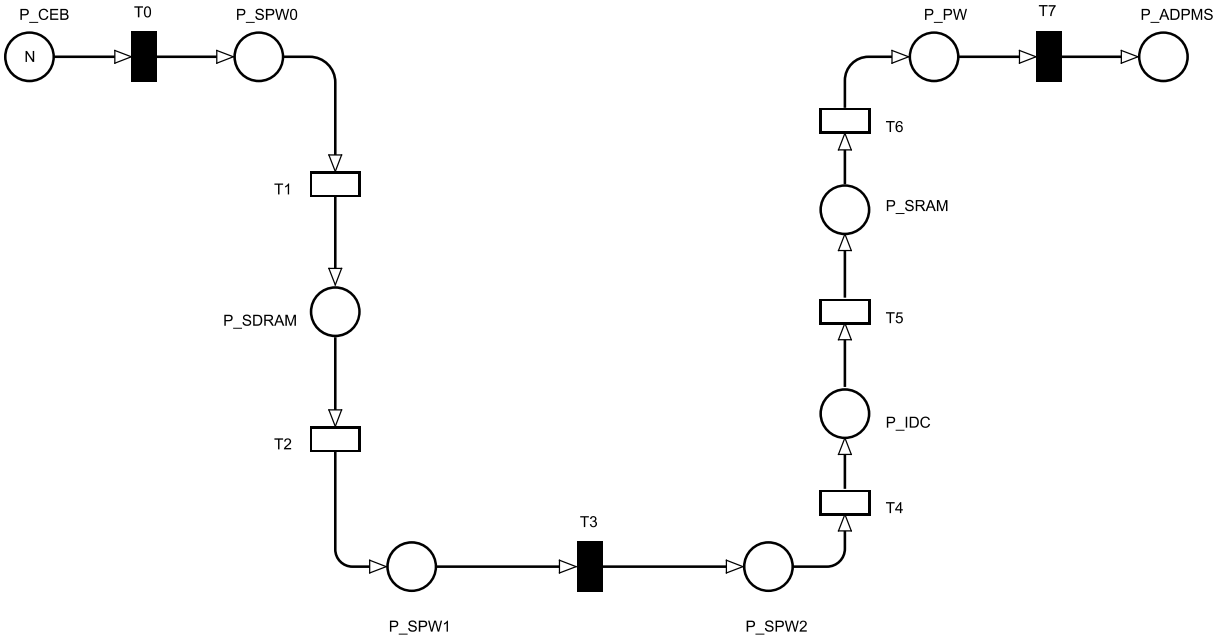

Figure 32 DPU pipeline model expressed as Petri net.

For a reader understanding pipelined model, presented Petri net implementation shall be immediately appealing. Tokens (or scientific data tiles, as in current interpretation) are moved among subsequent places (direction determined by arcs, i.e.: P_CEB as a scientific data source, P_SPW0 as a CCB-CEB Space Wire interface internal buffer, P_SDRAM as SDRAM buffer) at pace defined by firing delays (also known as mean time to fire). N, number of tokens / tiles in P_CEB, is user defined according to specific simulation requirements and tool capabilities (TimeNet is limited to maximum 2^{16} tokens in simulation). Hence, N could be 10 000 as well as 2 949 120 (maximum number of tiles per observation period in evaluated observation scenarios), depending on what features of model simulation are easier to emphasize. In model from Figure 32 two types of transitions are used. Solid black rectangles denote deterministic transitions that fire periodically (of course, if tokens are available in source place). Empty white rectangles denote transitions with stochastic firing, described by exponential probability density function. Average (mean) time to firing is interpreted as function's λ factor. Transitions parameters (valid for all DPU performance models from Figure 32, Figure 33, Figure 34) are summed in Table 18.

Transition	Type	Throughput [tiles/s]	Mean time to fire [s]
T0	deterministic	192	5.21E-03
T1	exponential	25600	3.91E-05
T2	exponential	25600	3.91E-05
T3	deterministic	800	1.25E-03
T4	exponential	12800	7.81E-05
T5	exponential	9600	1.04E-04
T6	exponential	9600	1.04E-04
T7	deterministic	1200	8.33E-04

Table 18 Petri net transitions firing parameters - compression-on configuration.

Performing transient simulation on abovementioned model for N=10000 yields following results (Table 19):

N [tiles]	time to transfer from P_CEB to P_ADMPS [s]
10 000	52.15
2 949 120	15379

Table 19 Simulation results - tile transfer performance (for model from Figure 32).

Obtained result, about 15379 seconds for transfer (with compression) of all scientific data tiles generated during worst case scenario (in terms of volume of performed measurements in CME-Watch) is close to value estimated using simplified pipeline method. This confirms the validity of model, at presented level of sophistication.

6.2.2 Increasing level of the detail

Second of presented models (Figure 33) is updated with real buffering capabilities, when compared to first one, simple, pipeline like petri net. This feature makes the model more real, as for example, technically speaking, SpaceWire controller cannot buffer more than one tile (in petri net context, P_SPW0 cannot contain more than one token). Moreover, SpaceWire controller cannot buffer even one whole tile, so presence of token in P_SPWi (i = {1,2,3}) shall be interpreted as tile in transfer rather than tile in storage. Nevertheless, limits on token holding capabilities of places can be implemented by use of transition feedback, inhibitor arcs. To take as an example, there is a feed backing inhibitor arc from P_SPW0 to T0. As there is no number denoting number of tokens in originating place that starts inhibiting action, it default to 1. Therefore if 1 token is present in P_SPW0 then T0 remains inactive until number of tokens in P_SPW0 drops to 0. On the other hand, T1 is inhibited by arc with SDRAMmax label and T5 is inhibited by arc with SRAMmax label. In full scale DPU model, mentioned labels shall have values as expressed in Table 20 and Table 21.

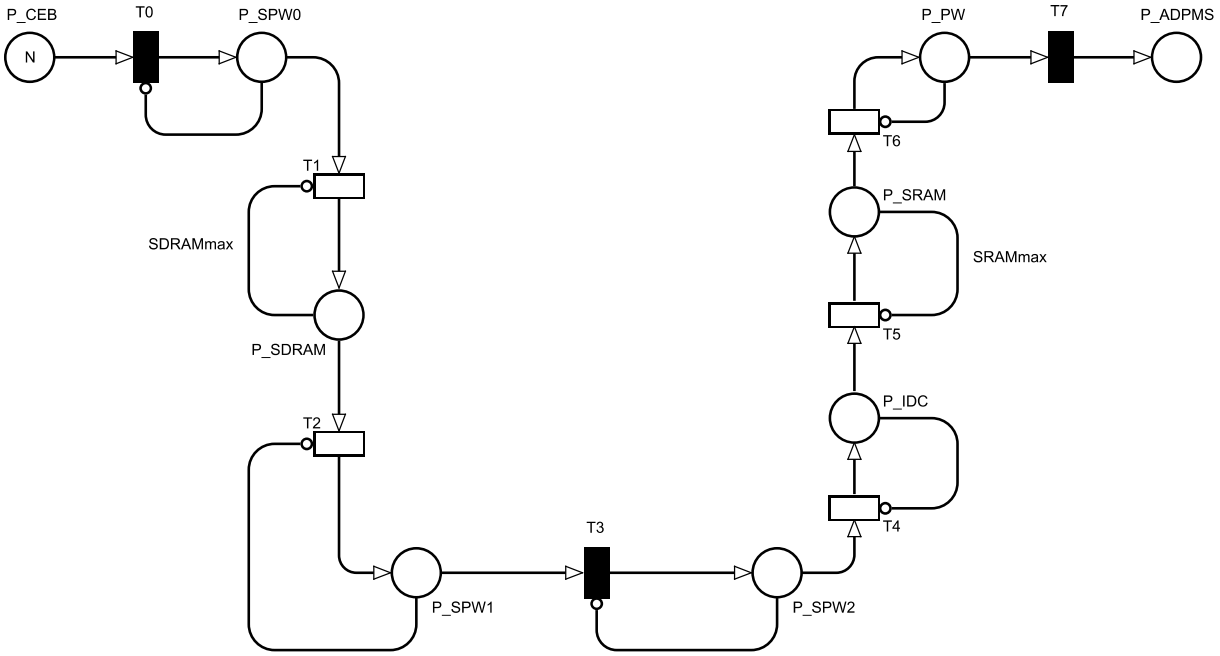

Figure 33 DPU dynamic model – added buffer limits.

Label	Value
SDRAMmax	50000
SRAMmax	150

Table 20 Inhibitor arc labels – compression-on configuration.

Label	Value
SDRAMmax	50000
SRAMmax	50

Table 21 Inhibitor arc labels – compression-off configuration.

Performing transient simulation on second model for N=10000 yields following results (Table 22):

N [tiles]	time to transfer from P_CEB to P_ADMPS [s]
10 000	52.55
2 949 120	15497

Table 22 Simulation results - tile transfer performance (for model from Figure 33).

Although second model is a little bit more realistic, yielded simulation result is very similar to one received in first Petri net model as well as simplistic pipeline analysis. This fact can be explained by huge difference between mean firing ratio T0 (slow) and the rest of transitions (very fast to medium). When token arrives in P_SPW0, then it is literally sucked by the rest of the system, and is transferred at much higher pace toward ADPMS than is transferred from CEB towards CCB. This part of model needs further improvements.

Careful reader will notice that in such model, buffers are not really used (mean utilization, stochastically understood as expected token value at given place is less than 0.1 token for all places inside CCB including P_SDRAM and P_SRAM).

6.2.3 Proposed performance DPU model

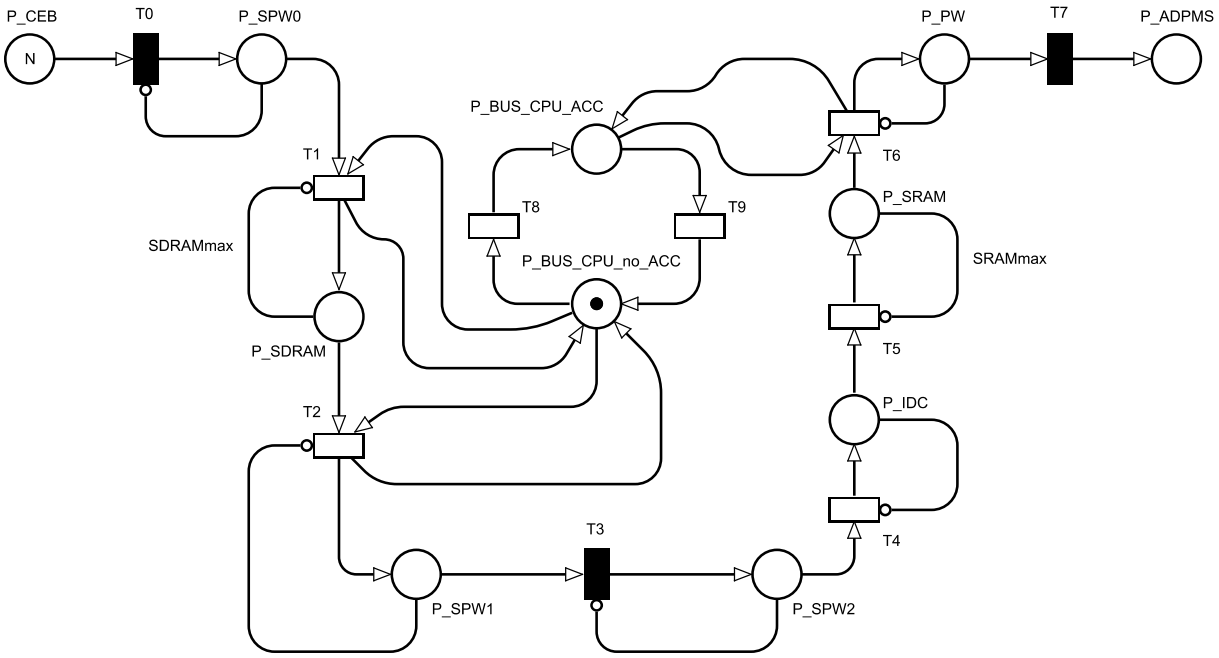

Figure 34 DPU dynamic model with CPU flow control - compression on

Third model presented on Figure 34 brings more details of DPU operation. Two new places and transitions are introduced. Subnet of P_BUS_DPU_ACC, P_BUS_CPU_no_ACC, T8 and T9 models behavior of GR712RC AMBA AHB bus. This is crucial part of the system as at the same time is transfers a lot of data from entry interface to SDRAM storage and to science processing part of FPGA and the same time has to control, manage and set pace of all the activities inside CCB and ASPIICS Coronagraph by accessing the same AHB bus (that is the only way for CPU to interact with external systems). Taking into account fact that AHB accesses are blocking there is an obvious competition (on access to AHB) among different DPU functionality chains (science vs SPS control vs maintenance vs TC/TM control). Getting back to newly introduced subnet, there is one token in it, that represents CPU state. Presence of token in place P_BUS_CPU_ACC denotes the CPU has gained access over AHB and perform its activities while token in place P_BUS_CPU_no_ACC denotes that CPU is either idle or performs activities using it's cache memories only and GR712RC AHB is free for science data transfer.

There is a trick used to model blocking behavior of AMBA inside GR712RC – that is T1 (transfer ongoing from SpW to SDRAM) can occur only by using token from P_BUS_CPU_no_ACC which in turn, makes T2 (transfer from SDRAM to SpW) blocked. Whole trick works also the other way around. Transition T1 and T2, are enabled by presence of token in P_BUS_CPU_no_ACC, will only fire where there is a token in P_BUS_CPU_no_ACC place. Firing of T1 or T2, beside moving token representing scientific data tile along its journey inside DPU, it will also put back AHB availability token back to P_BUS_CPU_no_ACC place.

Similar situation is with enabling T6, with small but important difference, that this transition can fire only when CPU takes control over GR712RC AMBA bus – CPU, using GR71RC – FPGA Space Wire link with RMAP, in order to configure Packet Wire DMA transfer engine. Reader may notice, that transitions T4 and T5 are not gated in anyway. This makes sense, as there are no places with buffering capabilities between P_SDRAM and P_SRAM. Therefore, the moment when the scientific data tile leaves SDRAM, heading to FPGA for being processed, its whole path, traversing Space Wire link and IDC, must be configured and ready for handling the data. And, indeed, that is the way DPU is intended to be operated.

In final DPU Performance Model whole scientific dataflow is controlled by CPU, namely it's access on GR712RC AMBA bus and its duties to manage the FPGA dataflow, are very similar to expected physical implementation. Thing interesting to measure, is the influence of CPU related AMBA activity inside GR712RC on overall DPU capability to stream and process scientific data to ADPMS – in other words, a processing performance. DPU processing performance can be conveniently expressed as time necessary to process and stream to ADPMS all the scientific tiles of data, generated by CEB during worst case observation scenario. It is also useful to evaluate system simulation in context of boundary condition of maximum DPU operation time (as whole Coronagraph Instrument will be switched on during fraction of orbital time: ~25-30%).

Results of final DPU Performance Model simulation are shown in Table 23. Values in column T8 and T9 are mean firing times of respective transitions (which are modeled as stochastic, with exponential probability density function, expressed in seconds). AMBA blocking ratio is the ratio of time token is in place P_BUS_CPU_ACC to time of token is in places P_BUS_CPU_ACC and P_BUS_CPU_no_ACC (part of time AMBA AHB in GR712RC is blocked, on average). "Time to send 10 000 tiles" column is the result obtained for simulation of transfer of 10 000 tiles generate in CEB, through DPU model, up to ADPMS. Data is

extrapolated for transfer of 2 949 120 tiles in column “Time to send all tile in CME-watch scenario”.

T8 [s]	T9 [s]	AMBA blocking ratio [%]	Time to send 10000 tiles [s]	Time to send all tiles in CME-watch scenario [s]
1.00E-04	1.11E-05	10%	52.55	15497.63
1.00E-04	4.29E-05	30%	52.85	15586.10
1.00E-04	1.00E-04	50%	53.45	15763.05
1.00E-04	2.33E-04	70%	55.16	16267.35
1.00E-04	9.00E-04	90%	64.56	19039.52
1.00E-03	1.11E-04	10%	52.75	15556.61
1.00E-03	4.29E-04	30%	54.05	15939.99
1.00E-03	1.00E-03	50%	58.16	17152.08
1.00E-03	2.33E-03	70%	70.27	20723.47
1.00E-03	9.00E-03	90%	138.99	40989.82
1.00E-02	1.11E-03	10%	72.37	21342.78
1.00E-02	1.80E-03	15%	56.11	16547.51
1.00E-02	2.50E-03	20%	57.61	16989.88
1.00E-02	4.29E-03	30%	64.14	18915.66
1.00E-02	1.00E-02	50%	87.26	25734.02
1.00E-02	2.33E-02	70%	145.00	42762.24
1.00E-02	9.00E-02	90%	436.09	128608.17
1.00E-01	1.11E-02	10%	105.87	31222.33
1.00E-01	1.80E-02	15%	69.41	20469.84
1.00E-01	2.50E-02	20%	67.81	19997.98
1.00E-01	4.29E-02	30%	77.92	22979.54
1.00E-01	1.00E-01	50%	112.26	33106.82
1.00E-01	2.33E-01	70%	197.68	58298.20
1.00E-01	9.00E-01	90%	597.4	176180.43

Table 23 Final DPU Performance Model simulation results. Compression-on configuration.

Simulations results presented in Table 23 are also plotted on Figure 35 (time versus AMBA blocking ratio, for four time granularities of AMBA access periods).

Figure 35 visualizes results of simulated DPU performance capabilities using presented final Model. Blue-shaded area at the bottom of a figure denotes time limit for DPU operation (assumed to be 6 hours = 21 600 seconds). Scientific data streaming to ADPMS is considered successful when finishes before time limit elapses. Figure contains plots, each showing time to stream data generated by worst-case CME-watch scenario, to ADPMS, depending on GR712RC AMBA bus blocking ratio (fraction of total time that AMBA bus is **not used** for transferring scientific data tiles). Simulations are performed for four cases of T8 mean time to

fire, showing four orders of magnitude to AMBA bus blocking and not-blocking periods time granularity. AMBA blocking ratio is calculated using following formula:

$$AMBA\ blocking\ ratio = \frac{T9}{T8 + T9} \times 100\%$$

It can be easily noticed that the more AMBA bus is blocked for other activities than transferring tiles rom CEB further into DPU and ADPMS the more time it takes to complete whole data dump to on-board computer. This is actually very intuitive, as when tiles reception is blocked at GR712RC then tile inflow to DPU is limited, and no advantage of fast internal DPU interfaces can improve resulting system capabilities. More important discovery has been made in respect to impact of GR712RC AMBA bus modes switching time granularity. The bigger the granularity (larger the time chunks the bus is in given modes) the lower the DPU capability to stream the data. It can be explained by fact, that all presented models, including final model, are inherently a pipeline but with more sophisticated data flow control structure.

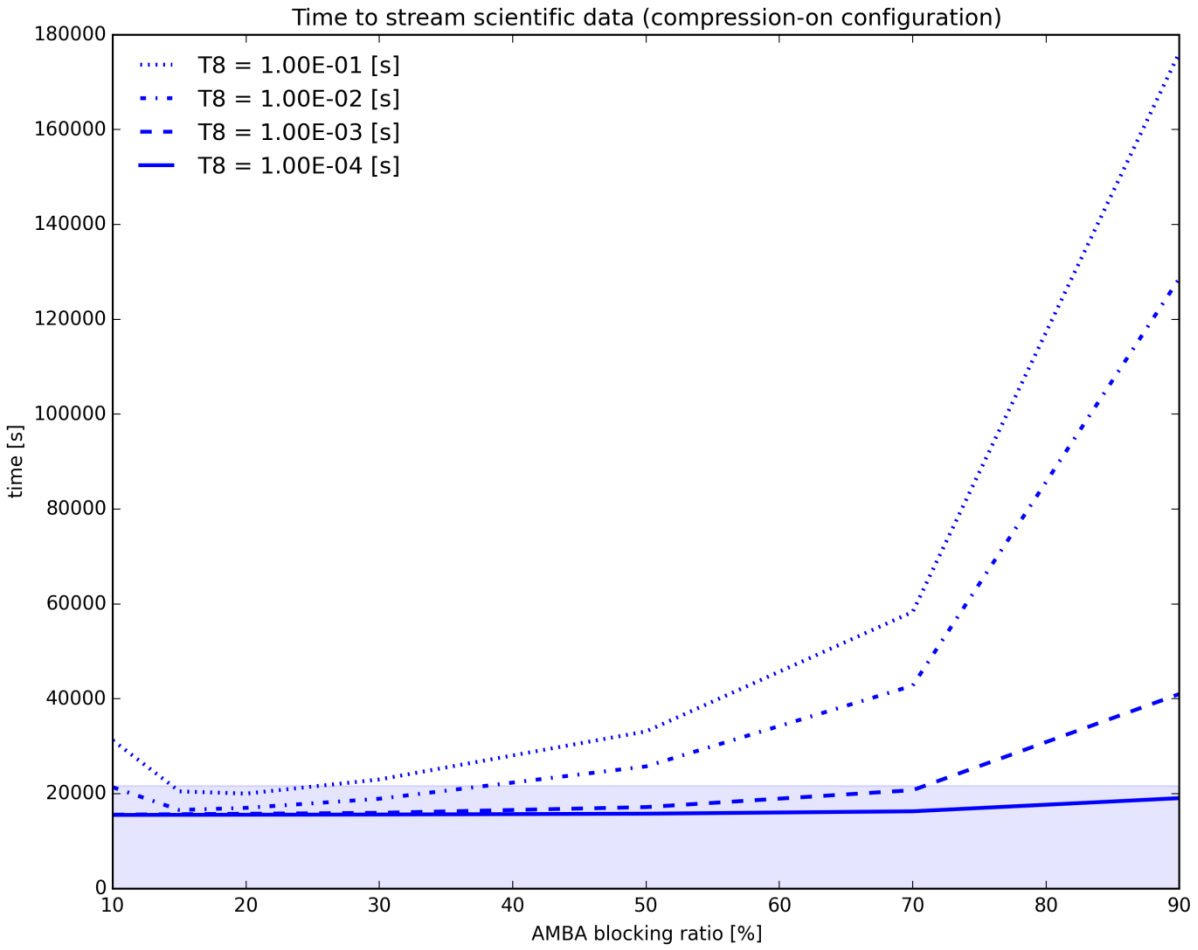

Figure 35 Simulated time to send all tiles of CME-watch observation program for final model, compression-on

Longer periods of AMBA being in one of bus modes, means i.e.: filling the buffer in one part of a system (i.e.: CPU part) while other buffer is not being served in other part of DPU system (i.e.: in FPGA part). This effect leads to tiles pile-up in buffers and to inefficiencies of DPU transfer capabilities. To sum above discussion up, it seems that keeping bus mode switching granularity low is a key factor in ensuring high DPU streaming capabilities, even for high AMBA blocking ratios.

Figure 36 introduces final DPU performance model in configuration for transferring raw data directly to ADMPS without any processing or compression. Final model compression-off operation is basically the same as the one in compression-on configuration, with only few changes:

- Place P_IDC is removed T4.
- Transition T4 is removed.
- Mean time to fire for transitions T5, T6, T7 has to be updated according to Table 24.
- Place P_SRAM buffer limit SRAMmax has to be updated according to Table 21.

T5, T6, T7 and SRAMmax updates are due to fact that tiles are no longer compressed (they are about 3 times bigger than compressed) so transfer rates or buffer size expressed per tile is smaller even if when expressed per bit remains unchanged.

Transition	Type	Throughput [tiles/s]	Mean time to fire [s]
T0	deterministic	192	5.21E-03
T1	exponential	25600	3.91E-05
T2	exponential	25600	3.91E-05
T3	deterministic	800	1.25E-03
T5	exponential	3200	3.13E-04
T6	exponential	3200	3.13E-04
T7	deterministic	400	2.50E-03

Table 24 Petri net transitions firing parameters - compression-off configuration

Results of time necessary to transfer all scientific simulations for various T8 and T9 is summed up in Table 25 and plotted in Figure 37. Final DPU performance model in compression-off configuration exhibits very similar behavior to Model 3 in compression-on configuration. It shows very similar system performance to AMBA blocking ratio dependency. In case of final model, compression-off configuration negative impact of mode switching time granularity on system performance is even stronger than for compression-on configuration. This peculiar feature of DPU model is attributed to limited capacities of both SDRAM and SRAM buffers. When either of buffers is topped, it is no longer able to accept

new tiles (by means of labelled inhibitor arcs added especially for this purpose), which in turn leads to deteriorated tile flow capabilities.

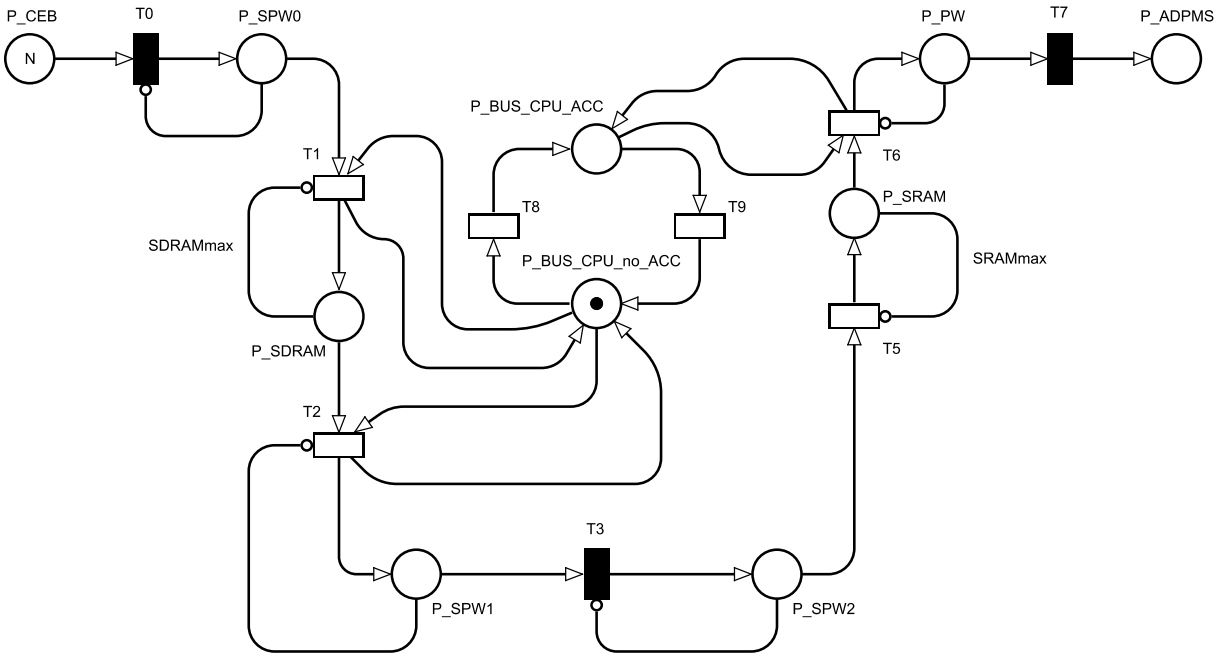

Figure 36 DPU dynamic model with CPU flow control - compression off

T8 [s]	T9 [s]	AMBA blocking ratio [%]	Time to send 10000 tiles [s]	Time to send all tiles in CME-watch scenario [s]
1.00E-04	1.11E-05	10%	58.06	17122.59
1.00E-04	4.29E-05	30%	52.85	15586.10
1.00E-04	1.00E-04	50%	53.45	15763.05
1.00E-04	2.33E-04	70%	55.16	16267.35
1.00E-04	9.00E-04	90%	64.46	19010.03
1.00E-03	1.11E-04	10%	66.37	19573.31
1.00E-03	4.29E-04	30%	54.05	15939.99
1.00E-03	1.00E-03	50%	58.06	17122.59
1.00E-03	2.33E-03	70%	70.37	20752.96
1.00E-03	9.00E-03	90%	139.99	41284.73
1.00E-02	1.11E-03	10%	142.6	42054.45
1.00E-02	1.80E-03	15%	117.68	34705.24
1.00E-02	2.50E-03	20%	100.27	29570.83
1.00E-02	4.29E-03	30%	77.35	22811.44
1.00E-02	1.00E-02	50%	87.56	25822.49
1.00E-02	2.33E-02	70%	144.7	42673.77
1.00E-02	9.00E-02	90%	430.79	127045.14
1.00E-01	1.11E-02	10%	276.56	81560.86
1.00E-01	1.80E-02	15%	189.44	55868.13
1.00E-01	2.50E-02	20%	150.03	44245.65
1.00E-01	4.29E-02	30%	106.72	31473.01

1.00E-01	1.00E-01	50%	114.02	33625.87
1.00E-01	2.33E-01	70%	191.34	56428.46
1.00E-01	9.00E-01	90%	630.6	185971.51

Table 25 DPU Performance Model 3 simulation results. Compression-off configuration.

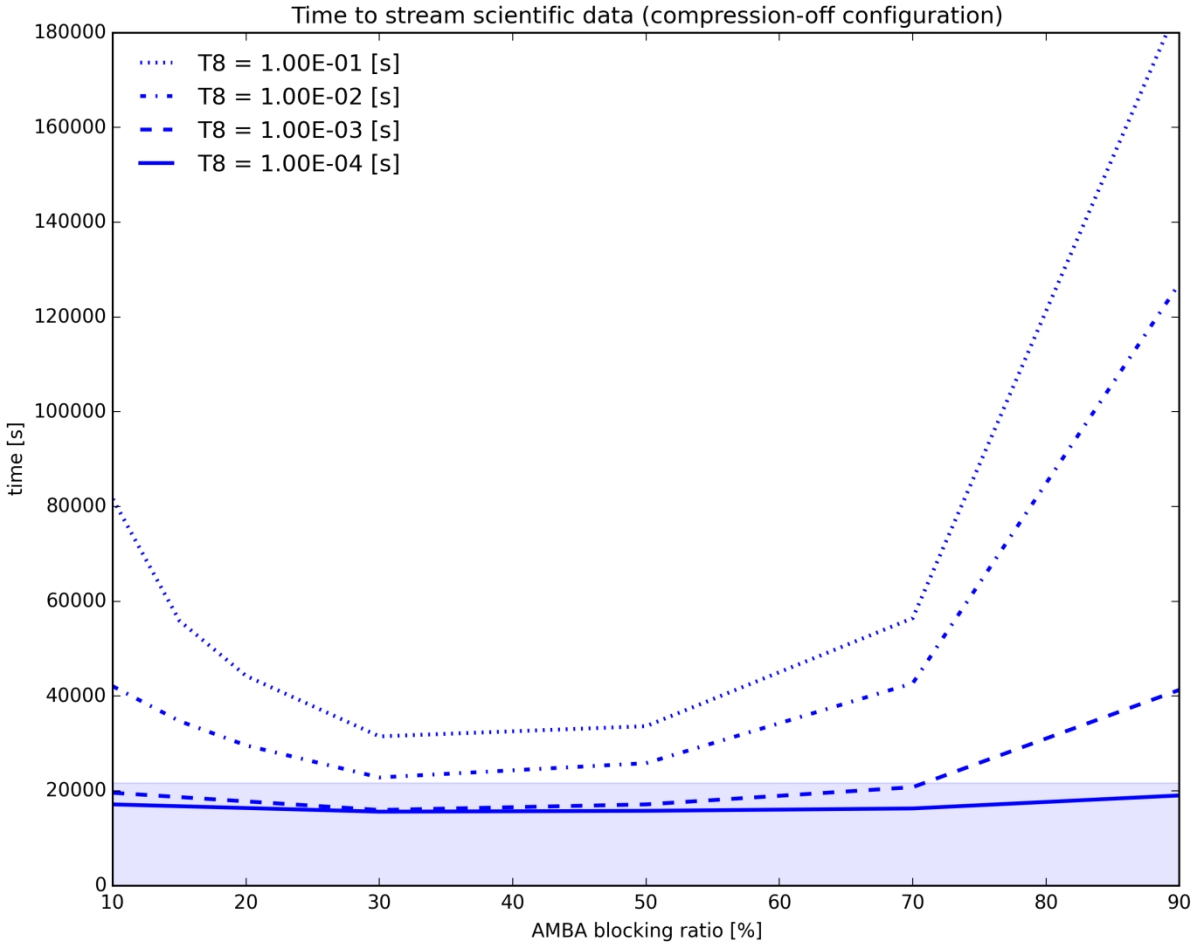

Figure 37 Simulated time to send all tiles of CME-watch observation program, compression-off

Simulations for final DPU performance model compression-on and compression-off configurations are performed on exactly the same T8 and T9 parameters set, therefore are easily comparable. Inspection of simulation results plotted on Figure 35 and Figure 37 reveals that final mode in compression-off configuration is generally more sensitive to AMBA blocking ratio. Model is efficient in tile transfer only when time spent by GR712RC AMBA bus on tile transfer is more or less equal to time spent on other activities, including managing FPGA data flow (AMBA block ratio ≈ 50%) and mode switching granularity is low. Final model in compression-on configuration can be operated efficiently in low mode switching granularities and in, both, low and medium AMBA blocking ratio (< 50%, meaning that GR712RC AMBA bus is not used for scientific data transfer less than half of bus time). As pointed out before, these effects are attributed to limited buffer size. SRAM buffer tops-out

when SDRAM buffer is being purged, SDRAM buffer tops-out when SRAM buffer is being flushing its contents into ADPMS. These effects can be observed in simulation by i.e. measurement of expected tile (in simulation terms – token) number in buffers (P_SDRAM and P_SRAM). Additionally, way of measuring total transmission time is explained on the same examples.

On following figures two cases are evaluated in details, both of final model compression-off configuration. First case, is the simulation for AMBA blocking ratio = 10%, with T8 = 1.00E-04 and T9 = 1.11E-05. This is the case when resulting DPU processing performance meets user requirements Figure 38, shows how resulting time to process all scientific data tiles is evaluated – by extrapolating time to send 10 000 tiles to time necessary to transfer tiles generated in CME-watch scenario.

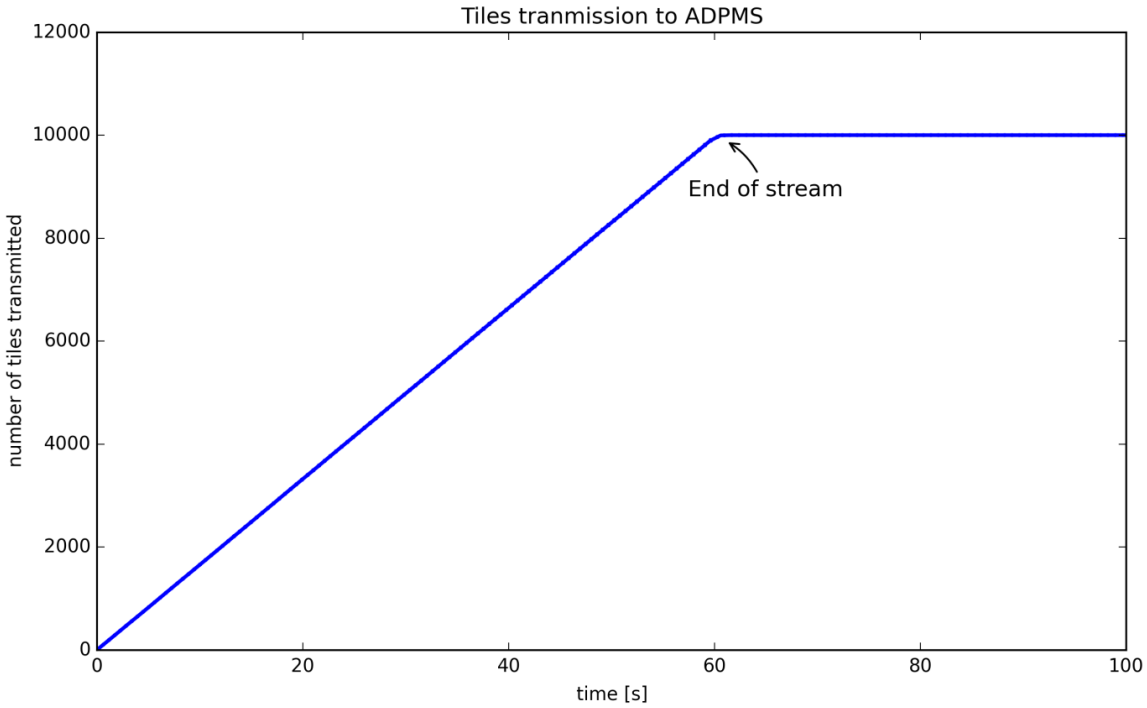

Figure 38 Simulated tiles transmission to ADPMS (N=10000, T8=1.00E-04, T9=1.11E-05, AMBA_blocking_ratio = 10%, compression-off configuration)

Figure 39 and Figure 40 explains how number of tiles varies over time in SDRAM and SRAM buffers respectively. Blue-shaded area denotes buffer size. SDRAM is filled with narrow stream of tiles that couldn't be fetched into FPGA due to limited time spent by CPU on managing FPGA dataflow (10% of AMBA bus time). When tiles inflow stops, SDRAM buffer is quickly flushed trough FPGA to ADPMS. SDRAM filling speed can be estimated – simulation shows it reaches about 756 tiles in 52 seconds which gives 14.53 tiles /s. For this

case it has been extrapolated that for CME-watch scenario it will take about 17122s to transfer all tiles, SDRAM buffer will need to hold more than 200 000 tiles. This is obviously not possible as SDRAM buffer maximum is 50 000 tiles, so in reality it will take more time to transfer the tiles than suggested by simple extrapolation.

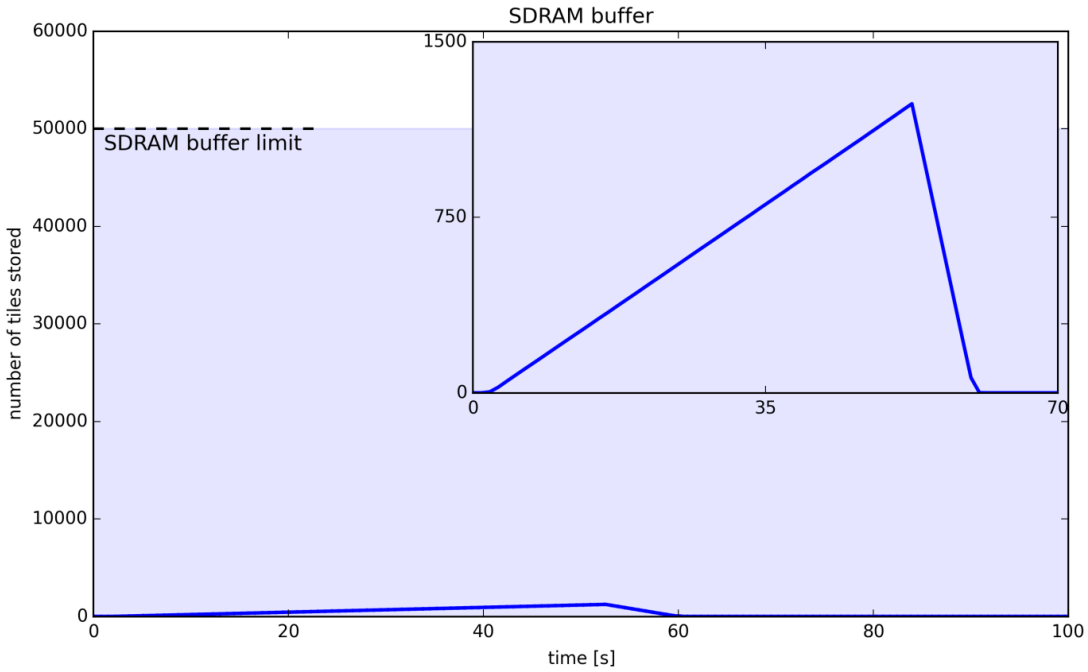

Figure 39 Simulated tiles pile-up in SDRAM (N=10000, T8=1.00E-04, T9=1.11E-05, AMBA_blocking_ratio=10%, compression-off configuration)

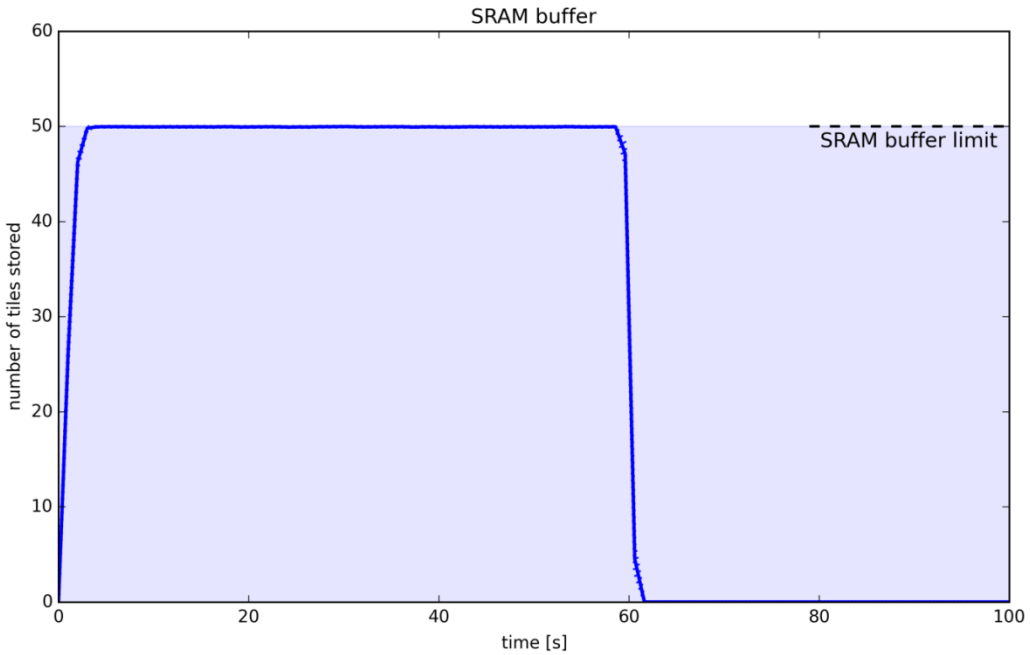

Figure 40 Simulated tiles pile-up in SRAM (N=10000, T8=1.00E-04, T9=1.11E-05, AMBA_blocking_ratio=10%, compression-off configuration)

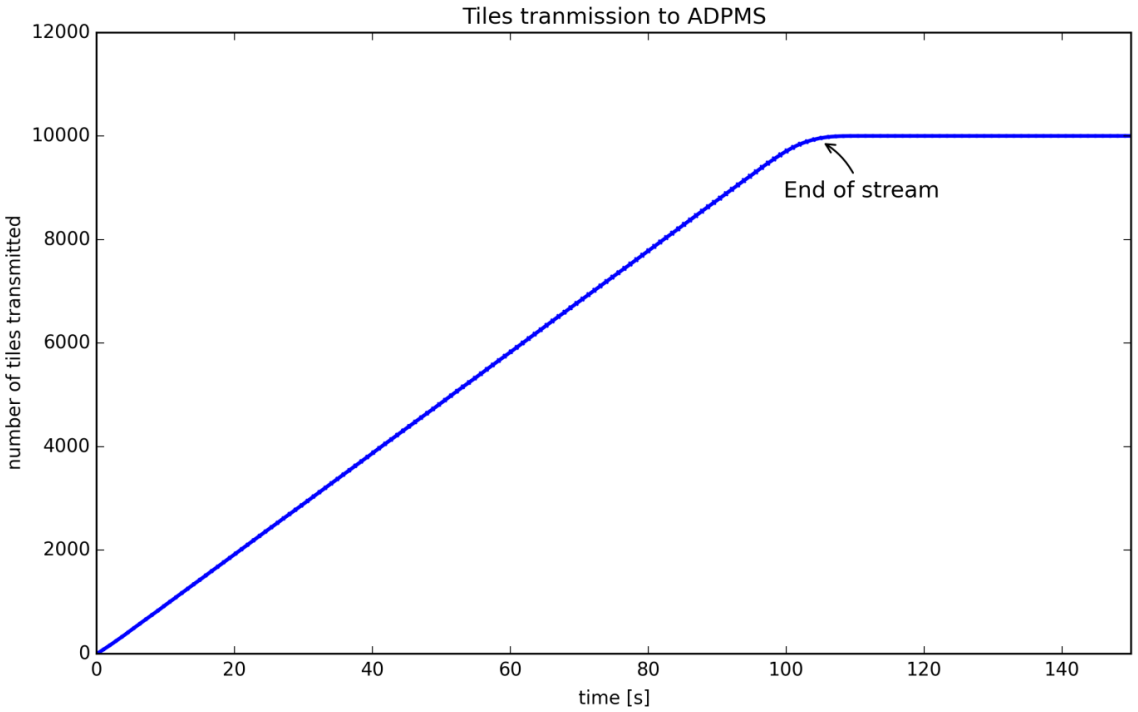

Figure 41 Simulated tiles transmission to ADPMS (N=10000, T8=1.00E-01, T9=1.00E-01, AMBA_blocking_ratio=50%, compression-off configuration)

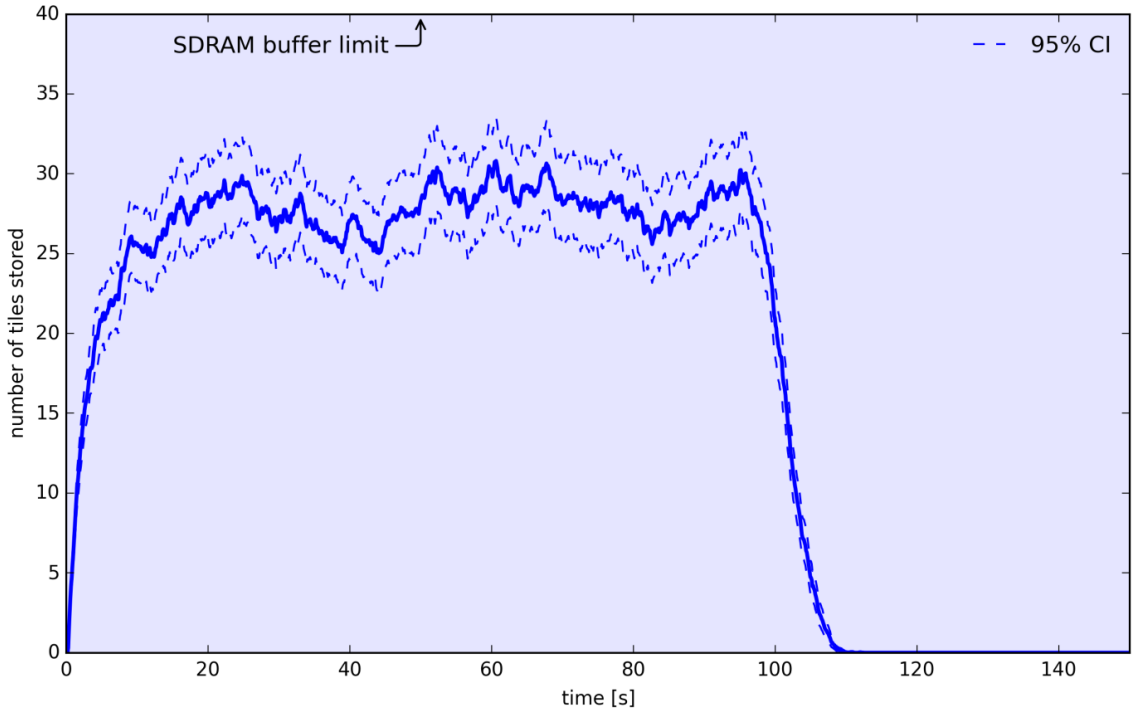

Figure 42 Simulated tiles pile-up in SDRAM (N=10000, T8=1.00E-01, T9=1.00E-01, AMBA_blocking_ratio=50%, compression-off configuration)

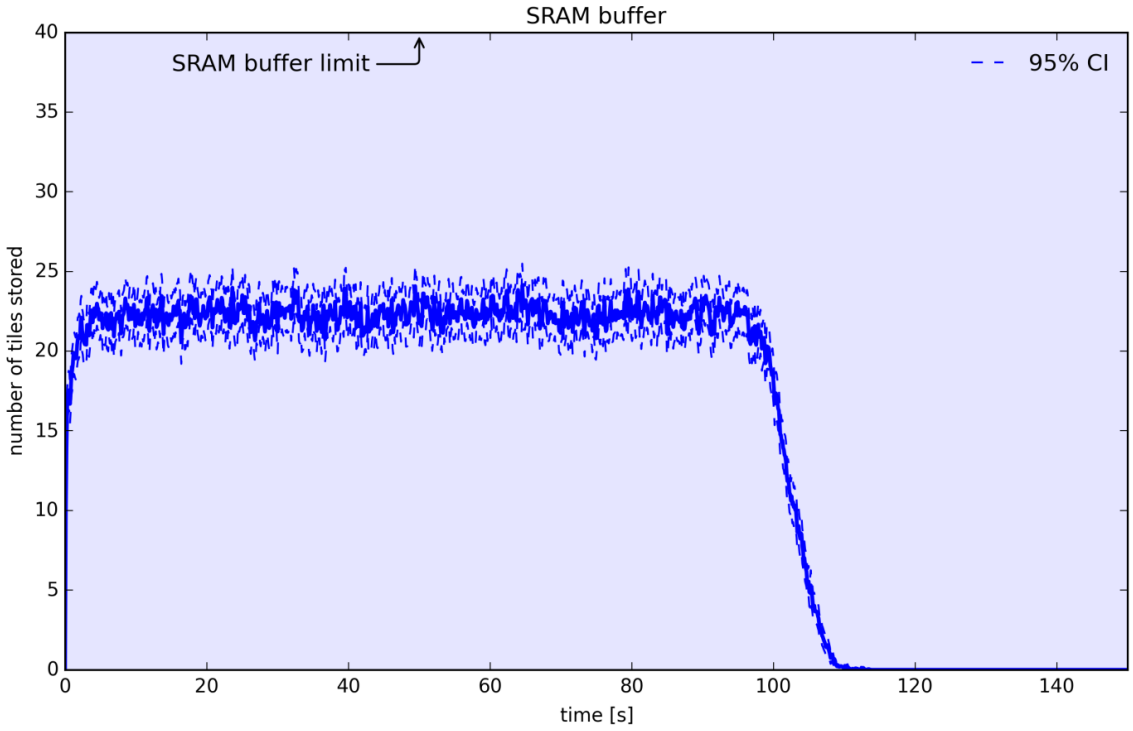

Figure 43 Simulated tiles pile-up in SRAM (N=10000, T8=1.00E-01, T9=1.00E-01, AMBA_blocking_ratio=50%, compression-off configuration)

As pointed out before, SRAM buffer gets filled up very fast due to infrequent flushes to ADPMS managed by CPU. SRAM being filled up in turn blocks SDRAM.

Similar analysis can be performed for second case, the simulation for AMBA blocking ratio = 50%, with T8 = 1.00E-01 and T9 = 1.0E-01. Here Figure 42 and Figure 43 of SDRAM and SRAM buffers shows their utilization on steady level thorough whole simulation. Therefore buffer fill-up is not expected in time extrapolation process, therefore results obtained from Figure 41 are considered accurate.

6.3 Performance analysis results assessment

Two types of analysis has been performed in order to analyze performance of DPU architecture and its ability to meet scientific observation and data processing within time limits imposed by planned way of operation of Coronagraph Instrument (ASPIICS).

First type of analysis, classic, simplified one, based on pipeline model (chapter 6.1). This kind on analysis takes into account effective throughput of whole resulting from throughput of each communication link that DPU consists of, and compares it with data stream injected to the system. In case of DPU, injected data stream is lower than effective throughput and worst-case scenario data volume is easily processed and transferred to ADPMS for both

configurations: compression-off and compression-on. This model is very simplistic, and despite the result is very positive (data stream utilization of DPU system is almost always lower than 50%) it shall be treated with reserve. It is a rule of thumb estimation that system concept is feasible, expected performance could be achieved and details shall be evaluated further.

Second type of analysis (chapter 6.2), more detailed, takes pipeline model as a base and adds information on dynamic behavior of the system (bus blocking, buffer sizes). Using Petri Net language and related tools, dynamic behavior could be analyzed and evaluated. Third model, which is most advanced and detailed of proposed dynamic DPU models gives a lot more insight into system behavior, especially how operation (congestion) of AMBA bus in GR712RC influences resulting system performance. This node has been identified as potential bottleneck for DPU as different functional chains cross their path in mentioned AMBA bus. Although the on-chip bus has very high throughput it is also used by CPU to fetch instructions (processor cache contents update), execute housekeeping tasks and control routines, as well as (and this plays significant role) manage data flow in FPGA (which doesn't have separate controller and rely on GR712RC to set the pace on events happening inside). Simulations of time necessary to deliver all scientific data tiles to ADPMS (Figure 35 and Figure 37) show DPU transfer and processing performance dependence on GR712RC AMBA operations time granularity as well as fraction of time spent on transferring scientific tile to and from GR712RC and time spent to on managing FPGA data flow. Main and most important takeaway from performed analysis is that the smaller the AMBA operation granularity and the more balanced is the AMBA time division between SoC activities and FPGA activities, the better. Simulations show that when granularity is low (order of time necessary to send a data tile on AMBA in GR712RC) and time division is balanced around 50%, for both configurations with and without compression, DPU will be able to process (if applicable) and stream the data to ADPMS in under 6 hours for presented worst case observation scenario (CME-watch). This gives a guideline, perhaps, how on-board software operations shall be designed and scheduled, in order to maintain required processing performance od DPU

7 CCB DPU reliability analysis

Chapter 6 has shown how Petri Nets combined with holistic system analysis methodology allows system engineer to analyze and theoretically evaluate achievable processing performance, taking into account dynamic nature of embedded control and data transfer system, confirming foreknown and revealing unknown system features and attributes.

In following chapter, CCB DPU is going to be analyzed in terms of its reliability, using very similar tools and the same scientific methodology to tackle system complexity. Topics and issues addressed in following paragraphs:

- identification and extraction of functional blocks relevant from reliability perspective
- calculation of expected functional blocks' fault rates
- identification of functional chains relevant from mission perspective
- classic (RBD / Bayes theorem) analysis of functional chains reliability
- Petri Net based analysis of functional chains reliability
- results comparison and more advanced reliability checks

7.1 Average CCB DPU component level fault rates on P3 HEO orbit

This subchapter presents components' error (SEU) rates estimation for expected PROBA-3 HEO orbit. SEU occurrence has been simulated for period of one year, in average conditions. In order to get bit (flip-flop, memory cell) fault rate result ([$bit^{-1}year^{-1}$]) has to be divided by 365 to get fault rate in [$bit^{-1}day^{-1}$].

For memories, an effective word fault rate might be more convenient to use, calculated by multiplying bit fault rate by number of bits in word (i.e. 8, 16, 32) and taking into account ECC scheme if implemented (which already has been performed for RTAX BRAM in chapter B.3, and in this chapter is done in similar manner for other protected memories).

Component bit fault rates, based on susceptibility information presented in Appendix B and PROBA-3 mission environment presented in chapter 4.5 and, are shown in Table 26.

Device	Effect	[SEU*bit^{-1}*year^{-1}]
GR712RC	Direct ionization	1.9348E-05
	Proton induced ionization	5.2159E-06
	Total	2.4564E-05
RTAX R+C	Direct ionization	4.0340E-05
	Proton induced ionization	1.0813E-06
	Total	4.1421E-05

RTAX BRAM	Direct ionization	2.9167E-04
	Proton induced ionization	4.8352E-06
	Total	2.9650E-04
3DSR4M08CS1647 SRAM	Direct ionization	1.1870E-04
	Proton induced ionization	7.9458E-06
	Total	1.2664E-04
3DSD2G16VS4364 SDRAM	Direct ionization	6.3655E-10
	Proton induced ionization	3.1854E-09
	Total	3.8220E-09
UT8QNF8M8 Flash	Direct ionization	5.6169E-20
	Proton induced ionization	0.0000E+00
	Total	5.6169E-20

Table 26 Components SEU rate estimation in PROBA-3 orbit

7.1.1 GR712RC

Table 26 states that GR712RC fault rate is 2.4564E-05 SEU / year. Bearing in mind that GR712RC processor core (available radiation data tackled LEON3-FT core issues only) has about 210 critical (susceptible bits) which results in 2.4564E-5 * 210 = 0.00515844 = 5.1584E-3 bit flips in a year.

Truth to be told, GR712RC consists of many more functional blocks than just LEON3-FT processor cores and includes external memory controllers (FTMCTRL), communication link controllers (GRSPW and APBUART, SPI, AHB/APB Bridge), timers and general purpose registers for I/O pin activities. No radiation information is available that explicitly related to any of these functional block.

There is a possibility to perform estimation of fault rates related to these blocks, based on following assumptions:

- all of GR712RC functional blocks are implemented from Gaisler's GRLIB IP-core library (confirmed in [103], [112])
- all of GR712RC functional blocks are created in the same technological process (true, all block are placed on one semiconductor die)
- functional block fault rate is proportional to its complexity, hence, die area it utilizes (it is possible, although there should be fault rate dependence on implementation of sensitive logic circuit solutions, in fact, bigger, more complex logic is more likely to generate sensitive, unmitigated circuit parts)

Taking into account above assumption, fault rate of each of existing functional black can be related to known LEON3-FT fault rates, proportionally to relative area utilization on GR712RC die (Table 27).

Functional Block	Logic area[1)]	Logic area normalized[2)]	Fault rate [upset*block^{-1}*year^{-1}]	Fault rate [upset*block^{-1}*day^{-1}]
LEON3-FT[3)]	60000	100.0%	5.1584E-03	1.4133E-05
FTMCTRL	5000	8.3%	4.2987E-04	1.1777E-06
INTERRUPT CTRL	1500	2.5%	1.2896E-04	3.5332E-07
AHB/APB bridge	2500	4.2%	2.1494E-04	5.8886E-07
TIMERS*4 +Watchdog	2600	4.3%	2.2353E-04	6.1242E-07
TIMERS*4	2000	3.3%	1.7195E-04	4.7109E-07
GRSPW2+RMAP	25000	41.7%	2.1494E-03	5.8886E-06
GRSPW2	15000	25.0%	1.2896E-03	3.5332E-06
SPI	2500	4.2%	2.1494E-04	5.8886E-07
APBUART	2500	4.2%	2.1494E-04	5.8886E-07
GRGPREG	1500	2.5%	1.2896E-04	3.5332E-07
Note: 1): ASIC gates equivalent [24] 2): normalized to LEON3-FT 3):LEON3-FT core variant with 16kB instruction +16kB data caches, with FPU IEEE-754, with MMU				

Table 27 GR712RC functional blocks fault rate estimation for PROBA-3 HEO

7.1.2 RTAX 2000

7.1.2.1 R -cells

In PROBA-3 orbit, on average, RTAX 2000 – which contains 10752 R-cells [RD-18], may experience upset ratio (for whole device utilization): 4.1421E-05 * 10752 = 0.4453 logic upsets / year. For two year mission, assuming that CCB is online whole time, it is estimated that RTAX2000 may experience about 0.9 logic upset in total.

7.1.2.2 BRAM

RTAX 2000 BRAMs contains 294912 bits which results total device BRAM bit upset ratio of 2.2017E-04 * 294912 = 64.9307 memory bit upsets / year. For two year mission, assuming that CCB is online whole time, it is estimated that RTAX2000 may experience about 130 bit upsets in total.

word size [bit]	depth	user memory [bit]	block RAMs	physical memory [bit]	efficiency
8	4096	32768	16	65088	50,34%

16	2048	32768	16	65088	50,34%
32	1280	40960	15	61020	67,13%

Table 28 RTAX BRAM memory EDAC organization schemes

Useful information from perspective of system design is word, rather than bit, upset ratio, and effects of implemented mitigation schemes. RTAX 2000 BRAM words can be organized in three different organization and mitigation schemes: 8 bits (if encoded - 12 bits), 16 bits (if encoded - 29 bits), 32 bits (if encoded - 47 bits) as it is summed up in Table 28. Methods for calculating mitigated word upset rate, based on bit upset rate has been shown in chapter B.3. Word level results are shown in Table 29 and Table 30.

User data word size	**Unmitigated word error rate [upset*word^{-1}*year^{-1}]**	**Mitigated word error rate [upset*word^{-1}*year^{-1}]**
8	2.3695E-3	5.7907E-06
16	4.7334E-3	3.5502E-05
32	9.4445E-3	9.4191E-05

Table 29 RTAX BRAM word upset rates in [upset*word^{-1}*year^{-1}]

User data word size	**Unmitigated word error rate [upset*word^{-1}*day^{-1}]**	**Mitigated word error rate [upset*word^{-1}*day^{-1}]**
8	6.4986E-06	4.3551E-11
16	1.2997E-05	2.6790E-10
32	2.5994E-05	7.1331E-10

Table 30 RTAX BRAM word upset rates in [upset*word^{-1}*day^{-1}]

7.1.2.3 IP cores fault rates

Knowledge of R-cell and BRAM upset rates allows to determine IP-core fault rate as a cumulated error in logic and memory blocks. Formula for effective IP core fault estimation is following:

$$P_{IP-core} = N_{R-cell}P_{R-cell} + N_{BRAM}P_{BRAM}$$

where:

$P_{IP-core}$ is IP-core fault rate

N_{R-cell} is number of R-cells used by given IP-core

P_{R-cell} is R-cell upset rate

N_{BRAM} is number of BRAM bits / words used

P_{BRAM} is BRAM bit or word upset rate

IP core	**Logic resources**	**R-cells**	**BRAM words**
GRSPW2 + RMAP	4500	1501	256
AHB CONTROLLER	500	167	0

AHB / APB BRIDGE	200	67	0
GRPWTX	2200	734	0
AMBA-FIFO DATA I/F	1000	334	512
IDC	6000	2001	256
COMPRESSED DATA DMA ENGINE	1000	334	0
CACHE MEMORY CONTROLLER	1000	334	0
SPI	900	301	0
SPS DATA PROCESSING	500	167	0

Table 31 FPGA resources utilization per IP-core

Logic upsets per IP core:

IP core	Logic Upsets [1/core*day]	Logic Upsets [1/core*year]
GRSPW2 + RMAP	1.70E-04	6.22E-02
AHB CONTROLLER	1.90E-05	6.92E-03
AHB / APB BRIDGE	7.60E-06	2.78E-03
GRPWTX	8.33E-05	3.04E-02
AMBA-FIFO DATA I/F	3.79E-05	1.38E-02
IDC	2.27E-04	8.29E-02
COMPRESSED DATA DMA ENGINE	3.79E-05	1.38E-02
CACHE MEMORY CONTROLLER	3.79E-05	1.38E-02
SPI	3.42E-05	1.25E-02
SPS DATA PROCESSING	1.90E-05	6.92E-03

Table 32 Estimated IP-core logic upset rate for P3 HEO

Block RAM upsets per IP core:

IP core	BRAM word upsets [1/core*day]	BRAM word upsets [1/core*year]
GRSPW2 + RMAP	1.83E-07	6.67E-05
AHB CONTROLLER	0.00E+00	0.00E+00
AHB / APB BRIDGE	0.00E+00	0.00E+00
GRPWTX	0.00E+00	0.00E+00
AMBA-FIFO DATA I/F	0.00E+00	0.00E+00
IDC	3.65E-07	1.33E-04
COMPRESSED DATA DMA ENGINE	1.83E-07	6.67E-05
CACHE MEMORY CONTROLLER	0.00E+00	0.00E+00
SPI	0.00E+00	0.00E+00
SPS DATA PROCESSING	0.00E+00	0.00E+00

Table 33 Estimated IP-core BRAM word upset rate for P3 HEO

Total expected IP-core upset count:

IP core	Total upsets [1/core * second]	Total upsets [1/core * day]	Total upsets [1/core * year]
GRSPW2 + RMAP	1.97E-09	1.71E-04	6.22E-02
AHB CONTROLLER	2.19E-10	1.90E-05	6.92E-03
AHB / APB BRIDGE	8.80E-11	7.60E-06	2.78E-03
GRPWTX	9.64E-10	8.33E-05	3.04E-02
AMBA-FIFO DATA I/F	4.39E-10	3.79E-05	1.38E-02
IDC	2.63E-09	2.27E-04	8.30E-02
COMPRESSED DATA DMA ENGINE	4.41E-10	3.81E-05	1.39E-02
CACHE MEMORY CONTROLLER	4.39E-10	3.79E-05	1.38E-02
SPI	3.95E-10	3.42E-05	1.25E-02
SPS DATA PROCESSING	2.19E-10	1.90E-05	6.92E-03

Table 34 Estimated total IP-core upset rate for P3 HEO

7.1.3 Memories

7.1.3.1 SDRAM

Although 3DSD2G16VS4364 SDRAM is quite robust: 3.8220E-09 / 365 = 1.0471e-11 bit upsets / day, there is a possibility to implement EDAC protection scheme on this memory as well. As this memory is going to be used in 32 bit wide access scheme, effective Reed-Solomon coding mechanism can be used. Reed-Solomon EDAC is capable of correcting two 4-bit nibble errors in 32-bit word or 16-bit checksum. To be exact, implemented Reed-Solomon coding (RS(6,4,2)) in form of two 16- bit data (+8 bit checksum) codewords, interleaved nibble-wise. As a result coding can correct two 4-bit errors, when each error is located in a different nibble, and not in the same original codeword. [112] describes how data is coded in SDRAM. It is worth noticing that SDRAM data bus is 48 bits long: 32-bits for data (DATA[..] bus) and 16 bits-for check-bits (CB[..] bus) – there are three 16-bit wide SDRAM chips used to implement CCB DPU memory.

Nibble	Contents
DATA[31 : 28]	codeword 0, data symbol
DATA[27 : 24]	codeword 1, data symbol
DATA[23 : 20]	codeword 0, data symbol
DATA[19 : 16]	codeword 1, data symbol
DATA[15 : 12]	codeword 0, data symbol
DATA[11 : 8]	codeword 1, data symbol
DATA[7 : 4]	codeword 0, data symbol
DATA[3 : 0]	codeword 1, data symbol
CB[15 : 12]	codeword 0, check symbol
CB[11 : 8]	codeword 1, check symbol
CB[7 : 4]	codeword 0, check symbol

CB[3 : 0]	codeword 1, check symbol

Table 35 SDRAM Reed-Solomon EDAC coding

Knowing above, and letting P_{bSEU} be SDRAM bit flip probability, mitigated and unmitigated RAM words error probabilities can be derived. For unmitigated word, error probability is probability that any of 32 data bits is flipped (check bits doesn't matter at this time):

$$P_{word} = 1 - (1 - P_{bSEU})^{32}$$

For Reed – Solomon EDACed word error, first, nibble error must be estimated (let P_n be nibble error probability, any error – from 1 to 4 simultaneous bitflips):

$$P_n = 1 - (1 - P_{bSEU})^4$$

Reed – Solomon EDAC coder can handle (memory word doesn't get corrupted) following cases:

- total number of zero upsets: zero errors in codeword 0 and codeword 1:

$$P_{0\ upset} = (1 - P_n)^{12}$$

- total number of one upset: one error in codeword 0 xor one error in codeword 1

$$P_{1\ upset} = 12 * P_n * (1 - P_n)^{11}$$

- total number of two upsets: one error in codeword 0 and one error in code word 1

$$P_{2\ upset} = 36 * P_n^2 * (1 - P_n)^{10}$$

Taking into account above considerations, Reed-Solomon protected 32-bit word unrecoverable error probability can be expressed as:

$$P_{RS\ upset} = 1 - P_{recoverable\ errors} = 1 - \left(P_{0\ upset} + P_{1\ upset} + P_{2\ upset}\right)$$

$$P_{RS\ upset} = 1 - (1 - P_n)^{12} - 12 * P_n * (1 - P_n)^{11} - 36 * P_n^2 * (1 - P_n)^{10}$$

As a result, SDRAM word error rates, in PROBA-3 HEO, for, both, unmitigated and mitigated cases are shown in Table 36.

User data word size	**Unmitigated word error rate [upset*word^{-1}*year^{-1}]**	**Mitigated word error rate [upset*word^{-1}*year^{-1}]**
32	1.2230E-07	7.0117E-15

Table 36 SDRAM word upset rates in [upset*word^{-1}*year^{-1}]

7.1.3.2 SRAM

For 3DSR4M08CS1647 SRAM SEU rate is high compared to other components: 1.2664E-04 / 365 = 3.4695E-07 bit upsets / day. Although, physically speaking, SRAM memory can be implemented in 8-bit and 32-bit organization schemes, FTMCTRL memory controller implemented in FPGA, operates in a way that always 32-bits (4 consecutive bytes) are read. If EDAC is enabled (BCH coding available (39,7)) then each 32bit word (in both 8- and 32-bit organization schemes) is protected by 7 bit checksum. Implemented BCH coding can mitigate 1 bit upset in whole 39-bit codeword [103].

BCH(39,7) coder can handle (so memory word doesn't get corrupted) following cases (let P_b be bit error probability):

- total number of zero upsets in codeword:

$$P_{0\ upset} = (1 - P_b)^{39}$$

- total number of one upset in codeword:

$$P_{1\ upset} = 39 * P_b * (1 - P_b)^{38}$$

Taking into account above considerations, BCH(39,7) protected 32-bit word unrecoverable error probability can be expressed as:

$$P_{BCH\ upset} = 1 - P_{recoverable\ errors} = 1 - \left(P_{0\ upset} + P_{1\ upset}\right)$$

$$P_{BCH\ upset} = 1 - (1 - P_b)^{39} - 39 * P_b * (1 - P_b)^{38}$$

Both, 8 bit and 32 bit accesses can have implemented EDAC, but, as mentioned before, physically these are always 32 bit reads and writes to memory chip, so word error rates remain the same (Table 37):

User data word size	Unmitigated word error rate [upset*word^{-1}*year^{-1}]	Mitigated word error rate [upset*word^{-1}*year^{-1}]
8	4.0445E-03	1.1847E-05
32	4.0445E-03	1.1847E-05

Table 37 SDRAM word upset rates in [upset*word^{-1}*year^{-1}]

7.1.3.3 Flash

For UT8QNF8M8 Flash SEU rate is extremely low being 5.6169E-20 / 365 = 1.5388E-22 bit upsets / day. Memory is connected to GR712RC memory controller, which is functionally the same as FPGA memory controller FTMCTRL. It offers exactly the same memory organization schemes and EDAC mechanism as described in previous chapter (7.1.3.2)

[103].Therefore, EDACed (BCH(39,7)) word upset probability is the same as previously estimated:

$$P_{BCH\ upset} = 1 - (1 - P_b)^{39} - 39 * P_b * (1 - P_b)^{38}$$

Flash, EDACed, word error rates are following (Table 38):

User data word size	Unmitigated word error rate [upset*word^{-1}*year^{-1}]	Unmitigated word error rate [upset*word^{-1}*year^{-1}]
8	1.7974E-18	<1.0000E-27
32	1.7974E-18	<1.0000E-27

Table 38 Flash word upset rates in [upset*word^{-1}*year^{-1}]

Flash memory is considered as close to impossible to upset (bear in mind that Flash upset is understood not as Flash memory cell flip, but an upset in readout or write circuits of Flash memory - as a result, faulty reads or writes are expected but not contents corruption). Therefore, for sake of simplicity, and due to fact that Flash, after switching from Boot Software to Application Software is seldom used, it is not further analyzed as it does not influence CCB reliability in measurable way.

7.2 CCB DPU functional blocks' fault rates on P3 HEO orbit

Partial results of functional block error rates are obviously interesting but key information is the resulting system error rate (or system unreliability). There are several approaches to perform such analysis, the simplest one being the RBD analysis supported by Bayes theorem, and more sophisticates one using Petri Nets, revealing greater flexibility and more holistic analysis of CCB DPU.

Few assumptions, valid for both cases, have to be made in order to organize the process of analysis:

- all errors (SEUs) are independent
- there are no common mode failures in DPU itself
- CCB DPUs (and PCUs but it doesn't matter) are cold redundant
- CCB (analyzing CCB nominal and redundant branches separately) power system error is a common mode type of error but is not taken into account in analysis at the moment

Table 39 summarizes error rates of all identified functional blocks within DPU: FPGA IP-cores, GR712RC CPU core and peripherals, blocks of memory space.

Type	Block	Error rate [error*block^{-1}*day^{-1}]
CPU functional block	LEON3-FT	1.4133E-05
	FTMCTRL	1.1777E-06
	INTERRUPT CTRL	3.5332E-07
	AHB/APB bridge	5.8886E-07
	TIMERS*4 +Watchdog	6.1242E-07
	TIMERS*4	4.7109E-07
	GRSPW2+RMAP	5.8886E-06
	GRSPW2	3.5332E-06
	SPI	5.8886E-07
	APBUART	5.8886E-07
	GRGPREG	3.5332E-07
FPGA IP-core blocks	GRSPW + RMAP	1.7100E-04
	AHB CONTROLLER	1.9000E-05
	AHB / APB BRIDGE	7.6000E-06
	GRPWTX	8.3300E-05
	AMBA-FIFO DATA I/F	3.7900E-05
	IDC	2.2700E-04
	COMPRESSED DATA DMA ENGINE	3.8100E-05
	CACHE MEMORY CONTROLLER	3.7900E-05
	SPI	3.4200E-05
	SPS DATA PROCESSING	1.9000E-05
Memory	SDRAM (EDACed) 100M words	5.2628E-12
	SRAM (EDACed) 100k words	8.9197E-06

Table 39 Identified functional blocks error rate summary in [errors*block^{-1}*day^{-1}]

7.3 CCB DPU functional execution chains

Knowing the component or functional block fault rate is crucial first step in understanding whole system reliability. Next step, incorporating topological and architectural information, is proper identification of functional execution (or control) chains. Operation of each of such chains delivers distinct set of services offered by DPU. Identification of functional chains allows for more than just a system global analysis. It allows to investigate which parts of system contribute more to global (un)reliability and what is given chain reliability in respect to importance of delivered services, to severity of possible failures and criticality to mission success.

The notion of functionality chain will be used in both, classic and Petri net based analysis. There are 3 functionality chains identified in CCB DPU that play very important role in maintaining dependability of CCB DPU and ASPIICS instrument as a whole (explained in Table 40).

Chain	Description
TC / TM control	this path is formed by functional blocks that take part in process of receiving telecommands from ADPMS and forming housekeeping

	telemetry replies, executing control loops on hardware being controlled by CCB, gathering housekeeping information, and involves: UART interface, processor core, SDRAM memory controller and memory itself, GPIO and SPI control and communication peripherals inside GR712RC (depicted on Figure 44)
SPS data	this path is formed by functional blocks that take part in process of gathering Shadow Position Sensor data, processing it and reporting back to ADPMS as a partial input data for whole fine guidance and navigation activities of flight in formation, and involves: UART interface, processor core, SDRAM memory controller and memory itself, GR712RC SpaceWire and FPGAs Space Wire, SPI (SPS communication) and SPS data processing IP-cores (depicted on Figure 45).
science data	this path is formed by functional blocks that take part in process of scientific data acquisition, buffering, selection, compression and feed to ADPMS, and involves: processor core, SDRAM memory controller and memory itself, two GR712RC SpaceWire peripherals (for FPGA and CEB communication), FPGAs Space Wire, compression engine and it's input and output data handling blocks, FPGA memory controller and SRAM cache for scientific telemetry packet assembly and PacketWire transmitter IP-core for ADPMS data dump. (depicted on Figure 46)

Table 40 Identified CCB execution paths

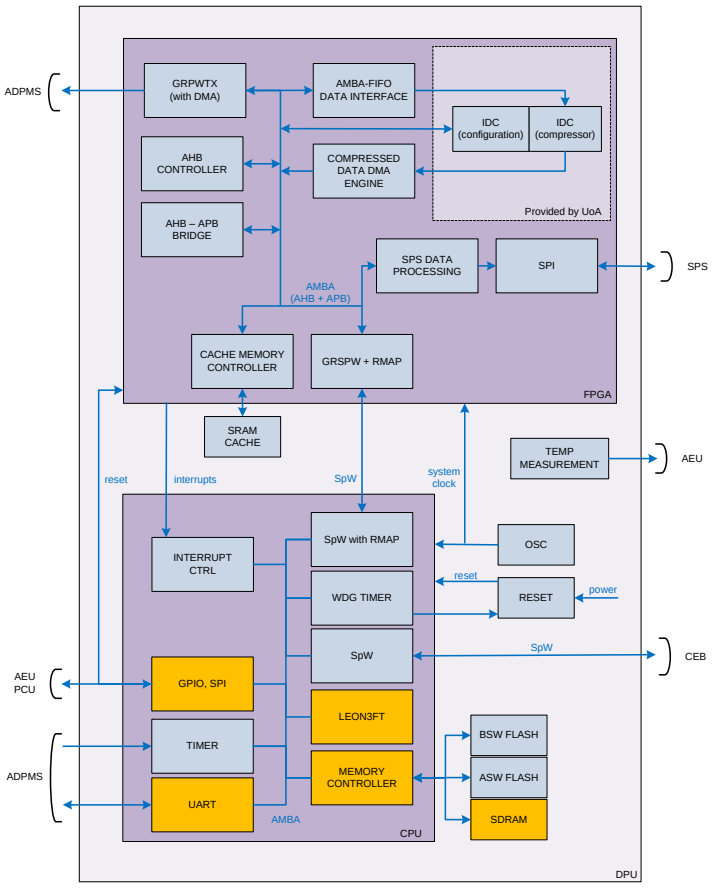

Figure 44 TC/TM control chain

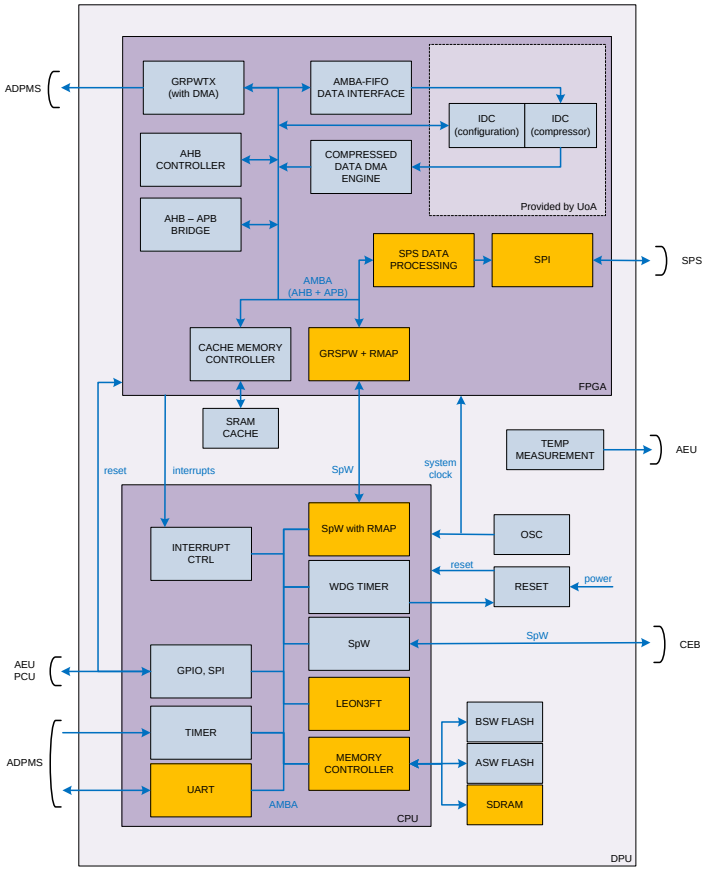

Figure 45 SPS data chain

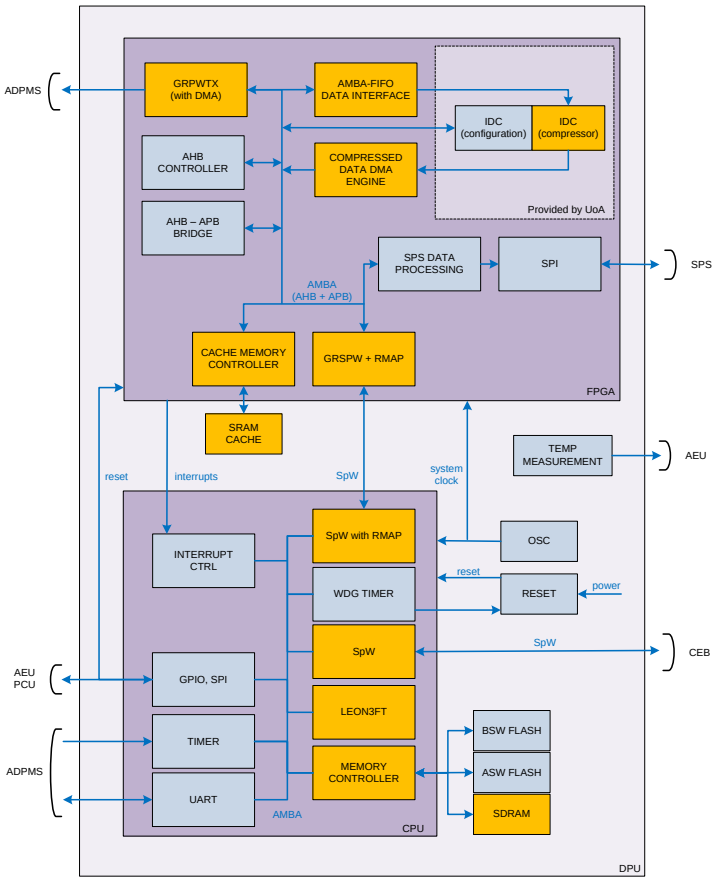

Figure 46 Science data chain

7.4 Standard reliability analysis of DPU CCB

Having all these functional block error rates allows to perform initial check of system error rate. Reliability Block Diagrams (RBDs) is the simplest way of rough checking reliability or unreliability of system. Each block on RBD represents a functional module. Blocks connected in series and parallel, represent a relation of how failures in given block affect the bigger whole. Calculation of resulting reliability or unreliability of control chain or execution path is straightforward, as "success" of control path is equivalent to "successes" of individual functional blocks that form the path. Let P_n be probability of functional block *n* error, R_n be reliability of functional block *n*, R_s be reliability ("success") of execution path, then:

$$R_i = 1 - P_i \, , i \in (n, s)$$

$$R_s = \prod_{i \in path} R_i$$

Calculations based on functional block error rates from Table 39 and paths definition from Table 40 brings path error rate estimation as in table below:

Chain	Error rate [chain^{-1} * day^{-1}]	Error rate (chain^{-1} * year^{-1})
TC/TM control	1,5900E-05	5,8033E-03
SPS data	2,4597E-04	8,9780E-02
science data	6,2870E-04	2,2948E-01
DPU error rate envelope	7,1200E-04	2,5988E-01

Table 41 Functional paths error rates

It is easy to notice that main CCB control chain (TC/TM), realized fully by GR712RC microprocessor exhibit low expected error rate. This is due to fact that microprocessor is third best radiation hardened device, connected with SDRAM (second best radiation hardening) & Flash (best radiation hardening).

Paths that utilize FPGA logic are more prone to errors generated by latched SETs in combinatorial part of the device. SPS data path, crucial, as being part of guidance and navigation control loop of whole satellite constellation has error rate of close to 1 error for 10 years (of 1/10 error per year) which seems to be acceptable (SPS is responsible for fine pointing part of formation flying).

Scientific data path, as it operates using most of CPU and FPGA resources, has error rate close to total envelope error rate estimated for DPU (case when any kind of error within any functional block causes DPU error) which is bit less than 0.3 errors per year or one error every three and a half years of operations on PROBA-3's Highly Elliptical Orbit.

Above calculations are valid when assuming that CCB / DPU constantly on, which is clearly not the case of CCB nominal operation, but in fact, CCB will be on for about quarter of orbital time, when close to apogee and where Earth's magnetic shielding is weaker. In author's opinion, for rough, pessimistic, estimation it is safer to leave it this way.

7.5 Dynamic reliability analysis of DPU CCB

It is best to start dynamic reliability analysis of CCB DPU by verification whether Petri Net based modeling yields similar results as classic methods. Two Petri Nets from Figure 47 are used for this purpose. Smaller one, in the bottom of mentioned figure, uses two places P_TCTM_chain_check and P_TCTM_chain_check_failed and an exponential firing probability density transition T_TCTM_chain_fail to simulate DPU unreliability (probability of failure in time) based on TCTM chain error rate (per day), from Table 40, calculated in classic way.

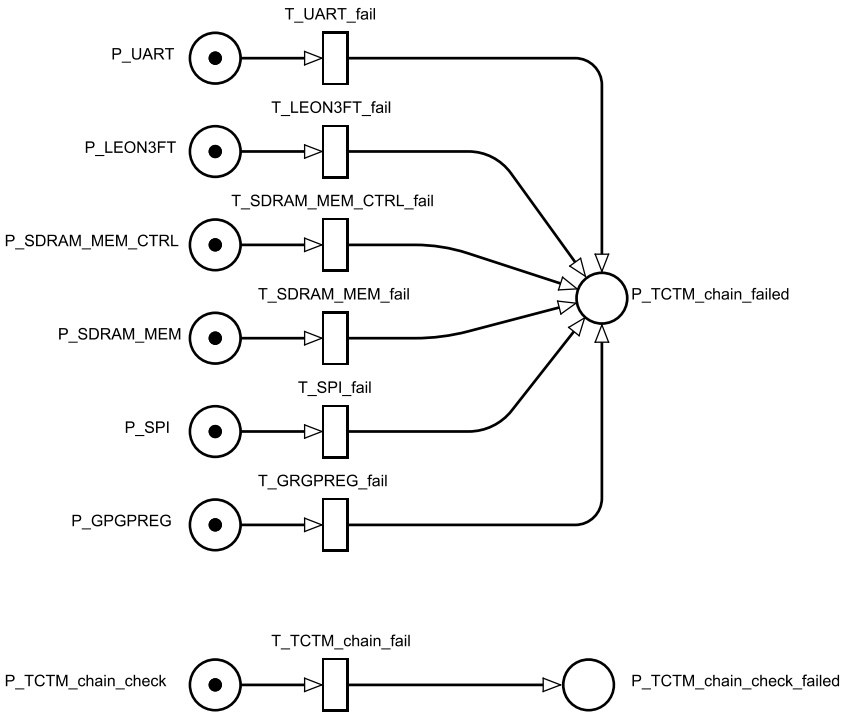

Figure 47 Petri Net modeling TCTM functional chain

Larger net, in the top of Figure 47, is constructed in a way, that each of functional blocks, that form, in this example TCTM chain, contributes separately its own failure rate. So token present in P_UART shall be represented as GR712RC UART functional block operating seamlessly, transfer of any tokens to P_TCTM_chain_failed is interpreted as a failure in one of the building blocks of TCTM chain, and probability of firing of any of transitions, hence,

fault occurrence, is based on well-known exponential probability density function with mean time to fire taken from Table 39.

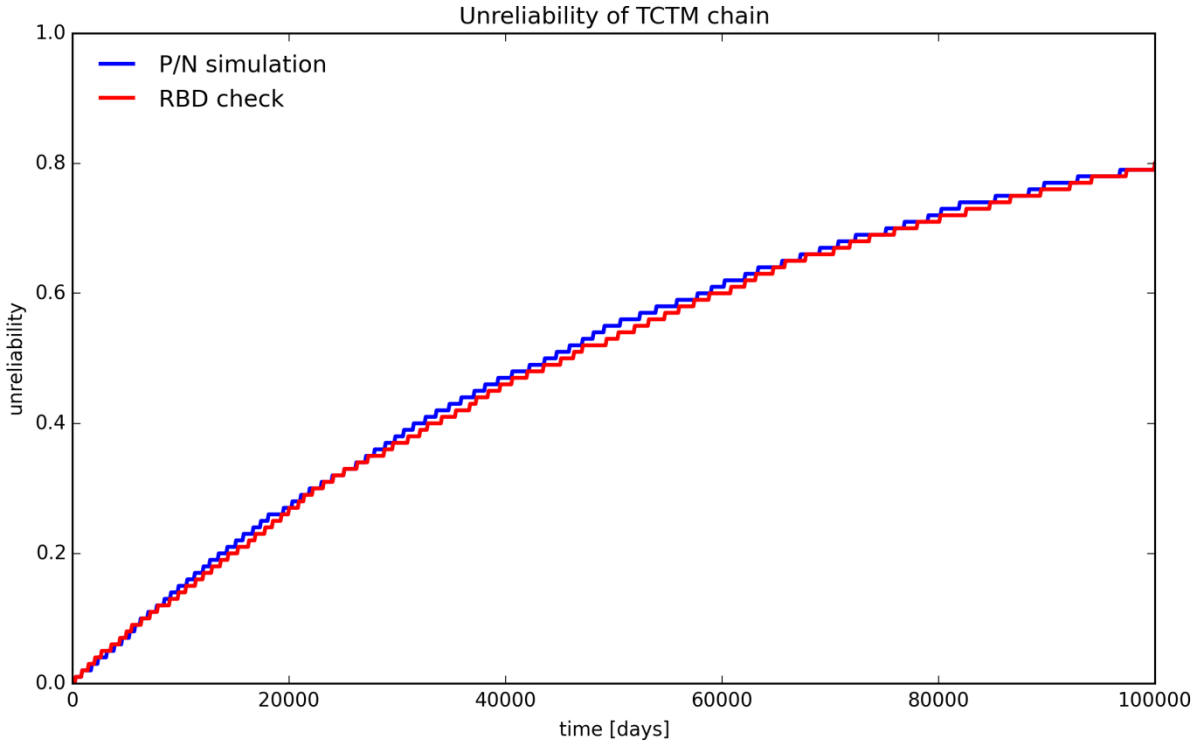

Figure 48 Petri Nets simulation result for TCTM chain

Result of both Petri Nets simulation is shown on Figure 48. Red curve represents the simulation outcome of Petri Net providing reference to Reliability Block Diagram ("classic" or "static" method). Blue curve represents simulation outcome of Petri Net employing all the functional blocks of TCTM chain. It is clearly visible that both curves are mutually corresponding – which is an experimental confirmation that both static and dynamic methods yields comparable result on fundamental, simplistic level.

Dynamic reliability Petri Net model from Figure 47 can be easily extended to contain all the previously identified functional chains in very similar fashion. Such Petri Net, with TCTM, SPS and Scientific chains, as well as, global CCB reliability envelope is shown Figure 49. Nets corresponding to each of functional chains are colored for clarity. Blue is the TCTM chain, red is the SPS chain and green is the Scientific chain. Tokens have the same meaning as in previous Petri Net. Failure of each of functional chain is also considered a CCB failure in order to create reliability envelope for whole CCB DPU (in place P_CCB_chains_failed). It is done simply by adding an immediate transition from each of places indicating given chain failure.

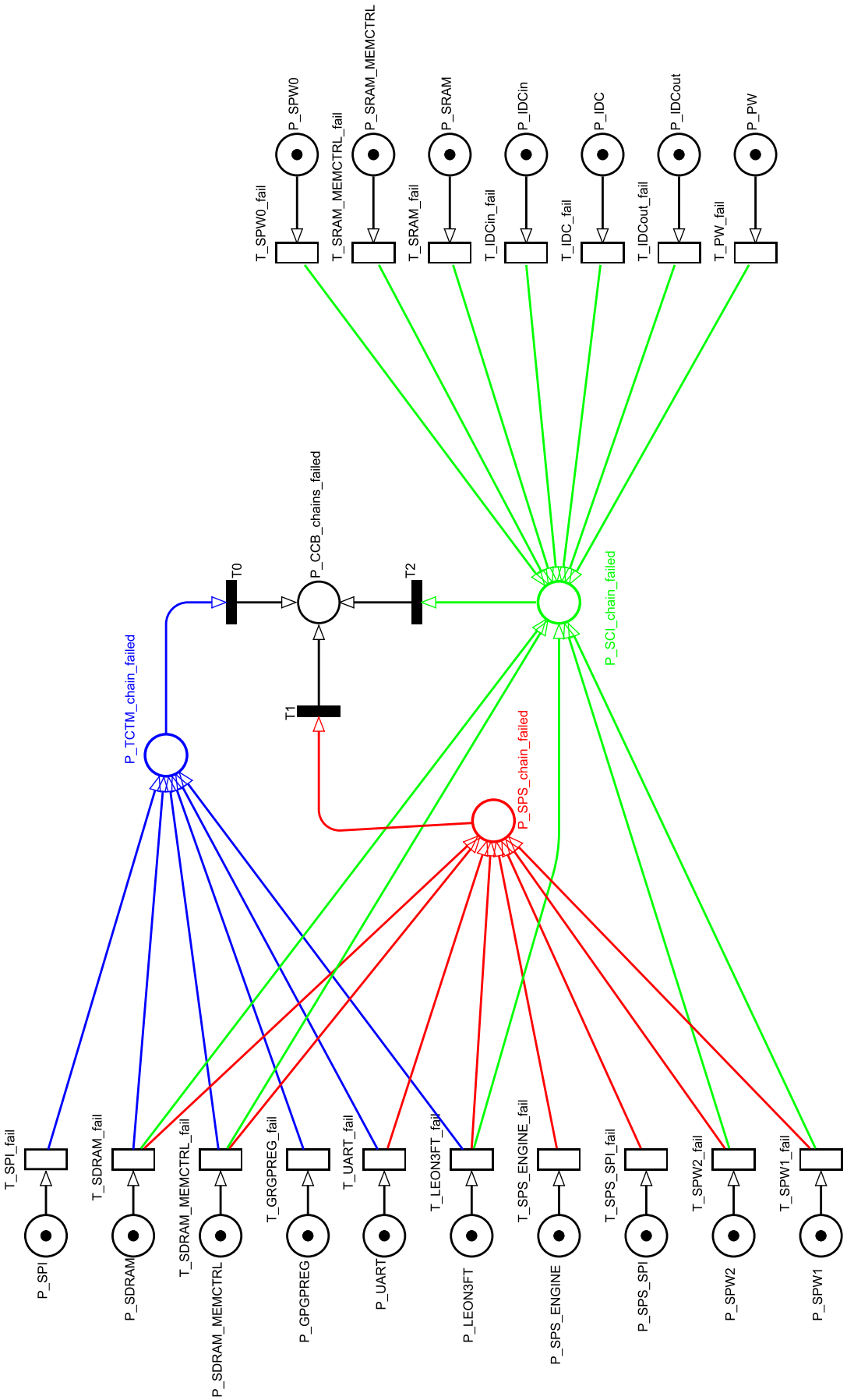

Figure 49 Petri Net modeling CCB DPU functional chains

Figure 50 shows results of simulation of net from Figure 49. Color scheme is maintained. Presented simulation results allow to identify which functional chains are major contributors to overall system unreliability. It is clear now that Scientific functional chains is the main source of faults that affects system as a whole. Reason for that is twofold. On one hand, scientific chain is the most complex (longest) one, which involves most of the existing DPU functional blocks. On the second hand, many of these functional blocks are implemented in FPGA which is the weakest part of system due to relatively high effective upset rates on logic and embedded RAM modules. Relatively high unreliability of scientific chain doesn't necessarily mean high system fault rate in more absolute measures. Readers attention is pointed to fact that unreality on Figure 50 is plotted against days. Taking into account that PROBA-3 mission time is 2 years or about 730 days, then expected CCB unreliability can be read from the figure, and is about 40%. In other words, this is the probability that the CCB will exhibit a fault in 2 years operation time.

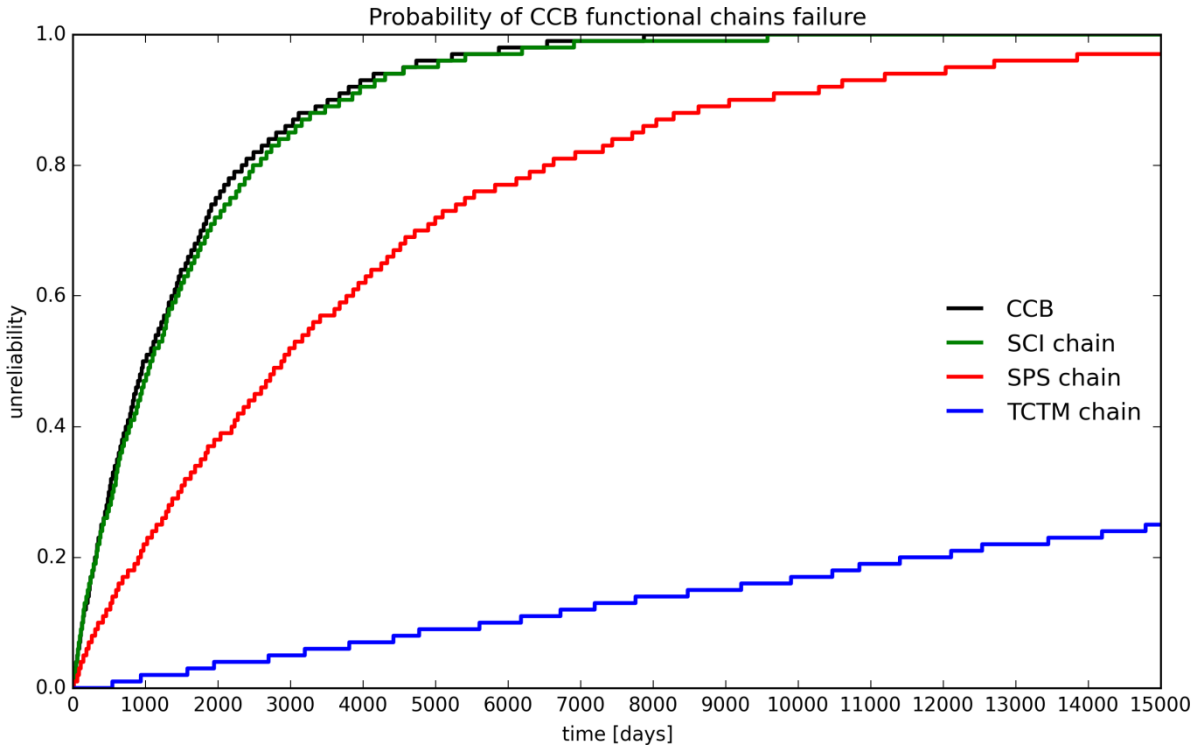

Figure 50 Petri Nets simulation result for CCB DPU chains

At this stage, it is possible to model even higher level of abstraction and incorporate the cold redundancy that is implemented in CCB. Such example redundancy model, useful for CCB DPU analysis is proposed on Figure 51. CCB DPU redundancy model accurately reflects cold redundancy operation philosophy, in which first unit, called nominal, is switched on and operated until it permanently fails. After nominal unit failure (and careful anomaly

investigation to ensure the owners of the equipment that further action are safe) redundant unit is switch on and operated until if fails, rendering the equipment inoperable.

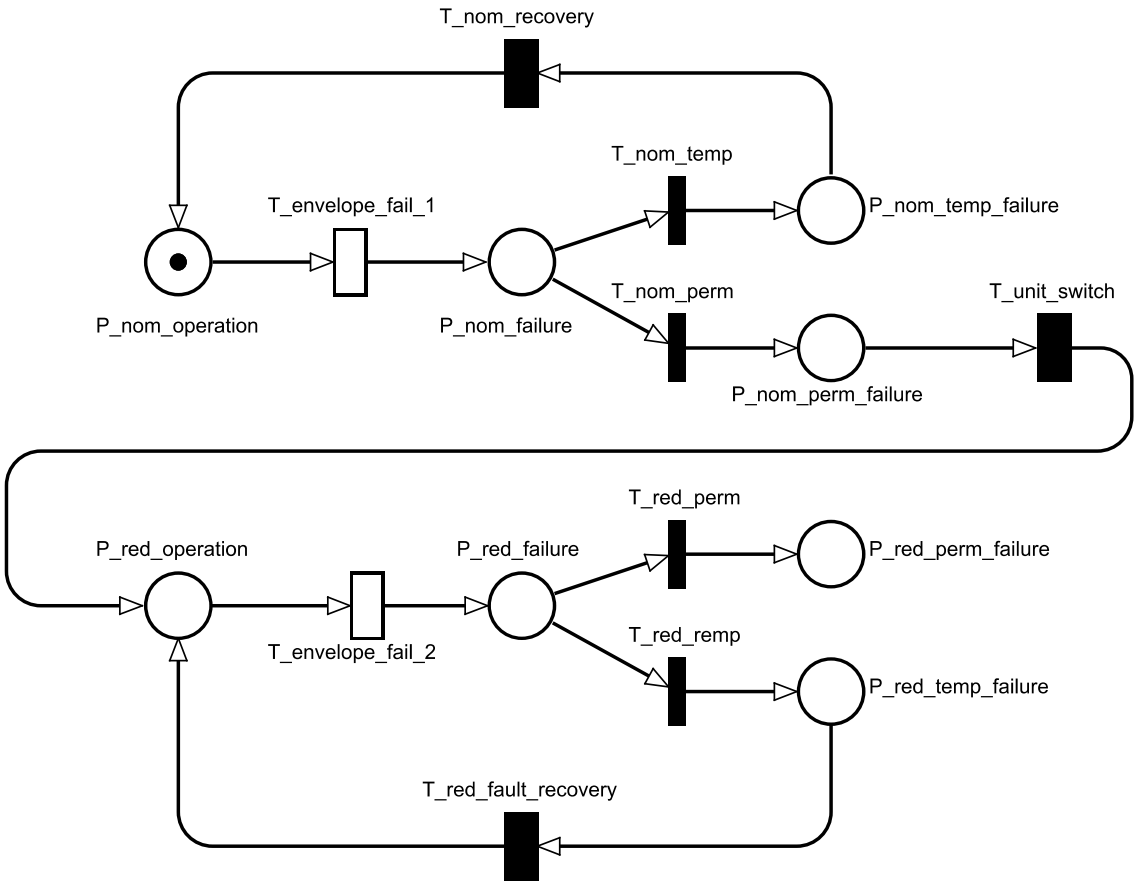

Figure 51 Petri Net modeling CCB DPU redundancy

Attentive reader will find this description transferred directly to abovementioned Petri Net on Figure 51. Presence of token in given place is interpreted as CCB being in this mode (active, or faulty, nominal or redundant). Place P_nom_operation represents configuration when CCB nominal branch is on. This could change if this branch fails, that is transition T_envelope_fail_1 fires (its firing rate is the CCB envelope fault rate obtained in previous step (Figure 49 and Figure 50) and moves the token to place P_nom_failure representing failed nominal branch. Token will not stay in this place long as it will be moved by one of two immediate transitions, either T_nom_temp (representing temporary, recoverable faults) or T_nom_perm (representing permanent branch failure). The probability of each of these two immediate transitions firing is user defined depending on occurrence ratio of permanent and temporary faults, if known at all. In case of present CCB DPU analysis, that ratio is 0.5 so both transitions have the same likelihood of firing, which is very conservative approach: permanent faults may happen to CCB but are very, very unlikely i.e.: SEL have extremely

low cross-section. Moreover latch-up in very unlikely case of occurrence shall be removed by power supply overcurrent protection.

In case that nominal branch enter temporary failure mode, system is switched on after, deterministic and defined period used for telemetry analysis and engineering investigation. Presented model assumes 60 days as firing delay of T_nom_recovery. In similar fashion, deterministic transition of T_unit_switch models the time necessary for mindful and careful switchover to redundant branch (also 60 days, but the number is in fact arbitrary as there are no known to author publications that elaborate on average times of after anomaly recovery). Redundant branch, as being architecturally and operationally identical to nominal one, is modeled in exactly the same manner as the latter and as described above.

Redundancy modeling Petri Net from Figure 51 simulation results are shown on following figures. Figure 52 and Figure 53 show the reliability (probability that branch will be operating at a time) of nominal and redundant branches respectively. Interestingly, the figure depicting redundant branch reliability takes into account the fact that it is unlikely for redundant branch to be switched on early after mission start as nominal branch is unlikely to fail fast. Figure 54 shows the probability that either nominal or redundant branch of CCB is on, evidencing significant increase in system dependability (bit less than facto of 3).

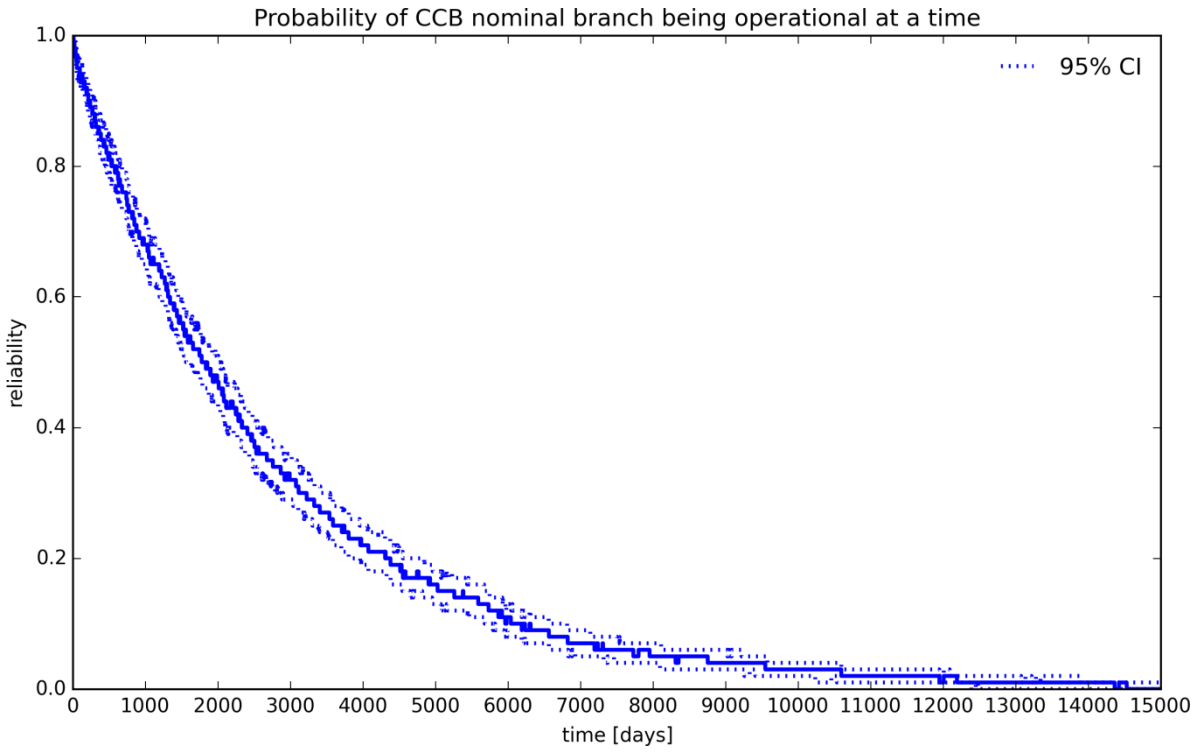

Figure 52 Petri Net simualtion of CCB nominal branch reliability

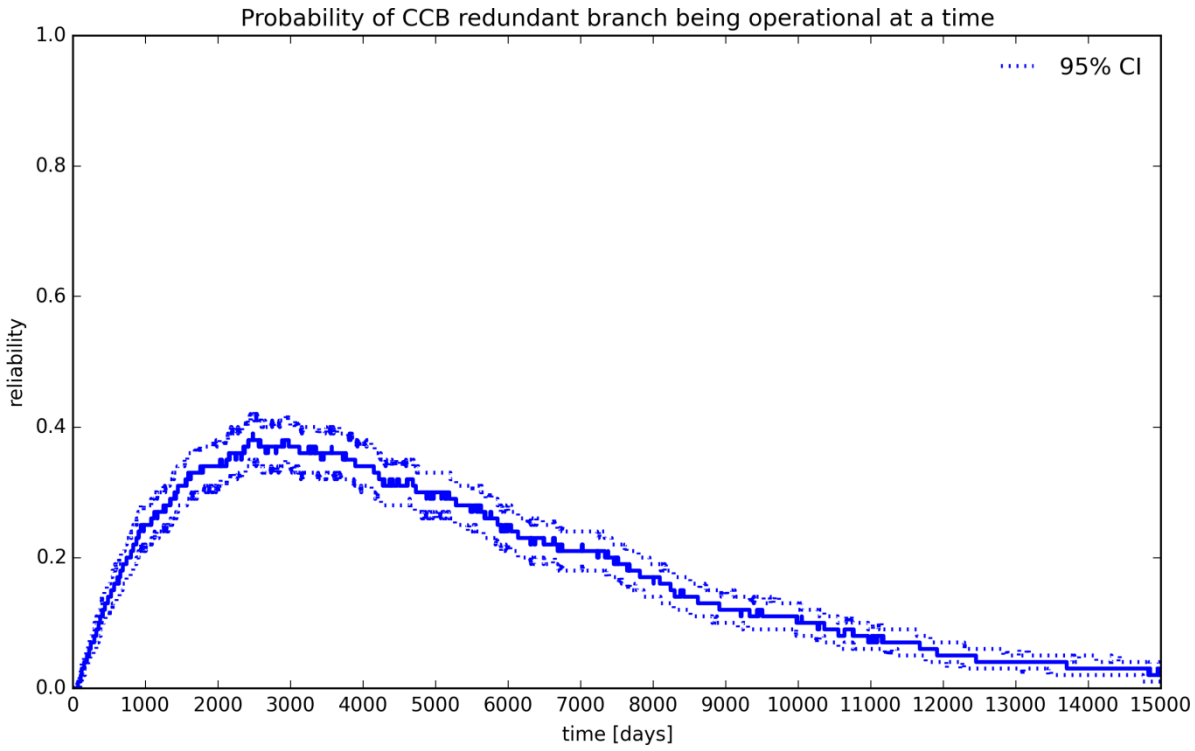

Figure 53 Petri Net simulation of CCB redundant branch reliability

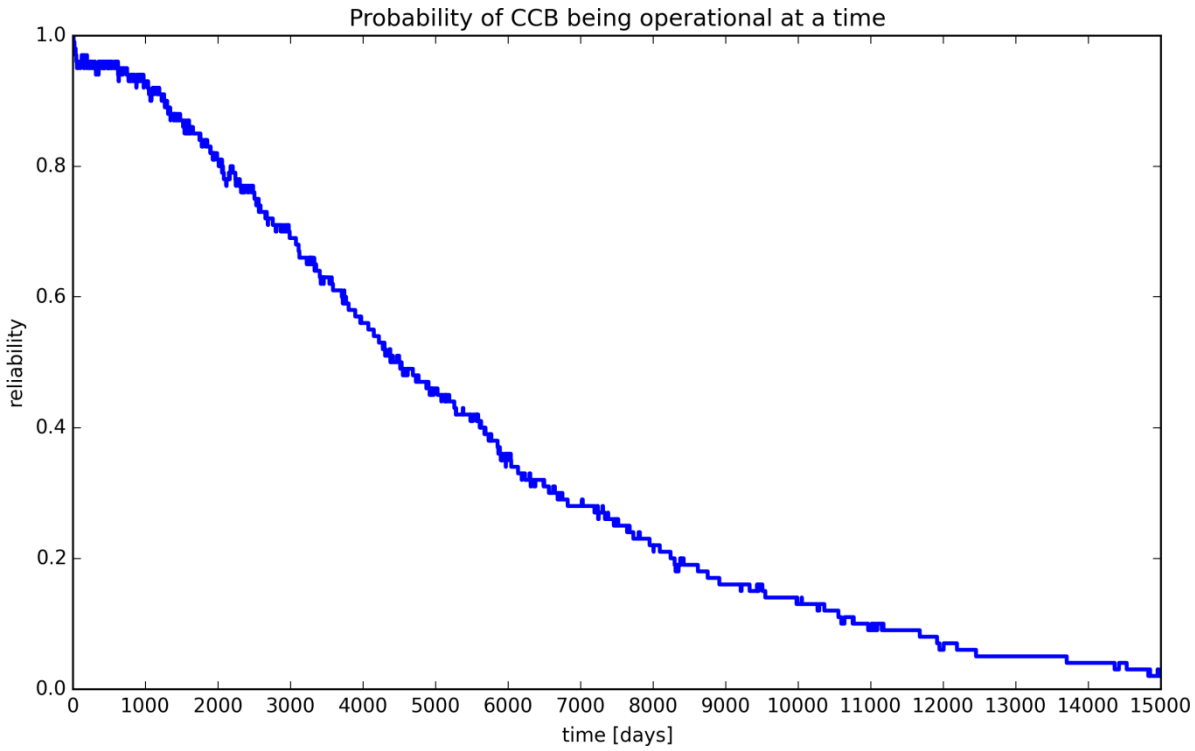

Figure 54 Resulting reliability of CCB both branches

Readers attention is also pointed to result that CCB, during its envisioned orbital lifetime of 2 years, has about 95% probability of successful operation (top-left part of Figure 54).

7.6 Reliability analysis results assessment

In very similar fashion as in case of CCB DPU performance analysis, the DPU reliability has been evaluated. First thing, a block of interest fault rates in P3 HEO orbit have been calculated and functional chains have been identified. With this basic information, a classic analysis has been done, using RBD approach. This has defined of baseline reference of what are expected reliability figures for PROBA-3 CCB DPU.

In next step a dynamic model, using Petri Net has been introduced. First simulation goal was to reproduce the classic results – to perform rough sanity check. Next, more advance model, allowed to simulate the expected reliability of each of functional chains, to get the envelope reliability data and to visualize how each of chains contribute to overall system reliability.

The last reliability model analysis added system redundancy and measured how it affects resulting CCB DPU global failure rate. Redundancy model, if filled with trustworthy data on time spent on anomaly investigation and contingency action and also on proportion of temporary upsets to permanent faults, is ready to provide user information on modeled system availability, mean time to failure, mean time t repair and mean time between failures.

What is worth emphasizing, is that presented models simulation time, in order to show meaningful results (i.e.: significant decrease in system reliability), has to be run for for several thousands of days while nominal PROBA-3 mission will be little bit more than 700 days, so at least order of magnitude more. This is due to fact that components that are envisioned for use in CCB DPU are of highest possible quality (MIL-STD or ECSS) qualified and component is in radiation hardened version. Therefore the expected SEU rates, are very low. Also, SELs are very unlikely with threshold levels well above 100 MeV*cm^2/mg.

8 Summary

The following chapter contains summary of whole presented dissertation, including:

- recapitulation of methodology that has been proposed, hypothesis that has been stated and goals that has been set
- assessment of these statements in light of provided evidence and results
- critical discussion of obtained results
- ideas on how presented work could lead to further developments in the discipline of systems engineering and what improvement are still necessary

8.1 Results review

Presented dissertation focuses on new and efficient ways of modeling complex avionic systems that are used in unmanned space vehicles, namely scientific probes. The dissertation is a side effect of work performed in CBK PAN on PROBA-3 ASPIICS Coronagraph Control Box, mainly in field of general system engineering but also in more detailed fashion: reliable, spaceborne computer system design and verification.

Dissertation is intended with all the necessary background information regarding challenges of complex system design and how to tackle complexity itself. Also, current avionics satst-of-the-art approaches are introduced, together with information on space equipment design peculiarities, followed right after by general overview on modeling methodologies present and used in modern aerospace and space industry. Additionally, reader is introduced to ASPIICS CCB architecture and components base, as well as, its key reliability and performance requirement that are supposed to guarantee the dependability of a system as a whole.

Main substantive contents of the dissertation are:

- holistic, mixing static and dynamic system features, modeling approach (chapter 5.1)
- guidelines on how to implement the methodology in practice (chapter 5.3)
- CCB DPU performance modeling (chapter 6), including:
    - classic performance analysis
    - PN performance model for plausibility check
    - more detailed PN models unveiling deep system features (bus blocking and preemption, buffer filling, control and data flow), achieving Goal 1.
- CCB DPU reliability modeling (chapter 7), including:

- classic reliability evaluation
- PN reliability model to classic method comparison
- PN detailed reliability model including CCB cold redundancy and proving robustness of proposed instrument controller architecture and implementation, achieving Goal 2.

Copyrights related to: the modeling methodology and its application guidelines, all PN models of CCB DPU, PN simulation results and CCB DPU architecture (with exception of component base selection where other CBK personnel was involved as well) solely belong to author of this dissertation.

Additionally, in context of the dissertation and present work since 2014, five (three regarding ASPIICS development and two regarding modeling of space equipment using Petri Nets) original papers has been published in peer reviewed conference proceedings (details in Appendix D).

8.2 Final conclusions

In the light of presented evidence **hypothesis** stated in chapter 5.2 **is confirmed**.

Work and results presented in chapters 6 and 7 clearly shows that, indeed, it is possible to:

…design and evaluate a dependable (performance and reliability measures or attributes have been simulated) *on-board control and data processing unit for unmanned spacecraft* (PROBA-3 CCB DPU) *utilizing system model* (static and dynamic system descriptions have been expressed in each of Petri Nets models) *associating environmental* (radiation, scientific data streams) *and functional* (system architecture and components interconnection and characteristics) *information in Petri Net* (mathematical tool used for simulations).

Petri Nets, together with presented design methodology turned out to be very useful in visualizing various aspects of system behavior, confirming or falsifying engineers' "gut feelings". In particular there are several solid evidence supporting the usefulness of Petri Net based, dynamic models:

- DPU performance model revealed that DPU has to be treated as streaming, not buffering, device as in worst-case operation scenario of "CME-Watch" there will be no time to send all scientific data tiles after the end of observations

- with respect to last point, DPU amount of on-board SDRAM memory has been reduced significantly, which in terms of Flight Units delivery is equivalent of buying 9 SDRAM memory modules less (savings of more than 50 000 € in components costs)
- DPU performance model analysis provided insight in how the Application Software controlling the DPU manage the scientific data flow and circulation of tiles from CEB, to compression engine and to ADPMS Mass Memory Modules.
- simulated DPU reliability model provided clear visualization of ratio that each of functional chains contributes to overall CCB DPU reliability
- DPU reliability model, extended to contain redundancy and recovery times, offers much greater insight into expected mean time to event (failure, repair) or availability values, than the classic modeling

There are also some shortcoming of presented analyses that shall be duly reported:

- DPU Petri Net performance model in order to simulate the tile transfer uses enormous number of tiles which, in turn, immediately explodes the possible state space that has to be simulated – in today's abundance of computer processing power is not an issue preventing the user to obtain meaningful results but definitely it's not an elegant approach
- DPU Petri Net reliability model, when fed the faults rates of DPU components does not yield any meaningful results as components used for DPU construction are simply too radiation hardened for their bits to flip. Also, if we bear in mind that ASPIICS will only operate only quarter of its orbital time and fault rates are calculated for device operating all the time then one shall not expect any to fault to happen disregarding modeling methodology in use (although it is better to err on safe side). That is basic reason why fault occurrence simulation time has been stretched to ridiculous 15000 days while discussed mission time is 720 days and real, cumulated, operating time of ASPIICS is about 180 days.

8.3 Way forward

Petri Nets are not new to industry. They are used, not very widely, but with increasing importance in logistics, network and computer architectures design and analysis, workflow systems. Although, so far, the Petri Nets, haven't found their way as a viable and useful design support in space systems design.

In author's opinion, presented work is a small step towards providing new system engineering tool for use in aerospace or space industry. It is definitely long and tedious process as space business is most conservative, change reluctant, economy branch that is out there. In space and aerospace, every harmonization process, product introduction or project development takes enormous amount of time and effort.

All in all, time and effort, spent on pushing additional modeling tool as Petri Net is likely to pay back in the future. Any kind modeling, if allows to prototype the concept early, in order to fail and pivot or to consolidate and move forward without doubt, is an invaluable support, that is directly traceable to savings in time and money spent.

"Essentially, all models are wrong, but some are useful"

George E. P. Box

Quote of famous English statistician accurately describes the limitation that are inherent to any kind of modeling and in the underline the advantages that are offered by early and incremental modeling in process of design, manufacturing, assembly, integration, verification and validation of complex system.

There are several next steps that shall be taken to popularize Petri Nets a modeling tool in space domain:

- Petri Net tool set has to be improved (tools in a sense of net capture and simulation software not underlying mathematical principles) as at the moment it is academic grade level (stability, user interface, limitations not meeting industrial standards)
- correlation of models, simulation results and physical reliability and performance measures must be collected to ensure the users that modeling results are to be trusted
    - initially, presented modeling methodology shall be used for trade-off, comparative, analysis of various solution, emphasizing trends rather than providing absolute values of interesting system measures
    - in time, gathered experience and analyses heritage shall enable the absolute system measures calculations (especially important for reliability / availability)

9 References

[1] “Dictionary and Thesaurus | Merriam-Webster.” [Online]. Available: http://www.merriam-webster.com/. [Accessed: 30-Aug-2016].

[2] A. D. Hall, *A Methodology for Systems Engineering*. Van Nostrand, 1962.

[3] P. B. A. II, *A Framework for Complex System Development*, 1 edition. Boca Raton: CRC Press, 2000.

[4] E. Rechtin, *Systems Architecting: Creating and Building Complex Systems*. Prentice Hall, 1991.

[5] D. G. Firesmith, P. Capell, C. B. Hammons, D. Latimer, T. Merendino, and D. Falkenthal, *The Method Framework for Engineering System Architectures*. Boca Raton: Auerbach Publications, 2008.

[6] A. Kossiakoff and W. N. Sweet, *Systems Engineering Principles and Practice*, 1 edition. New York: Wiley-Interscience, 2002.

[7] C. Girault and R. Valk, *Petri Nets for Systems Engineering: A Guide to Modeling, Verification, and Applications*. Springer Science & Business Media, 2003.

[8] G. Magistrati, “Technical Dossier on Data Systems and On-Board Computers.” European Space Agency, 06-Jan-2012.

[9] J. Miró, “Technical Dossier on Avionics Embedded Systems.” European Space Agency, 12-Jan-2011.

[10] “SAVOIR Functional Reference Architecture.” European Space Agency, 10-Feb-2015.

[11] M. Suess, R. Trautner, R. Vitulli, J. Ilstad, and D. Thurnes, “Technical Dossier on On-Board Payload Data Processing.” European Space Agency, 08-Feb-2012.

[12] “3D Plus.” [Online]. Available: http://www.3d-plus.com/. [Accessed: 30-Aug-2016].

[13] J. Eickhoff, *Onboard Computers, Onboard Software and Satellite Operations: An Introduction*, 2012 edition. Berlin ; New York: Springer, 2011.

[14] “Atmel Rad Hard Components.” [Online]. Available: http://www.atmel.com/products/rad-hard/default.aspx. [Accessed: 30-Aug-2016].

[15] “Cobham Gaisler GR740.” [Online]. Available: http://www.gaisler.com/index.php/products/components/gr740. [Accessed: 30-Aug-2016].

[16] R. L. Alena, J. P. Ossenfort, K. . Laws, A. Goforth, and F. Figueroa, “Communications for Integrated Modular Avionics,” in *2007 IEEE Aerospace Conference*, 2007, pp. 1–18.

[17] N. C. Audsley and M. Burke, “Distributed fault-tolerant avionic systems-a real-time perspective,” in *1998 IEEE Aerospace Conference*, 1998, vol. 4, pp. 43–60 vol.4.

[18] P. Binns, “A robust high-performance time partitioning algorithm: the digital engine operating system (DEOS) approach,” in *Digital Avionics Systems, 2001. DASC. 20th Conference*, 2001, vol. 1, p. 1B6/1-1B6/12 vol.1.

[19] P. B. Hugge and J. D. Lang, “Advanced design for quality avionic systems: a new systems development guide,” in , *AIAA/IEEE Digital Avionics Systems Conference, 1994. 13th DASC*, 1994, pp. 52–57.

[20] E. Schüler and M. Weinhardt, “XPP-III,” in *Dynamic System Reconfiguration in Heterogeneous Platforms*, N. S. Voros, A. Rosti, and M. Hübner, Eds. Springer Netherlands, 2009, pp. 63–76.

[21] J. Roselló, P. Silvestrin, G. L. Risueño, R. Weigand, J. V. Perelló, J. Heim, and I. Tejerina, “AGGA-4: Core device for GNSS space receivers of this decade,” in *2010 5th ESA Workshop on Satellite Navigation Technologies and European Workshop on GNSS Signals and Signal Processing (NAVITEC)*, 2010, pp. 1–8.

[22] F. Koebel and J. F. Coldefy, “SCOC3: a space computer on a chip,” in *2010 Design, Automation Test in Europe Conference Exhibition (DATE 2010)*, 2010, pp. 1345–1348.

[23] “Xilinx Space Qualifed FPGAs.” [Online]. Available: https://www.xilinx.com/applications/aerospace-and-defense/space.html. [Accessed: 30-Aug-2016].

[24] “Microsemi Space Qualifed FPGAs.” [Online]. Available: http://www.microsemi.com/products/fpga-soc/rad-tolerant-fpgas. [Accessed: 30-Aug-2016].

[25] A. Fernández León, “Technical Dossier on Microelectronics: ASIC and FPGA.” European Space Agency, Jan-2012.

[26] J.-F. Castet and J. H. Saleh, “Satellite and satellite subsystems reliability: Statistical data analysis and modeling,” *Reliab. Eng. Syst. Saf.*, vol. 94, no. 11, pp. 1718–1728, Nov. 2009.

[27] J.-F. Castet and J. H. Saleh, “Beyond reliability, multi-state failure analysis of satellite subsystems: A statistical approach,” *Reliab. Eng. Syst. Saf.*, vol. 95, no. 4, pp. 311–322, Apr. 2010.

[28] B. F. James, O. W. Norton, and M. B. Alexander, “The natural space environment: Effects on spacecraft,” Nov. 1994.

[29] E. Daly and A. Zadeh, “Technical Dossier on Radiation Environments, Monitoring, Effects, Analysis Tools, Testing, Experimentation and Methodologies.” European Space Agency, 20-Jan-2010.

[30] D. M. Harland and R. Lorenz, *Space Systems Failures: Disasters and Rescues of Satellites, Rocket and Space Probes*. Springer Science & Business Media, 2007.

[31] M. Tafazoli, “A study of on-orbit spacecraft failures,” *Acta Astronaut.*, vol. 64, no. 2–3, pp. 195–205, Jan. 2009.

[32] L. P. Sarsfield, “The Cosmos on a Shoestring,” 1998. [Online]. Available: http://www.rand.org/pubs/monograph_reports/MR864.html. [Accessed: 31-Aug-2016].

[33] “IEEE Standard for Environmental Specifications for Spaceborne Computer Modules,” *IEEE Std 11564-1997*, p. 0_1-, 1997.

[34] A. B. Campbell and A. R. Knudson, “Charge Collection Measurements for Energetic Ions in Silicon,” *IEEE Trans. Nucl. Sci.*, vol. 29, no. 6, pp. 2067–2071, Dec. 1982.

[35] R. D. Leach and M. B. Alexander, “Failures and anomalies attributed to spacecraft charging,” Aug. 1995.

[36] G. Santin, “Space environments and effects analysis for ESA missions,” *Nucl. Phys. B - Proc. Suppl.*, vol. 150, pp. 377–381, Jan. 2006.

[37] H. C. Koons, J. E. Mazur, R. S. Selesnick, J. B. Blake, and J. F. Fennell, “The Impact of the Space Environment on Space Systems,” Jul. 1999.

[38] K. Bedingfield, R. D. Leach, and M. B. Alexander, “Spacecraft System Failures and Anomalies Attributed to the Natural Space Environment,” Aug. 1996.

[39] JPL.NASA.GOV, “NASA JPL Papers & Presentations on Components and Radiation.” .

[40] J. R. Schwank, M. R. Shaneyfelt, J. A. Felix, P. E. Dodd, J. Baggio, V. Ferlet-Cavrois, P. Paillet, G. L. Hash, R. S. Flores, L. W. Massengill, and E. Blackmore, “Effects of Total Dose Irradiation on Single-Event Upset Hardness,” *IEEE Trans. Nucl. Sci.*, vol. 53, no. 4, pp. 1772–1778, Aug. 2006.

[41] H. J. Barnaby, “Total-Ionizing-Dose Effects in Modern CMOS Technologies,” *IEEE Trans. Nucl. Sci.*, vol. 53, no. 6, pp. 3103–3121, Dec. 2006.

[42] E. Daly, C. Stroom, and J. Sørensen, “Technical Dossier on Thermal and Space Environment Software Tools and Interfaces.” European Space Agency, 26-Mar-2002.

[43] L. Ding, H. Guo, W. Chen, Z. Yao, Y. Yan, D. Chen, A. Paccagnella, S. Gerardin, M. Bagatin, L. Chen, H. Sun, and R. Fan, “Analysis of TID Failure Modes in SRAM-Based FPGA Under Gamma-Ray and Focused Synchrotron X-Ray Irradiation,” *IEEE Trans. Nucl. Sci.*, vol. 61, no. 4, pp. 1777–1784, Aug. 2014.

[44] B. L. Gregory and C. W. Gwyn, “Radiation effects on semiconductor devices,” *Proc. IEEE*, vol. 62, no. 9, pp. 1264–1273, Sep. 1974.

[45] J. R. Srour and J. M. McGarrity, “Radiation effects on microelectronics in space,” *Proc. IEEE*, vol. 76, no. 11, pp. 1443–1469, Nov. 1988.

[46] G. C. Messenger and M. Ash, *Single Event Phenomena*. Springer Science & Business Media, 2013.

[47] V. Zajic and P. Thieberger, “Heavy ion linear energy transfer measurements during single event upset testing of electronic devices,” *IEEE Trans. Nucl. Sci.*, vol. 46, no. 1, pp. 59–69, Feb. 1999.

[48] M. D. Berg, K. A. LaBel, H. Kim, M. Friendlich, A. Phan, and C. Perez, “A Comprehensive Methodology for Complex Field Programmable Gate Array Single Event Effects Test and Evaluation,” *IEEE Trans. Nucl. Sci.*, vol. 56, no. 2, pp. 366–374, Apr. 2009.

[49] M. Alderighi, F. Casini, S. d’Angelo, M. Mancini, S. Pastore, and G. R. Sechi, “Evaluation of Single Event Upset Mitigation Schemes for SRAM based FPGAs using the FLIPPER Fault Injection Platform,” in *22nd IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems, 2007. DFT ’07*, 2007, pp. 105–113.

[50] G. Allen, L. D. Edmonds, G. Swift, C. Carmichael, C. W. Tseng, K. Heldt, S. A. Anderson, and M. Coe, “Single Event Test Methodologies and System Error Rate Analysis for Triple Modular Redundant Field Programmable Gate Arrays,” *IEEE Trans. Nucl. Sci.*, vol. 58, no. 3, pp. 1040–1046, Jun. 2011.

[51] P. E. Dodd and L. W. Massengill, “Basic mechanisms and modeling of single-event upset in digital microelectronics,” *IEEE Trans. Nucl. Sci.*, vol. 50, no. 3, pp. 583–602, Jun. 2003.

[52] H. Quinn, P. Graham, J. Krone, M. Caffrey, and S. Rezgui, “Radiation-induced multi-bit upsets in SRAM-based FPGAs,” *IEEE Trans. Nucl. Sci.*, vol. 52, no. 6, pp. 2455–2461, Dec. 2005.

[53] H. Baig, J.-A. Lee, and Z. A. Siddiqui, “A Low-Overhead Multiple-SEU Mitigation Approach for SRAM-based FPGAs with Increased Reliability,” *IEEE Trans. Nucl. Sci.*, vol. 61, no. 3, pp. 1389–1399, Jun. 2014.

[54] M. Alderighi, A. Candelori, F. Casini, S. D’Angelo, M. Mancini, A. Paccagnella, S. Pastore, and G. R. Sechi, “Heavy ion effects on configuration logic of Virtex FPGAs,” in *On-Line Testing Symposium, 2005. IOLTS 2005. 11th IEEE International*, 2005, pp. 49–53.

[55] G. Asadi, S. G. Miremadi, H. R. Zarandi, and A. Ejlali, “Fault injection into SRAM-based FPGAs for the analysis of SEU effects,” in *2003 IEEE International Conference on Field-Programmable Technology (FPT), 2003. Proceedings*, 2003, pp. 428–430.

[56] M. Berg, C. Poivey, D. Petrick, D. Espinosa, A. Lesea, K. LaBel, M. Friendlich, H. Kim, and A. Phan, “Effectiveness of internal vs. external SEU scrubbing mitigation strategies in a Xilinx FPGA: Design, test, and analysis,” in *9th European Conference on Radiation and Its Effects on Components and Systems, 2007. RADECS 2007*, 2007, pp. 1–8.

[57] M. Ceschia, M. Violante, M. S. Reorda, A. Paccagnella, P. Bernardi, M. Rebaudengo, D. Bortolato, M. Bellato, P. Zambolin, and A. Candelori, “Identification and classification of single-event upsets in the configuration memory of SRAM-based FPGAs,” *IEEE Trans. Nucl. Sci.*, vol. 50, no. 6, pp. 2088–2094, Dec. 2003.

[58] C. Bernardeschi, L. Cassano, and A. Domenici, “Formal approaches to SEU testing in FPGAs,” in *2013 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, 2013, pp. 209–216.

[59] H. Abbasitabar, H. R. Zarandi, and R. Salamat, “Susceptibility Analysis of LEON3 Embedded Processor against Multiple Event Transients and Upsets,” in *2012 IEEE 15th International Conference on Computational Science and Engineering (CSE)*, 2012, pp. 548–553.

[60] J. H. Adams, A. F. Barghouty, M. H. Mendenhall, R. A. Reed, B. D. Sierawski, K. M. Warren, J. W. Watts, and R. A. Weller, “CREME: The 2011 Revision of the Cosmic Ray Effects on Micro-Electronics Code,” *IEEE Trans. Nucl. Sci.*, vol. 59, no. 6, pp. 3141–3147, Dec. 2012.

[61] T. P. Ma and P. V. Dressendorfer, *Ionizing Radiation Effects in MOS Devices and Circuits*. John Wiley & Sons, 1989.

[62] P. J. Haas, *Stochastic Petri Nets*. New York, NY: Springer New York, 2002.

[63] G. Balbo, “Introduction to Generalized Stochastic Petri Nets,” in *Formal Methods for Performance Evaluation*, M. Bernardo and J. Hillston, Eds. Springer Berlin Heidelberg, 2007, pp. 83–131.

[64] S. Bernardi and S. Donatelli, “Stochastic Petri nets and inheritance for dependability modelling,” in *10th IEEE Pacific Rim International Symposium on Dependable Computing, 2004. Proceedings*, 2004, pp. 363–372.

[65] R. M. Fricks and K. S. Trivedi, *Modeling Failure Dependencies In Reliability Analysis Using Stochastic Petri Nets*. .

[66] M. Malhotra and K. S. Trivedi, “Dependability modeling using Petri-nets,” *IEEE Trans. Reliab.*, vol. 44, no. 3, pp. 428–440, Sep. 1995.

[67] C. Constantinescu, “Dependability evaluation of a fault-tolerant processor by GSPN modeling,” *IEEE Trans. Reliab.*, vol. 54, no. 3, pp. 468–474, Sep. 2005.

[68] S. M. M. Fernandes and P. R. M. Maciel, “Parameterized GSPN Model and Extended Dependability Block Diagram for Reliability Evaluation of Embedded Systems,” in *IEEE International Conference on Systems, Man and Cybernetics, 2006. SMC ’06*, 2006, vol. 4, pp. 3046–3051.

[69] R. Blommestijn and J. Fuchs, “Technical Dossier on System Modelling and Simulation Tools.” European Space Agency, 14-Jul-2012.

[70] J. Eickhoff, *Simulating Spacecraft Systems*. Springer Science & Business Media, 2009.

[71] S. Distefano and A. Puliafito, "Dependability Evaluation with Dynamic Reliability Block Diagrams and Dynamic Fault Trees," *IEEE Trans. Dependable Secure Comput.*, vol. 6, no. 1, pp. 4–17, Jan. 2009.

[72] S. Distefano and A. Puliafito, "Dynamic Reliability Block Diagrams VS Dynamic Fault Trees," in *2007 Annual Reliability and Maintainability Symposium*, 2007, pp. 71–76.

[73] S. Distefano and L. Xing, "A new approach to modeling the system reliability: dynamic reliability block diagrams," in *RAMS '06. Annual Reliability and Maintainability Symposium, 2006.*, 2006, pp. 189–195.

[74] S. Ha, N. Ku, M.-I. Roh, and J.-H. Cha, "Reliability Analysis Method Using Dynamic Reliability Block Diagram Based on DEVS Formalism," in *AsiaSim 2013*, G. Tan, G. K. Yeo, S. J. Turner, and Y. M. Teo, Eds. Springer Berlin Heidelberg, 2013, pp. 219–230.

[75] J. R. Belland and D. Wiseman, "Using fault trees to analyze safety-instrumented systems," in *2016 Annual Reliability and Maintainability Symposium (RAMS)*, 2016, pp. 1–6.

[76] E. Chung and J. S. Hanks, "Fault tree analyses as a tool for flight control system architecture design," in *2016 Annual Reliability and Maintainability Symposium (RAMS)*, 2016, pp. 1–6.

[77] Z. Hamza and T. Abdallah, "Mapping Fault Tree into Bayesian Network in safety analysis of process system," in *2015 4th International Conference on Electrical Engineering (ICEE)*, 2015, pp. 1–5.

[78] Y. Hiraoka, T. Murakami, K. Yamamoto, Y. Furukawa, and H. Sawada, "Method of Computer-Aided Fault Tree Analysis for High-Reliable and Safety Design," *IEEE Trans. Reliab.*, vol. 65, no. 2, pp. 687–703, Jun. 2016.

[79] A. Bouti and D. A. Kadi, "A state-of-the-art review of fmea/fmeca," *Int. J. Reliab. Qual. Saf. Eng.*, vol. 1, no. 4, pp. 515–543, Dec. 1994.

[80] B. C. Wei, "A unified approach to failure mode, effects and criticality analysis (FMECA)," in *Reliability and Maintainability Symposium, 1991. Proceedings., Annual*, 1991, pp. 260–271.

[81] Z. Taylor and S. Ranganathan, “Wiley: Designing High Availability Systems: DFSS and Classical Reliability Techniques with Practical Real Life Examples.” [Online]. Available: http://eu.wiley.com/WileyCDA/WileyTitle/productCd-1118551125,subjectCd-EE23.html. [Accessed: 31-Aug-2016].

[82] B. R. Haverkort and A. P. A. van Moorsel, “Using the probabilistic evaluation tool for the analytical solution of large Markov models,” in *, Proceedings of the Sixth International Workshop on Petri Nets and Performance Models, 1995*, 1995, pp. 206–207.

[83] J. Panerati, S. Abdi, and G. Beltrame, “Balancing system availability and lifetime with dynamic hidden Markov models,” in *2014 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, 2014, pp. 240–247.

[84] A. Gillespie, M. W. Monaghan, and Y. Chen, “Comparison modeling of system reliability for future NASA projects,” in *Reliability and Maintainability Symposium (RAMS), 2012 Proceedings - Annual*, 2012, pp. 1–7.

[85] P. A. Tobias and D. C. Trindade, *Applied Reliability*. Springer Science & Business Media, 1986.

[86] J. J. Marin and R. W. Pollard, “Experience report on the FIDES reliability prediction method,” in *Annual Reliability and Maintainability Symposium, 2005. Proceedings.*, 2005, pp. 8–13.

[87] P. Charpenel, F. Davenel, R. Digout, M. Giraudeau, M. Glade, J. Guerveno, N. Guillet, A. Lauriac, S. Male, D. Manteigas, R. Meister, E. Moreau, D. Perie, F. Relmy-Madinska, and P. Retailleau, “14th European Symposium on Reliability of Electron Devices, Failure Physics and AnalysisThe right way to assess electronic system reliability: FIDES,” *Microelectron. Reliab.*, vol. 43, no. 9, pp. 1401–1404, Sep. 2003.

[88] D. O. Koval, X. Zhang, J. Propst, T. Coyle, R. Arno, and R. S. Hale, “Reliability methods applied to The IEEE Gold Book Standard Network,” *IEEE Ind. Appl. Mag.*, vol. 9, no. 1, pp. 32–41, Jan. 2003.

[89] J. W. Harms, “Revision of MIL-HDBK-217, Reliability Prediction of Electronic Equipment,” in *Reliability and Maintainability Symposium (RAMS), 2010 Proceedings - Annual*, 2010, pp. 1–3.

[90] J. G. McLeish, “Enhancing MIL-HDBK-217 reliability predictions with physics of failure methods,” in *Reliability and Maintainability Symposium (RAMS), 2010 Proceedings - Annual*, 2010, pp. 1–6.

[91] M. Pecht, D. Das, and A. Ramakrishnan, “The IEEE standards on reliability program and reliability prediction methods for electronic equipment,” *Microelectron. Reliab.*, vol. 42, no. 9, pp. 1259–1266, Sep. 2002.

[92] T. Murata, “Petri nets: Properties, analysis and applications,” *Proc. IEEE*, vol. 77, no. 4, pp. 541–580, Apr. 1989.

[93] F. Bause and P. S. Kritzinger, *Stochastic Petri Nets: An Introduction to the Theory*. Vieweg+Teubner Verlag, 1996.

[94] M. Berg, C. Poivey, D. Petrick, K. LaBel, M. Friendlich, S. Stansberry, and H. Kim, “Risk Reduction for Use of Complex Devices in Space Projects,” *IEEE Trans. Nucl. Sci.*, vol. 54, no. 6, pp. 2137–2140, Dec. 2007.

[95] D. Caihong, “Application of Petri net to fault diagnosis in satellite,” *J. Syst. Eng. Electron.*, vol. 12, no. 2, pp. 92–96, Jun. 2001.

[96] C.-K. Chen, “A Petri net design of FPGA-based controller for a class of nuclear I&C systems,” *Nucl. Eng. Des.*, vol. 241, no. 7, pp. 2597–2603, Jul. 2011.

[97] J.-F. Ereau and M. Saleman, “Modeling and simulation of a satellite constellation based on Petri nets,” in *Reliability and Maintainability Symposium, 1996 Proceedings. International Symposium on Product Quality and Integrity., Annual*, 1996, pp. 66–72.

[98] H. J. Genrich and K. Lautenbach, “Special Issue Semantics of Concurrent ComputationSystem modelling with high-level Petri nets,” *Theor. Comput. Sci.*, vol. 13, no. 1, pp. 109–135, Jan. 1981.

[99] W. Reisig and G. Rozenberg, *Lectures on Petri Nets I: Basic Models: Advances in Petri Nets*. Springer Science & Business Media, 1998.

[100] W. Reisig and G. Rozenberg, *Lectures on Petri Nets II: Applications: Advances in Petri Nets*. Springer Science & Business Media, 1998.

[101] V. Volovoi, “Modeling of system reliability Petri nets with aging tokens,” *Reliab. Eng. Syst. Saf.*, vol. 84, no. 2, pp. 149–161, May 2004.

[102] R. Zurawski and M. Zhou, “Petri nets and industrial applications: A tutorial,” *IEEE Trans. Ind. Electron.*, vol. 41, no. 6, pp. 567–583, Dec. 1994.

[103] “GRLIB IP Core User’s Manual.” Cobham Gaisler, Jan-2016.

[104] D. Heynderickx, B. Quaghebeur, E. Speelman, and E. Daly, “ESA’s Space Environment Information System (SPENVIS) - A WWW interface to models of the space environment and its effects,” in *38th Aerospace Sciences Meeting and Exhibit*, American Institute of Aeronautics and Astronautics.

[105] M. Kruglanski, N. Messios, E. de Donder, E. Gamby, S. Calders, L. Hetey, and H. Evans, “Space Environment Information System (SPENVIS),” presented at the EGU General Assembly Conference Abstracts, 2009, vol. 11, p. 7457.

[106] M. Kruglanski, N. Messios, E. D. Donder, E. Gamby, S. Calders, L. Hetey, H. Evans, and E. Daly, “Last upgrades and development of the space environment information system (SPENVIS),” in *2009 European Conference on Radiation and Its Effects on Components and Systems (RADECS)*, 2009, pp. 563–565.

[107] “Solar Cycle Progression.” [Online]. Available: http://www.swpc.noaa.gov/products/solar-cycle-progression. [Accessed: 02-Sep-2016].

[108] E. Graas, J.-M. Gillis, and J.-S. Servaye, “PROBA-3 ASPIICS Coronagraph Control Box Requirements Specification.” Center Spatial de Liege, 24-Feb-2016.

[109] E. Graas, J.-M. Gillis, and J.-S. Servaye, “PROBA-3 ASPIICS Coronagraph Electrical Box - Coronagraph Control Box Image Data Handling User Requirements.” Center Spatial de Liege, 02-Dec-2016.

[110] S. Ilsen and R. Kassel, “PROBA-3 Payload PacketWire Data Interface Description.” Quinetiq, 12-Apr-2014.

[111] J.-M. Gillis and J.-S. Servaye, “PROBA-3 ASPIICS Coronagraph Electrical Box - Coronagraph Control Box Image Data Handling Performance Analysis.” Center Spatial de Liege, 31-Mar-2015.

[112] “GR712RC Dual-Core LEON3FT SPARC V8 Processor User’s Manual.” Cobham Gaisler, Jan-2016.

[113] E. Renotte, A. Alia, A. Bemporad, J. Bernier, C. Bramanti, S. Buckley, G. Capobianco, I. Cernica, V. Dániel, R. Darakchiev, M. Darmetko, A. Debaize, F. Denis, R.

Desselle, L. de Vos, A. Dinescu, S. Fineschi, K. Fleury-Frenette, M. Focardi, A. Fumel, D. Galano, C. Galy, J.-M. Gillis, T. Górski, E. Graas, R. Graczyk, K. Grochowski, J.-P. A. Halain, A. Hermans, R. Howard, C. Jackson, E. Janssen, H. Kasprzyk, J. Kosiec, S. Koutchmy, J. Kovačičinová, N. Kranitis, M. Kurowski, M. Ładno, P. Lamy, F. Landini, R. Lapáček, V. Lédl, S. Liebecq, D. Loreggia, B. McGarvey, G. Massone, R. Melich, A. Mestreau-Garreau, D. Mollet, Ł. Mosdorf, M. Mosdorf, M. Mroczkowski, R. Muller, G. Nicolini, B. Nicula, K. O'Neill, P. Orleański, M.-C. Palau, M. Pancrazzi, A. Paschalis, K. Patočka, R. Peresty, I. Popescu, P. Psota, M. Rataj, J. Rautakoski, M. Romoli, R. Rybecký, L. Salvador, J.-S. Servaye, C. Solomon, Y. Stockman, A. Swat, C. Thizy, M. Thomé, K. Tsinganos, J. Van der Meulen, N. Van Vooren, T. Vit, T. Walczak, A. Zarzycka, J. Zender, and A. Zhukov, "Design status of ASPIICS, an externally occulted coronagraph for PROBA-3," 2015, vol. 9604, p. 96040A–96040A–15.

[114] F. Sturesson, J. Gaisler, R. Ginosar, and T. Liran, "Radiation characterization of a dual core LEON3-FT processor," in *2011 12th European Conference on Radiation and Its Effects on Components and Systems (RADECS)*, 2011, pp. 938–944.

[115] M. Berg, H. Kim, M. Friendlich, C. Perez, C. Seidleck, K. LaBel, and R. Ladbury, "SEU Analysis of Complex Circuits Implemented in Actel RTAX-S FPGA Devices," *IEEE Trans. Nucl. Sci.*, vol. 58, no. 3, pp. 1015–1022, Jun. 2011.

[116] M. Berg, J.-J. Wang, R. Ladbury, S. Buchner, H. Kim, J. Howard, K. LaBel, A. Phan, T. Irwin, and M. Friendlich, "An Analysis of Single Event Upset Dependencies on High Frequency and Architectural Implementations within Actel RTAX-S Family Field Programmable Gate Arrays," *IEEE Trans. Nucl. Sci.*, vol. 53, no. 6, pp. 3569–3574, Dec. 2006.

[117] M. Berg, J. J. Wang, R. Ladbury, S. Buchner, H. Kim, J. Howard, K. LaBel, A. Phan, T. Irwin, and M. Friendlich, "An Analysis of Single Event Upset Dependencies on High Frequency and Architectural Implementations within Actel RTAX-S Family Field Programmable Gate Arrays," *IEEE Trans. Nucl. Sci.*, vol. 53, no. 6, pp. 3569–3574, Dec. 2006.

[118] S. Rezgui, J. J. Wang, Y. Sun, D. D'Silva, B. Cronquist, and J. McCollum, "SET characterization and mitigation in RTAX-S antifuse FPGAs," in *2009 IEEE Aerospace conference*, 2009, pp. 1–14.

[119] M. Berg, M. Friendlich, and K. Hak, “RTAX-S Field Programmable Gate Array Single Event Effects (SEE), High-Speed Test Plan- Phase I.” NASA Goddard Spaceflight Center Radiation Effects Group.

[120] J. J. Wang, “RTAX-S EDAC-RAM Single Event Upset Test Report.” Microsemi / Actel, 06-Apr-2004.

[121] “Using EDAC RAM for RadTolerant RTAX-S FPGAs and Axcelerator FPGAs.” Microsemi / Actel, 06-Feb-2014.

[122] “DETAIL SPECIFICATION 4Mbit SRAM – 512k x 8 – 5V SOP 44-08 – Package W1a Part Num: 3DSR4M08CS1647.” .

[123] “DETAIL SPECIFICATION 2Gbit SDRAM – 3.3V –128M x 16 SOP 54-08 – Package B4 Part Num: 3DSD2G16VS4364.” .

[124] “UT8QNF8M8 64Mbit NOR Flash Memory Datasheet.” Cobham Aeroflex, 09-Sep-2013.

[125] “Aeroflex Microelectronic Solutions: Product Short Form.” Cobham Aeroflex, Jan-2014.

[126] R. German, C. Kelling, A. Zimmermann, and G. Hommel, “TimeNET-a toolkit for evaluating non-Markovian stochastic Petri nets,” in , *Proceedings of the Sixth International Workshop on Petri Nets and Performance Models, 1995*, 1995, pp. 210–211.

[127] C. Kelling, R. German, A. Zimmermann, and G. Hommel, “TimeNET: evaluation tool for non-Markovian stochastic Petri nets,” in *Computer Performance and Dependability Symposium, 1996., Proceedings of IEEE International*, 1996, p. 62-.

[128] A. Zimmermann, “Modeling and evaluation of stochastic Petri nets with TimeNET 4.1,” in *2012 6th International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS)*, 2012, pp. 54–63.

[129] A. Zimmermann, “Reliability Modelling and Evaluation of Dynamic Systems with Stochastic Petri Nets (Tutorial),” in *Proceedings of the 7th International Conference on Performance Evaluation Methodologies and Tools*, ICST, Brussels, Belgium, Belgium, 2013, pp. 324–327.

[130] A. Zimmermann, *Stochastic Discrete Event Systems: Modeling, Evaluation, Applications*. Springer Science & Business Media, 2008.

[131] A. Zimmermann, Jö. Freiheit, and A. Huck, “A Petri net based design engine for manufacturing systems,” *Int. J. Prod. Res.*, vol. 39, no. 2, pp. 225–253, Jan. 2001.

[132] A. Zimmermann and G. Hommel, “Modelling and Evaluation of Manufacturing Systems Using Dedicated Petri Nets,” *Int. J. Adv. Manuf. Technol.*, vol. 15, no. 2, pp. 132–138.

[133] G. Alves, P. Maciel, and R. Massa, “Evaluating Supply Chains with Stochastic Models,” in *IEEE International Conference on Service Operations and Logistics, and Informatics, 2007. SOLI 2007*, 2007, pp. 1–6.

[134] E. Andrade, P. Maciel, G. Callou, and B. Nogueira, “A Methodology for Mapping SysML Activity Diagram to Time Petri Net for Requirement Validation of Embedded Real-Time Systems with Energy Constraints,” in *Third International Conference on Digital Society, 2009. ICDS ’09*, 2009, pp. 266–271.

[135] C. Araujo, E. Sousa, P. Maciel, F. Chicout, and E. Andrade, “Performance Modeling for Evaluation and Planning of Electronic Funds Transfer Systems with Bursty Arrival Traffic,” in *Intensive Applications and Services, International Conference on*, Los Alamitos, CA, USA, 2009, vol. 0, pp. 65–70.

[136] G. Callou, P. Maciel, E. Carneiro, B. Nogueira, E. Tavares, and M. Oliveira Jr., “Integrated Circuit and System Design. Power and Timing Modeling, Optimization and Simulation,” L. Svensson and J. Monteiro, Eds. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 379–388.

[137] G. R. de A. Callou, P. R. M. Maciel, E. C. de Andrade, B. C. e S. Nogueira, and E. A. G. Tavares, “A Coloured Petri Net Based Approach for Estimating Execution Time and Energy Consumption in Embedded Systems,” in *Proceedings of the 21st Annual Symposium on Integrated Circuits and System Design*, New York, NY, USA, 2008, pp. 134–139.

[138] M. Marinho, P. Maciel, E. Sousa, T. Maciel, and A. Guimares, “Stochastic model for performance evaluation of test planning,” in *2010 IEEE International Conference on Systems Man and Cybernetics (SMC)*, 2010, pp. 3690–3697.

[139] M. Marinho, P. Maciel, E. Sousa, T. Maciel, and E. Andrade, “Performance Evaluation of Test Process Based on Stochastic Models,” in *Proceedings of the 2010 Spring Simulation Multiconference*, San Diego, CA, USA, 2010, p. 141:1–141:6.

[140] E. Tavares, B. Silva, and P. Maciel, “An Environment for Measuring and Scheduling Time-Critical Embedded Systems with Energy Constraints,” in *Proceedings of the 2008 Sixth IEEE International Conference on Software Engineering and Formal Methods*, Washington, DC, USA, 2008, pp. 291–300.

Appendix A PROBA3 ASPIICS Coronagraph

Contents of this chapter originates in significant majority, in unmodified form, from [113].

The “sonic region” of the Sun corona remains extremely difficult to observe with spatial resolution and sensitivity sufficient to understand the fine scale phenomena that govern the quiescent solar corona, as well as phenomena that lead to coronal mass ejections (CMEs), which influence space weather. Improvement on this front requires eclipse-like conditions over long observation times. The space-borne coronagraphs flown so far provided a continuous coverage of the external parts of the corona but their over-occulting system did not permit to analyze the part of the white-light corona where the main coronal mass is concentrated. The proposed PROBA-3 Coronagraph System, also known as ASPIICS (Association of Spacecraft for Polarimetric and Imaging Investigation of the Corona of the Sun), with its novel design, will be the first space coronagraph to cover the range of radial distances between ~1.08 and 3 solar radii where the magnetic field plays a crucial role in the coronal dynamics, thus providing continuous observational conditions very close to those during a total solar eclipse.

A.1 Introduction

PROBA-3 is first a mission devoted to the in-orbit demonstration of precise formation flying techniques and technologies for future European missions, which will fly ASPIICS as primary payload. The instrument is distributed over two satellites flying in formation (approx. 150m apart) to form a giant coronagraph capable of producing a nearly perfect eclipse allowing observing the sun corona closer to the rim than ever before.

The coronagraph instrument is developed by a large European consortium including about 20 partners from 7 countries under the auspices of the European Space Agency. This paper is reviewing the recent improvements and design updates of the ASPIICS instrument as it is stepping into the detailed design phase.

A.1.1 Mission objectives

PROBA-3 is a mission devoted to the in-orbit demonstration (IOD) of precise formation flying (F^2) techniques and technologies for future ESA missions. It is part of the overall ESA IOD strategy and it is implemented by the Directorate of Technical and Quality management (D/TEC) under a dedicated element of the General Support Technology Programme (GSTP). In order to complete the end-to-end validation of the F^2 technologies and following the practice of previous PROBA-X missions, PROBA-3 includes a primary payload that exploits

the features of the demonstration. In this case it is a giant 150 m sun coronagraph capable of producing a nearly perfect eclipse allowing observing the sun corona closer to the rim than ever before. The coronagraph is distributed over two satellites flying in formation. The so called coronagraph satellite (CSC) carries the "detector" and the so called occulter satellite (OSC) carries the Sun occulter disc (figure below). A secondary payload will be embarked on the occulter satellite: the DARA solar radiometer.

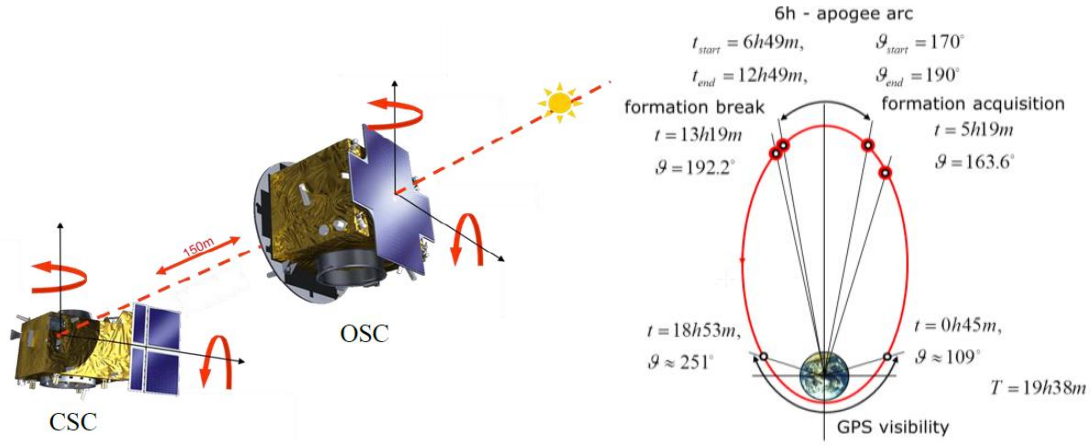

Figure 55 PROBA-3 formation flying overview and orbit

A.1.2 ASPIICS Coronagraph

The region within the sonic point (around 2 – 3 solar radii from the solar centre), where the solar wind is accelerated and coronal mass ejections (CMEs) are initiated, remains extremely difficult to observe with spatial resolution and sensitivity sufficient to understand these phenomena. This requires eclipse-like conditions for long periods of time. However natural eclipses do not allow studying the coronal dynamics and eruptive phenomena during a sufficient amount of time to analyze its magnetic structure and the ubiquitous processes of dissipation of the free magnetic energy. Space-borne coronagraphs were designed and flown to provide a continuous coverage of the external parts of the corona but their over-occulting system did not permit to analyze the part of the white-light corona where the main coronal mass is concentrated.

The proposed PROBA-3 Coronagraph, also known as ASPIICS (Association of Spacecraft for Polarimetric and Imaging Investigation of the Corona of the Sun), will be the first space coronagraph to cover the range of radial distances between ~1.08 and 3 solar radii (R_{Sun}), thus providing continuous observational conditions very close to those during a total solar eclipse, but without the effects of the Earth's atmosphere. The ASPIICS unprecedented field of view makes it uniquely suited for studies of the solar corona, as it will fill the crucial observational

gap between the fields of view of low-corona EUV imagers and conventional space coronagraphs. ASPIICS will combine observations of the corona in both natural and polarized white light [540nm – 570nm] with images of prominences in the He I line (587.6nm) and Fe XIV line (530.3nm).

ASPIICS will provide novel solar observations to achieve two major solar physics science objectives: 1°) to understand physical processes that govern the quiescent solar corona and 2°) to understand physical processes that lead to coronal mass ejections (CMEs) and determine space weather.

A.2 Instrument design

ASPIICS optical design follows the general principles of a classical externally occulted Lyot coronagraph. The external occulter (EO), hosted by the Occulter Spacecraft (OSC), blocks the light from the solar disc while the coronal light passes through the circular entrance aperture of the Coronagraph Optical Box (COB), accommodated on the Coronagraph Spacecraft (CSC). A general sketch of the Coronagraph System is shown in figure below.

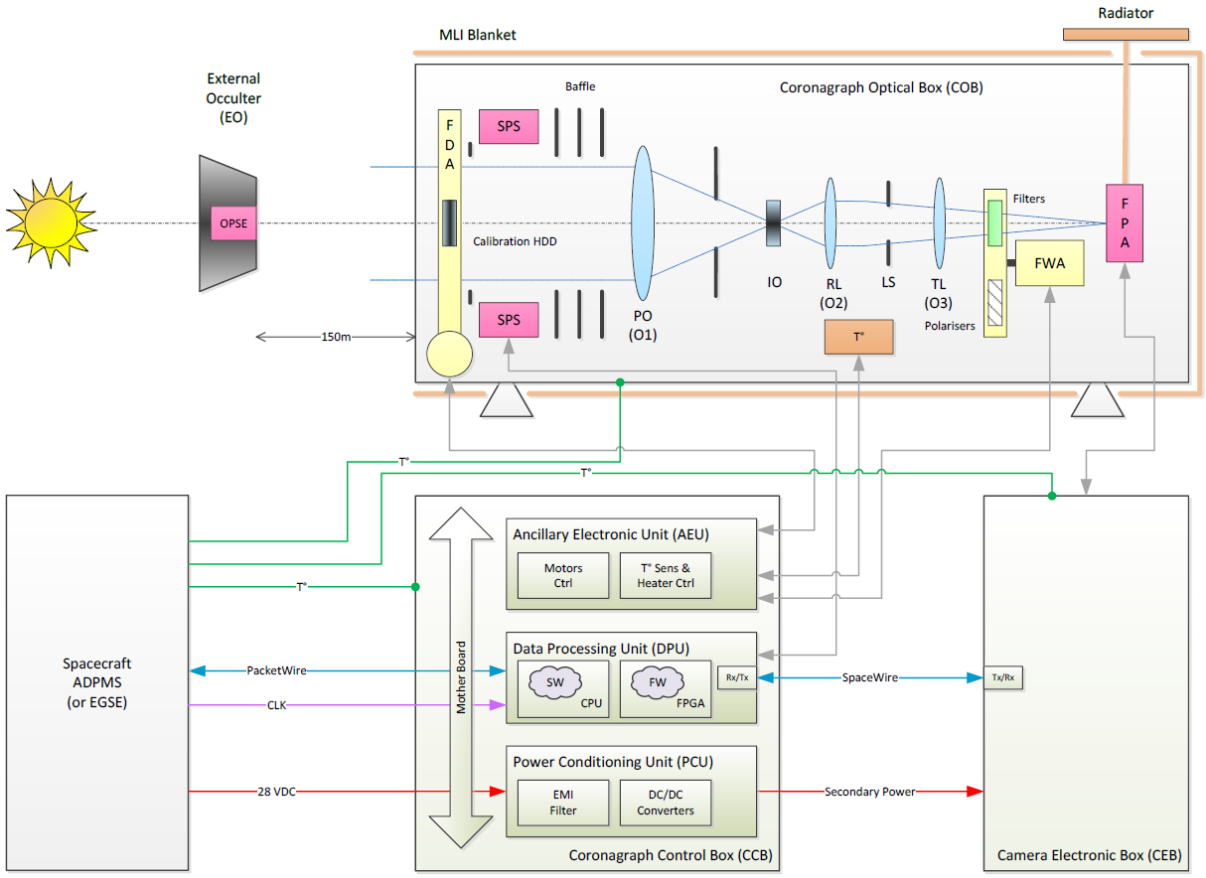

Figure 56 Coronagraph system functional block diagram

The Coronagraph instrument is made of four units:

- The Coronagraph Optical Box (COB),
- The Coronagraph Control Box (CCB),
- The Camera Electronic Box (CEB), and
- The Occulter Position Sensor Emitters (OPSE), located on the OSC.

The science objectives have driven the following key design requirements for the coronagraph:

- To perform white-light coronal observations within a useful field of view from 1.08 to 3 solar radii R_{sun} with a minimum spatial resolution of 5.6 arcsec (plate scale $\leq$ 2.8 arcsec/pixel).
- To perform coronal polarimetric imaging in the [540 – 570nm] band, by measuring the linear polarisation state along at least three different directions of polarization: 0, +60, -60 degrees.
- To perform narrow-band imaging of prominences and the surrounding coronal material in the He I D3 emission line at 587.6 nm ($\Delta\lambda$ = 2.0 nm)
- To perform narrow-band imaging of the corona in the Fe XIV emission line at 530.3 nm ($\Delta\lambda$ = 2.0 nm).

A.2.1 Optics

The Coronagraph optical system consists in a Primary Objective (PO) that forms an image of the external occulter (EO) onto the internal occulter (IO). The IO is slightly oversized to block the diffraction from the EO and to take into account the possible co-alignment error between the two spacecraft's. The field lens O2 makes a real conjugate image of the entrance pupil on the Lyot stop, in order to block the diffraction coming from the pupil's edges. The imaging lenses (O3 + Tele lens) make an image of the solar corona on the detector.

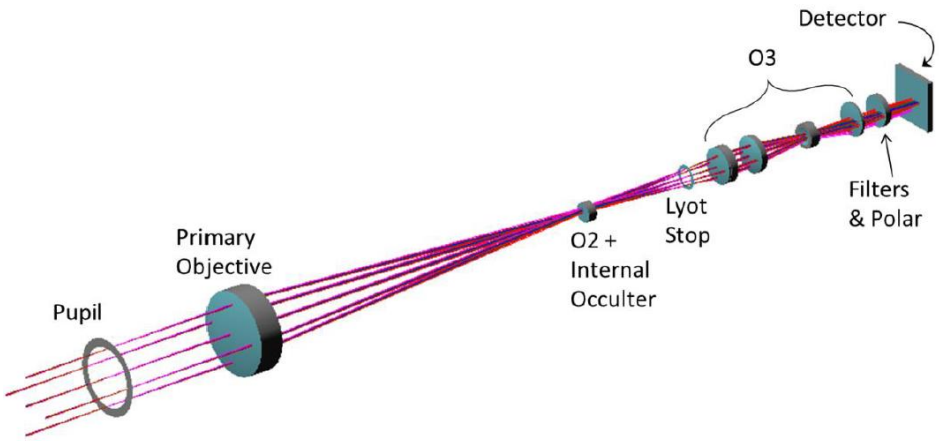

Figure 57 Coronagraph System Lens Objectives

The PO is a doublet composed of a BK7G18 lens in optical contact with a N-SF8 lens. The molecular bonding of the two lenses limits the creation of ghosts. The front lens is made of radiation hard glass to avoid any darkening effects. The primary objective has been optimized for a finite object distance (EO at 144.348 m) and for a field angle corresponding to the edge of the EO (1.02 R_{Sun}). The field lens O2 is a SF6 plano-convex lens. The lenses group O3 is composed of four individual lenses. It creates an image of the sun corona and reduces the divergence of the beam when crossing the filters and polarizers to avoid chromatic issues.

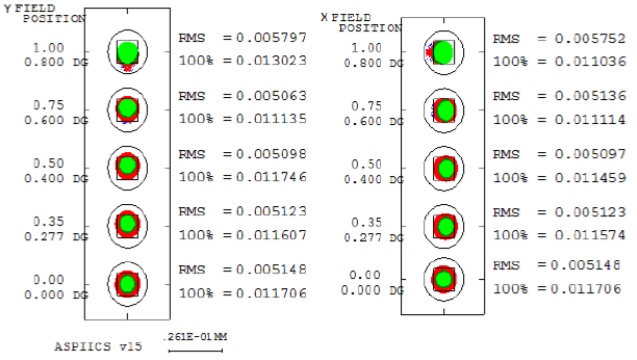

Figure 58 Image spot diagrams at focal plane.

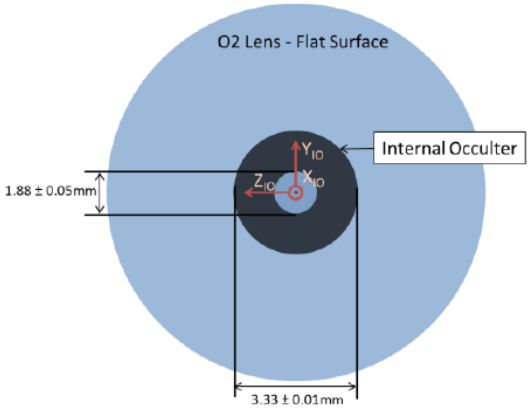

Figure 59 IO design (typical dimensions)

The IO is designed to block the diffraction produced by the edge of the EO. It consists of a coating deposited on the O2 Lens, with a central hole so that the instrument can acquire images of the OPSE lights (see below). The minimum dimension of the IO is when its size is directly the size of the conjugate image of the EO (≈ 1.02 R_{Sun} = 0.272°). In this configuration, the IO radius shall be Rio = 1.631mm, corresponding to an angular dimension of 1014.6 arcsec (equivalent to 1.057 R_{Sun} but this is given for information only since it has no real meaning, the IO being not conjugated with the Sun). During sensitivity analysis, several

over-occultation scenarios have been considered, to take into account different tolerances (EO misalignment, stray-light, etc.) yielding to an actual IO size between 1.798 mm and 1.874 mm in radius. Finally, the Lyot stop is designed to block the diffracted light produced by the pupil edges. It is mounted on the relay lens O3 barrel. The Lyot Stop aperture is 11.4 mm in diameter.

In terms of image quality, the optical tolerance analysis demonstrated RMS spots ≤ 5.8 μm at the border of the field of view (diffraction limit: 19.36 μm at 540nm). The overall calculated transmission of the instrument is > 70% in the useful wavelength range.

The principal source of stray light is the Sun light diffracted by the EO edge. Subsequently, the main contributors to the stray light are the ghosts from reflections between the optics and the scattering by the optics. The stray light induced by the scattering on the mechanical parts is several orders of magnitude lower than the two previous contributors.

A filter wheel mechanism (see below) allows placing the following filters and polarizers in front of the detector:

- Broad-band filter 540-570 nm,
- 3 Polarizing Broad-band filter 540-570 nm (filter combined with a linear polarizer),
- Narrow-band filter Fe XIV at 530.3 nm ($\Delta\lambda$ = 2 nm FWHM),
- Narrow-band filter He I D3 at 587.6 nm ($\Delta\lambda$ = 2 nm FWHM),

A high density diffuser (HDD) composed of a ground fused silica plate is used to produce a flat field on the detector plane when lit by the Sun, with a flux compatible with the dynamics of the coronagraph. The HDD is located in the lid of the Front Door Assembly.

A.2.1.1 Detector

The Focal Plane Assembly (FPA) consists of a front-side illuminated CMOS Active Pixel Sensor (APS), the proximity electronics of the APS, a harness from the FPA to the Camera Electronic Box (CEB), the FPA mechanical parts, a passive radiator and its thermal strap. The selected image sensor is a CMOS APS, manufactured by CMOSIS (Belgium) that has been developed for the ISPHI instrument of Solar Orbiter. No additional qualification testing or other development aimed at improving the performance of the proposed sensor will be envisaged in the context of this mission.

The sensor is configurable through a SPI line. The pixel array of the sensor requires to be clocked externally. The output of the sensor is analogue. The sensor is powered from the secondary voltage of the CCB. The APS proximity electronics is a front-end printed circuit

board aimed at supporting the image sensor electrically but also mechanically. It contains elements for biasing (capacitors, resistors), decoupling (capacitors), and communication to the CEB (drivers, buffers). A radiator is used to cool down passively the FPA so that the sensor can operate in a lower temperature range for noise reduction purposes. The radiator will be connected to the mechanical housing by means of a thermal strap.

A.2.1.2 Image data handling

The high dynamic in the coronal images is managed electronically while minimizing telemetry requirements. The proposed strategy is to take several images, subdivided in blocks (also called "image tiles"), with different exposure times (typically 0.1, 1 or 10 seconds) and recombine them on the ground (figure below). The image tile data are transferred to the CCB "on-the-fly" as they are available so that they don't need to be stored inside the CEB memory after formatting.

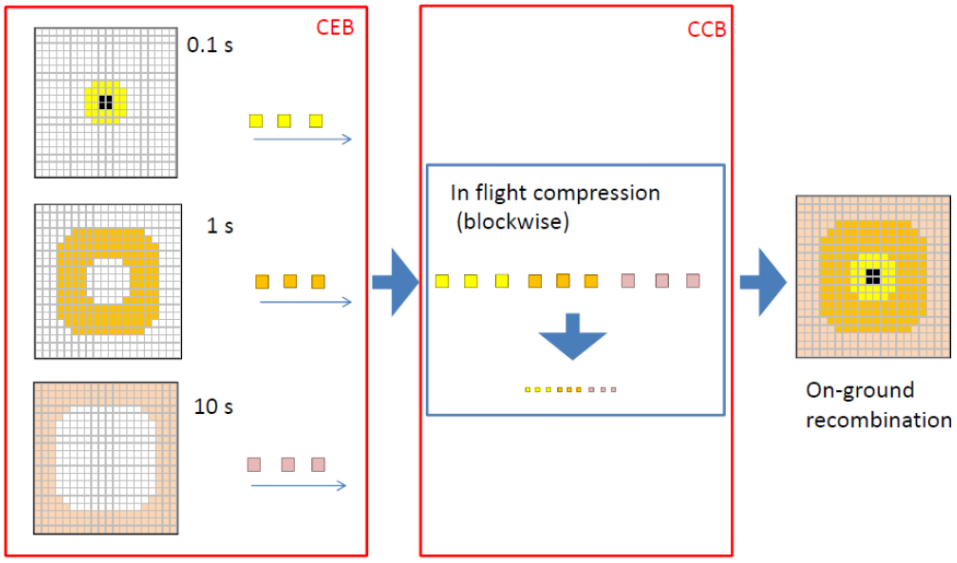

Figure 60 Image data handling principle

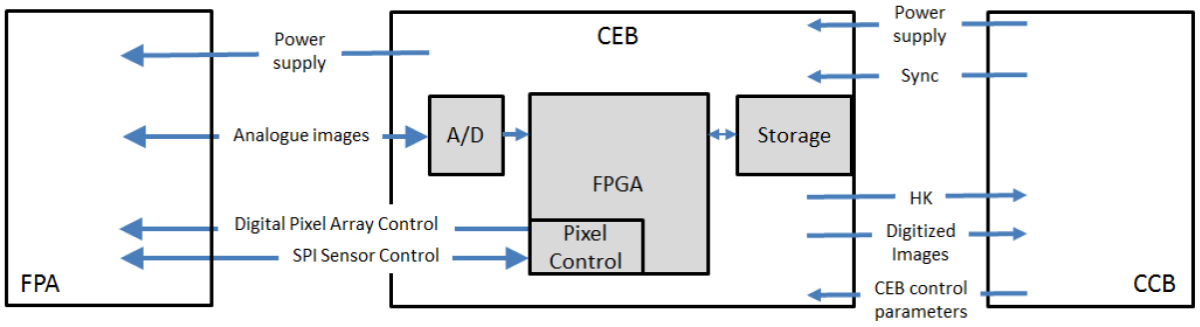

Figure 61 CEB functional diagram

A.2.1.3 Camera electronic box

The Camera Electronic Box (CEB) is dedicated to controlling the FPA. It interfaces with the FPA on one side, to which it sends the necessary control signals and receives in return the analogue pictures from the CMOS APS; with the CCB on the other side, from which it receives its configuration and to which it sends housekeeping and digitized pictures. Upon trigger from the CCB, the CEB starts an image acquisition and is able to acquire, handle and buffer up to three full exposures (2048×2048 pixels) with configurable exposure times. The CEB divides the full exposures acquired by the FPA into square blocks of 64×64 pixels and transfers to the CCB only those blocks defined in the current observation mask, while discarding other blocks. The CEB also performs basic image quality checks (overexposed/underexposed pixels counting and flagging).

A.2.1.4 Mechanical structure

The coronagraph structure is based on a tube and a box mounted on 3 feet (Figure 62). The feet are made of titanium alloy and consist of 3 bipods (more exactly 2 bipods and 2 monopods). They ensure isostatic mounting to decouple mechanically the optical box from the spacecraft optical bench. The feet also ensure thermal insulation to limit heat load from and towards the optical bench (~6mW/K per foot). The monopods are located under the tube and the bipods on the equipment box sides. The tube holds the first part of the optical elements.

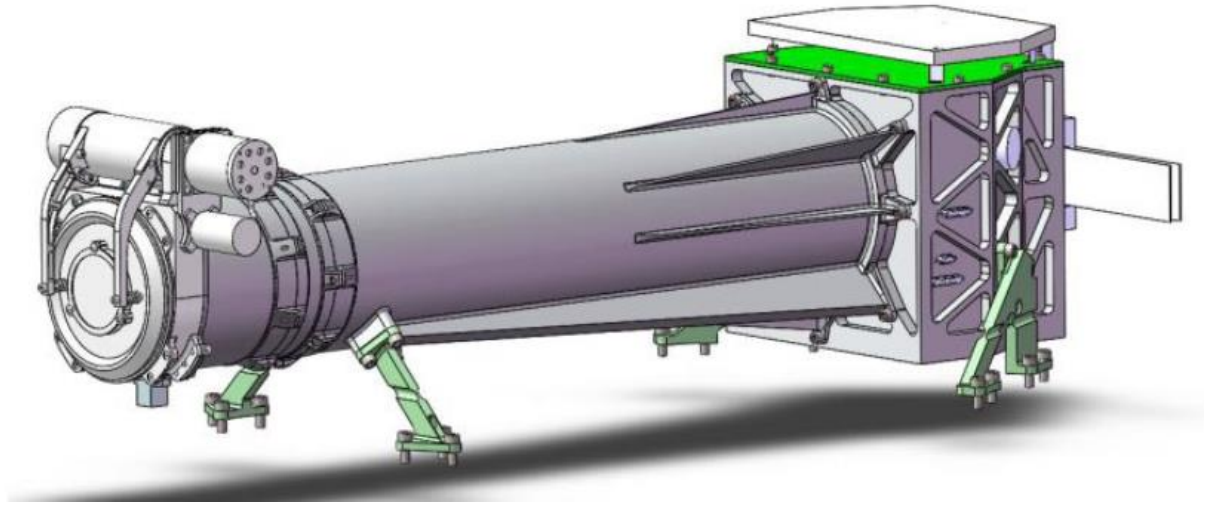

Figure 62 Coronagraph COB overview.

The entrance pupil is located at the very front of the instrument to avoid any mechanical element in front of it. In the center of the tube, the primary objective is inserted. This objective images the Corona on an intermediate focus and the external occulter on the internal

occulter. It is important to maintain the internal occulter at an accurate position to ensure that it continuously covers the external occulter image. A suitable distance is maintained between the pupil and the primary objective in order to protect the lenses from direct view of bright elements in space, to protect it from radiation and to ensure that the temperature between the objective and the internal occulter remains stable and uniform. The full tube is thermally controlled by a heater. At the front of the tube, the Front Door Assembly is attached in order to protect the tube interior and more particularly the front lenses from contamination. In flight the door will also reduce the risk of long exposure to direct Sun (un-occulted), this can be detrimental for the detector and cause important increase of temperature. The door will also be used in flight for calibration of the optical system. In the middle of the cover lid, a diffuser will spread the light of the Sun in the entire field of view in order to distribute it uniformly on the detector. A number of vanes are inserted in the tube in order to prevent stray light from source out of the field of view to enter the optical system.

The second part of the structure is a box holding the different optical elements. The equipment box also contains the filter wheel. This wheel holds the filters and polarizers for the different observation modes. The filters in the wheel are tilted in order to ensure that no ghost is generated by the flat surfaces. The FPA is mounted on the back wall of the box. It includes the detector matrix and thermal links to the externally mounted radiator. The Equipment Box is also thermally regulated.

A.2.2 Mechanisms

A.2.2.1 Filter Wheel Assembly

The Filter Wheel Assembly (FWA) is a 6-position mechanism designed to position the combination of filters and polarizers described above within the science beam between O3 and the FPA. A stepper motor with a 5:1 gear box rotates the wheel during position change and ensures fixation of the wheel during exposures. Sets of 4 cams are placed at the rear side of the filter wheel. They engage magneto-resistive sensors when the appropriate working filter is placed into the beam. To reduce vibration during launch and during rotation of the disc the counter bearing assembly is designed.

A.2.2.2 Front Door Assembly

The Front Door Mechanism is designed to protect telescope optics from contamination on the ground, during launch and some flight operations and to avoid thermal loads of inner part of coronagraph. The Front Door Assembly (FDA) has two operational positions and one locked

position. During launch, the Lid is locked by a wax pin-puller and protects the optics and detector from direct sun light. During the non-observation phases of Coronagraph instrument, the lid is in the closed position. Before and after the Coronagraph observation the Lid is moved by stepper motor via gearbox to the Open position.

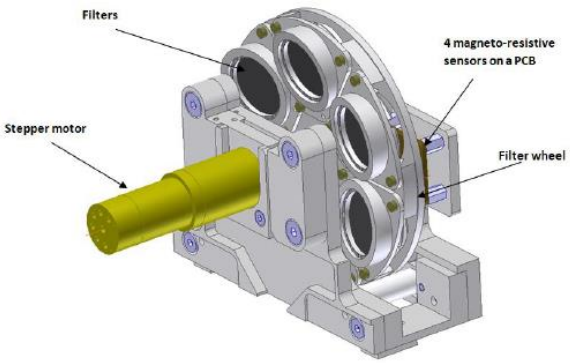

Figure 63 Filter Wheel Assembly (FWA)

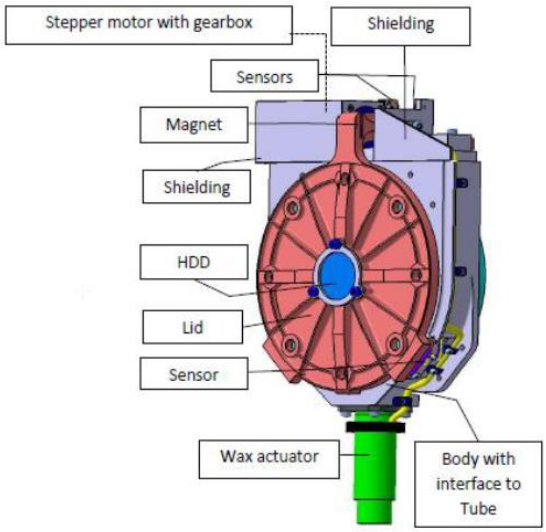

Figure 64 Front Door Assembly (FDA)

A.2.3 External occulter

The external occulter (EO) is a critical element for coronagraphs as it is the major stray light source. The external occulter shape must be optimized in order to reduce the light that is diffracted by its edge and then scattered by the telescope optics.. According to the preliminary studies, the current baseline optimization is a truncated cone (see Figure 65), from 70 to 100 mm thick and with a semi-angle that has to respect the following constraints:

- the truncated cone surface must be in the shadow of the EO outer edge with respect to the solar disk light.
- the cone surface must be as close as possible (compatibly with pointing uncertainties) to the line connecting the IEO outer edge and the pupil edge

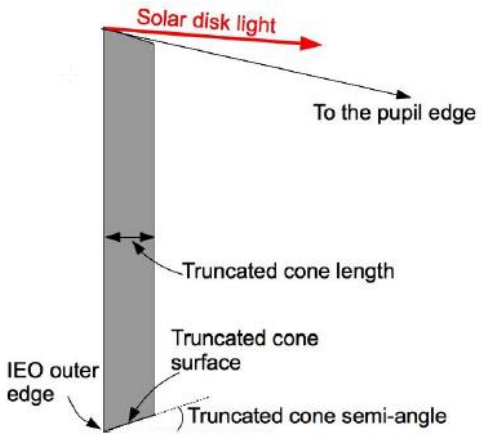

Figure 65 EO geometry

According to investigation results, such optimizations would reduce by at least a factor two the level of diffracted light inside the entrance pupil of the coronagraph. The EO outer edge shall be as smooth as possible, in order to prevent solar disk light scattering from manufacturing imperfections. The EO surface must be black coated and with a Lambertian surface finishing.

A.2.4 Formation flying metrology - Shadow position sensor

The Shadow Positioning Sensors (SPS), together with the Occulter Position Sensors Emitter (OPSE) form the ASPIICS metrology units. The SPS verifies the safe centering of the entrance pupil of the coronagraph within the shadow cone formed by the occulting disc. Initially planned for a high sensitivity relative measurement of the umbra location with reference to the center of the entrance pupil of the coronagraph instrument, it has evolved towards a sensor giving an absolute location with a high accuracy. Another function required on the SPS is to signal to the satellite that the umbra moves away from nominal position, this is to prevent the risk of full Sun illumination inside the coronagraph instrument.

Eight light sensors are used. The sensors are equally distributed on a 55 mm radius circle centered on the entrance pupil. The output current response expected by the SPS is computed in the penumbra around the location of SPS on the entrance pupil plane.

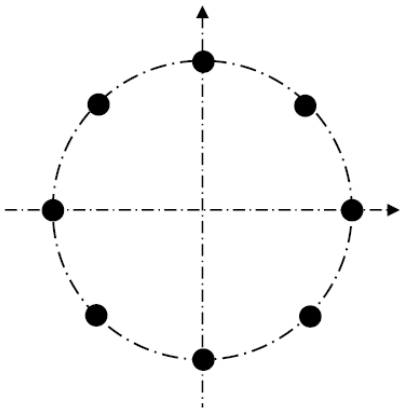

Figure 66 SPS photodiode arrangement around entrance pupil.

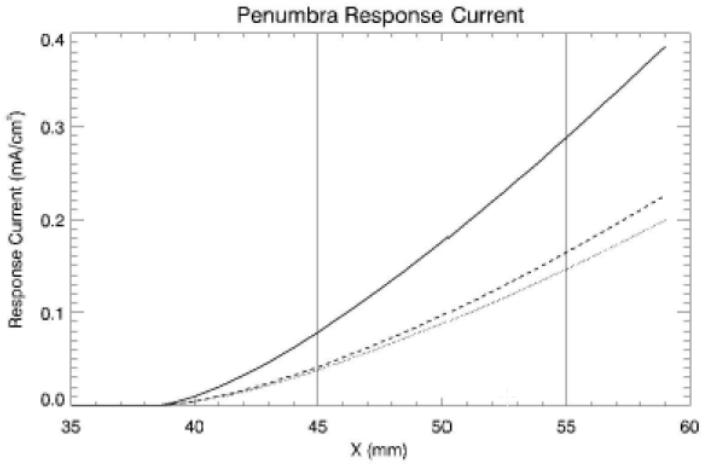

Figure 67 SPS response current in the penumbra on the entrance pupil plane as a function of the distance from the telescope optical axis.

From the 2D distribution of the SPS response currents, it is then possible to estimate the sensitivities necessary to detect the occulter displacements in agreement with performance requirements, i.e.

- for lateral displacements, a sensitivity of 0.45% with diodes at 55 mm is needed to meet the lateral measurement accuracy of 50μm (3-σ) in each axis;
- for longitudinal displacements, a sensitivity of 0.048% with diodes at 55 mm is needed to meet the longitudinal measurement accuracy of 1mm (3-σ).

This shows that the requirement on longitudinal displacement is more challenging (by about one order of magnitude) than the requirement on lateral displacement. The SPS is developed as a joint effort of INAF (Torino, Italy) and SensL Ltd. (Cork, Ireland). The purpose of the OPSE is to verify the positioning of the occulting disc in the field-of-view of the coronagraph that is the alignment of both spacecrafts independently of the pointing to the Sun. The OPSE consists of a set of 3 light emitters mounted on the rear side of the external occulting disc. Their images produced by the coronagraph have a characteristic pattern that uniquely defines the position along the transverse axes, with respect to the instrument coordinate system. Moreover an estimate of the inter-satellite distance (ISD) and of the orientation of the external occulter is also delivered. The 3 OPSE are located close to the centre of the disk, limiting by this way the size of the central hole of the IO. The images of the OPSEs are received by the FPA and are sent to ground for analysis, like other images. (The information delivered by the OPSE is therefore not usable onboard in real time.) The use of the OPSE supposes a coronagraph fully operational, door open, with the entrance pupil in the umbra or low penumbra. Each of the three OPSE enclosures integrates two LEDs of two types, so four LEDs per enclosure. The use of two LEDs with different central wavelengths decreases the main risk of a light wavelength falling out of the spectral band of the coronagraph. Indeed, at least one of the two LEDs must emit in the band pass of the white-light filter (540-570 nm) of the coronagraph, and the major problem is that the central wavelength of a LED slightly shifts with temperature. The temperature of the rear side of the occulting disc is estimated close to -117 °C but with a large uncertainty. The LEDs selected, at the moment, are the one from Phase B: the models VS575N and VS590N from OPTRANS CORP (Japan) which emits respectively at 575 nm and 590 nm at +25 °C. Preliminary tests have been performed at IMT-Bucharest to characterize these two LEDs (and others) with temperature: wavelength drift, output power drift and forward current-voltage drift. The drift of central wavelength of VS575N for an operating temperature close to -133 °C (140 K) leads to a working wavelength

close to 559 nm which is in the wavelength band of the CI. The peak wavelength of this LED is shifted by 0.07 nm/°C to the lower wavelength when the temperature decreases. The same coefficient was found for the VS590N. (This is lower than the 0.2 nm/°C considered in Phase B and could lead to use only one LED instead of the 2 LEDs planned initially.) The OPSE image properties have been studied to verify the light distribution at the focal plane and the vignetting of the IO; the radiometric budget has also been estimated. From these studies the SNR has been evaluated for different output powers of the LEDs (from 0.15 mW to 1 mW at room temperature). By considering the VS575N LED that has an output power of ~ 0.15 mW measured by IMT, the SNR is always above 100 for the LED emission angle of ± 6°. Note that a PWM at 50% was considered in this analysis. The accuracy requirement for the measurement of the lateral and longitudinal displacement has also been retrieved. In order to reach an accuracy on the lateral displacement measurement of 300 µm (3σ) the centroiding accuracy has to be 1.3 µm (1/8 of a pixel). For the longitudinal displacement measurement of 210 mm (3σ), the cenroiding accuracy for OPSE located at 200 mm from the centre of the EO, as it is in the current design, the centroiding accuracy has also to be 1.3 µm. Some centroiding algorithms have been studied to check the monitoring of the OPSE Point Spread Function (PSF) movements, with the expected SNR. The most common procedure, centre of gravity and the fit to analytical models are well suited to guarantee the expected performances of the OPSE system.

A.2.5 Electronics and software

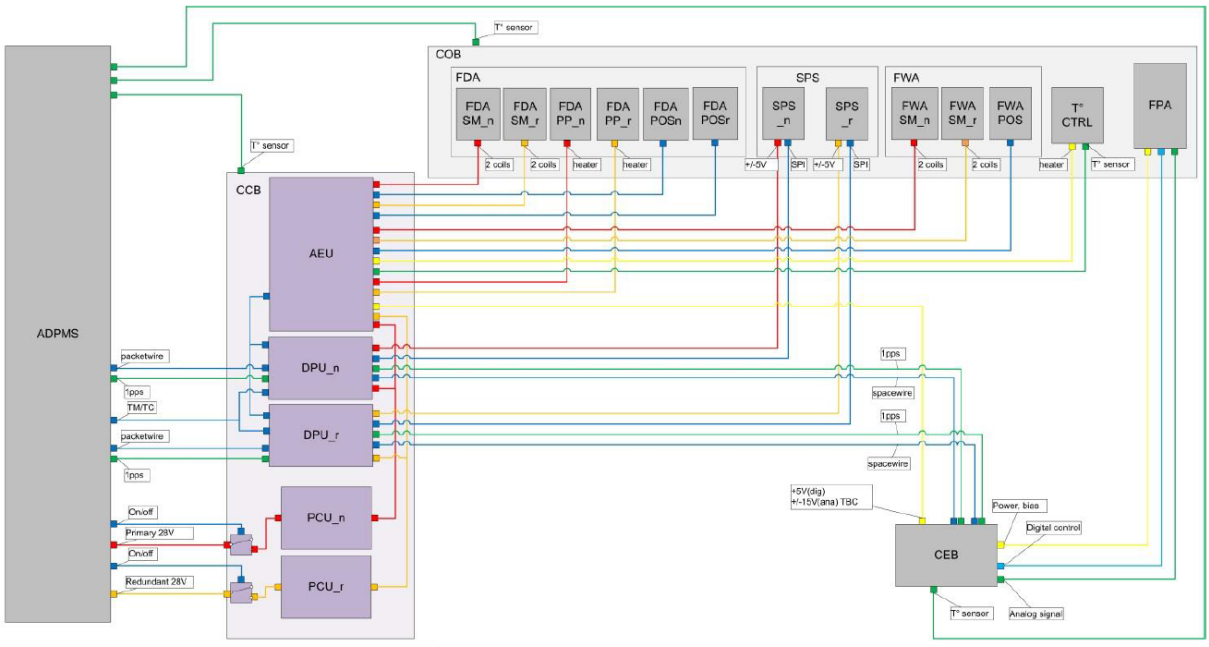

Figure 68 ASPIICS electrical architecture.

The DPU is responsible for Coronagraph control and management functionality. It consists of two main functional parts, a System-on-Chip microcontroller GR712RC that hosts complete 2-core LEON3 microprocessor (CPU) with peripherals and a configurable logic device (FPGA) to implement data processing functionality and processor peripherals which are missing in GR712RC. DPU has several types of memory. Its purpose is to store and execute application code and to store and process scientific data. Memories in principle are connected to microprocessor. Additionally, there is envisioned a cache memory connected to FPGA that will be used for scientific telemetry packet assembly. CPU will have SRAM or MRAM for operation system, application execution needs. Up to 1GB of SDRAM can be implemented to act as scientific mass memory (volatile) before sending the scientific data to ADPMS (through DPU FPGA). This could be a memory space where additional scientific algorithms execute, if necessary. Small, non-volatile flash memory will be used to hold boot loader and, both, basic and application software images. An RTAX 2000 FPGA contains the CCSDS compressor engine (see below), some cache buffers and Packet Wire interface and additional peripherals. In order to give CPU full control over what happens in FPGA, for communication between CPU an FPGA SpaceWire with RMAP will be used. With such solution, AMBA bus connecting all IP cores in external FPGA is "mapped" in address space of microprocessor, so there is a straightforward access to all IP cores, to configure them and program DMA transfers. Interrupts from IP cores residing in FPGA will be fed to processor via GPIOs. Data arrives in CPU via SpaceWire, and it has either to be stored in temporary memory and later fed to FPGA or, as in fact, we will have to use separate SpaceWire interfaces in microprocessor, to receive the data from CEB and to control the FPGA, a DMA mechanism is involved in data reception, so no temporary storage would be involved.

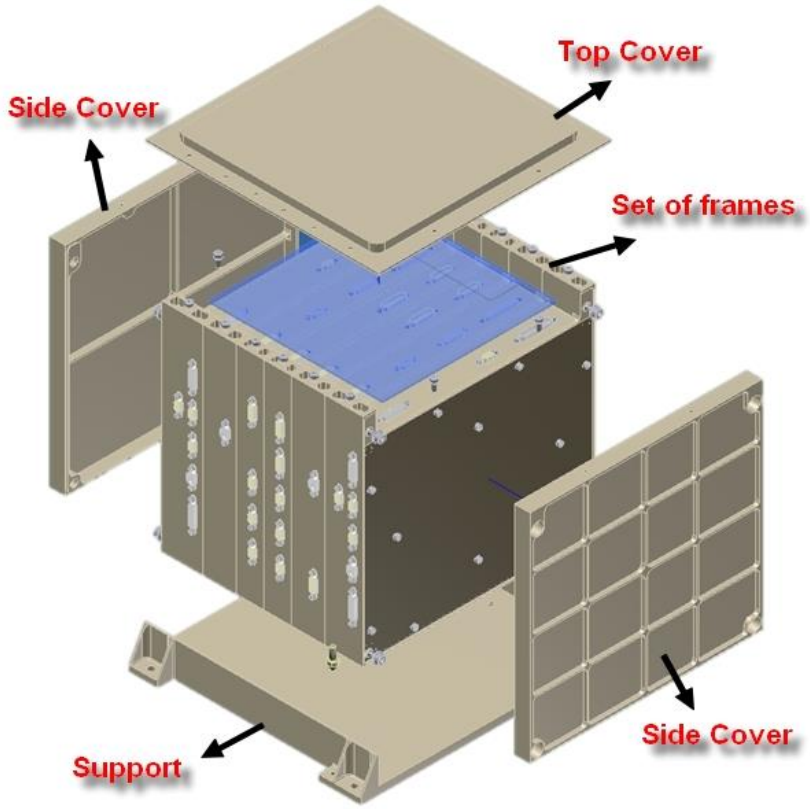

Figure 69 Coronagraph Control Box (CCB) housing

A.2.5.1 On-board Software

CCB software is structured into two separate programs: Boot Software (BSW) and Application Software (ASW). Both programs are compiled separately and are independent meaning that only one can be executed at a time. Software is executed by CCB Data Processing Unit (DPU) based on Leon3 processor. In general BSW provides functionalities related to bootloader. Main functionalities are summarized below:

- Performing initial built-in self-test (BIST)
- Generation of boot report
- FDIR handling with dedicated Safe Mode
- Detailed self-tests of CCB modules
- ASW booting/updating
- Simplified packetization and PUS packets handling

BSW will provide possibility to boot selected ASW image from two images stored in MRAM. First image will be the default image preloaded on ground. There will be no possibility of updating this image during flight. Second image will be updatable with BSW software management functions. BSW will provide only possibility to update whole image. Main functionalities of the ASW are summarized below.

- Commanding Coronagraph modules (AEU, FDA, FWA, LCVR, PCU, COB, CEB, CCSDS-RICE)
- Instrument mode management
- Image acquisition when requested by ADPMS
- Image data handling and buffering
- Packetization and PUS handling
- Shadow Position Sensor (SPS) algorithms calculations

A.2.5.2 Image data compression

ASPIICS implements both lossless and near-lossless compression capabilities, selectable by ground command. The Image Data Compressor (IDC) is based on the CCSDS 121.0-B-2 algorithm and consists of an IP Core firmware to be implemented on the CCB RTAX 2000 FPGA. The top level architecture of the IDC IP Core consists of two major processing units (Figure 70): preprocessor and adaptive entropy coder along with a Main Control unit which hosts the configuration registers. The Image Data Compressor IP Core configuration registers are memory mapped, accessible by the AMBA APB bus through a Slave APB interface and

therefore they can be accessed by standard serial link interfaces like SpaceWire-RMAP or SPI. Compression ratios near 3 and up to 9 can be reached for 12 bits/pixel dynamic range with the lossless and near-lossless compression options, respectively.

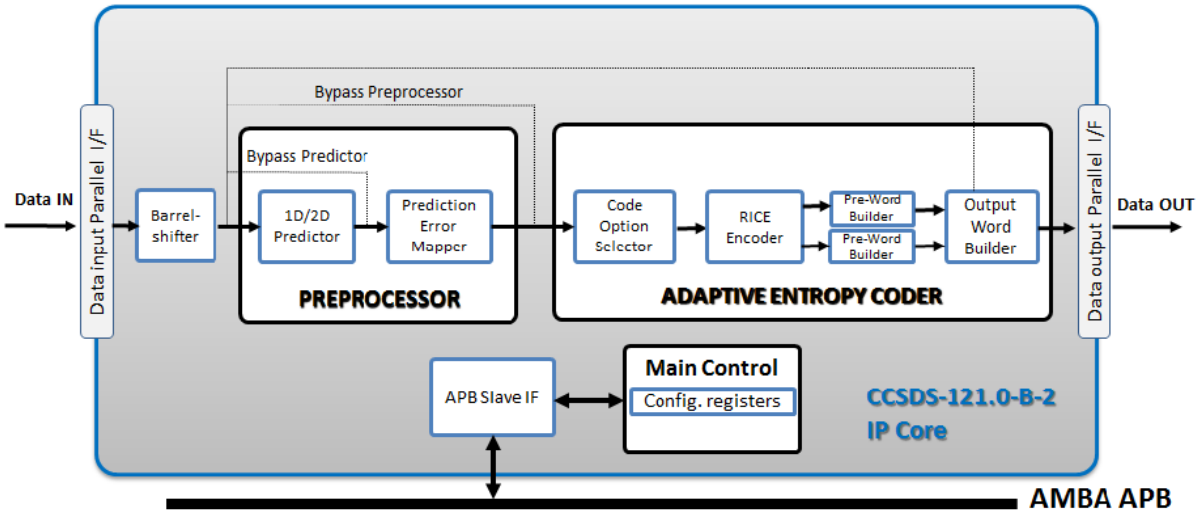

Figure 70 Image data compression IP core block diagram.

Appendix B Components susceptibility

B.1 GR712RC processor

There is very limited amount of information available to potential GR712RC users for free. Most comprehensive and accessible source of information about GR712RC radiation effects is RADECS2011 paper [59], [114]. This scientific paper contains results from several ion irradiation tests performed at Heavy Ion Irradiation Facility in Louvain-La-Neuve, Belgium. There are several measurements taken during processor operation under ion stream and execution of two test programs IU-test (exercising Integer Unit of processor) and Paranoia test (exercising instruction and data caches of processor). Creators of GR712RC estimate there are about 200 sensitive bits in processor core. Upsets of these bits (flip-flops buried deep in logic structure of processor) are not mitigated by implemented protection mechanisms. Sensitive bit number estimated are done by dividing residual (not corrected by EDAC mechanisms) fault saturation cross section acquired during IU and Paranoia Test by fault saturation cross section of flip-flops measured on test chip).

Data point set estimated from worst-case susceptibility (Paranoia test shows higher GR712RC susceptibility so it is treated as a baseline) presented in Fig 5 in [114] are shown in Table 42:

LET	cross-section*cm^2/device	cross-section*cm^2/bit
5	4,00E-07	2,00E-09
10	7,00E-07	3,50E-09
15	9,00E-07	4,50E-09
40	7,00E-06	3,50E-08
65	7,50E-06	3,75E-08
80	8,00E-06	4,00E-08

Table 42 GR712RC device and bit cross-section data sets

Data points from Table 42 can be Weibull-fitted with following parameters (Table 43):

Weibull fit parameter	Value
S (shape curve)	1.9
L (onset LET)	1.0
W (slope width)	20
σ_{sat} (saturation cross-section)	4.00E-8

Table 43 GR712RC cross-section Weibull fit parameters

Cross-section curve for GR712RC is shown on Figure 71.

There are no official cross-section curve fit parameters available nor tables with data points used for interpolation (as of the date of publishing of this dissertation), so values in Table 42 and Weibull fit parameters are read out and selected manually.

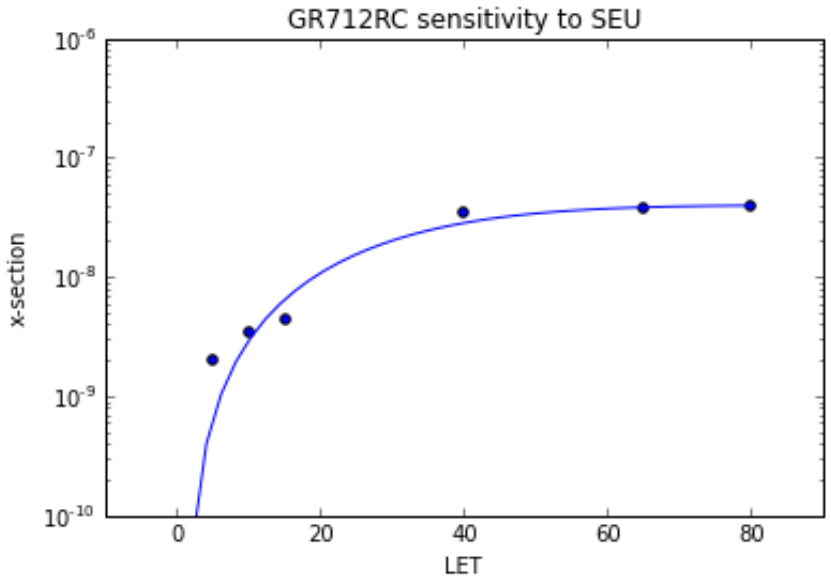

Figure 71 GR712RC bit SEU cross-section

Results for total SEU estimation in GR712RC for GEO-MIN orbit (Table 44):

Device	Effect	(SEU*bit^{-1}*year^{-1})
GR712RC	Direct ionization	1.1625E-05
	Proton induced ionization	2.8278E-07
	Total	1.1908E-05

Table 44 GR712RC GEO-MIN calibration estimate SEU*bit-1*year-1

Value obtained in Table 44 in calibration process can be extrapolated to whole GR712RC device (bearing in mind number of sensitive bits):

1.1908E-05 * 210 = 2.5E-3 [SEU*device^{-1}*year^{-1}]

2.5E-3 non-mitigated bit flips in GR712RC processor operating one year in geosynchronous orbit in solar minimum condition is value close to 2.7E-3 reported by [114] table V. Therefore, it can be assumed that cross-section curve from Figure 71 is validated as GR712RC susceptibility information.

B.2 RTAX 2000 FPGA R-cells and C-cells

RTAX2000 R- and C-cell susceptibility to radiation induced is very well documented and verified by independent NASA teams [24], [115]–[118]. Peculiar thing about RTAX devices is that, while flip-flops in R-cells are intrinsically quite robust (are TMRed), C-cells are upset relatively easy, generating Single Event Transients (SET). SETa are then latched into to the TMRed flip-flops of R-cells, indirectly but effectively, upsetting them. Whole process is obviously frequency dependent – higher the operation frequency, more likely are SETs to be

latched into R-cells. All in all, effective cross-section of operating R-cell is much higher than for static flip-flop.

Figure 72 sums up radiation susceptibility of RTAX R-cells.

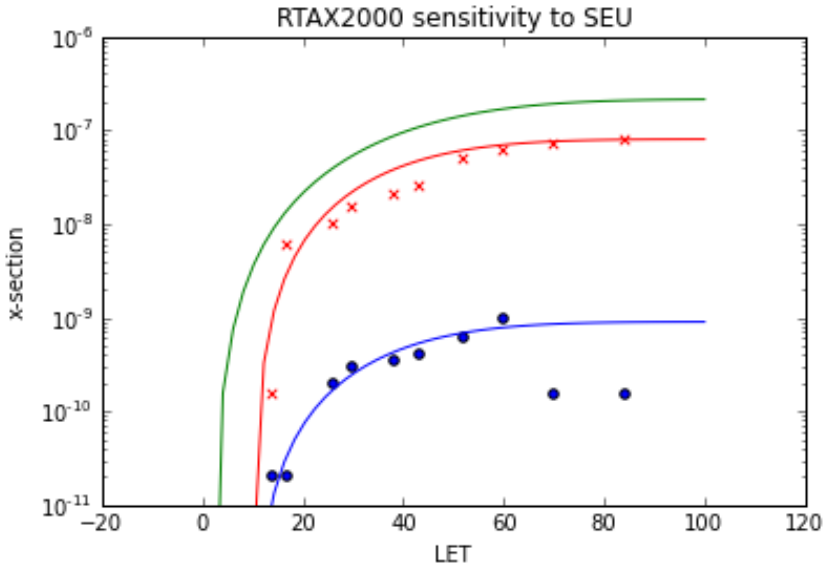

Figure 72 RTAX2000 R-cells SEU cross-sections

Curves on Figure 72 are:

- Blue (dots) - Weibull fit of RTAX2000 SEU sensitivity according to Microsemi initial measurements
- Red (x's) - Weibull fit of RTAX200 SEU sensitivity on independent NASA experimental data
- Green (outermost envelope) - Weibull fit of worst-case scenario of RTAX2000 SEU sensitivity suggested by NASA & Microsemi in [115], [119] figure 10.

All analyses of RTAX R-cells are based on worst-case cross-section of green curve (Table 45).

Weibull fit parameter	Value
S (shape curve)	2.3
L (onset LET)	2
W (slope width)	48
σ_{sat} (saturation cross-section)	2.14E-7

Table 45 RTAX2000 R-cell worst-case cross-section Weibull fit parameters

GEO-MIN calibration:

Device	Effect	[SEU*bit^{-1}*year^{-1}]
RTAX2000 R-cell	Direct ionization	2.9684E-05
	Proton induced ionization	6.0202E-08
	Total	2.9745E-05

Table 46 RTAX2000 R-cells GEO-MIN calibration estimate SEU*bit^{-1}*year^{-1}

Obtained 2.9745E-5 SEU in R-cell a year, translates to 2.97E-5 / 365 = 8.15E-8 (SEU*bit^{-1}*day^{-1}) which is comparable with 5E-8 (SEU*bit^{-1}*day^{-1}) reported by [119] chapter 6.3. It is assumed that worst-case cross-section from Figure 72 (green) and Table 45 is calibrated and is validated as RTAX2000 R-cell susceptibility information.

B.3 RTAX 2000 FPGA Block RAM

Weibull fit parameters from Table 47 describe RTAX2000 BRAM bit SEU susceptibility (bit cross-section curve on Figure 73), that yields fault rate of 2.3210E-04 SEU*bit^{-1}*year^{-1} or 2.3210E-04/ 365 = 6.1698E-07 SEU*bit^{-1}*day^{-1} (Table 48) for standard GEO-MIN conditions and 100 mils Al shielding [120]

Weibull fit parameter	Value
S (shape curve)	1
L (onset LET)	1
W (slope width)	10
σ_{sat} (saturation cross-section)	3.5E-8

Table 47 RTAX2000 BRAM bit SEU cross-section Weibull fit parameters estimation

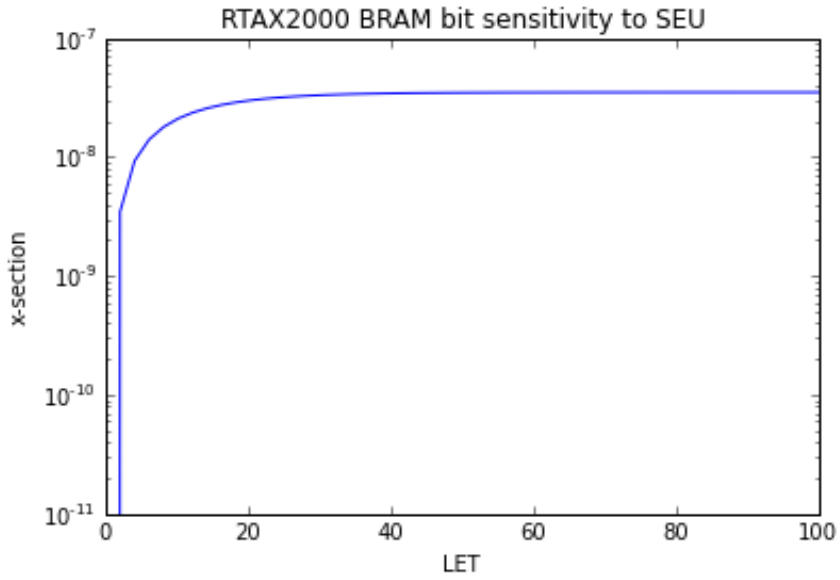

Figure 73 RTAX2000 BRAM bit upset cross-sections

Device	Effect	[SEU*bit^{-1}*year^{-1}]
RTAX2000 BRAM bit	Direct ionization	2.3190E-04
	Proton induced ionization	1.9625E-07
	Total	2.3210E-04

Table 48 RTAX2000 BRAM bit GEO-MIN calibration estimate SEU*bit^{-1}*year^{-1}

BRAM block can be implemented in FPGA, either, as mitigated or unmitigated memory. If memory word is EDACed, then there is an combinatorial dependence between bit upsets and whole word upsets. EDAC mechanism implemented in RTAX Block RAM is Hamming coding (single error correction, double error detection), where 8 bit logic word is coded on 12 physical bits, 16 bit logic word is coded on 29 physical bits and 32 bit logic word is coded on 47 physical bits [121]. Probability that the word is upset is equal to probability that it exhibits two or more upsets, or equally, 1 – probability of zero or one bit upsets. Let P_{word} be probability (or rate, which is equivalent, explained in chapter **Błąd! Nie można odnaleźć źródła odwołania.**) of word upset (not covered by mitigation mechanism) and P_{bit} be probability of bit flip (upset), then:

for 8-bit word (encoded on 12 bits):

$$P_{word} = 1 - ((1 - P_{bit})^{12} + 12 * (1 - P_{bit})^{11} * P_{bit})$$

for 16-bit word (encoded on 29 bits):

$$P_{word} = 1 - ((1 - P_{bit})^{29} + 29 * (1 - P_{bit})^{28} * P_{bit})$$

for 32-bit word (encoded on 47 bits):

$$P_{word} = 1 - ((1 - P_{bit})^{47} + 47 * (1 - P_{bit})^{46} * P_{bit})$$

Taking into account abovementioned consideration leads to BRAM EDACed word upset ratios (derived combinatorically from upset ratio) presented in Table 49.

word size	word GEO-MIN upset rate [SEU*word^{-1}*day^{-1}]
8-bit	2.5124E-11
16-bit	1.5455E-10
32-bit	4.1149E-10

Table 49 RTAX200 BRAM GEO-MIN word upset rate versus word size estimated from bit cross-section

RTAX2000 BRAM cells susceptibility, similarly to R-cells, is well documented in [RD-10]. [120] states the results of RTAX2000 BRAM testing also by showing cross-section for 8bit EDACed word (Table 50 and Figure 74):

Weibull fit parameter	Value
S (shape curve)	1.5
L (onset LET)	30
W (slope width)	10
σ_{sat} (saturation cross-section)	3.91E-9

Table 50 RTAX2000 BRAM 8bit EDACed word upset cross-section Weibull fit parameters [120]

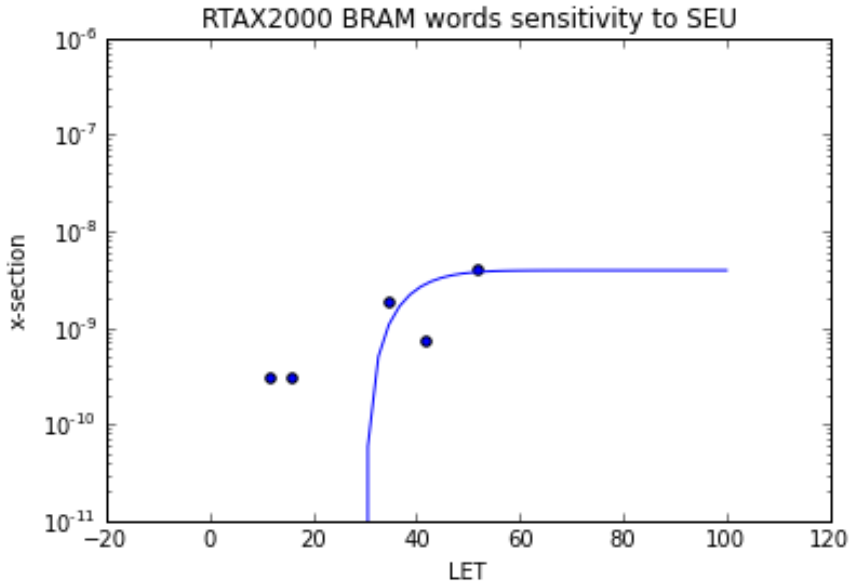

Figure 74 RTAX2000 BRAM 8bit EDAC word SEU cross-section [120]

Simulation of 8 bit upset word in GEO-MIN conditions, based on information from Figure 74 yields word upset ratio of 3.6610E-11, which is comparable to raw bit susceptibility based results from Table 49.

[120] reports that expected fault rates GEO-MIN of EDACed word upset rates are:

word size	word GEO-MIN upset rate [SEU*word^{-1}*day^{-1}]
8-bit	2.5533E-11
16-bit	1.5707E-10
32-bit	4.1821E-10

Table 51 RTAX200 BRAM GEO-MIN word upset rate versus word size

Table 51, with official Microsemi calculated GEO-MIN conditions EDACed word upset rates, holds value of the same order of magnitude as simulation results from Table 49. To sum up, for RTAX BRAM, it can be assumed that susceptibility information for bits and words, mitigated and unmitigated is available and validated in GEO-MIN conditions.

B.4 Memories

Memory chips used in DPU do not offer as deep documentation on radiation effects as RTAX FPGA does, but still, offer more details than GR712RC processor. Significant disadvantage is that there are no GEO-MIN fault rates for any of used memories so no real calibration is possible. Hence, worst-case approach in cross-section estimation is used again to ensure decent estimation margins.

B.4.1 SRAM

SRAM memory used in CCB DPU is 3D-plus 3DSR4M08CS1647 512Kx8 (1 memory chip used). In [122] SEU susceptibility information of LET_{th}: 0.7 MeV*cm^2/mg and saturated cross-section 6E-8 cm^2/bit can be found. Proposed Weibull fit parameters for cross section are presented in Table 52.

Weibull fit parameter	Value
S (shape curve)	2
L (onset LET)	0.7
W (slope width)	10
σ_{sat} (saturation cross-section)	6E-8

Table 52 SRAM bit SEU cross-section Weibull fit parameters

B.4.2 SDRAM

SDRAM memory used CCB DPU is 3D-plus: 3DSD2G16VS4364 128M x 16 2 Gbit (3 memory chips used). Documentation [123] states only following SEU susceptibility information LET_{th}: 2 MeV*cm^2/mg, saturated cross-section 3E-11 cm^2/bit. Proposed Weibull fit parameters for cross section are presented in Table 53.

Weibull fit parameter	Value
S (shape curve)	2
L (onset LET)	2
W (slope width)	10
σ_{sat} (saturation cross-section)	3E-11

Table 53 SDRAM bit SEU cross-section Weibull fit parameters

B.4.3 Flash

Flash memory used in CCB DPU is Aeroflex UT8QNF8M8 64Mbit NOR Flash (2 chips used). Documentation [124] states that Flash cells are very robust, being SEU immune up to 102 MeV*cm^2/mg. Although Flash cells in UT8QNF8M8 are extremely unlikely to upset, memory control logic is less robust: LET_{th}: 29 MeV*cm^2/mg, saturated cross-section 5.0E-13 cm^2/bit, reported in [125] on page 5. Proposed Weibull fit parameters for cross section are presented in Table 54.

Weibull fit parameter	Value
S (shape curve)	2
L (onset LET)	29
W (slope width)	10
σ_{sat} (saturation cross-section)	5.0E-13

Table 54 Flash logic SEU cross-section Weibull fit parameters

B.4.4 Memories GEO-MIN results

Memories cross-section used for all the estimations are plotted on Figure 75, where blue curve (outermost) is for SRAM, red curve (middle) is for SDRAM and green (innermost) is for Flash memory.

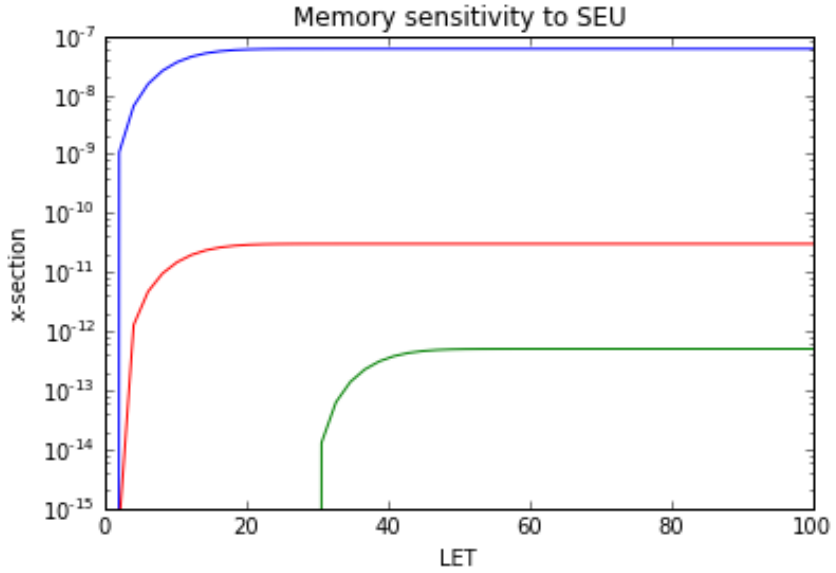

Figure 75 Memories bit SEU cross-sections

As it has been mention in the beginning of this chapter, although no calibration of cross-section data is possible at the moment (no expected GEO-MIN SEU rates available), in case of further investigation, GEO-MIN fault rate results are presented in Table 55 Table 56 and Table 57 below.

Device	Effect	[SEU*bit^{-1}*year^{-1}]
3DSR4M08CS1647 SRAM	Direct ionization	1.2190E-04
	Proton induced ionization	4.0170E-07
	Total	1.2230E-04

Table 55 SRAM GEO-MIN calibration estimate SEU*bit^{-1}*year^{-1}

Device	Effect	[SEU*bit^{-1}*year^{-1}]

3DSD2G16VS4364 SDRAM	Direct ionization	6.6720E-10
	Proton induced ionization	1.7875E-10
	Total	8.4595E-10

Table 56 SDRAM GEO-MIN calibration estimate SEU*bit^{-1}*year^{-1}

Device	**Effect**	**[SEU*bit^{-1}*year^{-1}]**
UT8QNF8M8 Flash	Direct ionization	5.8816E-20
	Proton induced ionization	0.0000E+00
	Total	5.8816E-20

Table 57 Flash GEO-MIN calibration estimate SEU*bit^{-1}*year^{-1}

Appendix C Petri Net tools used in the dissertation

TimeNet

TimeNet version 4.3 is used for Petri net capture, simulation and net figure export to vector graphics. For academic purposes tool is available from System and Software Engineering, Technische Universität Ilmenau, Ilmenau, Germany [126]–[132]

Tool development manager:	prof. Armin Zimmermann
Website:	https://www.tu-ilmenau.de/sse/timenet/

Table 58 TimeNet modeling software details

Mercury

Mercury version 4.6.1 is used for Petri net transient simulation. For academic purposes tool is available from Modeling of Distributed and Concurrent Systems (MoDCS) Research Group at Federal University of Pernambuco, Recife, Brazil [68], [133]–[140]

Tool development manager:	prof. Paulo Romero Martins Maciel
Website:	http://www.modcs.org/

Table 59 Mercury modeling software details

Appendix D Publications related to the dissertation

E. Renotte, E. C. Baston, A. Bemporad, G. Capobianco, I. Cernica, R. Darakchiev, F. Denis, R. Desselle, L. De Vos, S. Fineschi, M. Focardi, T. Górski, **R. Graczyk**, J.-P. Halain, A. Hermans, C. Jackson, C. Kintziger, J. Kosiec, N. Kranitis, F. Landini, V. Lédl, G. Massone, A. Mazzoli, R. Melich, D. Mollet, M. Mosdorf, G. Nicolini, B. Nicula, P. Orleański, M.-C. Palau, M. Pancrazzi, A. Paschalis, R. Peresty, J.-Y. Plesseria, M. Rataj, M. Romoli, C. Thizy, M. Thomé, K. Tsinganos, R. Wodnicki, T. Walczak, and A. Zhukov, "*ASPIICS: an externally occulted coronagraph for PROBA-3: Design evolution*," 2014, vol. 9143, Space Telescopes and Instrumentation 2014: Optical, Infrared, and Millimeter Wave p. 91432M–91432M–15.

R. Graczyk, P. Orleanski, M.-C. Palau, and K. Pozniak, "*Dependability modeling of dynamically reconfigurable space equipment*," in 2014 20th International Conference on Microwaves, Radar, and Wireless Communication (MIKON), 2014, pp. 1–4.

E. Renotte, A. Alia, A. Bemporad, J. Bernier, C. Bramanti, S. Buckley, G. Capobianco, I. Cernica, V. Dániel, R. Darakchiev, M. Darmetko, A. Debaize, F. Denis, R. Desselle, L. de Vos, A. Dinescu, S. Fineschi, K. Fleury-Frenette, M. Focardi, A. Fumel, D. Galano, C. Galy, J.-M. Gillis, T. Górski, E. Graas, **R. Graczyk**, K. Grochowski, J.-P. A. Halain, A. Hermans, R. Howard, C. Jackson, E. Janssen, H. Kasprzyk, J. Kosiec, S. Koutchmy, J. Kovačičinová, N. Kranitis, M. Kurowski, M. Ładno, P. Lamy, F. Landini, R. Lapáček, V. Lédl, S. Liebecq, D. Loreggia, B. McGarvey, G. Massone, R. Melich, A. Mestreau-Garreau, D. Mollet, Ł. Mosdorf, M. Mosdorf, M. Mroczkowski, R. Muller, G. Nicolini, B. Nicula, K. O'Neill, P. Orleański, M.-C. Palau, M. Pancrazzi, A. Paschalis, K. Patočka, R. Peresty, I. Popescu, P. Psota, M. Rataj, J. Rautakoski, M. Romoli, R. Rybecký, L. Salvador, J.-S. Servaye, C. Solomon, Y. Stockman, A. Swat, C. Thizy, M. Thomé, K. Tsinganos, J. Van der Meulen, N. Van Vooren, T. Vit, T. Walczak, A. Zarzycka, J. Zender, and A. Zhukov, "*Design status of ASPIICS, an externally occulted coronagraph for PROBA-3*," 2015, vol. 9604, Solar Physics and Space Weather Instrumentation VI, p. 96040A–96040A–15.

R. Graczyk, P. Orleański, and K. Poźniak, "*Petri net-based dependability modeling methodology for reconfigurable field programmable gate arrays*," 2015, vol. 9662, Photonics

Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments, p. 96623Y–96623Y–11.

E. Renotte, S. Buckley, I. Cernica, F. Denis, R. Desselle, L. De Vos, S. Fineschi, K. Fleury-Frenette, D. Galano, C. Galy, J.-M. Gillis, E. Graas, **R. Graczyk**, P. Horodyska, N. Kranitis, M. Kurowski, M. Ladno, S. Liebecq, D. Loreggia, I. Mechmech, R. Melich, D. Mollet, M. Mosdorf, M. Mroczkowski, K. O'Neill, K. Patočka, A. Paschalis, R. Peresty, B. Radzik, M. Rataj, L. Salvador, J.-S. Servaye, Y. Stockman, C. Thizy, T. Walczak, A. Zarzycka, and A. Zhukov, "*Recent achievements on ASPIICS, an externally occulted coronagraph for PROBA-3*," 2016, vol. 9904, Space Telescopes and Instrumentation 2016: Optical, Infrared, and Millimeter Wave, p. 99043D–99043D–15.