WHITE PAPER

MOSAIC

From Federated to Centrelized Architectures

The Impact of Open Standards on Next Generation Data Handling Systems

Authors:

Ulf Kulau, Ran Qedar, Patrick Rosenthal, Friedrich Schoen, Joachim Krieger, Ivan Masar

DSI-AS, SPiN, TAS-D, FOKUS, STI, TTTech

ID: openDHS-WP-0001
Date: January 17, 2020
Version: DRAFT

Contents

1. Introduction

1.1. Scope

In this document we describe how open standards could impact the way space industry is building on-board digital systems (computers, sensors, actuators, payloads, etc) commonly referred to as data handling system. Starting from the state-of-the-art, the authors describe how space industry can benefit from technologies developed in other technological domains. Open standards for industrial embedded systems are based on a fine grain modularity at board level. In space data handling systems modularity is at box level. As shown in the subsequent sections applying an open industrial standard for modular embedded system in the space domain will result in a scalable data handling system architecture with lower mass and volume compared to the traditional federated approach. In addition significant cost savings for both, users and industry are expected as the effort to specify and integrate functional modules will be much less than in the traditional approach. Of course the selected industrial standard cannot be used without any modifications. It has to be adapted to the specific environmental conditions in the space domain.

1.2. Outline

Over the last several decades, open standards have become ever more important for a wide range of embedded and specialized computer applications, big and small. What do we mean by an open standard? Definitions vary, but for the embedded computer world it usually means a succinct definition of everything a vendor needs to know to build equipment (and write software) that will work with compatible products offered by other vendors. The first really impactful open standard was that of the IBM Personal Computer, first released in 1981. The ISA bus used therein was easy to understand, easy to design and build to, and it is fair to say there would not be the massive amount of powerful, inexpensive off-the-shelf hardware and software available today that exists for both personal and industrial computers without the hardware and software being open. The initial personal computer hardware wasn't particularly rugged and didn't need to be, so standards more suited for industrial, communications, transportation, and military applications emerged, including the first PCI-ISA, i.e. PICMG 1.0 standard, followed by SHB Express, CompactPCI, ATCA, MicroTCA, COM Express, VME, and a lots of others.

One of the great values of an open standard is that it is not controlled by any single company, and the development and updating of these standards is controlled by a large group of interested parties working under the umbrella of an industry consortium using well defined and well tested processes. Developers or vendors of open standards compliant products may be large companies with broad technical skills or small organizations that are expert in a few areas only. Any vendor, large or small, can participate and profit from the large global ecosystem for these products. Users benefit because they are not beholden to a single supplier, as is often the case when proprietary technologies are used. Suppliers of proprietary products have monopoly over their customers and technology upgrades are often slow to arrive. Users of open standards can pick and choose their vendors, who must compete on both price and performance continuously.

Today, traditional space industry is under pressure to compete with new players in this small market segment. This *newspace* approach is based on Commercial Off The Shelf product (COTS) products and development is based on commercial processes. SpaceX is one of *new* companies that demonstrates this new approach with innovative products. A great benefit for all component suppliers is the high reusability of modules when considering a common standard on component level.

2. Acronyms and Definitions

2.1. Acronyms

AIT	Assembly, Integration and Test
AOCS	Attitude and Orbit Control
APEX	APlication EXecutive
API	Application Programming Interface
ARINC	Aeronautical Radio Incorporated
AUTOSAR	AUTomotive Open System ARchitecture
CCA	Conduction Cooled Assembly
CCU	Core Control Unit
CFS	Core Flight System
COTS	Commercial Off The Shelf product
CPM	Core Processing Module
CPU	Central Processing Unit
DHS	Data Handling System
DSP	Digtal Signal Processor
EBB	Elegant Breadboard
ECSS	European Cooperation For Space Standardization
EQM	Engineering Qualification Model
ESA	European Space Agency
ESOC	European Space Operations Centre
FDIR	Failure Detection, Isolation and Recovery
FMEA	Failure Mode Effect Analysis
GNSS	Global Navigation Satellite System
HSSL	High Speed Serial Link
IMA	Integrated Modular Avionics
ITAR	International Traffic in Arms Regulations

LCL	Latch up Current Limiter
LRU	Line Replaceable Unit
MILS	Multiple Independent Levels of Safety and Security
MMFU	Mass Memory Formatting Unit
OBC	On-board Computer
OBC-SA	On-Board Computer System Architecture
OSAL	Operating System Abstraction Layer
PCB	Printed Circuit Board
PCU	Payload Control Unit
PICMG	PCI Industrial Computer Manufacturers Group
PFC	Platform Controller
PHY	PHYsical layer transceiver
PPS	Pulse Per Second
PUS	Packet Utilization Standard
RDC	Remote Data Concentrator
RIU	Remote Interface Unit
RTEMS	Real-Time Executive for Multiprocessor Systems
RTOS	Real-Time Operating System
SAVOIR	Space AVionics Open Interface aRchitecture
SEL	Single Event Latchup
SGM	Safe Guard Memory
SMP	Symmetric Multi-Processing
SUV	Supervisor
TCS	Thermal Control Subsystem
TMR	Tripple Modular Redundancy
TRL	Technology Readiness Level
TSN	Time Sensitive Network
TSP	Time and Space Partitioning
TTE	Time Triggered Ethernet
VITA	VMEbus International Trade Association

2.2. Definitions

Table 2.1.: Definitions

Term	Description
System	The data handling system
SubSystem	A system is broken down into several subsystems, e.g. Power subsystem, AOCS subsystem
Equipment	An equipment is an element of a subsystem, in the context of this study a box hosting several components
Unit	same as equipment
Rack	Specific implementation of a unit/equipment composed of several boards
Component	A component is an element of an equipment, in the context of this study a printed circuit board or a software application
Module	same as component
Board	Specifice implementation of a module, typically a Printed Circuit Board (PCB)
CCU	Core Control Unit is a specific implementation of an equipment dealing with control tasks, e.g. platform control, instrument control
CPM	Core Processing Module refers to a module with processing capabilities, i.e. a Central Processing Unit (CPU) with memory and peripherals. Typically a Core Processing Module (CPM) is the core component of the Core Control Unit (CCU)]
App	An App is a unit of composition with contractually specified interfaces and explicit context dependencies only. An App can be deployed independently and is subject to composition by third parties. An App can be executed on a CPM.

3. Background

3.1. State-of-the-Art: Federated Architectures

Avionics is a term used to describe electronic systems used in aviation. The word itself is result of a blend between the words aviation and electronics. The first use of electronic equipment in aircraft dates back to the Second World War, where simple radars and radios were incorporated in military aircraft. Since then the number and complexity of avionic systems has grown exponentially. From its origin to the end of the twentieth century the avionics development has followed a federated architecture. In this architecture each function of the avionic system is a self contained black box that includes every resource needed for it to achieve its functionality. This black box, a Line Replaceable Unit (LRU), contains all the Hardware, Software and interfaces required for that function. The system integrator is unaware of the inside structure of the LRU, because its development is usually performed by an independent contractor. The selfcontained nature of the LRUs allows them to be modular and quickly replaceable at an operating location. This modularity decreases maintenance costs since a damaged LRU can be replaced quickly by a new stocked one. Furthermore a LRU ensures fault containment given that a failing unit will just stop providing its function and will not affect others. LRUs are designed following a set of standards, mainly Aeronautical Radio Incorporated (ARINC) specifications, which specify their interfaces and physical characteristics. Several manufacturers military and space organization have also defined their own proprietary standards.

As the number of functions required by a satellite started growing, the shortcomings of the Federated Architecture became self-evident. Each new function will add a new LRU to the satellite, requiring more power, mass, and volume from the aircraft or spacecraft. This direct correlation between number of functions and the mass/power consumption enforces satellites with high number of functions to be impractical.

A typical satellite avionics architecture is characterised by dedicated computers (core processing module (CPM)) for different tasks like satellite attitude and orbit control, instrument control and payload data processing. The spectrum of CPU's used for these computers ranges from Sparc based CPUs (e.g. Leon family) to application specific processors implemented in FPGAs or ASICs. Also many different busses, networks and point-to-point connections are used to transfer data between these core processing modules. Traditionally, the MIL1553 bus is used for deterministic command and control while SpaceWire and Spacefibre are used for point-to-point interconnections when higher data rates are required. In addition, many different types of payload specific interfaces are used, e.g. Ethercat for robotics applications or High Speed Serial Links (HSSLs) to transfer image data to the mass memory unit.

Accordingly, the data handling system of a spacecraft is very heterogeneous. It consists of the following elements (see also Figure 3.1):

- Several units with its own CPUs
  - Radhard platform controller (OBC)
  - Instrument Control Unit (ICU), dedicated solution for instruments

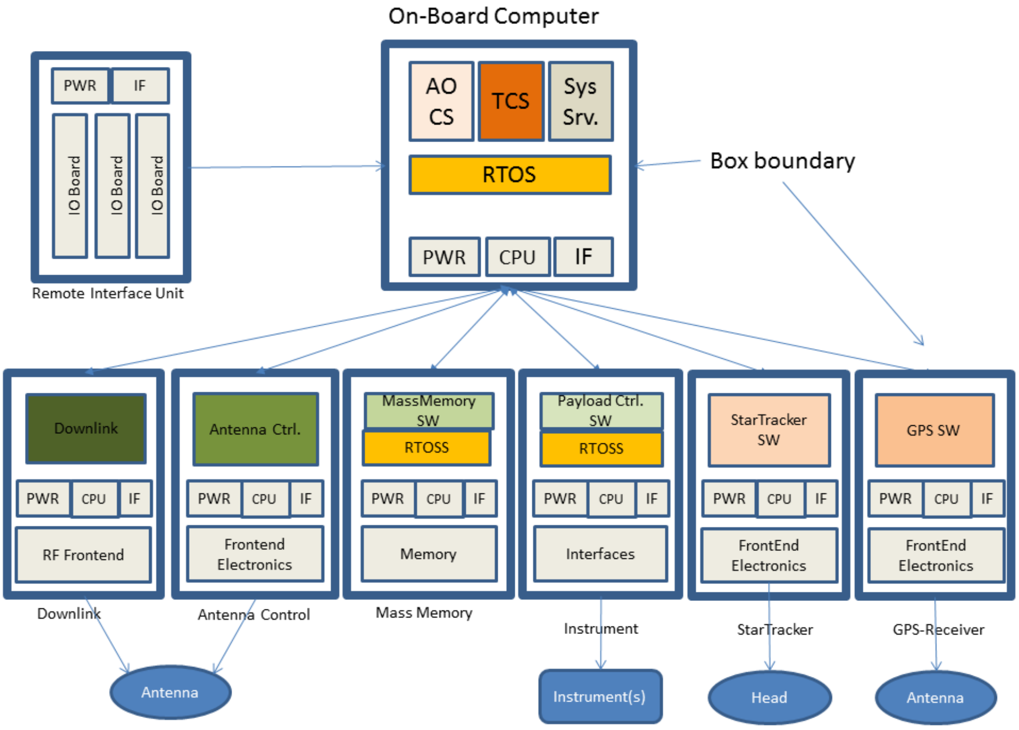


Figure 3.1.: Data Handling System Architecture of a Satellite

- Mass Memory (provides a file system for payload data)
- StarTracker
- GPS receiver

- Several units without CPUs
  - S-Band communication
  - X-Band Downlink
  - Antenna Control
  - Remote Interface Unit to connect all kind of sesnsore and actuators of the spacecraft (for AOCS, thermal control, equipment status monitoring)

- Four different communication links
  - Mil1553 for Command & Control
  - SpaceWire for *high speed* point-to-point connections
  - Ethercat for robotic applications
  - Special HSSLs for payload data (e.g. optical cameras. radars, scientific instruments, etc.)

In addition, with increasing on-board processing demand and processing flexibility during the mission lifetime (e.g. change of protocols or algorithms) also payload computers benefit from a modular architecture, namely software defined payload computers. A modular architecture allows to align the hardware configuration according to the sensor and needed processing power. The specific mission needs are implemented in software/firmware. Modules of a software defined payload computer comprise (see also Figure 3.2):

- RF front-/back-end
    - Single/multi-channel with fixed/configurable frequency/bandwidth
    - Optimised for sensitivity, throughput
    - Receiver/transmitter/transponder boards
- Analogue imaging chain
    - Analogue front end for single/multiple optical sensors (CCD/CMOS based)
    - Sensor control
    - Optimised for noise, sensitivity, linearity
- Digital processor
    - Low level signal de-/modulation and de-/encoding
    - Low level signal conditioning
    - Mission data processing
    - (Lossless) compression
    - Transmit signal generation
    - Configuration of low level signal processing

Placing the Digital/Analogue conversion on the analogue board to avoid routing of analogue signals over the backplane implies the need of high-speed serial data transfer, particularly when involving multi-channel sensors with higher bit resolution.

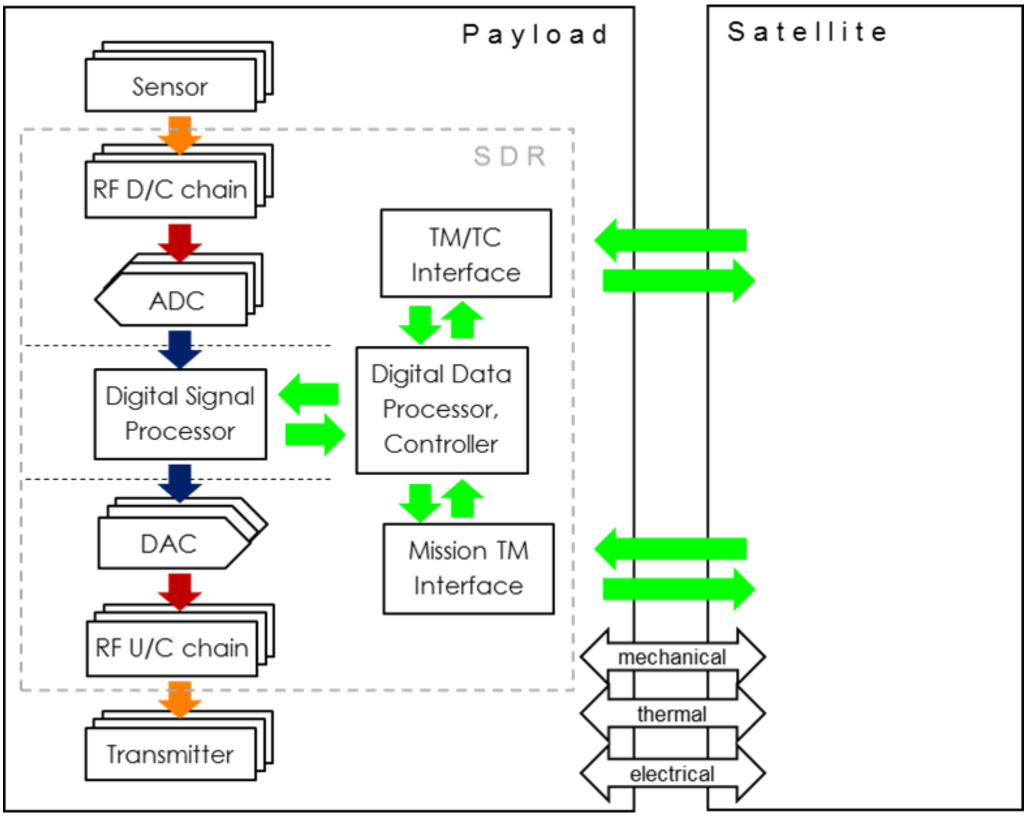


Figure 3.2.: Data Handling System Architecture of a Satellite

In the field of operating systems and software, the situation is similar. For a classic CPM often the real-time operating system RTEMS is used. The variant qualified for space does not support processes, i.e. applications that run in their own address space. The different applications on a CPM are all running in the same address space. Any failure in one application often requires a reboot of the whole computer.

Due to the limited hardware resources modern software architectures based on a middleware for example cannot be used. Same is true for modern software development based on object-oriented languages.

For payload data processing, the situation is slightly different as the primary focus is on computing power and requirements on availability are less stringent as for platform controllers. Therefore, COTS components are sometimes used for both hardware (e.g. Digtal Signal Processors (DSPs)) and operating system (e.g. VxWorks) .

To summarize, the data handling system of a satellite is characterized by

1. Computer optimized for a specific task (platform or payload control or data processing) and
2. Links for data exchange that are specifically designed for space.

Therefore, the architecture is very heterogeneous wrt. computers and communication links. With this architecture new requirements from users on additional functionality will result in an even more complex architecture.

This has some consequences for the software: the heterogeneity in hardware impacts the software complexity. The number of interfaces the software has to support and new functionality is directly linked to the lines of source code.

Heterogeneous hardware and complex software results in tremendous testing effort: for each interface separate test equipment has to be provided and the number of source code lines determins the testing effort.

To summarize, today's data handling system architecture *limits the growth in functionality* which is required to be competitive in a market that is penetrated by non-space companies that are using latest technologies for hardware and software but also for the development process (for hardware and software).

3.2. SAVOIR Reference Architecture

The Space AVionics Open Interface aRchitecture (SAVOIR) working group at European Space Agency (ESA) has defined a reference architecture for a data handling system. The current status of this discussion is presented in Fig. 3.3.

Basically SAVOIR distinguishes between five functional units:

1. On-Board Computer
2. Data Concentrator
3. Intelligent Sensors & Actuators (e.g. GNSS receiver, Star Tracker)
4. Data Storage
5. Payload & Instruments

All five functional units include a CPU (ranging form microcontroller to multicore CPUs for payload data processing). The communication links have different requirements on data throughput, determinism and real-time capabilities.

1. Platform C&C Link (deterministic, real-time)

2. Payload C&C Link (deterministic, real-time)

3. Mission Data Link (high throughput)

SAVOIR does not propose the implementation of the different elements.

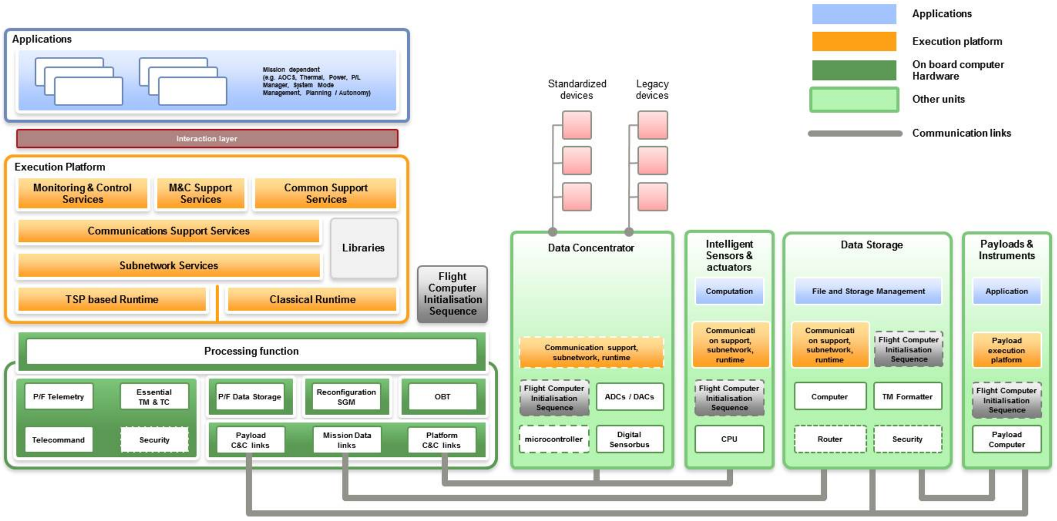

Applications
Mission dependent (e.g. AOCS, Thermal, Power, P/L Manager, System Mode Management, Planning / Autonomy)
Interaction layer
Execution Platform
Monitoring & Control Services
M&C Support Services
Common Support Services
Communications Support Services
Libraries
Subnetwork Services
TSP based Runtime
Classical Runtime
Flight Computer Initialisation Sequence
Processing function
P/F Telemetry
Essential TM & TC
Telecommand
Security
P/F Data Storage
Reconfiguration SGM
OBT
Payload C&C links
Mission Data links
Platform C&C links
Standardized devices
Legacy devices
Data Concentrator
Communication support, subnetwork, runtime
Flight Computer Initialisation Sequence
ADCs / DACs
microcontroller
Digital Sensorbus
Intelligent Sensors & actuators
Computation
Communication support, subnetwork, runtime
Flight Computer Initialisation Sequence
CPU
Data Storage
File and Storage Management
Communication support, subnetwork, runtime
Flight Computer Initialisation Sequence
Computer
TM Formatter
Router
Security
Payloads & Instruments
Application
Payload execution platform
Flight Computer Initialisation Sequence
Payload Computer
Applications
Execution platform
On board computer Hardware
Other units
Communication links

Figure 3.3.: SAVOIR Referenz-Architektur

SAVOIR also proposes a functional architecture as shown in the following Fig. 3.4.

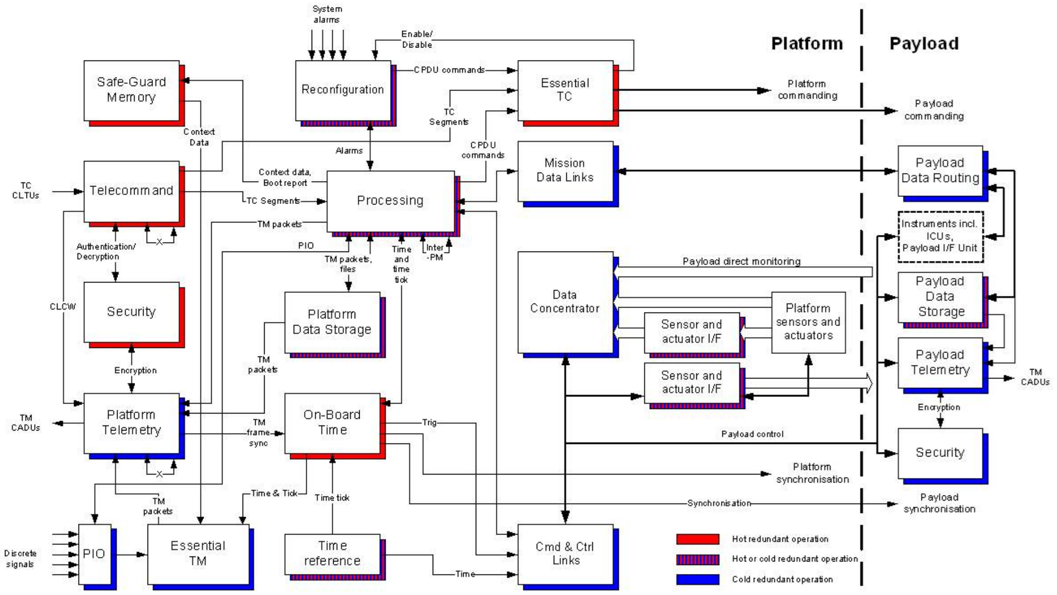


Figure 3.4.: SAVOIR Functional Breakdown

3.3. First Implementations of Modular Computer Systems based on Open Standard

3.3.1. General

In this section we present the results from a DLR study finished end of 2018. The study was called *On-Board Computer System Architecture* (OBC-SA). The objective of the study was to define a modular and scalable hardware architecture for future data handling systems with special focus on robotics missions.

3.3.2. OBC-SA Reference Architecture

During phase 1 of the project On-Board Computer System Architecture (OBC-SA) different architectural concepts were analysed as well as network topologies suited for high speed deterministic data transfer.
The result of phase 1 is the architecture shown in Fig. 3.5. This architecture is basically compliant to the SAVOIR architecture (s. Fig. 3.3) proposed by the SAVOIR working group at ESTEC. Three functional blocks were added to the SAVOIR Reference Architecture: two for the communication with the ground station and one reconfiguration unit switching from primary to redundant units in case one unit fails. The redundant units are not shown in this figure.
All functional blocks are connected to the On-board Computer (OBC) through different communication channels.

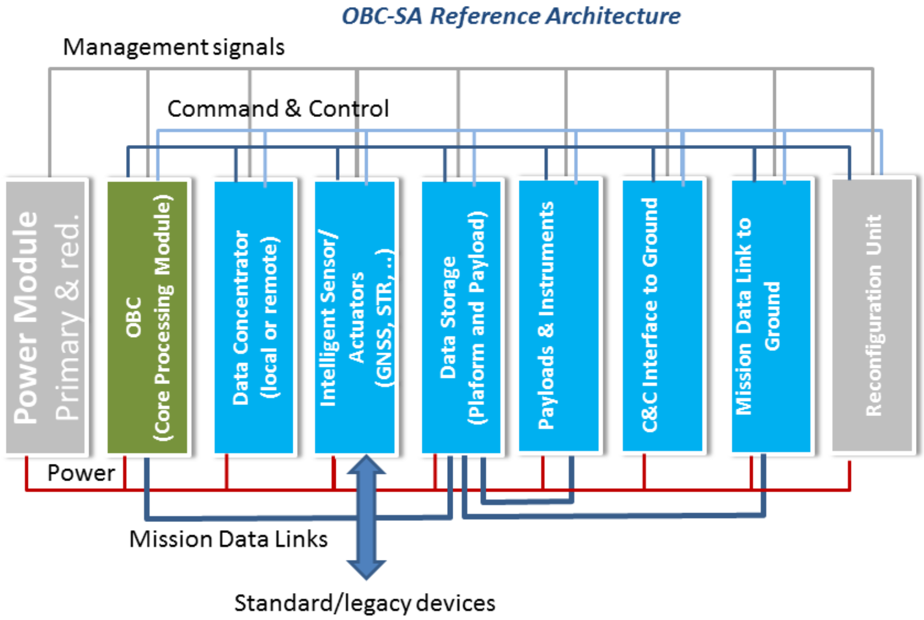


Figure 3.5.: OBC-SA Reference Architecture

The architecture was implemented based on the CPCI Serial Space, an international standard for modular embedded systems. This architecture is characterised as follows:

- Rack based architecture
- 3U formfactor
- Single supply voltage (12V) provided by a power conversion unit (internaly redundant)
- Standby power (5V) for vital functions
- Single point failure free design through dual star architecture, i.e. two system slots, each system slot is connected to each peripheral slot
- Command and Control Link based on SpaceWire
- Separate reconfiguration unit monitoring all boards in the box and performs switching from primary to redundant boards (shelf ctrl)
- CPM with SpW router integrated
- Remote data concentrator unit connected via redundant CAN bus

In phase 2 of OBC-SA Elegant Breadboard (EBB) models of the following modules were developed and validate up to Technology Readiness Level (TRL) 4:

- power supply,
- CPM based on GR740 (quad-core CPU)
- CPM based on P4080 (eight core CPU)
- FPGA board based on RTG 4
- Reconfiguration Unit (shelf controller)
- Remote Data Concentrator

During phase 3 all boards were further qualified and TRL 6 was achieved which is equivalent to an Engineering Qualification Model (EQM), i.e. all environmental tests were passed sucessfully.
The phase 3 test configuration is shown in Fig. 3.6. This CCU implements as high performance computer in hot redunancy. One CPM (hosted in the system slot on the left) is based on a GR740 CPU with integrated SpaceWire router. On the right the second CPM based on the NXP P4080 is shown. It occupies a peripheral slot. The SpaceWire router in the right system slot is implemented based on a rad-hard FPGA (RTG4). The software on both CPMs is identical (but compiled for the target CPU) and based on the *space***APPS** concept (Section 3.4).
A detailed list of components that has been developed during phase 3 is given in the ANNEX (Tbl. A.1)

3.4. *space*APPS - A new generation of modular software

The software complexity has increased continously over the past 50 years from almost no software in the early seventies to 200000 lines of code for a state of the art earth observation satellite. The productivity of writing software for space applications needs to be increased by a factor of two to four to remain competitive in a rapidly changing

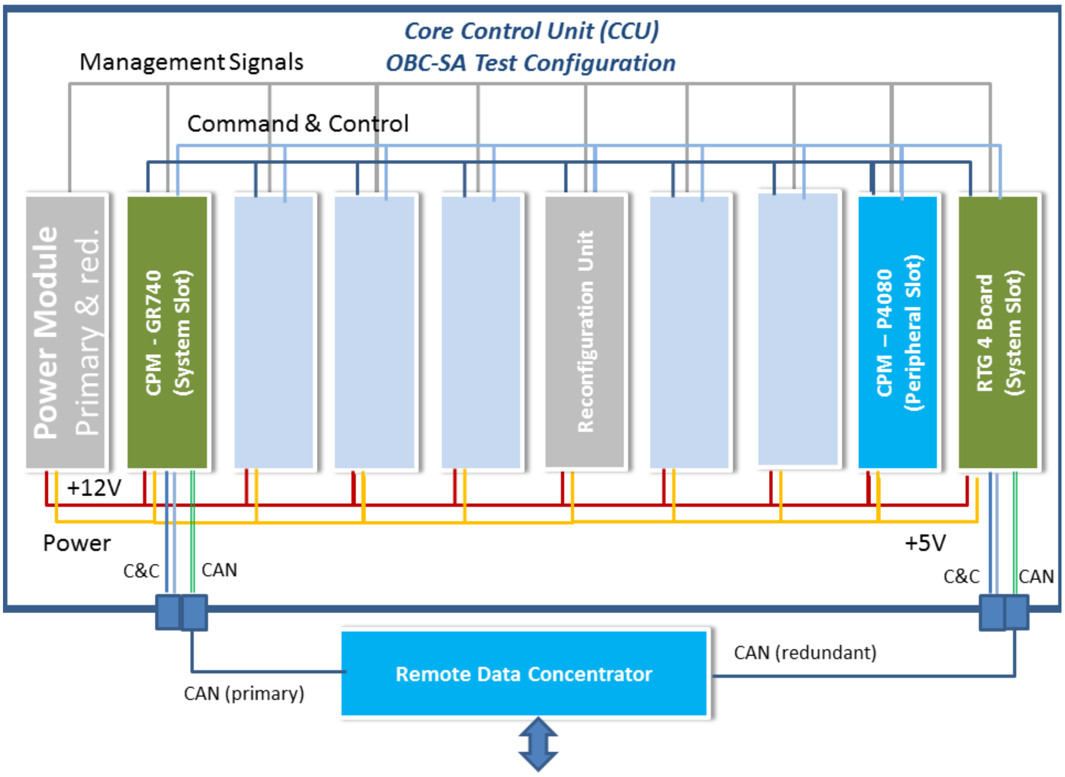


Figure 3.6.: OBC-SA Phase 3 Test Configuration

market. Mega constellations (1000 an more satellites) require a high level of autonomy and efficient procedures to update the software onboard the satellites or adapt the software to changing mission requirements.

Software modularity and partial qualification are key elements to keep the software costs at an acceptable level. The *space***APPS** concept developed in the frame of the OPS-SAT mission aiming at missions were flexibility is one of the success factors.

*space***APPS** implements a novel software architecture for satellites inspired by the Apps concept of modern smartphones. In traditional satellites the on-board software is a monolithic block that performs all tasks to control the satellite. In case of a failure in one element of this block the on-board computer has to be rebooted. The possibilities to patch the failed element of the software are very limited. In most cases a complete software image has to be uploaded to the satellite. This is a long running procedure with some risk.

In the *space***APPS** concept, the on-board computer comes with a predefind set of standard apps implementing the basic functionality of a satellite. For example, this includes all basic Packet Utilization Standard (PUS) services typically used by European Space Operations Centre (ESOC) to operate the satellite. Satellite specific apps like Attitude and Orbit Control (AOCS), Thermal Control Subsystem (TCS) and Failure Detection, Isolation and Recovery (FDIR) extend the basic set of apps taking into account the mechanical structure and data handling system configuration. Mission specific apps like instrument control and basic data processing are added to reflect the mission requirements. All these apps directly affect the satellite safety and therefore, have to be developed in accordance with the strict rules of the European Cooperation For Space Standardization (ECSS).

This new software concept is going to be tested in-flight with the OPS-SAT mission initiated by ESOC The flexible and highly configurable OPS-SAT concept together with the provided processing capabilities of the OPS-SAT payload processing unit allows us to add

a fourth group of apps for user specific processing of data (e.g, camera images). These apps are executed in a secured container, i.e. first, these apps will have only access to a predefined set of on-board ressources (e.g. processing time, memory, image data, satellite position) and second in case one of these apps fails the error remains inside the container and is not propagated to other apps. Due to this fault containment, these apps need not to be developed in accordance with the ECSS.
Thus, users can quickly implement innovative data processing algorithms and test them in a realistic environment. All these apps run as separate service which can be stopped, started or updated when necessary. As the size of these apps are typically much smaller than the size of a traditional software, the update process is much faster and less risky.
In particular the following objectives shall be demonstrated:

1. Demonstrate that the modular apps concept allows to adapt the on-board software to mission requirements at reasonable costs
2. Demonstrate that applications with different criticality levels can co-exist on the same execution platform
3. Demonstrate that updating individual apps is possible in safety critical environment
4. Demonstrate that fast turnaround times can be realized for data processing algorithms

The software framework developed in the frame of OPS-SAT aiming at missions with a high level of flexibility. The software is highly modular with very limited dependencies. Thus, the mission objective can be implemented as a set of different applications (apps). Each app is contributing to the overall objective and communicates with other apps via well defined communication channel (like apps on a mobile phone). This concept allows to upload a single app rather than an entire image during the mission in order to

- Provide a new version of an app after failure corrections or
- Update the functionality of an app in order to adapt to new mission objectives
- Provide completely new functionality

In all cases, it is essential that failure propagation from one component to other components is prevented. Modern operating systems provide mechanisms to execute applications in a separate protected address space. In case on app crashes this does not affect other apps on the platform. For higher levels of protection and determinism Time and Space Separation kernels can be used. There are several products on the market, both commercial (PikeOS, VxWorks653, Lynx, ...) and open source (xTratum/RTEMS, AIR, ...). In order to allow portability accross all these platforms an Operating System Abstraction Layer (OSAL) has been introduced. Software engineers developing apps for mobile devices are using the Application Programming Interface (API) of the underlaying operating system to monitor and control the ressources of the mobile device in an easy way. In aeronautics ARINC653 specifies the APlication EXecutive (APEX) layer in order to provide a standard interface for developers. For the spaceApps framework we developed an API that has been inspired by APEX and NASA's Core Flight System (CFS). This *space***APEX** simplifies the app development as the API provide an standardized access to most services used in space. Like in mobile devices a third layer is needed that provides the basic but domain specific services. In *space***APPS** this third element is called System Support Services. To summarize, the key elements for software modularity are:

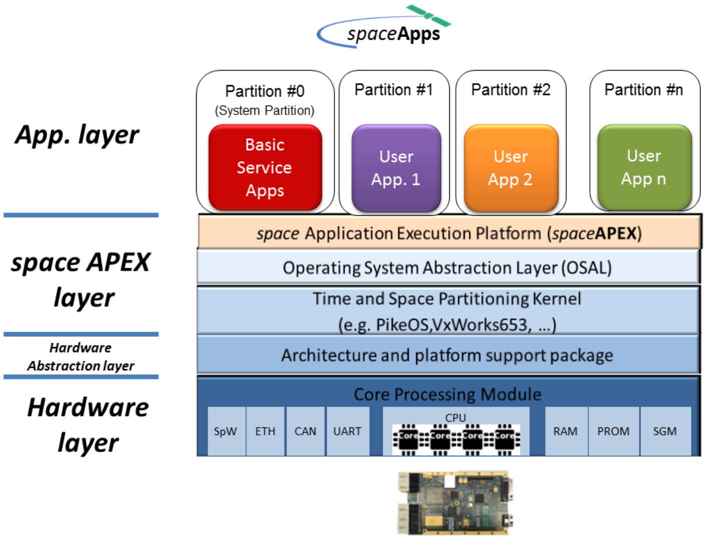

spaceApps
App. layer
space APEX layer
Hardware Abstraction layer
Hardware layer
Partition #0 (System Partition)
Basic Service Apps
Partition #1
User App. 1
Partition #2
User App 2
Partition #n
User App n
space Application Execution Platform (spaceAPEX)
Operating System Abstraction Layer (OSAL)
Time and Space Partitioning Kernel (e.g. PikeOS,VxWorks653, ...)
Architecture and platform support package
Core Processing Module
SpW
ETH
CAN
UART
CPU
Core
Core
Core
Core
RAM
PROM
SGM

Figure 3.7.: Software Layers

1. An Operating System Abstraction Layer (OSAL).
2. An API supporting the development of application components (*space***APEX**) and
3. The definition of domain specific services (System Support Services),

This layered approach for spaceApps is shown in Fig. 3.7. The OSAL resides on top of the operating system. It provides an API to an abstract operating system making it easiler and quicker to develop code for multiple software and hardware platforms. Currently, the OSAL is available for Linux, Vx- Works, VxWorks653 and PikeOS.
*space***APEX** provides a standard interface (API) for all apps. This API includes functions like:

- PUS encoding/decoding
- Partition Management;
- Process Management;
- Thread management
- Memory management
- File handling
- Interpartition communication;
- Intrapartition communication;
- Health Monitoring;
- Event handling
- Access to data pool

The System Support Services implement the basic PUS services frequently used in European space projects according to ECSS-E-ST-70-41. This includes:

- Service 1: Telecommand Verification
- Service 2: Device Command Distribution

- Service 3: Housekeeping and Diagnostic Data Reporting
- Service 4: Parameter Statistics Reporting
- Service 5: Event reporting
- Service 6: Memory Management
- Service 8: Function Management
- Service 9: Time Management
- Service 10: NOT USED
- Service 11: On Board Operations Scheduling
- Service 12: On Board Parameter Monitoring
- Service 13: Large data transfer
- Service 14: Packet Forwarding Control
- Service 15: On Board Storage and Retrieval
- Service 16: NOT USED
- Service 17: Connection Test
- Service 18: On-board Operations Procedures Service
- Service 19: Event Action
- Service 140: Parameters Handling
- Service 148: On-board Macro Procedures
- Service 151: Orbit Position Schedule

These services are implemented as individual apps. Tbl. 3.1 shows the basic Apps needed to operate a satellite.

Table 3.1.: Re-usable Software Components

Icon	**Name**	**PUS Service**
⚙	Input/Output Handler (TC reception and TM distribution) (IO Handler (IOH))	1,2,17
⚙	Service Interface (access to on-board resources via separate port, not used in flight configuration) Service Interface (SIF)	-
⚙	Data Management (storage and retrieval, statistics, monitoring) (Data Management (DM))	12,140
⚙	Event Handler (event reporting, event/action) (Event Handler (EVH))	5,19
⚙	Logging Handler (paket storage and retrieval) (Logging Handler (LOH))	14,15

continued next page ...

Icon	Name	PUS Service
⚙	Execution of mission timelines (Mission Timeline Handler (MTH))	11
⚙	Cyclic housekeeping reporting (MTH)	3
⚙	Execution of TC sequences based on orbit position (Orbit Position Schedule Handler (OPSH))	151
⚙	Execution of On-Board Control Procedures (On-Board Control Procedure Handler (OBCPH))	18
⚙	Telecommand Sequencer (MACRO)	148
⚙	System supervision (Supervisor (SUV))	6,8
⚙	Time Management (SUV)	9

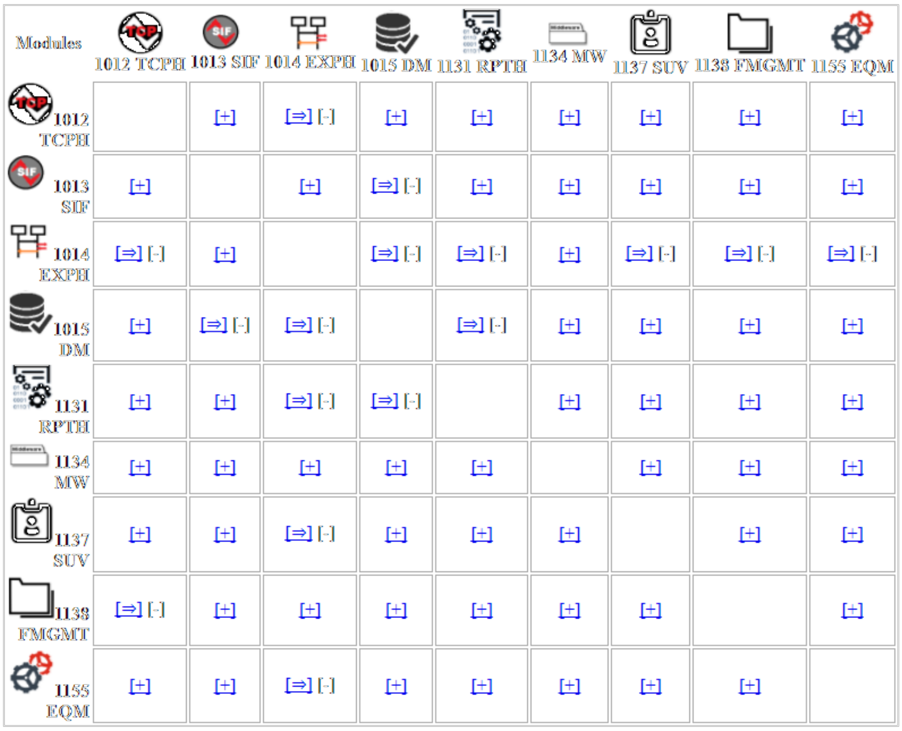
All Apps are connected to a switch matrix which allows a very flexible interconnection scheme between Apps. Typically, the switch matrix is configured during design time (see Fig. 3.8). All Apps in the System Support Services are highly configurable through corresponding tables for PRIDs14, event IDs, event action lists etc. These system configuration tables are defined during design time and can be modified during run-time. Thus, mission specific adaptations can be performed without changing the code.

Depending on the available hardware I/O ports corresponding equipment handlers can be configured. In the present version equipment handlers for Ethernet, UART, SpaceWire, MIL1553 and CAN are available.

The Supervisor App (Supervisor (SUV)) controls all other Apps and therefore is highly mission specific. The spacecraft configuration and status vector is handled by the supervisor as well as the satellite modes. It is the highest FDIR instance on-board the spacecraft. Events that cannot be handled here are forwarded to the ground operators. Other mission specific extentions include payload control components for example.

Depending on the mission requirements either the full set or a subset of Apps can be used to implement the required functionality. Additional services can be easily implemented based on the provided API.

This framework is a key element to build realiable application software (e.g. platform or payload control) to be executed on the various on-board computers (Platform Controller (PFC) and Payload Control Unit (PCU)).



Modules	1012 TCPH	1013 SIF	1014 EXPH	1015 DM	1131 RPTH	1134 MW	1137 SUV	1138 FMGMT	1155 EQM
1012 TCPH		[+]	[⇒] [-]	[+]	[+]	[+]	[+]	[+]	[+]
1013 SIF	[+]		[+]	[⇒] [-]	[+]	[+]	[+]	[+]	[+]
1014 EXPH	[⇒] [-]	[+]		[⇒] [-]	[⇒] [-]	[+]	[⇒] [-]	[⇒] [-]	[⇒] [-]
1015 DM	[+]	[⇒] [-]	[⇒] [-]		[⇒] [-]	[+]	[+]	[+]	[+]
1131 RPTH	[+]	[+]	[⇒] [-]	[⇒] [-]		[+]	[+]	[+]	[+]
1134 MW	[+]	[+]	[+]	[+]	[+]		[+]	[+]	[+]
1137 SUV	[+]	[+]	[⇒] [-]	[+]	[+]	[+]		[+]	[+]
1138 FMGMT	[⇒] [-]	[+]	[+]	[+]	[+]	[+]	[+]		[+]
1155 EQM	[+]	[+]	[⇒] [-]	[+]	[+]	[+]	[+]	[+]	

Figure 3.8.: App. Interconnection Matrix

4. Towards modular and scalable DHS

4.1. General Considerations

As shown in the previous section, if the Space industry wants to advance it should dismiss the black box architecture and embrace the sharing of physical resources among different functions.
A number of technical and economic advantages could be realized if the different elements of a federated architecture were integrated into a centralized architecture in a modular manner:

- Cost savings by the reduction of equipments and wiring points (results also in an increase in hardware reliability)
- Cost reduction in terms of reusability
- Better integration of functions - more flexibility
- Implementation of fault tolerance simplified
- Volume and mass reduction

But:

- Independence of individual components compromised - increased potential of error propagation from one component to another component
- Integration increases complexity and diagnostics
- Allocation of responsibility more difficult

The first step towards the sharing of resources was given during the project of the Boeing 777 with the Integrated Modular Avionics (IMA) concept.
There has therefore been a push for IMA, in which common computing platforms can be leveraged for different functions. Besides saving power, mass and volume also cost savings are to be expected wrt. software. The reasons are:

1. The software has to deal with a much smaller spectrum of computing platforms,
2. Using standards will allow to combine solutions from different vendors either hardware or software,
3. Modular certification will be possible as hardware or software components can be reused without any modification.

The ideal future avionics systems would combine the complexity management advantages of the federated approach, but would also realize the functional integration and hardware efficiency benefits of an integrated system. Hammett Robert. Flight Critical Electronics System Design, IEEE AESS Systems Magazine, June 2003, p.32

4.1.1. Integration of Functionality

The integration of functionality requires more powerful processors. With the quadcore LEON4 a processor is available now that has a much higher computing power as what has been used in space until today. The Symmetric Multi-Processing (SMP) variant of Real-Time Executive for Multiprocessor Systems (RTEMS) fully supports the multicore architecture. Thus, it would be possible to combine software functionality from different equipments on a single CPU board.
As shown in Figure 3.1 all equipments have their processing capability, i.e. CPU plus software or FPGA. In both cases a separation of hardware and software is possible. This will allow to transfer the software onto a single CPM. In this case the processing capability inside the equipment is no longer necessary and can be removed. Fig. 4.1 shows this approach.

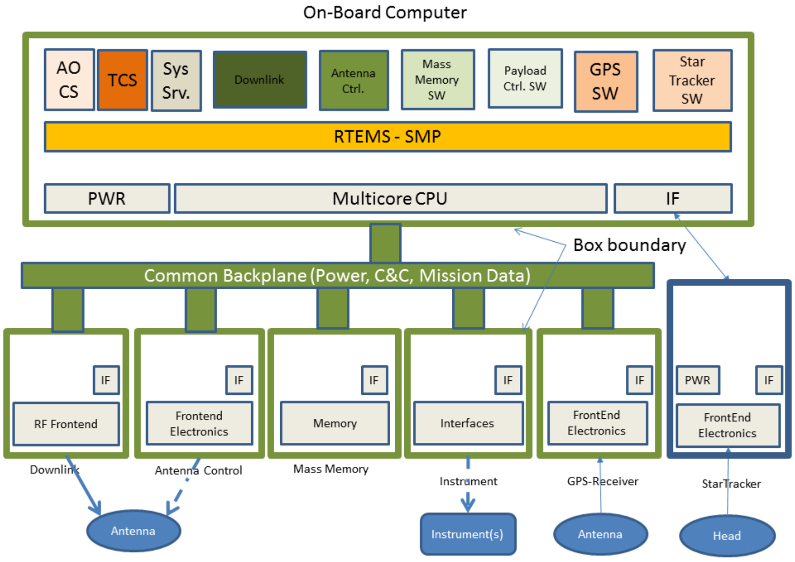


Figure 4.1.: Centralized Architecture

This approach fully utilizes the processing power of the multicore CPU but it does not provide the required independence of individual equipments in the federated architecture. Temporal and special separation of individual applications are needed to achieve the same level of independence as with the federated architecture.
IMA architecture is a reference avionics architecture standardized by ARNIC653. It was first presented by Honeywell for cockpit on the Boeing 777 aircraft in 1995, and is extensively implemented in avionics design of Airbus 380, Boeing787 Dreamliner, Boeing C130, F-22, Gulfstream G280 etc.
Although the implementation of IMA can be different from manufacturers, the key concept conformed by IMA system designers is the same which is the spatial and temporal partition with the sharing of the computing resource. With this concept the same level of independence can be achieved as with the federated architecture.
The transition from the traditional software architecture to an IMA architecture is shown in Figure 4.2. The APEX standardized by Aeronautical Radio Incorporated (ARINC) 653 is to facilitate distributed software applications development. The application - either on top of an Real-Time Operating System (RTOS) or without RTOS - deployed into a partition of the Time and Space Separation kernel ensures that errors in one partition do not propagate throughout the entire system. Due to the isolation of applications in partitions

and a standard way of inter-partition communication defined in ARINC 653 it is possible to develop applications individually with very limited dependencies to other applications.

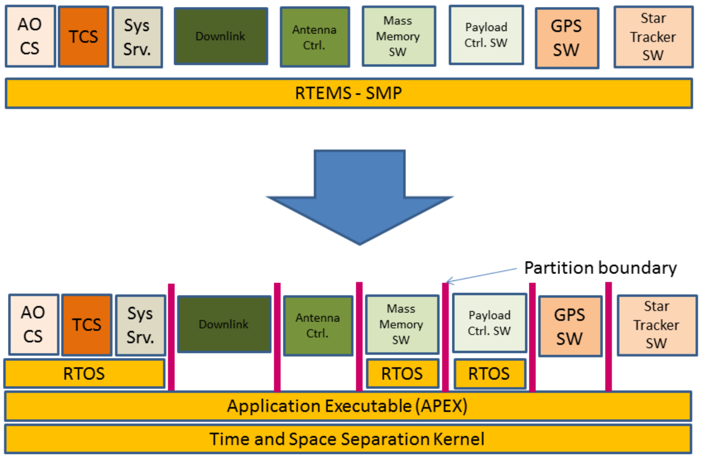


Figure 4.2.: Time and Space Separation

In the IMA approach the system integrator provides the application developer with an execution platform running on virtual hardware. The resource and time partitions define the boundaries in which the application shall be executed, i.e. the max. size of memory to be used, the processing resources to be used and the schedule.
Thus, the applications can be developed largely independent, as the interfaces and the execution constraints are clearly defined. Modular certification at software level becomes possible which contributes to overall cost savings.

4.1.2. Hardware Modularization

Once the software has been transfered from an equipment to a central processing unit the processing capability inside an equipment is no longer needed. Thus, the core functionality of these equipments is reduced to a level that it can be easily implemented on a single board or even fully integrated with the central processing unit.
If the board design of the core functionality follows a common standard (from factor, eletrical interfaces, communication links, etc.) it can be easily integrated with the platform computer(see Figure 4.1).
Using this approach at board level based on an international backplane standard will help to further reduce the overall mass and harness complexity.
The boards can be developed largely independent, as the interfaces are based on agreed international standards. Modular certification at hardware level becomes possible which contributes to overall cost savings.
The full Multiple Independent Levels of Safety and Security (MILS) compliant data handling architecture is shown in Fig. 4.3. The On-Board Computer (one single board) hosts all application software components clearly separated in partitions. Other functionalities are implemented as separate PCBs interfacing the Command and Control Link routed through the common backplane. A separate PCU converts the primary power to voltages required by the boards.

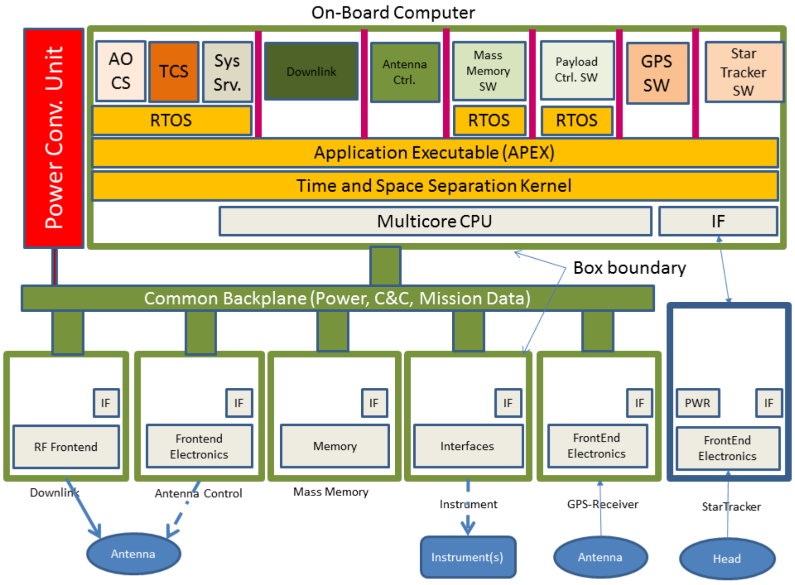


Figure 4.3.: MILS compliant Data Handling System Architecture

4.2. System Concept and Technology Selection

4.2.1. General Aspects

In OBC-SA but also in the NASA studies[1] a modular approach for the data handling system is described. This means, functionalities traditionally implemented as separate boxes are integrated as board into a rack (furtheron referred to as CCU). Fig. 4.4 shows a typical configuration for a platform controller.

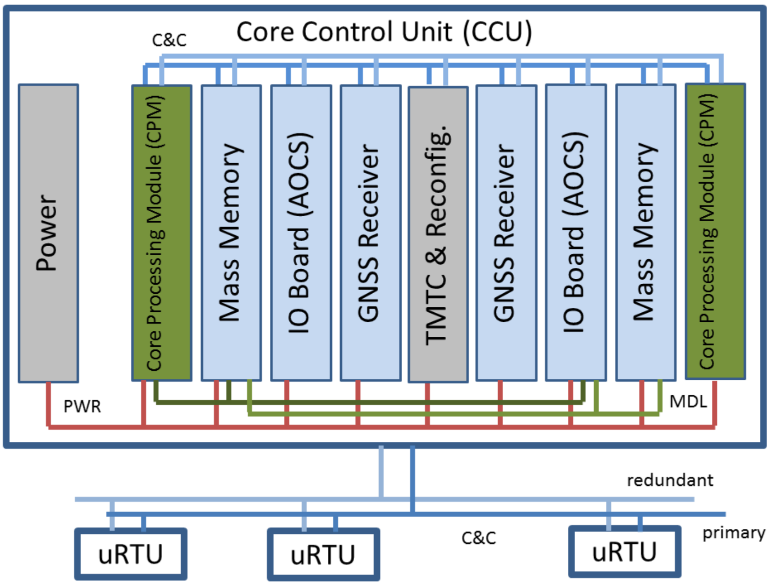


Figure 4.4.: Modular CCU

At both ends a CPM is located. Two mass memory boards, two Global Navigation Satellite System (GNSS) receiver boards and two IO board complete this configuration. The

[1] https://www.nasa.gov/feature/goddard/2017/brains-of-the-operation-nasa-team-develops-modular-avionics-systems-for-small-missions

GNSS receiver provides position, velocity and time incl. a Pulse Per Second (PPS) signal for synchronization purpose. The IO board provides the interface to AOCS sensors and actuators. Additional uRTUs may be added to interface with other sensors and actuators. Savoir defines the functionalities that are typically allocated to the OBC (or CPM).

4.2.2. Stand-alone vs Modular

Specific, highly integrated stand-alone solutions pay off when:

1. The market requires a high number of devices, i.e. more than 1 million.
2. The available space and mass are driving the design, i.e. in CubeSats.

In all other cases (particularly for small series of 5 .. several 1000 units), modular approaches are more cost efficient. Basic functionalities are provided as building blocks and the system integrator implements the system from predefined building blocks. Only very specific (i.e. mission specific) functionality or basic functionality that is not available at system definition time have to be developed.

As an example we consider two configurations: one with separate 3U boards for CPU, MM and GPS and one with all three functions integrated on a single PCB this results in the box dimensions shown in Tbl. 4.1.

Table 4.1.: Estimated Box Volume

Board Config.	Dimensions WxHxD	Volume
PCU + 3 boards	130x130x200	3.38 l
PCU + integrated board	230x70x170	2.74 l

The integrated approach has a volume that is 19% smaller than the modular approach, i.e. the savings in volume do not compensate the benefits of the modular approach.

Modular Avionic approaches have been around for 40 years but vary widely in implementation and the extent of both hardware and software levels of unification. The IMA concept, which replaces numerous separate processors and Line Replaceable Unit (LRU) with fewer, more centralized processing units, has led to significant weight reduction and maintenance savings in both military and commercial airborne platforms. Similar concepts have been developed for automotive (AUTomotive Open System ARchitecture (AUTOSAR)) and in the industrial automation domain. Besides saving mass and volume the major driver in the industrial domains is cost both for development and maintenance. Common to all these concepts is the use of standards for both hardware and software. Like in other domains space industry will have to cope with increasing system complexity but shorter development cycles and shrinking budgets. Proven concepts from other domains need to be investigated, adapted and applied in space programs to meet the customer expectations wrt. quality, time and cost in a global market. Recent developments in space avionics like spaceVPX, and CPCI Serial Space will help to achieve these objectives.

Open Standards for Modular Embedded Systems

The selection of the backplane standard is based on the definitions made in the previous sections. Basically, there are only two international standards for embedded computer systems available which address also the needs of the space industry:

1. SpaceVPX released Apr. 2015 by VMEbus International Trade Association (VITA)
2. CPCI Serial Space released Aug. 2017 by PCI Industrial Computer Manufacturers Group (PICMG)

A brief description of both standards is given hereafter.

CPCI Serial Space

CPCI Serial Space differentiates between 3(4) slot profiles (see Fig. 4.5):

1. Power
2. System
3. Peripheral
4. Shelf Controller (not part of the standard)

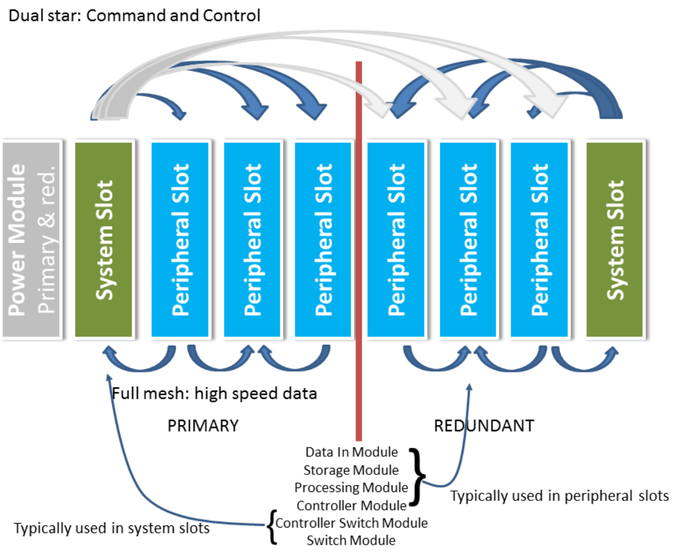


Figure 4.5.: CPCI Serial Space Overview

The C&C bus of each system slot is connected to each of the peripheral slots. Thus, a dual star architecture is implemented. In case one element fails the redundant element takes over. For high speed interconnects CPCI is using a full mesh scheme, i.e. each slot is connected to all other slots (max. 8) with 4 differential pairs. 10 Gbps data transfer rate over the backplane are possible. As shown in Fig. 4.6 together with a switch a dual star architecture is also possible. The switch can be allocated to any of the peripheral slots.
The pin assignment of the two different slots is shown in Fig. 4.7. The full mesh network is allocated to connectore P6. The connectors P2,P4 and P5 in a system slot provide the pins for the SpaceWire star architecture, i.e. each peripheral slot slot is connected to

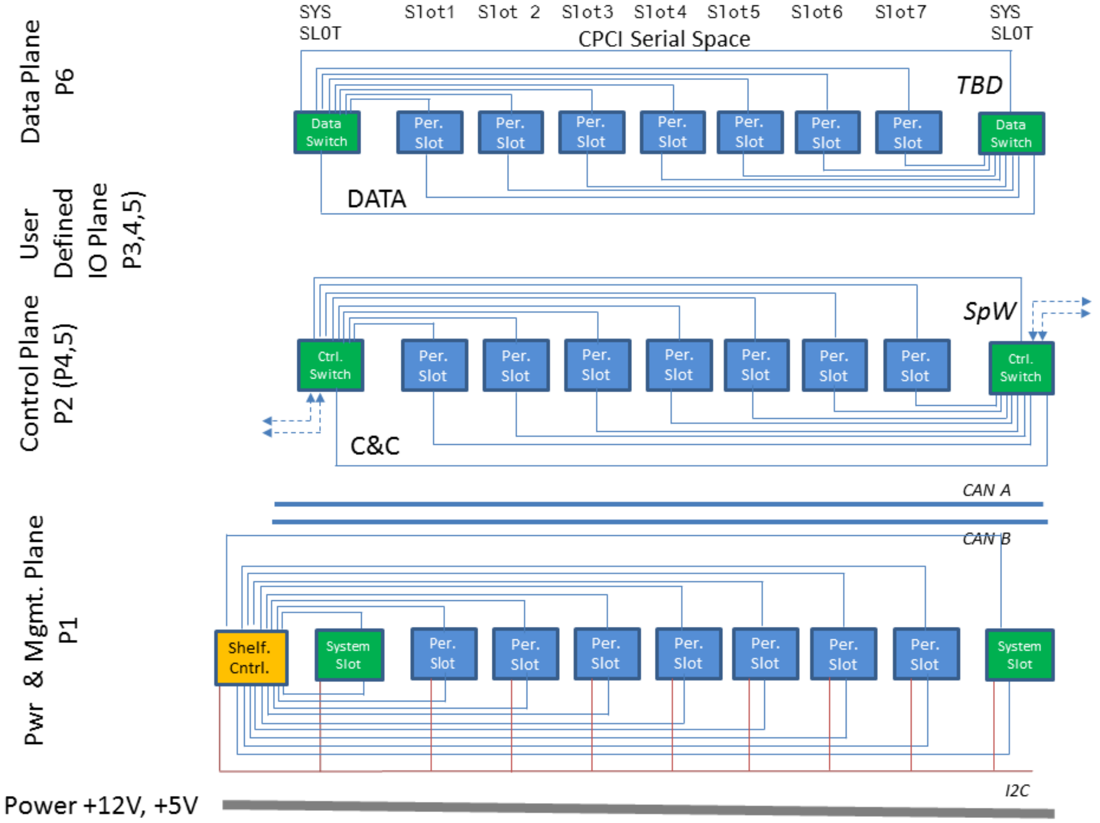


Figure 4.6.: CPCI Serial Space Backplane Interconnections

the system slot via connector P2. The connector P2 provides also the connectivity for the redundant CAN bus. All power and management signal are available on P1. Each peripheral slot provides up to 180 user defineable pins. These could be used for rear connectors or dedicated interconnections between peripheral slots.

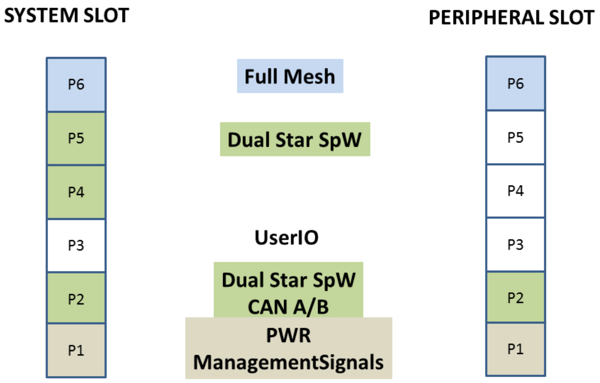


Figure 4.7.: CPCI Serial Space slot Profiles

Switching to redundant units is performed by a reconfiguration unit. CPCI Serial Space and also SpaceVPX have foreseen a separate reconfiguration unit. In CPCI Serial Space this reconfiguration unit is called ”shelf controller”. The standard describes this functionality as follows:

> A shelf controller can control the power supply of all boards separately. Also the shelf controller can check the status of the boards and can reset the boards individually. Two redundant CAN busses are available additionally as board management busses. Neither the shelf controller connector nor the shelf controller itself is specified in this specification.

SpaceVPX

In Space VPX the unit is called spaceUtilityModule. The purpose is similar to the shelf controller in CPCI. All signals are routed over the Switched Utility Plane (system management, reference clocks, reset and power) and the spaceUtilityModule performs switching between redundant units.
Space VPX differentiates between 9 slot profiles (see Fig. 4.8):

1. Power
2. DataIn Module
3. Processing Module
4. Storage Module
5. DataOut Module
6. Controller Module
7. Controller Switch Module
8. DataSwitch Module
9. SpaceUM Module

Table 4.2 shows how the redundancy concept defined in section 4.2.3 is implemented with the different standards.
Table 4.3 shows the mapping of these requirements onto different backplane standards:

Summary and Conclusion

Table B.1 provides a summarized comparison of the CPCI Serial Space and SpaceVPX standards:
Under technical perspective, both standards fullfill the requirements for future modular data handling systems. The number of different profiles in SpaceVPX requires an additional effort for tailoring, which is not needed when using CPCI Serial Space. Accordingly, a SpaceVPX backplane is always application specific whereas a CPCI Serial Space backplane can be considered as a COTS product.
Following the discussion of the embedded community in the Internet, the costs of SpaceVPX are considerably higher than those of CPCI Serial Space.
Therefore, the recommendation is to use **CPCI-S.1 R1.0**.

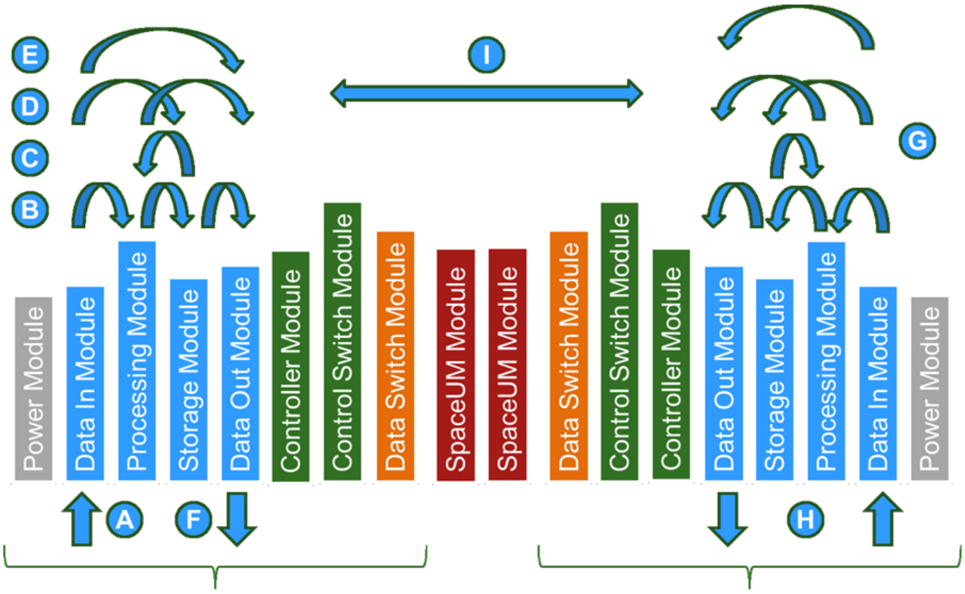


Figure 4.8.: SpaceVPX slot Profiles

Form Factor

There is a strong trend in the commercial market towards smaller form factors. In the CubeSat domain PC104 is the dominating form factor while in other industrial domains 3U (100mm x 160mm) is preferred. As space components and connectors are less integrated, therefore, the following criteria need to be assessed for the form factor trade-off:

Available Space for Connectors, useful area for space qualified EEE-parts, number of needed modules per form factor, forced separation of functions, used modules for simple functions, unit volume, volume used in the S/C, mass, AIT integration, accommodation in small, medium and large spacecraft.

Connectors

It is not expected, that smaller space qualified connectors will be used/developed in the near future, which would lead to a more effective use of the given space. 3U offers 78mm available space for connectors; 6U offers 211mm space, which is 2.7 times of 3U. For equipment/modules with many external connectors such as Remote Interface Units (RIUs) or Mass Memory Formatting Unit (MMFU) a form factor of 6U provides sufficient space to integrate the needed connectors. With a form factor of 3U two or three slots are needed, to integrate the same amount of connectors.

To overcome the drawback of a small front panel for connectors, rear connectors shall be considered - especially, when the connectors determine the board dimensions. However, compact connetors like microD Connectors have to be investigated.

Table 4.2.: Redundancy Concept

Feature	Implementation CPCI	Implementation VPX
Power Distribution	On power bus, redundancy in Power Conversion Unit	Dual-redundant power distribution (bussed) where each distribution is supplied from an independent power source.
Management	Shel controller that monitor health status of all boards and switches between redundant units	SpaceUM module that selects between the A and B management controllers for distribution to each of the slots controlled by the SpaceUM module.
Reset Control	Card-level reset control	Card-level reset control
Power Control	Card-level power control	Card-level power control
Timing / Syncr.	low-skew differential	low-skew differential
Utility Plane	I2C or CAN	I2C
High Speed	Dual redundant (dual star)	Dual-redundant Data planes (dual star)
C&C	Dual-Redundant (dual star)	Dual-Redundant Control planes (point-to-point cross-strapped)

Table 4.3.: Communication Channels

Type	Implementation	Compliance CPCI	Compliance VPX
C&C	CAN, SpaceWire, (Ethercat, TTEthernet, TSN)	C (dual star)	C (switch fabric in dedicated slot)
High Speed Links	SpaceFibre, RapidIO, ... (AFDX, TTEthernet, TSN)	C (full mesh connector)	C (switch fabric in dedicated slot)
(low level) Reporting	UART, SpI, I2C, SpaceWire	(Ethernet, TTEthernet, TSN)	C (CAN, I2C)

EEE-Parts

Furthermore, for equipment/module which are highly populated, a form factor of 6U increase the available space not only by a factor of two, but 2.7. Therewith the needed modules can be reduced and volume and mass can be saved, similar as for modules which are connector driven. For EEE-parts the integration factor will not dramatically increase if we not consider using only COTS parts. The useful area for 3U vs 6U is shown in Fig. 4.9:

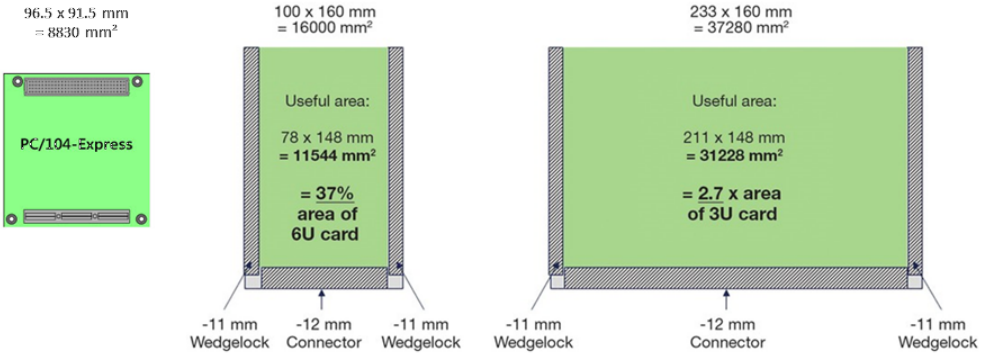


Figure 4.9.: PCB Formfactors

For constellations, were COTS parts can be up-screened with realistic cost impact, a substantial higher integration factor can be achieved. For such huge constellations, as OneWeb a form factor of 3U is sufficient for the population of EEE-parts.

Mezzanine Concept

The Mezzanine concept allows to add complex functionality to a base board. Typically, compute modules are implemented as Mezzanine boards which are plugged into the *carrier* board (see Fig. 4.10). The carrier board provides the standard IO functionality, additional memory, etc. Typically, the interfaces remain stable for a long period of time while functionality grows. More processing power is required to implement the new functionality. In case of a mezzanine architecture the compute module is replaced with a more powerfull one while the mother boards remains unchanged.

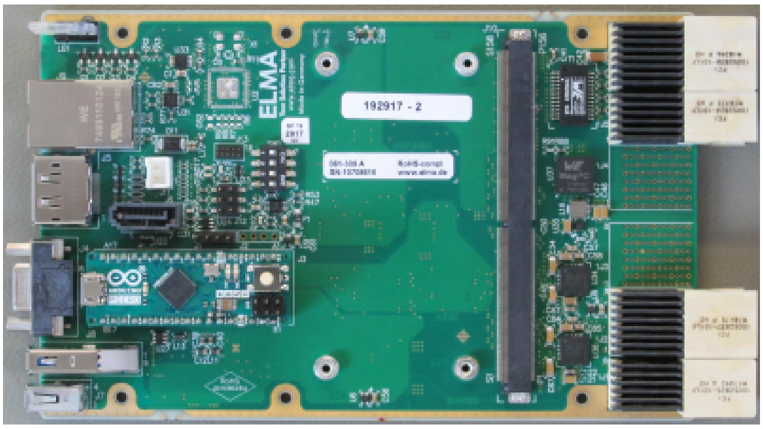


Figure 4.10.: Example of a Mezzanine Carrier Board for SMARC Boards

Dimensions and Mass

The dimensions and mass of a rack depends on the number of boards incl. PCU.

The width of a rack can be calculated as follows: $W = pw * n + 25mm$ where pw is the pitch width (e.g. 2.54 mm) and n is the number of boards (Examples shown in 4.4).

Nom height.	Dimensions W x H x D mm	Remark
3U	175 x 130 x 250	n = 6
6U	250 x 270 x 250	n = 9

Table 4.4.: Rack dimensions

The mass of a rack can be calculated as follows: $M = n * (bm + bbm + hm) + pm\ kg$ where bm is the mass of a board (e.g. 0.35 kg), bbm the backplane mass/slot, hm the housing mass/slot, pm power supply mass and n is the number of boards (Examples are given in Tbl. 4.5)

Item	Mass	Remark
Backplane/slot	0.150	
PowerSupply	0.500	
3U Board	0.350/0.600	
3U Housing/slot	1.000	

Table 4.5.: Mass

A typical rack 3U, 1 PCU, 5 boards (2 with Conduction Cooled Assembly (CCA)) has a total mass of approx. 8 kg.

Accommodation in the spacecraft

In general smaller boxes are always better to accommodate, independent of the size of the spacecraft.
A single 6U box has a higher weight than a single 3U box. This makes a 6U box more difficult to handle for Assembly, Integration and Test (AIT). In case the mass of a box exceeds 15kg, a crane for handling becomes necessary (as we have it already today for big boxes e.g. PCDUs or RIUs). Thus, the mass of a box should be less than 10kg.
For nano- and micro-satellites the accommodation force a small form factor of 3U or even PC104. Small, medium and large satellites provide enough space for the accommodation of a 6U form factors. For the latter it is more important to lower the volume and to reduce the number of boxes for the accommodation than having small boxes.
Volume in the spacecraft is not only length*width*height of the unit. For mounting the boxes and the harness additional space around the boxes is needed. The space around the box is not decreased by smaller boxes. E.g. two small boxes with a lower combined unit volume could need more volume in the spacecraft than one bigger box, even if the bigger box has a higher unit-volume.
Figure 4.11 shall illustrate this. The assumptions are the following:

- Scenario 1: 1 3U box with 9 boards and one PCU: Box footprint: 250x200 mm, handling area: 350x400 mm = 140000m2

- Scenario 2: 3 3U boxes each with 1 board and one PCU: Box footprint: 50x200, handling area: 150x400 mm = 60000m2 * 3 = 180000m2

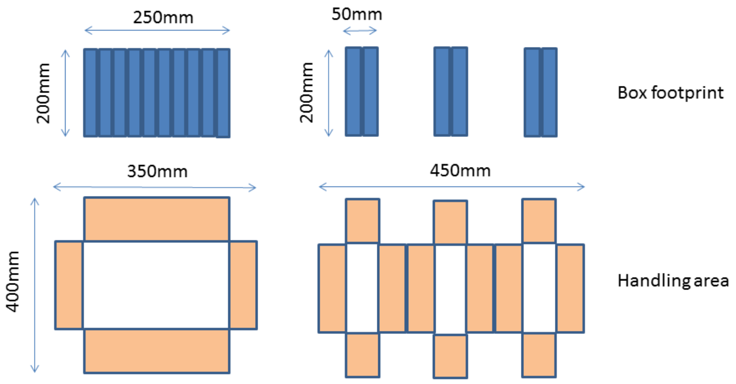


Figure 4.11.: Required space for small and large boxes

The area required for three boxes is approx. 1.3 times larger than for one box.

Summary and Conclusion

In the following table 4.6, the results from the previous analysis are summarized:

Table 4.6.: Comparision Formfactors

Criteria	**3U**	**6U**	**Comments**
Available Space for Connectors	-	+	2.7 times space for connectors in 6U
Useful area for space qualified EEE-parts	-	+	3U offers 37% useful area compared to 6U
Useful area for pure COTS parts in huge constellations	+	0	With the higher integration factor of COTS parts 3U provides sufficient area.
Number of modules	0	+	3U will lead to 3 times the modules compared to 6U
Separation of functions	-	0	Separation lead to technical difficulties
Used area for simple functions	+	0	

continued next page ...

Criteria	3U	6U	Comments
Unit Volume	0	+	Driven by the number of modules, 3U increase the unit volume compared to 6U
Volume in the S/C	-	+	Driven by the number of modules and boxes, 3U lead to essential more volume needed in the S/C compared to 6U
Mass	0	+	Driven by the number of modules and boxes, 3U increase the mass compared to 6U
AIT integration	+	0	Less weight for 3U form factor boxes
Accommodation in nano and micro spacecraft	+	- -	The limited available volume forces a 3U form factor.
Accommodation in small, medium and large spacecraft	0	+	Driven by needed volume of the boxes

For huge constellations, micro and nano satellites a form factor smaller or equal of 3U is mandatory and achievable with COTS components. For institutional missions (1 up to 3 small, medium or large satellites), using space qualified EEE-parts, a 6U form factor is beneficial. As the focus in this study is on smaller satellites a **form factor of 3U (160 x 100 mm)** is selected. The architecture can be accomplished with uRTU with a small 3U form factor.

4.2.3. Redundancy and Reconfiguration

In order to fulfill the requirements on a single-point-failure free implementation, each module inside needs to be doublicated as well as the communication channels. Together with the communication infrastructure described in the next section switching to the redundant module is performed in case one module fails.
Redundancy can be implemented as:

- Cold redundant
- Warm/Hot redundant
- Tripple Modular Redundancy (TMR)

Cold redundancy is the best option wrt. power consumption. In case of a failure the failed modules is switched off and the redundant module is switched on. It takes a few seconds until the second module is fully operational. During this period the module cannot be commanded neither from ground nor from on-board. This is only acceptable for

modules that do not directly contribute to the satellite safety. Typically, OBCs run in hot redundancy to guarantee continuous operation. This requires also continuous synchronization of the two modules and a Safe Guard Memory (SGM) holds the latest system status information. Thus, it is ensured that the redundant module starts at the same point of operation.
TMR is considered only for large satellites where power consumption, mass and volume are less constraining as in small satellites.
For this study **hot redundancy** will be implemented.

Reconfiguration Concept

The reconfiguration is based on three FDIR layers:

1. Component monitoring: Each component shall monitor vital parameters like voltage, current and temperture and report these to the next FDIR level.

2. Equipment monitoring and reconfiguration: The health status information from each component inside an equipment shall be monitored and in case of a failure recovery action (i.e. switchover to redundant module) shall be taken and an event shall be sent to the next FDIR level. The collected health status of all components shall be reported to the OBC.

3. OBC supervision: Based on the health status received, the OBC software decides on a recovery action or reports the event to the next higher level, i.e. to ground.

Management Signals

Management signals are depict in Fig. 4.12.

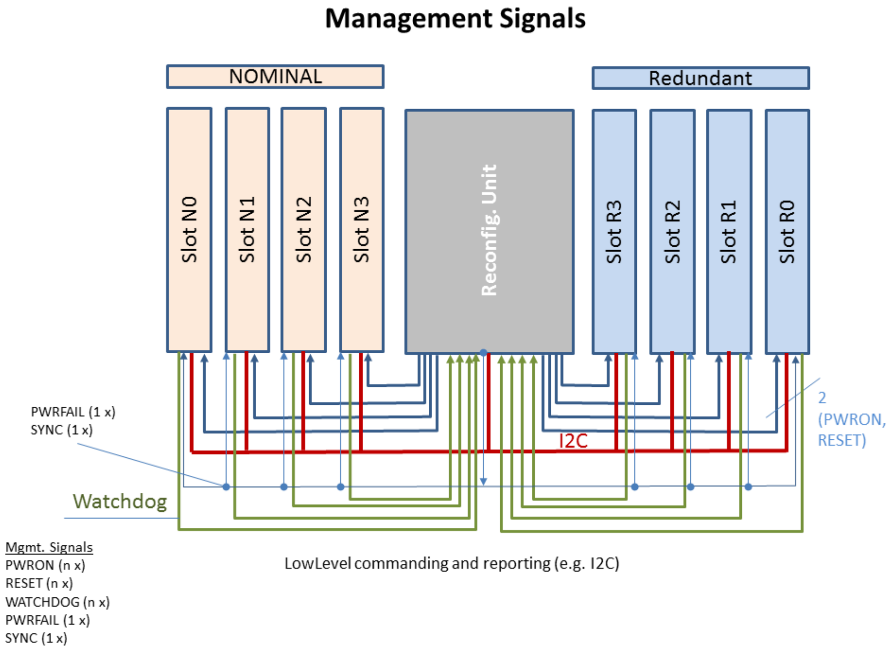


Figure 4.12.: Management Signals

The introduction of the management signals are based on a preliminary Failure Mode Effect Analysis (FMEA) (see Tbl. 4.7). This FMEA covers only functionality allocated to modules inside an equipment.

Summary and Conclusion

A **separate reconfiguration unit** controlling one box is the preferred solution as it is a generic function independent from the functionality of the box. In case the box only contains components with low criticality it can be omitted.

4.2.4. Communication infrastructure

In the SAVOIR reference architecture three different types of communication channels are identified:

1. Platform Command and Control
2. Payload Command and Control
3. Mission Data Link

The following considerations depict in Table 4.8 assume three traffic classes typically implemented with different physical links (in brackets Ethernet based links are listed used in other domains):

Command and Control Bus

For the command and control bus, SpaceWire has proven that it fulfills all requirements on determinism and the provided bandwidth allows data handling system to grow - compared to the solution used today based on the Mil1553 bus. SpaceWire is widely accepted in the European Space community and many equipments already provide SpaceWire interfaces. For small satellites with less demanding requirements on the number of commands and HK packet size CAN bus is recommended. The bus concept will drastically reduce the harness complexity.

In larger satellites CAN bus can be used to command actuators through uRTUs and collect data from sensors.

Ethernet based solutions are highly desired as everybody knows its basic concepts and test infrastructure commonly used in other areas can be used out of the box. With Time Triggered Ethernet (TTE) and Time Sensitive Network (TSN) protocolls are available that allow to combine all three traffic classes over a single pyhsical line. In addition Time Synchronization is one of the basic features of both TTE and TSN.

A single-point-failure-free design is a must in space, i.e. alternate communication paths between the source and destination devices need to be provided. As standard, SpaceWire and Ethernet do not allow rings or loops in the network as this would result in data frames circulating endlessly and flooding the network. The network infrastructure must therefore support redundancy protocols designed to negate the usual problems of putting loops into a SpaceWire or Ethernet network, maintaining a default data path and switching to an alternate one when a fault occurs.

Fig. 4.13 illustrates this architecture for the Command and Control bus with two Core Processing Module (CPM) representing a pair of redundant on-board computers that support link aggregation connected to a simple SpaceWire/Ethernet ring using Rapid Spanning Tree Protocol (RSTP).

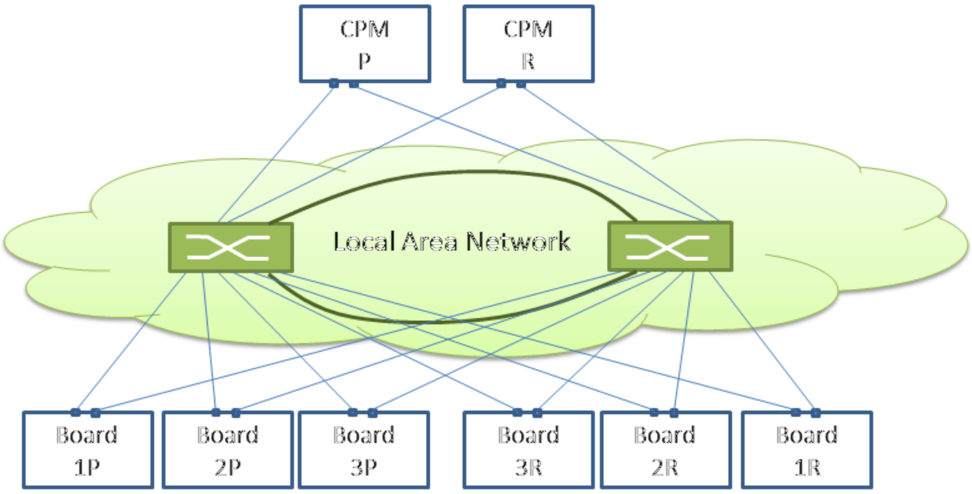


Figure 4.13.: Network Architecture

High Speed Mission Data Link

As of today high speed serial links are used for the mission data links, e.g. RapidIO or SpaceFibre. In other industrial domains Ethernet is replacing traditional field busses. E.g. in aircrafts AFDX is used while Ethernet enters cars for the multi-media infrastructure. In space Ethernet is used on the ISS and Ariane 6 is using TTEthernet as communication backbone as well as NASA's Orion capsule. The upcoming standard on TSN will add determinism to the Ethernet as well as time synchronization. This makes it very attractive for space as all traffic classes describe above can be handled via one physical link.

As the envisaged COTS devices used in this study provide Ethernet as standard interface we would like to propose to use Ethernet for the mission data link. Currently, the number of rad-hard chips for the physical layer (PHYsical layer transceiver (PHY)) is small and most of them are under ITAR restriction. The H2020 project SEPHY finished in 2018 was aiming at the development of a rad-hard PHY. The qualification of the chip needs to be done, but it is expected that in less than two years rad-hard devices from European suppliers will be available.

In order to open the door for future high speed data links (1 Gbps and more) we propose to implement the high speed mission data link based on Ethernet technology. It is planned to use chips that have been screened by Airbus which could be replaced later on with rad-hard devices.

Time Synchronization

With TTE and TSN time synchronization comes for free with the network while separate cables are required for all other solutions.

Intrabox Communication

Each module inside a box has to be connected to the C&C bus, the primary and the redundant one. Both SpaceWire and Ethernet are point-to-point communication links, i.e. a router/switch is required to connect all participants. The second router/switch is required to implement the redundancy.

For high speed communication links a full mesh network provides the highest flexibility and configurability.

Interbox Communication

In a spacecraft it is necessary to connect several boxes to the C&C bus but also dedicated high speed links between boxes are needed, e.g. to connect an instrument/camera to the mass memory or the data processing unit.

The C&C bus used between boxes shall be the sames as in the box. If full redundancy is required between all racks additional hardware (routers/switches) needs to be introduced. In case of simpler structures, direct connections between boxes may be used. uRTU may be connected via the redundant CAN bus as the bus topology simplifies the harness.

Test and Service Interface

Each CPM shall provide a test and service interface. Typically, serial links (UART, RS232, RS422, ...) are used for this purpose. In case the CPM provides an Ethernet interface, this can be used as it combines high bandwidth with a rich set of test utilities.

Summary and Conclusion

In general Ethernet would be the preferred solution as it allows to combine all traffic classes in one physical channel - when TTE or TSN is used. The availability of rad-hard devices is a major drawback as the reliability of the communication backbone should be close to 100%. This is not achievable with COTS devices.

In the European space community SpaceWire is well established and test equipment for SpaceWire is considered not to be problem. IP cores for end systems and routers are available from ESA. A deterministic variant of SpaceWire is under development.

For high speed data links SpaceFibre has been developed and is going to enter space equipment.

The summay of recommendations for different communication channels is given in Table 4.11

4.2.5. Power Chain

The power distribution over the backplane can be implemented in different ways depending on the application. The figures below show typical examples. Fig. 4.14 shows a common power line for all slots while in Fig. 4.15 each slot has its own power lines.

Each board needs to be protected against over current as advanced high performance semiconductor devices can be sensitive to Single Event Latchup (SEL) effect when exposed under radiation in the space environment.

Even if SEL is a very rare event, it can lead to a self destruction of the device and shall be mitigated to ensure the relevant reliability and life time of the application. Therefore, a safe design for a mission critical space application shall include a protection device called the Latch up Current Limiter (LCL).

The LCL monitors the power supply line of the radiation sensitive device and switches it off instantaneously in case of any *radiation induced SEL* or any other overvoltage in order to protect the device from over current and overheating.

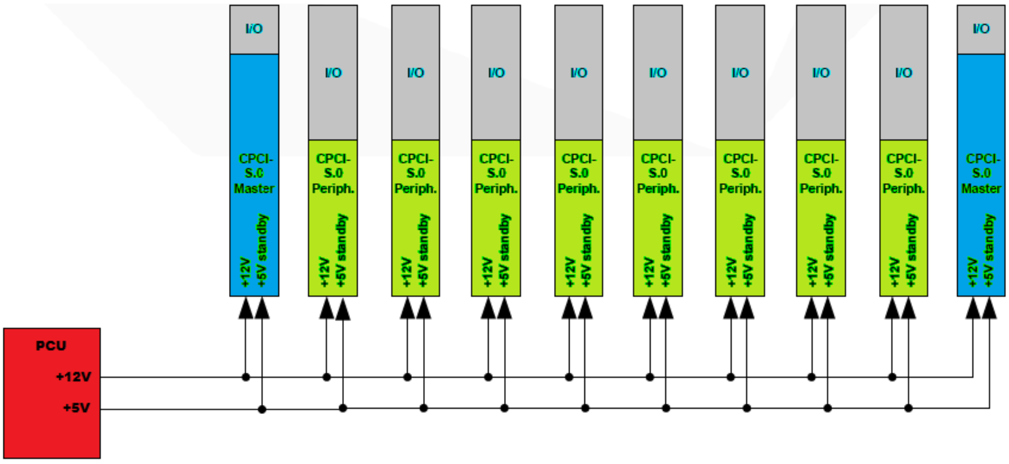


Figure 4.14.: CPCI Serial Space Power Distribution Example 1

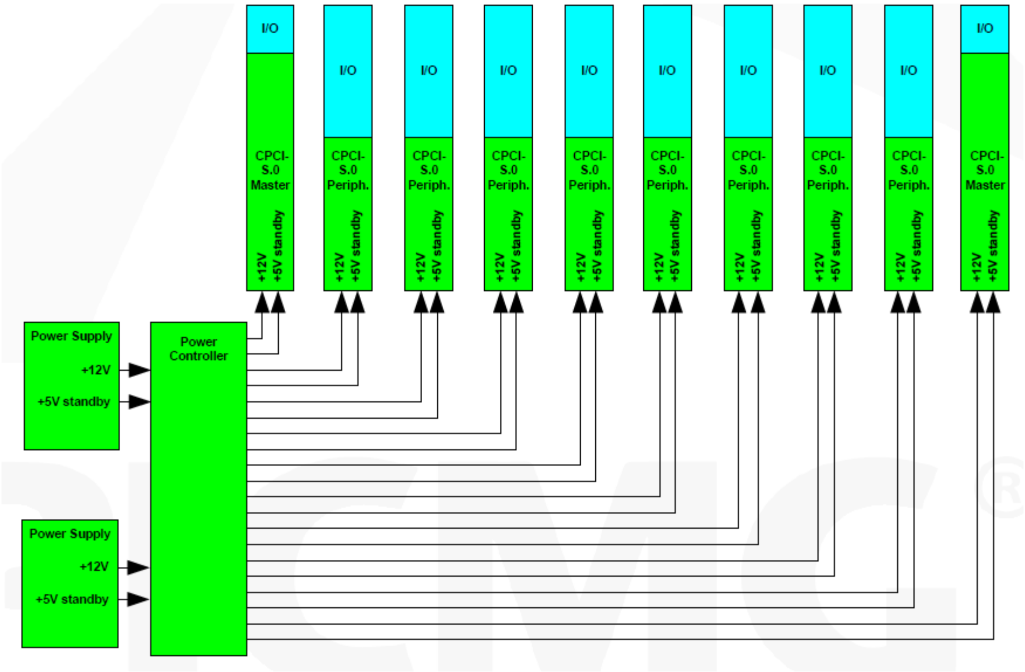


Figure 4.15.: CPCI Serial Space Power Distribution Example 2

Typically, the LCL module offers two adjustable threshhold currents: to control device Run & Standby currents and two operating modes: Automatic reconnection with adjustable delay or reconnection through ON/OFF command.

4.2.6. EEE Parts Availability

Preference shall be given to parts that do not have International Traffic in Arms Regulations (ITAR) restrictions.

4.2.7. Compliance to SAVOIR

The architecture of a centralized data handling architecture shall be compliant to SAVOIR (cf. Section 3.2).

4.3. Overall Summary and Conclusion

Based on the discussion above we propose a system architecture that is characterized as follows, while the software is executed on the CPM:

1. Modular,
2. rack-based,
3. 3U formfactor e.g. CPM, MM, GPS as separate modules
4. CPCI Serial Space backplane standard
5. C&C based on SpaceWire
6. High Speed based on Ethernet
7. CAN as C&C for small satellites

Table 4.7.: Preliminary FMEA

ID	Failure	Detection	Action
1	Primary Power fails	box dead, PCU needs to detect this and generate a signal to CPU boards for controlled shutdown	re-power
2	Main secondary power fails	no watchdog signals	re-power
3	Auxilary power fails	Cntrl. and montoring signals all low, low level HK reporting	restart PCU auxiliary power
4	Peripheral Board fails completely	no watchdog signals	reboot, switch to red. board
5	Software fails	no watchdog signal	reset board
6	C&C Router in system slot fails	timeout on cmds	Retry, reset, reboot, switch to redundant system slot
7	Primary C&C bus fails on one board	timeout on cmds	Retry, reset, reboot, switch to red. channel
8	SEU in memory	Internal CRC check raises event	reboot
9	SEU in silicon	current increased, on-board monitoring	reboot
10	Bit failure in cmd / telemetry	CRC check raises event	reset, reboot
11	High CPU load due to SW failure	Internal CPU load monitoring, temperature monitoring	reset
12	Running short of RAM	Internal RAM consumption monitoring	reset
12	uRTU fails	Timeout on cmds, no TM	Retry, reset, reboot

Table 4.8.: Traffic Classes

Traffic class	**Data rate**	**Used for**	**Implementation**
Real-Time deterministic	1 - 100 Mbps Latency: <100us, Jitter: <10us	C&C	MIL1553, CAN, SpaceWire, (Ethercat, TTEthernet, TSN)
Rate constrained	>1Gbps	High Speed Links	SpaceFibre, RapidIO, ... (AFDX, TTEthernet, TSN)
Best effort	<1 Mbps	Mgmt. & Reporting	UART, SpI, I2C, SpaceWire (Ethernet, TTEthernet, TSN)

Table 4.9.: Intrabox Communication

Type	**Speed/ Deterministic**	**Technology**	**Redundancy**
C&C	>200 Mbps deterministic	SpaceWire	Dual Star
C&C (low speed)	1 Mbps Deterministic	CAN	2 redundant busses
High Speed	>1 Gbps Non deterministic	Ethernet (, RapidIO, SpaceFibre)	No redundancy

Table 4.10.: Interbox Communication

Type	**Speed/ Determinism**	**Technology**	**Redundancy**
C&C	>200 Mbps deterministic	SpaceWire	Yes, 4 connectors on front panel
C&C (for connecting uRTUs)	1 Mbps Deterministic	CAN	Yes, 2 connectors on front panel
High Speed	>1 Gbps Non deterministic	Ethernet (,RapidIO, SpaceFibre)	no, connectors on front panel

Table 4.11.: Recommendations

Type	**Recommendation**	**Remark**
C&C	SpaceWire	SpW is widely accepted in the European Space community
C&C (for connecting uRTUs)	CAN	The bus structure simplifies harness and speed is sufficient (1 Mbit/s)
High Speed	Ethernet, SpaceFibre ,RapidIO, ...)	Ethernet is the preferred solution but today rad-hard devices are not available. Therefore, SpaceFibre could be used as intermediate solution
Interbox Management bus	I2C	
Service IF	Ethernet	

5. Proposed DHS Architecture

5.1. Hardware Architecture

According to the SAVOIR Functional Breakdown (Fig. 3.4) the Data Handling System (DHS) comprises the following functions:

- Telecommand, Telemetry, Security
- Reconfiguration, Save-Guard memory, Essential TC, Essential TM
- Processing
- Platform Storage
- On-Board Time, Time reference
- Cmd & Ctrl. Links
- Mission Data Links, Payload data routing
- Data Concentrator, Sensor and Actuator I/F
- Payload Data Storage
- Payload Telemetry, Security

These functions can be mapped to modules as shown in the Table 5.3:
The C&C Router Device can be integrated with another module, e.g. the Core Processing Module or the Platform Storage Module while the Mission Data Router Device can be integrated with the Payload Data Storage Module. An additional modules is required to provide power to all other modules. As a result of this integration we get the following list of boards, i.e. specific implementation of a module as PCB:
The column *Profile* specifies the slot profile of the given backplane standard: PWR=Power, P=Peripheral Slot ,S=System Slot, R=Reconfiguration Slot, RM=Remote Module
The total number of 18 boards requires two CPCI Serial Space based Core Control Units (CCU1, CCU2), each with two system slots, six peripheral slots, one reconfiguration slot and one power slot.

Function	Unit/Module/Device	Redundancy
Telecommand, Telemetry, Security	S-Band Module	hot redundant
Reconfiguration, Save-Guard memory, Essential TC, Essential TM	Reconfiguration Module	hot redundant
Processing	Core Processing Module	hot redundant
Platform Storage	Platform Storage Module	cold redundant
On-Board Time, Time reference	GNSS Receiver Module	cold redundant
Cmd & Ctrl. Links	C&C Router Device	hot redundant
Mission Data Links, Payload data routing	Mission Data Router Device	cold redundant
Data Concentrator, Sensor and Actuator I/F	IO Module, uRTU	cold redundant
Payload Data Storage	Payload Data Storage Module	cold redundant
Payload Telemtry, Security	Antenna Ctrl. Module, Downlink Module	cold redundant

Table 5.1.: Mapping SAVOIR functions to hardware components

All boards are connected to the C&C link based on SpaceWire, i.e. the position of an individual board (in CCU1 or CCU2) is not relevant for the design.
Fig. 5.1 shows the physical interconnection scheme for the C&C link between the two CCUs while Table 5.3 shows the allocation to the CCUs. Logically, each board can be reached from the system slots via two different paths. This is important for later design descisons on the number and allocation of the CPMs controlling the boards in the peripheral slots.
The traditional way of grouping functionalities into platform and payload becomes obsolete as the usage of an international backplane standard allows to allocate the boards according to other technical criteria, e.g. balancing power consumption or thermal dissipation. The entire DHS can be implemented with two boxes and a set of uRTUs (the exact number depends on the mission specific IO requirements) as shown in Fig. 5.2.

5.2. Intra and Interbox Communication

In case the C&C link is based on point-to-point interconnections (e.g. SpaceWire,SpaceFibre, Ethernet) a router is required. In this case a dual star interconnection scheme will provide the required single-point-failure free interconnection.
Fig. 5.3 shows the rack internal interconnection scheme. Two slots in the rack have to provide the routing capability. Each node is connected to the primary and the redundant router.

Board	**Profile**	**No. of Boards**
Power Conversion Board	PWR[1]	2
Reconfiguration Board	R[2]	2
Core Processing Module incl. C&C Router Device	S[3]	2
Platform Storage Board incl. C&C Router Device	S	2
GNSS Receiver Board	P[3]	2
IO Board	P	2
Payload Data Storage Board incl. Mission Data Router Device	S	2
Antenna Ctrl. Board	P	2
Downlink Board A	P	2
Downlink Board B	P	2
Total		**18**
uRTU	RM	3
S-Band Board	RM	2

Table 5.2.: Mapping SAVOIR functions to hardware components

The number of boards inside the rack is mainly determined by the number of physical links provided by the connector but also by the complexity of the router. In Fig. 5.3 the router has 10 ports which limits the number of boards inside the rack to seven. Two ports are reserved for external connections and routed to the frontpanel.

In order to use the same C&C link inside the racks and between racks (requirement /REQ-DES-COMM-001/) a separate router is required - if more than two racks need to be connected (see Fig. 5.4).

This interconnection scheme has some impact on the harness complexity, i.e. it shall be limited to connect only racks. For connecting smaller devices a bus type of interconnect is much better wrt. to harness complexity. In case of bus type of interconnection (e.g. CAN, MIL1553) only two separate links (primary and redundant) have to be provided.

This type of interconnect is the ideal choice for collecting data from MicroRIUs or RIUs (see Fig. 5.5) - if the data rate is sufficient. For cables with a length of 40 m a data rate of 1 Mbit/s is possible with the CAN bus. The number of nodes is limited to 128. Both figures are considered not to be a problem for satellites.

Other functionality may be integrated with the basic modules when they are compliant to the CPCI Serial Space Standard. This includes the GNSS receiver, IO boards to interface with the AOCS sensors and actuators, etc.

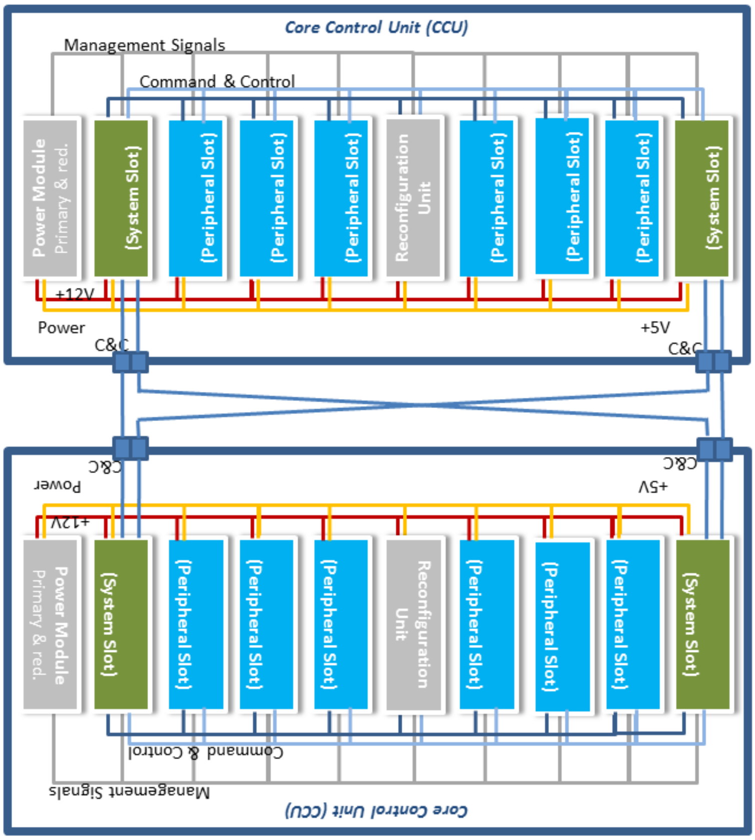


Figure 5.1.: C&C Physical Interconnection Scheme

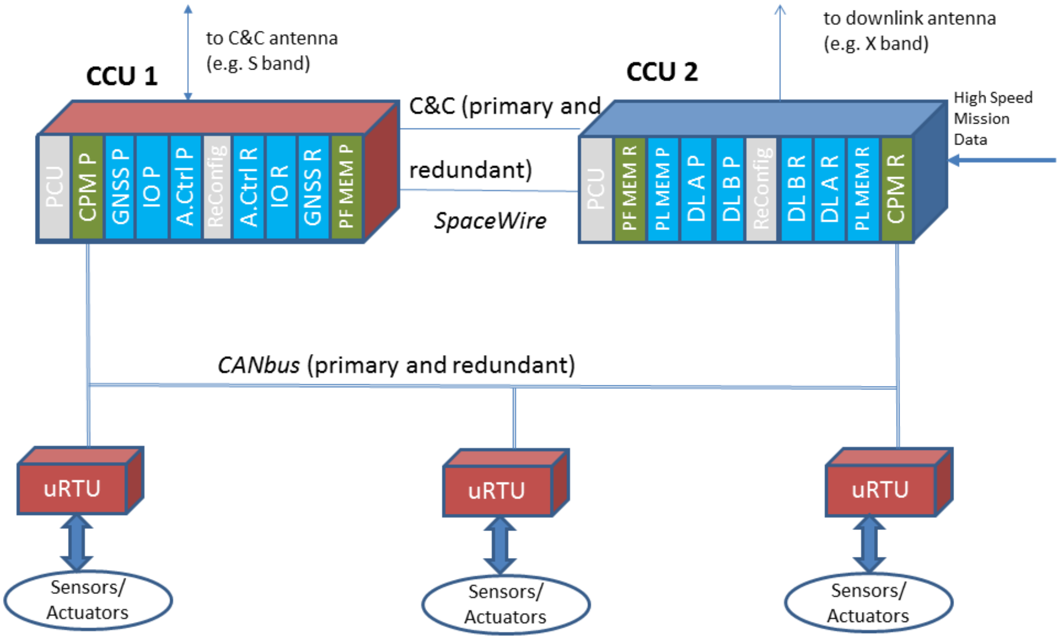


Figure 5.2.: Basic Configuration

Board	ID	CCU	Slot
Power Conversion Board	PCU	1	PWR1
Reconfiguration Board	ReConfig	1	R1
Core Processing Module incl. C&C Router Device	CPM P	1	SP
Platform Storage Board incl. C&C Router Device	PL MEM R	1	SR
GNSS Receiver Board	GNSS P/R	1	P1P,P1R
IO Board	IO P/R	1	P2P,P2R
Antenna Ctrl. Board	A.Ctrl. P/R	1	P3P,P3R
Power Conversion Board	PCU	2	PWR1
Reconfiguration Board	ReConfig	2	R1
Core Processing Module incl. C&C Router Device	CPM R	2	SR
Platform Storage Board incl. C&C Router Device	PL MEM P	2	SP
Payload Data Storage Board incl. Mission Data Router Device	PF MEM P/R	2	P1P,P1R
Downlink Board A	DL A P /R	2	P2P,P2R
Downlink Board B	DL B P/R	2	P3P,P3R

Table 5.3.: Allocation of boards to CCUs

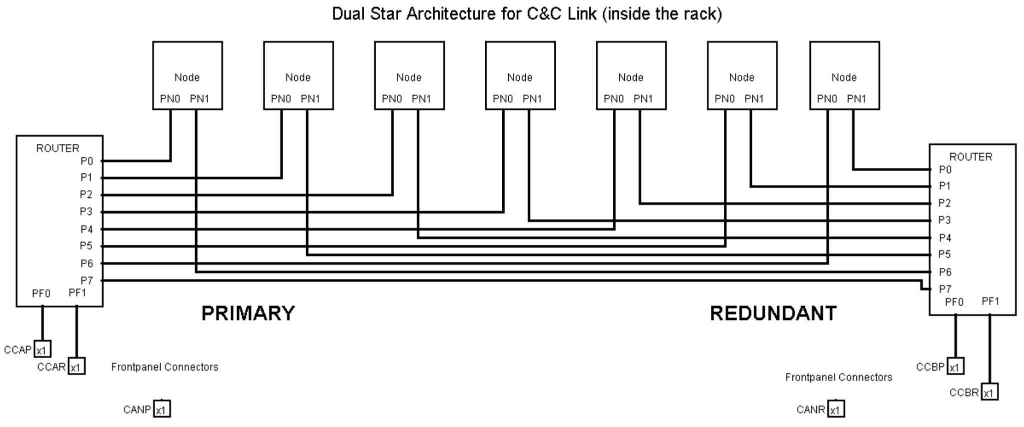


Figure 5.3.: Box internal Network Architecture

5.3. Software Architecture

The hardware architecture described in the previous sections assumes only one Core Processing Module. This means that software functions of different criticality have to be executed on one powerfull (multi-core) CPU. Combining software of different criticality requires a strict time and space separation at operating system level. Typically, this is provided by Time and Space Partitioning (TSP) operating systems like PikeOS, VxWorks653, Lynx, xTratum, AIR, ... Fig. 3.7 shows the software architecture to control the hardware

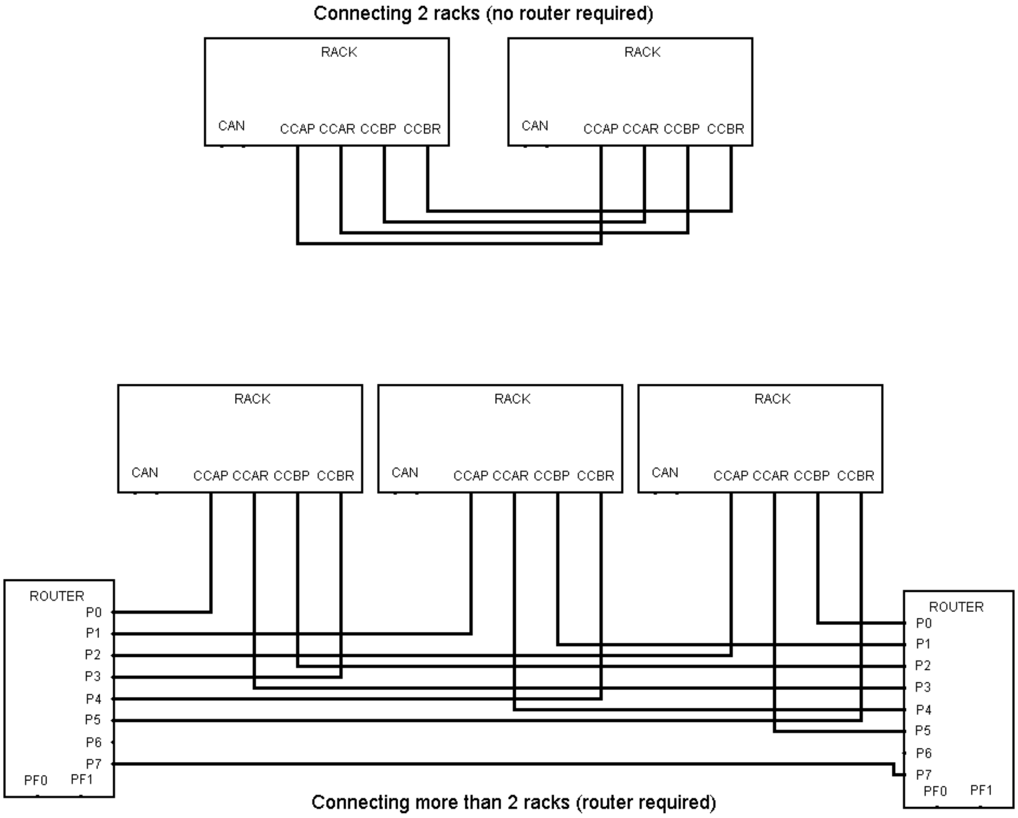


Figure 5.4.: Network Architecture between Racks

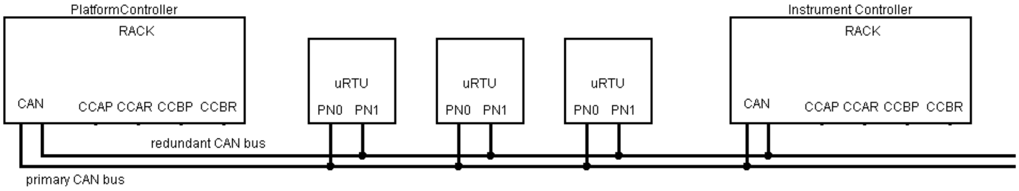


Figure 5.5.: CAN bus architecture

allocated to two CCUs. Each software component (or App) is executed in a separate space partition. A partition is a container (protected memory area) in which the App is executed. Any software failure will have no effect to other software executed in other containers. These partitions are scheduled according to the user requirements.

5.4. Physical Characteristics

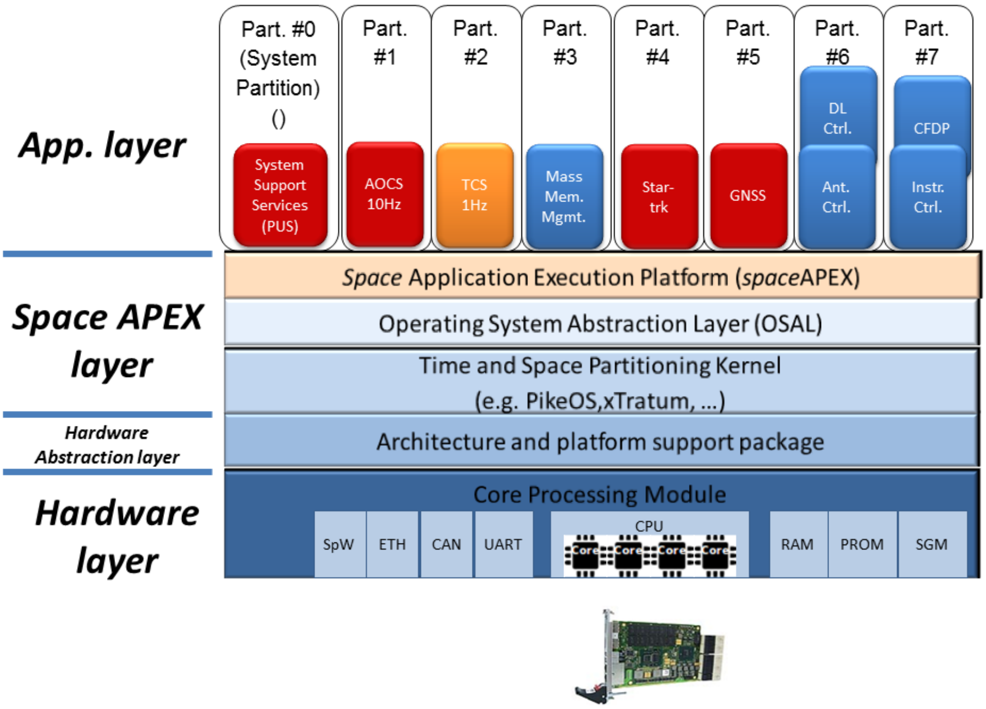

App. layer
Part. #0 (System Partition) ()
System Support Services (PUS)
Part. #1
AOCS 10Hz
Part. #2
TCS 1Hz
Part. #3
Mass Mem. Mgmt.
Part. #4
Star-trk
Part. #5
GNSS
Part. #6
DL Ctrl.
Ant. Ctrl.
Part. #7
CFDP
Instr. Ctrl.
Space APEX layer
Space Application Execution Platform (spaceAPEX)
Operating System Abstraction Layer (OSAL)
Time and Space Partitioning Kernel (e.g. PikeOS,xTratum, ...)
Hardware Abstraction layer
Architecture and platform support package
Hardware layer
Core Processing Module
SpW
ETH
CAN
UART
CPU
Core
Core
Core
Core
RAM
PROM
SGM

Figure 5.6.: Software Architecture

Item	Dimensions W x H x D mm	Mass	Power Consumption
CCU1	250 x 130 x 250	12	20W
CCU2	250 x 130 x 250	12	60W
uRTU	250 x 60 x 250	4	8W
Total	–	36	104W

Table 5.4.: Physical Characteristics

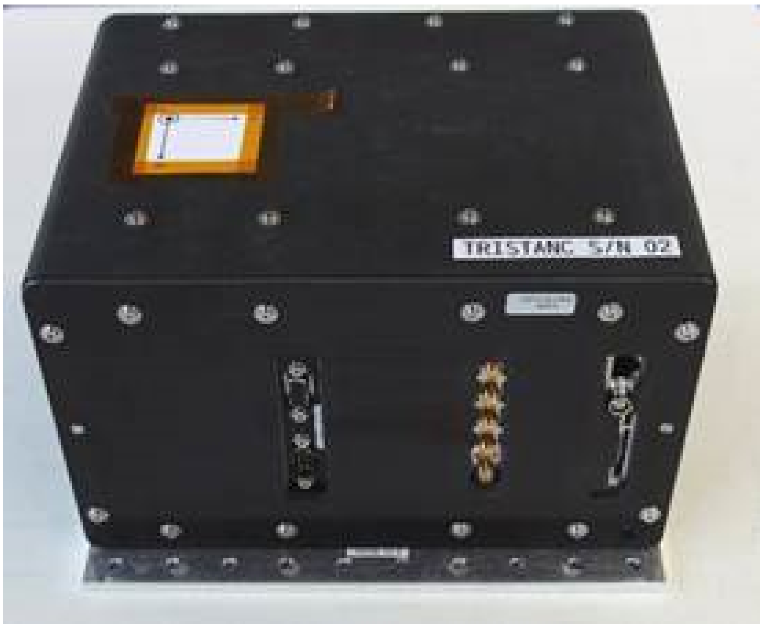
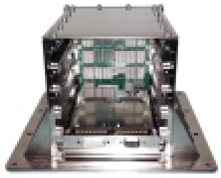
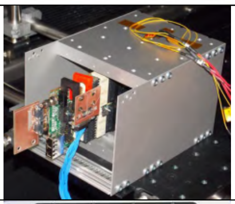
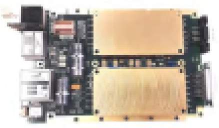
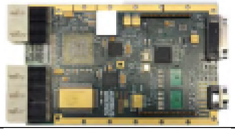
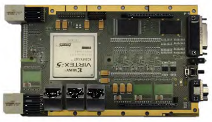
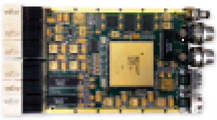
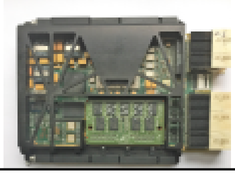


Figure 5.7.: Demonstrator Box (Source: TAS-D)

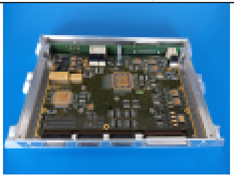
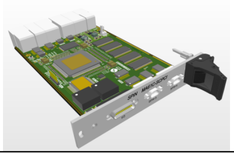
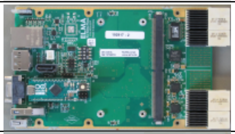
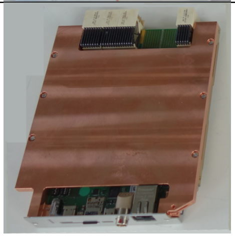
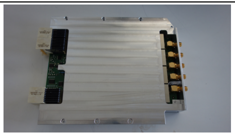
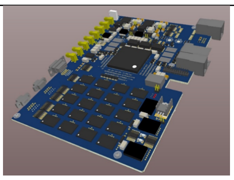
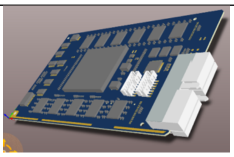
A. Available Components and Boards

In various DLR projects several cPCI Serial Space boards have been developed and tested:

Table A.1.: Available cPCI Serial Space compliant hardware components

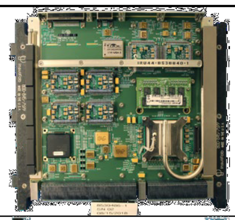
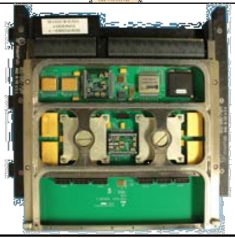
Titel	Bild	Firma	TRL	Mass [kg]	Power [W]
Box for ISS		Airbus	6	2.4	N/A
Box for Sat.		FOKUS	6	4.2	N/A
Box shielded (Tristan-C)		TAS-D	6	4.2	N/A
PowerConverter		Airbus	6	0.3	
Leon4 Board incl. SpW Router		Airbus	6	0.3	10
TMR Board		Airbus	6		
FPGA Board (RTG4)		FOKUS	6	0.3	10
P4080 Board (8-core CPU)		FOKUS	6	0.4	30

continued next page ...

Titel	Bild	Firma	TRL	Mass [kg]	Power [W]
Utility Board		FOKUS	4	0.2	1
Remote Data Concentrator (GR712 and various IO)		STI	6	0.85	10
MA61CSPCI-EM (GR712 and various IO)		SPiN	4	0.3	2
SMARC Carrier Board		ELMA			
Zynq (4+2-core ARM, US+ FPGA, 4GB RAM, equiv. >50GFlops)		TAS-D	6	1.6 incl. heat sink	11.4
RF board (4 channel precision Rx, small LEO)		TAS-D	6	0.85 incl. heat sink	6.5
cPCI MMM (RTG4 based Mass Memory Module)		DSI-AS	3	1.9 incl. mechanics	< 25W
cPCISS (High Performance Data Processing Unit (HPDPU))		DSI-AS	3	< 0.7	< 15W

The table below shows SpaceVPX components.

Table A.2.: SpaceVPX compliant hardware components

Titel	**Image**	**Company**	**TRL**	**Mass [kg]**	**Power [W]**
RAD5545 Single Board Computer		BAE System			
Reconfigurable Computing Module		BAE System			

B. SpaceVPX vs. cPCI Serial Space

Table B.1.: Backplane Standards

Criteria	Need	cPCI Serial Space	SpaceVPX	Remark
Backplane Layout	Low testing and qual. effort	Number of peripheral slots varies from 0 to 7	Number of slot types per backplane may vary for different configurations	
Form factor	6U (3U when technology evolves)	3U / 6U 6U one connector may be added for more power	3U / 6U Additional Connector for 6U	
Connector	qualifyable	Airmax VS Proven in harsh environment, less complex, less cost	MultiGitg RT Proven in space	
Total number Pins per board		Up to 184 pin pairs with serial data rates up to 12.5 Gb/s (3U/6U)	Up to 192 pin pairs and 48 single-ended (6U)	
Command & Control link	Single-point failure free, up to 100 Mbps	SpW dual star; switch in system slot	SpW Dual star; Switch fabric in dedicated slot	
High speed link	10 Gbps, flexible interconnection scheme	Full mesh connector 4 differential pairs per link.	Dual star with switch in system slot Full Mesh RapidIO Dual star; Switch fabric in dedicated slot	Higher needs could be covered by additional module to module connection
C&C or Management bus		CAN, I2C	I2C	

continued next page ...

Criteria	**Need**	**cPCI Serial Space**	**SpaceVPX**	**Remark**
User defined IO		72 pins in system slot 180 pins in peripheral slot Varies with each slot profile	User defined IOs can be used for additional connections between boards (project specific)	
Secondary Power Voltage	Low complexity preferred	12V and 5V auxiliary voltage, POL	+3.3V, +5V, +12V, 3.3 V AUX, and 48V, POL for big racks	DC/DC is easier to realize and to size in with number of supply voltages,
Rack controller	No controller preferred	Shelf controller possible, not part of the standard	Utility management module is foreseen but seems avoidable	
Slot types	-	Power-, System-, peripheral-slot	Power-, System-, Control-, Payload-, Storage, Switching-slot	VPX provides more flexibility. cPCI provides more standardization.
Number of slots	>8	9 Slots	Up to 16	
Power dissipation	Up to 40W per slot	Up to 80 W per slot	More than 80W per slot	Thermal analysis required for each box
Standardization, e.g. pin assignment and connectors.	Fully standardized to enable re-use and to avoid re-qualification	Fully standardized	Partly standardized, provides a lot of flexibility	In case of SpaceVPX additional standardization necessary.
Market	Common standard	Mainly Europe	US, driven by VPX used in defence	In space still not established.