Software Interlocks in the METASAT Project

MODULAR MODEL-BASED DESIGN AND TESTING FOR APPLICATIONS IN SATELLITES

Alfred Hönle and Leonidas Kosmidis

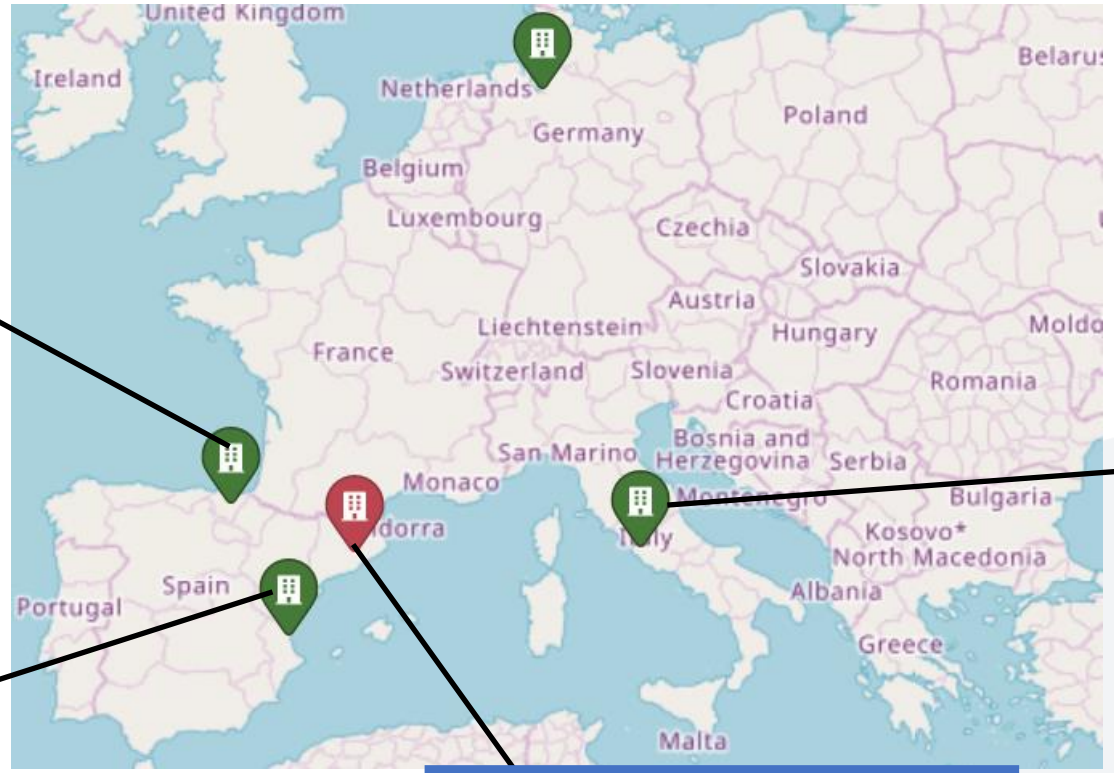
23/10/2024

Contents

METASAT Consortium

- 2-year Horizon Europe project: January 2023-December 2024
- TRL 3-4

OHB



ikerlan
MEMBER OF BASQUE RESEARCH & TECHNOLOGY ALLIANCE

Collins Aerospace

fentISS
FENT Innovative Software Solutions

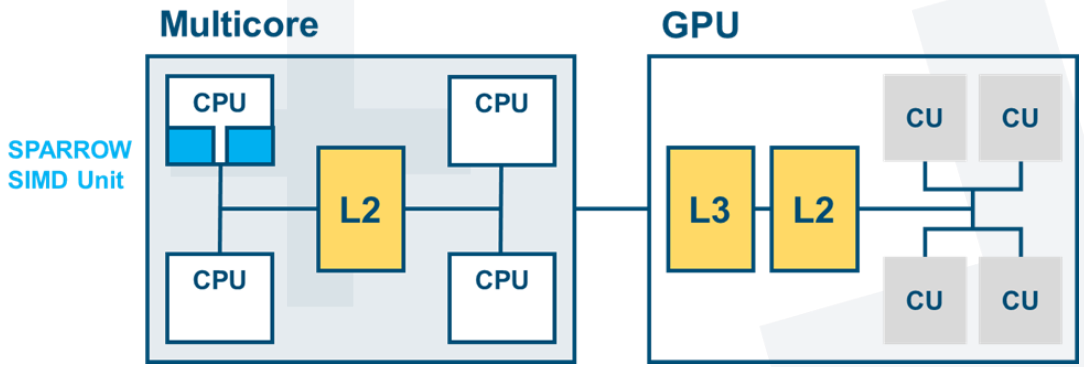
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

METASAT Overview

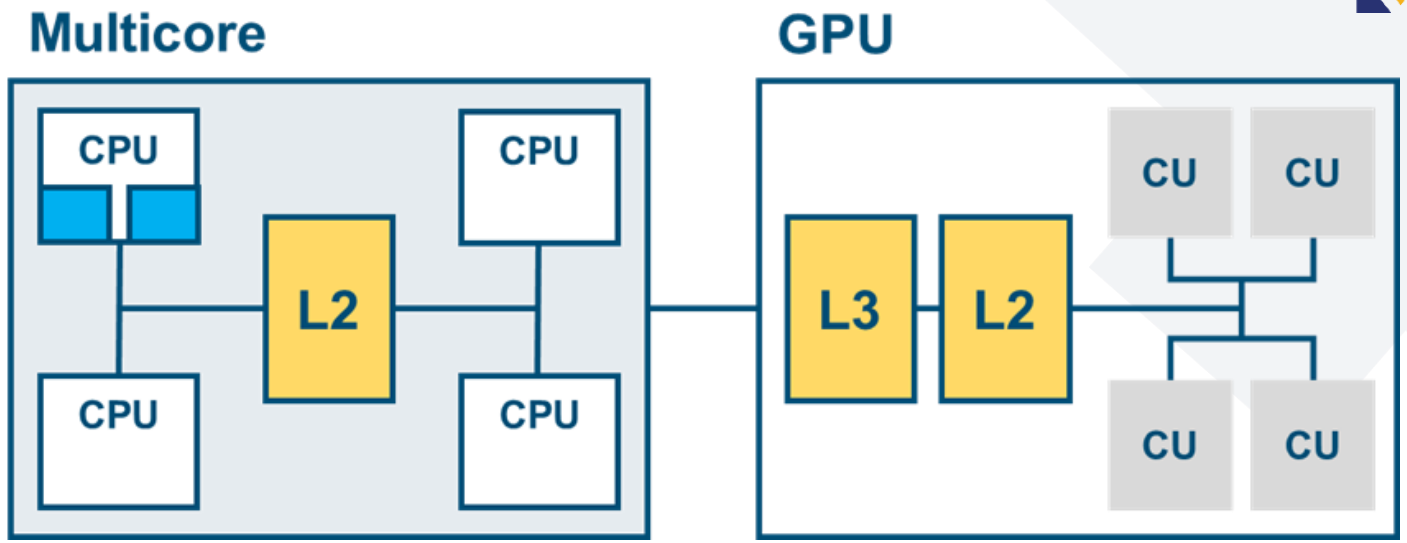
- Modern aerospace systems require new, advanced functionalities
  - Artificial Intelligence (AI)
  - High performance payload processing
  - High resolution sensor preprocessing etc
- Advanced functionalities require complex hardware and software compared to the existing space technologies
- High Performance Hardware technologies: Advanced Multi-cores, GPUs, AI accelerators
- Programming high performance hardware requires complex software: parallel and GPU programming

Hardware Selection

- No hardware with high-performance and architectural complexity exists for the space domain
- COTS Embedded Multicore and GPU devices provide these features but depend on non-qualifiable software stacks
  - GPU drivers available only for Linux
  - Blocking point for use in institutional missions
- Design a prototype hardware platform based on the RISC-V ISA



The METASAT Hardware Platform



- Qualifiable Software Stack: accelerators can be used from bare metal or RTEMS SMP
- Mixed-criticality: TSP support
- Added support in Model-based design tools
- Digital Twin

Space System Interlocks

- In the space domain: high demands on the overall system dependability
- One possible commonly agreed approach for handling this are architectures with redundancies
- Example for a dissimilar redundancy: the regular control and processing instance (e.g. the control SW of a satellite instrument) shall be monitored
  - by a strictly independent module that acts as an interlock …
  - … and steps in when a critical condition is met

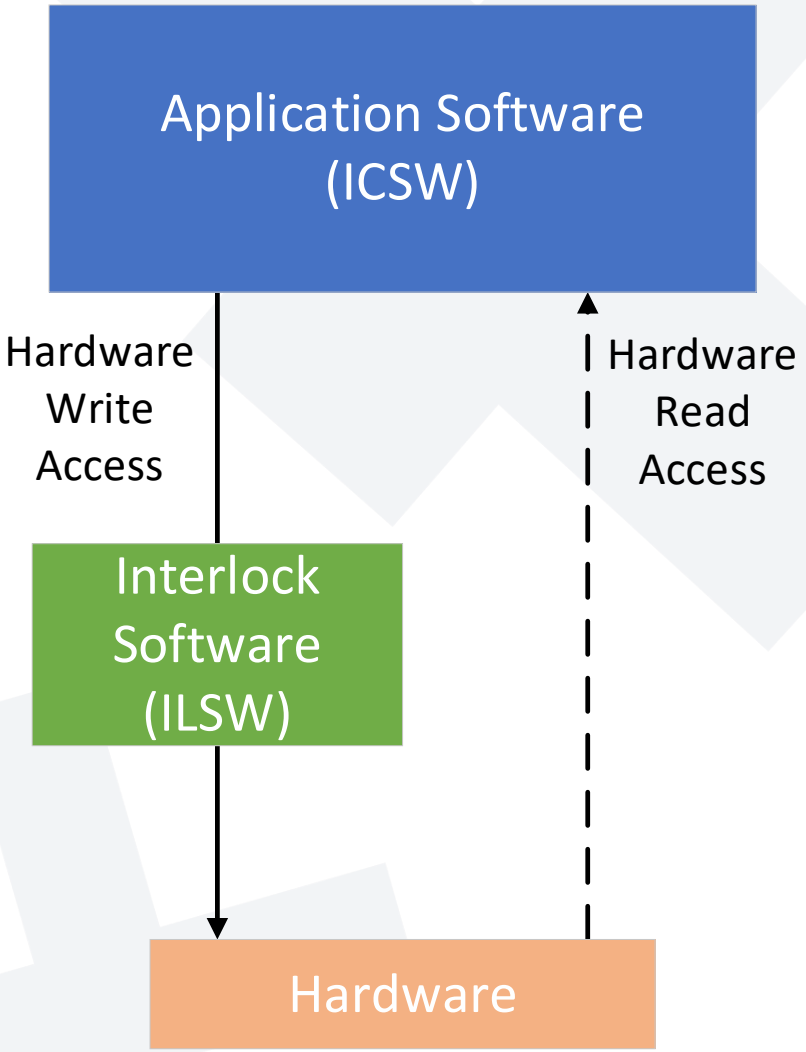
- Such an interlock module being implemented in hardware (like in the past at OHB) has several drawbacks
  - A HW solution typically has a rather simple failure isolation functionality
  - Adds extra resource budgets (extent of electronics and/or FPGA size grows)
  - Adaptations during the project may generate high costs and may pose a risk for the project schedule
  - The solution is rather inflexible when the operational conditions change in space

Motivation: Better in Software

- A software solution for interlocks would not have these disadvantages. Idea / scenario:
- Main application software with sophisticated system functionality
  - Complex; multi-tasking; executing on a real-time operating system
  - Interest of OHB to qualify it only to a medium level criticality for cost reasons
  - E.g. ECSS software criticality category “C”
- A second software module (“Software Interlock”) monitoring the application software and performing interlock functionality
  - Software Interlock: small, simple, bare-metal (i.e. no OS)
  - Be a compensating provision to the main application software
  - Has to be qualified one software criticality category higher than the ASW (i.e. SCC-B)
- Overall: more flexibility, and significantly lower V&V project effort assessed than with the complete application software being qualified as SCC-B

Software Interlocks

- Requirement:
  The interlock software shall not just be completely independent of the application software, but shall also be able to
  - block the application software from performing off-nominally
  - take over a kind of simple control then



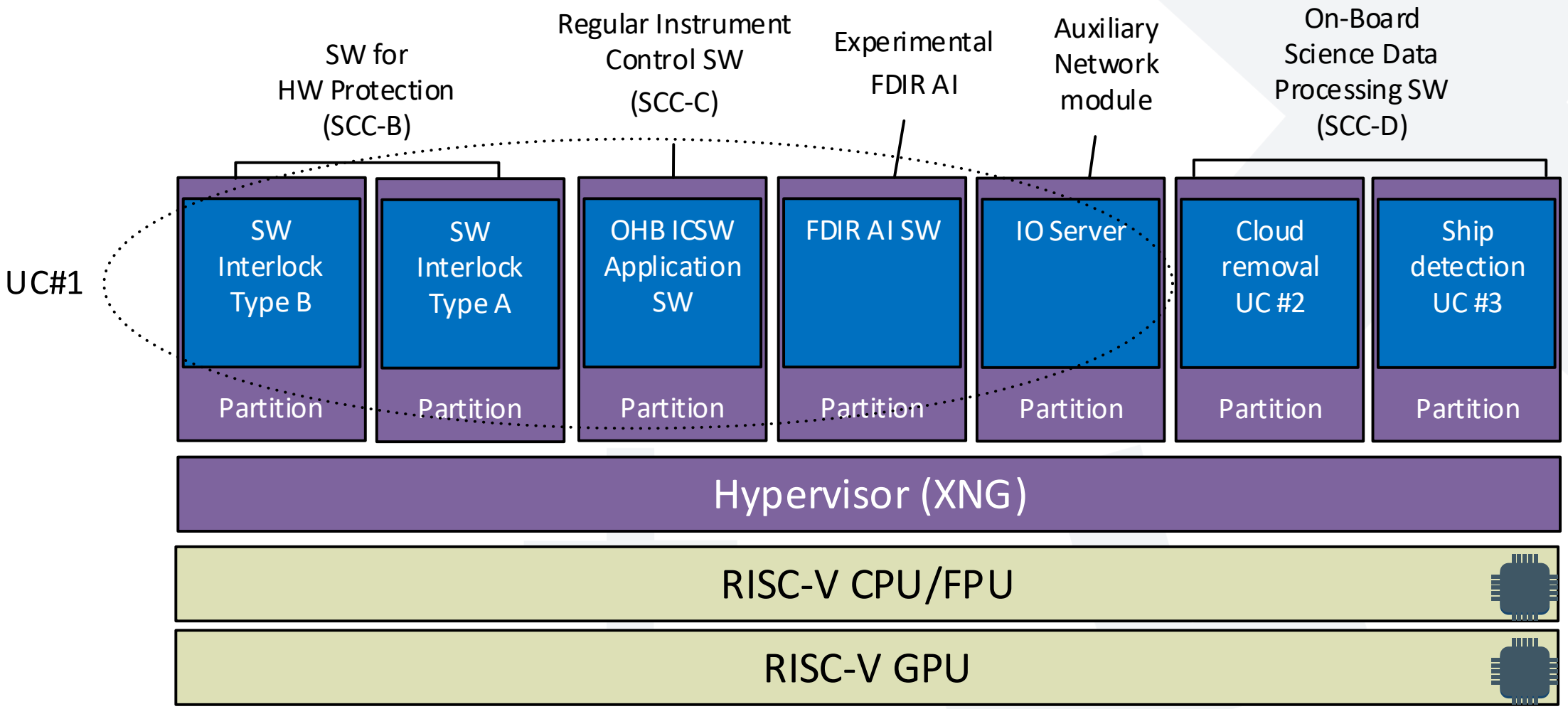
Hypervisor and Software Interlocks

- One solution -> utilization of a hypervisor software:
  keeps different application-level software parts *time* and *space* and *resource* separated in so called “partitions”
- Hypervisor typically qualified for SCC-B
- The Hypervisor guarantees that a regular partition cannot interfere with the other ones (except using well-defined communication channels)

- Due to this separation, and when executing on a powerful multi-core processor board, several totally different on-board software components, having dedicated software criticalities, can be operated in parallel

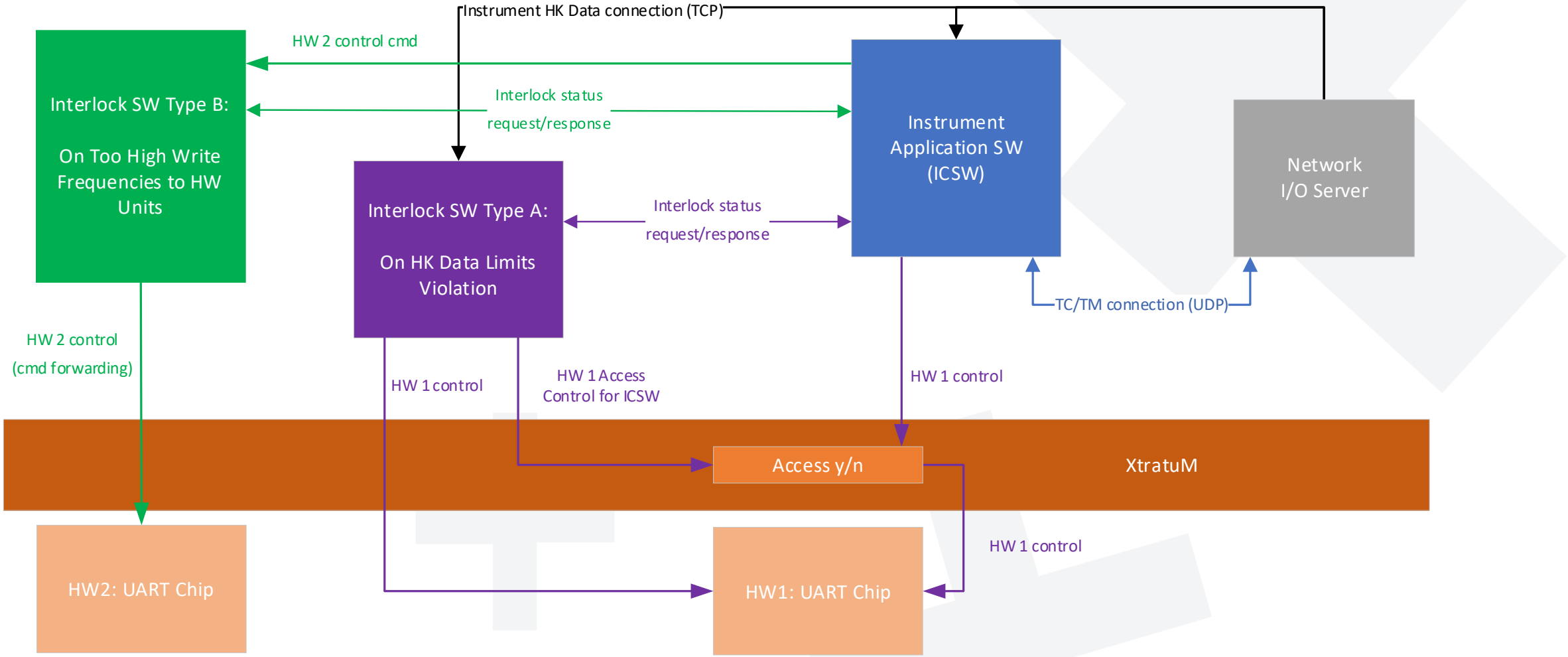
  -> the METASAT use cases

METASAT Use Cases Architecture



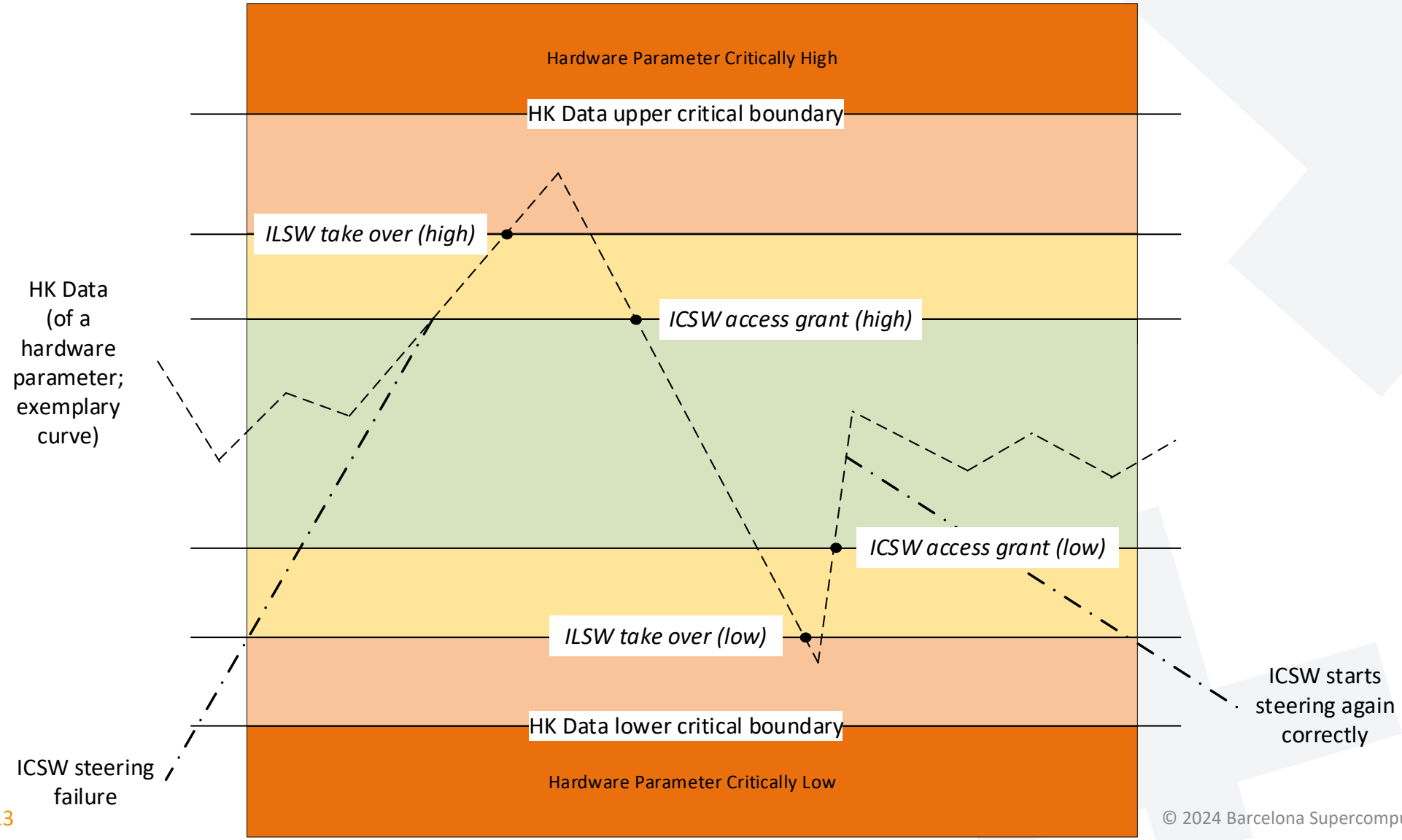
Software Interlocks (Exemplarily in UC#1)

(*patent pending*)



Exemplary for satellite instrument I/O hardware

SW Interlock Type A: Example Behaviour



METASAT Use Case #1 Setup: Validation

Satellite Instrument Environment Emulation (Outside the Computer on the Instrument)

Satellite Instrument Computer

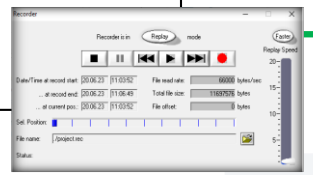
Satellite Platform & Ground Station Emulation regarding TM/TC

Real EnMAP Instrument Housekeeping Data from Orbit (Recorded)

Real EnMAP Instrument Housekeeping Data from Previous Tests on Ground (Recorded)

Simulated EnMAP-like Instrument Housekeeping Data for Extending Real Data

HK Data Replay EGSE



UC#1

SW for HW Protection (SCC-B)

Regular Instrument Control SW (SCC-C)

Experimental FDIR AI

Auxiliary Network module

On-Board Science Data Processing SW (SCC-D)

SW Interlock Type B

SW Interlock Type A

OHB ICSW Application SW

FDIR AI SW

IO Server

Cloud removal UC #2

Ship detection UC #3

Partition

Hypervisor (XNG)

RISC-V CPU/FPU

RISC-V GPU

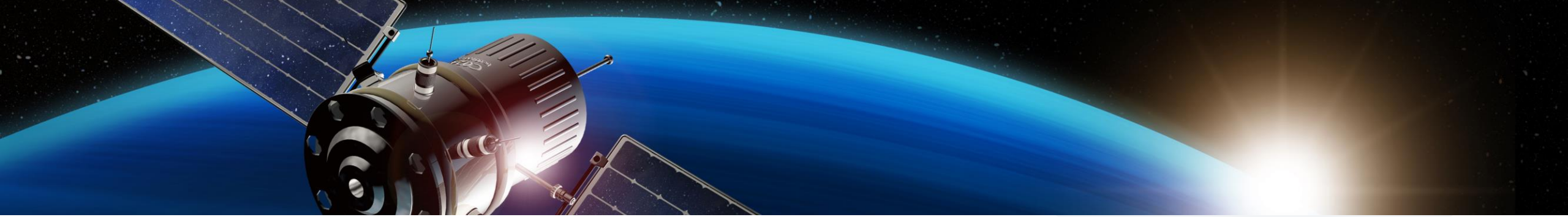
Test EGSE for Telemetry Display / Sending of Telecommands

Colours Legend:

- Satellite Housekeeping Data (as sufficient for the purpose of METASAT)
- Telemetry and Telecommanding for controlling and testing the Use Case
- Inter-partition communication for commanding and status/data request
- Inter-partition communication for IP network

METASAT Use Case #1 Summary

- METASAT system resembling a satellite instrument computer HW+SW
- RISC-V CPU based multi-core HW architecture
- Regular OHB Instrument Control Software (model-based; C++; RTEMS6 QDP), ported to METASAT, and with a TM/TC connection with “ground”
- Two different exemplary types of Software Interlocks realized
- In addition: experimental FDIR AI, and an I/O Server
- All software executing on a COTS Hypervisor
- Setup being demonstrated by means of replaying and on-board processing recorded HK data from the EnMAP satellite instrument in space (provided by courtesy of the DLR German Space Agency)
- The separation of the software in different partitions and the use of Software Interlocks is assessed to reduce the overall V&V effort



https://metasat-project.eu/
info@metasat-project.eu

https://twitter.com/MetasatProject

https://www.linkedin.com/company/metasat-project

METASAT has received funding from the European Union's Horizon Europe programme under grant agreement number 101082622.