Outline

- Introduction
- Common DPU & BSW activity
- The common DPU
- The common Basic Software
- Lessons learned and conclusions
Introduction
Science Objectives

- Exploration of the Jupiter system
  - Jovian atmosphere
  - Jovian magnetosphere
  - Jovian satellite and ring systems
- Exploration of habitable worlds
  - Ganymede as a planetary object and possible habitat
  - Europa’s recently active zones
  - Callisto as a remnant of the early Jovian system

Instrument suite

- **JANUS**: moons geology, cloud morphology and dynamics
- **MAJIS**: chemistry, atmospheric and surface composition
- **UVS**: atmosphere of moons and aurora of Jupiter
- **SWI**: Jupiter wind, moons atmospheric temperature and composition
- **GALA**: moons shape and topography
- **RIME**: moons surface study
- **3GM**: gravity field and moon interiors
- **J-MAG**: magnetic field
- **PEP**: plasma environment and study of neutral and ion composition of exospheres
- **RPWI**: plasma environment
Spacecraft and mission characteristics

**Spacecraft**
- Mass
  - Dry $\approx 2200$ kg
  - Propellant $\approx 2900$ kg
  - Total $> 5000$ kg
  - Instruments $= 260$ kg
- Power
  - Solar Array $\approx 725$ W EOL
  - Instruments GCO500 $= 180$ W
  - Instruments fly-by $= 230$ W ($360$ W for $\frac{1}{2}$ h)
- Memory $= 1$ Tbit EOL
- $\Delta v \approx 2400$ m/s
- Downlink data rate: $1.4$ Gb/24 h

**Mission**
- Launch date May 2022
- 5 gravity assists (Earth, Venus, Earth, Mars, Earth)
- Cruise phase $\approx 7.5$ years
- Science phase $\approx 3.5$ years
CDMU & Instruments

**CDMU:**
- OBC: processor, TM/TC, mass memory (and reconfiguration)
- SSMM: SpW router, science mass memory

**Instruments (SpW):**
- GALA, JANUS, MAJIS, RPWI, UVS, J-MAG, RIME, PEP-Lo, PEP-Hi, SWI and HAA (3GM)

**Instruments (Mil1553):**
- Kat (3GM)
Common DPU & BSW activity
Instruments at B1 phase

- **10 Instruments** developed by different institutes (PIs) at Instrument Consolidation Review (ICR)
  → **10 different designs** for the Digital Processing Unit (DPU).

Similar scenario observed also in Bepi-Colombo and Solar Orbiter where a similar suite of instruments is present, in particular:
- Only partial reuse of **Boot SW** or other SW libraries from previous missions
- **very different level of quality** of SW data packages between instruments
- In some cases, RTEMs OS included in **Boot SW** (mission critical, category B)!
- **Delays** and **cost overruns** also due to DPU and SW development/validation

### JUICE instr. status at ICR

<table>
<thead>
<tr>
<th>Instr</th>
<th>CPU</th>
<th>MIPS</th>
<th>Memory</th>
<th>SpW</th>
</tr>
</thead>
<tbody>
<tr>
<td>GALA</td>
<td>UT699</td>
<td>20</td>
<td>- 512 kB EEPROM - 128 MB SDRAM</td>
<td>100</td>
</tr>
<tr>
<td>JANUS</td>
<td>GR712RC</td>
<td>&lt;80</td>
<td>- 32 MB Flash - 256 MB SDRAM</td>
<td>-</td>
</tr>
<tr>
<td>JMAG</td>
<td>LEON3/RTAX</td>
<td>20</td>
<td>- EEPROM - SRAM</td>
<td>10</td>
</tr>
<tr>
<td>MAJIS</td>
<td>AT7913E</td>
<td>&lt;45</td>
<td>- EEPROM - SRAM</td>
<td>10 Rx, 100 Tx</td>
</tr>
<tr>
<td>PEP</td>
<td>OR1200</td>
<td>20</td>
<td>- 8 MB Flash - 16 MB SRAM</td>
<td>10</td>
</tr>
<tr>
<td>RIME</td>
<td>LEON3/RTAX4000</td>
<td>~20</td>
<td>- EEPROM - SRAM</td>
<td>10</td>
</tr>
<tr>
<td>RPWI</td>
<td>UT699</td>
<td>&lt;60</td>
<td>- 2 MB MRAM - 32 MB SRAM</td>
<td>-</td>
</tr>
<tr>
<td>SWI</td>
<td>Custom RTAX</td>
<td>~20</td>
<td>- 512 kB SRAM</td>
<td>-</td>
</tr>
<tr>
<td>UVS</td>
<td>8051</td>
<td>&lt;5</td>
<td>- 128 kB EEPROM - 128 kB SRAM</td>
<td>10 Rx, 40 Tx</td>
</tr>
</tbody>
</table>
How to improve?

- **Focus on the commonalities:**
  - Interface to the OBC/SSMM (i.e. SpW)
  - Redundancy scheme (i.e. 2 cold redundant DPUs, 2 SpW links)
  - Processor (i.e. LEON 2 or 3) and memories
  - EEE parts
  - Basic SW libraries (i.e. Boot SW, RTOS, Driver SW, PUS library)
  - SW Validation Facility

- **Flexibility & configurability**
  - CPU performance and memory size
  - Form factor and mechanical interface
  - Instrument-specific functions and interfaces

- **Constraints**
  - Power consumption
  - Radiation tolerance
  - Generic vs. specific
How the problem was addressed in JUICE

- **EXPRO** contract (ref. 4000113396/15/NL/BW)
  
  Common development of:
  
  - **DPU processor section design (CPU, SpW, memories):** schematics, EEE parts list, analyses, verification tests, HW prototype
  
  - **Basic SW (Boot, Standby & Drivers):** source code, unit/integration/validation tests, ECSS documentation

- **PI** still responsible for:
  
  - procurement of EEE parts
  
  - instrument-specific DPU functions
  
  - instrument DPU manufacturing and test
  
  - Boot & Standby SW validation in context
<table>
<thead>
<tr>
<th>Document Id</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOOT-SW-RS</td>
<td>DPU Boot SW Requirements Specification</td>
</tr>
<tr>
<td>BOOT-SW-ADD</td>
<td>DPU Boot SW Architectural Design Document</td>
</tr>
<tr>
<td>BOOT-SW-ICD</td>
<td>DPU Boot SW Interface Control Document</td>
</tr>
<tr>
<td>BOOT-UT-PLAN</td>
<td>DPU Boot SW Unit Test Plan</td>
</tr>
<tr>
<td>BOOT-VT-PLAN</td>
<td>DPU Boot SW Validation Test Plan and Specification</td>
</tr>
<tr>
<td>BOOT-SW-VCD</td>
<td>DPU Boot SW Verification Control Document</td>
</tr>
<tr>
<td>BOOT-SW-BUD</td>
<td>DPU Boot SW Budget Document</td>
</tr>
<tr>
<td>HDSW-SW-RS</td>
<td>DPU HW Driver SW Requirements Specification</td>
</tr>
<tr>
<td>HDSW-SW-ADD</td>
<td>DPU HW Driver SW Architectural Design Document</td>
</tr>
<tr>
<td>HDSW-SW-ICD</td>
<td>DPU HW Driver SW Interface Control Document</td>
</tr>
<tr>
<td>HDSW-UT-PLAN</td>
<td>DPU HW Driver SW Unit Test Plan</td>
</tr>
<tr>
<td>HDSW-VT-PLAN</td>
<td>DPU HW Driver SW Validation Test Plan and Specification</td>
</tr>
<tr>
<td>HDSW-SW-VCD</td>
<td>DPU HW Driver SW Verification Control Document</td>
</tr>
<tr>
<td>HDSW-SW-BUD</td>
<td>DPU HW Driver SW Budget Document</td>
</tr>
<tr>
<td>PSEC-HW-RS</td>
<td>DPU Processor Section Requirements Specification</td>
</tr>
<tr>
<td>PSEC-HW-ADD</td>
<td>DPU Processor Section Architectural Design Document</td>
</tr>
<tr>
<td>PSEC-HW-ICD</td>
<td>DPU Processor Section HW/SW Interface Control Document</td>
</tr>
<tr>
<td>PSEC-HW-LIST</td>
<td>DPU Processor Section EEE Parts List</td>
</tr>
<tr>
<td>PSEC-VV-PLAN</td>
<td>DPU Processor Section Verification Test Plan</td>
</tr>
<tr>
<td>PSEC-HW-VCD</td>
<td>DPU Processor Section Verification Control Document</td>
</tr>
<tr>
<td>PSEC-HW-PSA</td>
<td>DPU Processor Section Parts Stress Analysis</td>
</tr>
<tr>
<td>PSEC-HW-RAD</td>
<td>DPU Processor Section Radiation Analysis</td>
</tr>
<tr>
<td>PSEC-HW-WCA</td>
<td>DPU Processor Section Worst Case Analysis</td>
</tr>
<tr>
<td>PSEC-HW-FMECA</td>
<td>DPU Processor Section FMECA</td>
</tr>
</tbody>
</table>
Deliverables 2/2

<table>
<thead>
<tr>
<th>Document Id</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOOT-SW-DD</td>
<td>DPU Boot SW Design Document</td>
</tr>
<tr>
<td>BOOT-UT-REP</td>
<td>DPU Boot SW Unit Test Report</td>
</tr>
<tr>
<td>BOOT-SW-CF</td>
<td>DPU Boot SW Configuration File</td>
</tr>
<tr>
<td>BOOT-SW-UM</td>
<td>DPU Boot SW User's Manual</td>
</tr>
<tr>
<td>BOOT-VT-REP</td>
<td>DPU Boot SW Validation Test Report</td>
</tr>
<tr>
<td>HDSW-SW-DD</td>
<td>DPU HW Driver SW Design Document</td>
</tr>
<tr>
<td>HDSW-UT-REP</td>
<td>DPU HW Driver SW Unit Test Report</td>
</tr>
<tr>
<td>HDSW-SW-CF</td>
<td>DPU HW Driver SW Configuration File</td>
</tr>
<tr>
<td>HDSW-SW-UM</td>
<td>DPU HW Driver SW User's Manual</td>
</tr>
<tr>
<td>HDSW-VT-REP</td>
<td>DPU HW Driver SW Validation Test Report</td>
</tr>
<tr>
<td>PSEC-HW-DD</td>
<td>DPU Processor Section Design Document</td>
</tr>
<tr>
<td>PSEC-HW-SCH</td>
<td>DPU Processor Section Schematics</td>
</tr>
<tr>
<td>PSEC-HW-CF</td>
<td>DPU Processor Section Configuration File</td>
</tr>
<tr>
<td>PSEC-HW-UM</td>
<td>DPU Processor Section User's Manual</td>
</tr>
<tr>
<td>PSEC-VV-REP</td>
<td>DPU Processor Section Verification Test Report</td>
</tr>
<tr>
<td>DPU-FP</td>
<td>Final Presentation/Final Report</td>
</tr>
</tbody>
</table>

### ESA complementary documents:
- JUI-EST-SYS-MX-001 - BOOTSW-SGICD compliance
- JUI-EST-SYS-RP-027 - AUTOMATIC ASW START
- JUI-EST-SYS-RP-030 - PUS6 ADDRESSING
- JUI-EST-SYS-RP-031 - ACK FLAGS
- JUI-EST-SYS-RP-033 - ANOMALY IN SpW TDP
- JUI-EST-SYS-RP-034 - PUS 6 MEMORY ID 2
- BOOT-SW-RFD-001, 002, 003

### ISVV reports:
- JUI-ISVV-DPU-RDVR - DPU Req. and Design Verification Report
- JUI-ISVV-DPU-CVR - DPU Code Verification Report
- JUI-ISVV-DPU-TVR - DPU Test Evaluation Report
- JUI-ISVV-DPU-V300VR - DPU Boot Software V3.0.0 Verification Report

<table>
<thead>
<tr>
<th>Item identifier</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOOT-SRC-SW</td>
<td>DPU Boot SW Source Code (including make/link files and any other files needed to build the SW) and Executable</td>
</tr>
<tr>
<td>BOOT-UT-SW</td>
<td>DPU Boot SW Unit Test Scripts/SW Source Code</td>
</tr>
<tr>
<td>HDSW-SRC-SW</td>
<td>DPU HW Driver SW Source Code (including make/link files and any other files needed to build the SW) and Binary Library</td>
</tr>
<tr>
<td>HDSW-UT-SW</td>
<td>DPU HW Driver SW Unit Test Scripts/SW Source Code</td>
</tr>
<tr>
<td>BOOT-VT-SW</td>
<td>DPU Boot SW Validation Test Scripts/SW Source Code</td>
</tr>
<tr>
<td>HDSW-VT-SW</td>
<td>DPU HW Driver SW Validation Test Scripts/SW Source Code</td>
</tr>
<tr>
<td>PSEC-HW-HW</td>
<td>DPU Processor Section Prototype Hardware</td>
</tr>
</tbody>
</table>
The common DPU
### CPU
- GR712RC dual LEON3FT
- Clock freq. 20..100MHz

### Memory
- PROM: 32KiB
- MRAM: 2 or 8 MiB
- SRAM: 4, 8, 16 or 32 MiB
- SDRAM: 256 MiB

### SpaceWire
- Redundancy: 1 or 2 links
- Link speed: 40 or 100Mbps

### Instrument specific companion FPGA/ASIC
- Memory mapped I/O
- SpW RMAP
Key EEE components

**Dual LEON3-FT (CPU)**
- GR712RC
- Total dose: 300 kRad (Si)/106 MeV/cm2/mg
- 240-pin CQFP package

**SRAM (working memory)**
- UT8R4M39, UT8R2M39, UT8R1M39
- 100 krad(Si), SEL Immune: <110 MeV-cm2/mg, SEU error rate = 7.3x10-7 errors/bit-day
- 32-bit data, 7-bit BCH EDAC

**SDRAM (working memory)**
- UT8SDMQ64M48 3.0-Gigabit SDRAM
- 100 krad(Si), SEL Immune 111 MeV-cm2/mg, SEU Event Rate: 1.3E-10 events/bit-day
- 64Mx48-bit, allows 16-bit Reed-Solomon EDAC
- Single configuration: 256MiB effective size

**PROM (Boot memory)**
- UT28F256LVQLE - 32KiB PROM
- Total dose: 1Mrad (Si),
  Onset LET: 40 MeV-cm2/mg,
  SEL Immune > 110 MeV-cm2/mg

**MRAM (Application Storage Memory)**
- UT8MR2M8, UT8MR8M8
- Total dose: 1Mrad (Si),
  SEL Immune: 112 MeV-cm2/mg @125C
  SEU Immune: Memory Cell 112 MeV-cm2/mg @25C

**SpW interface (TM/TC interface)**
- Aeroflex UT54LVDS031LV/32LV
- Total-dose: 300 krad(Si) and 1Mrad(Si)
  Latchup immune (LET > 100 MeV-cm2/mg)
- Redundant SpW transceivers for SpW0/1
Performance & size

**SRAM 4 MiB**
- Max clock freq. 100MHz
- Max clock freq. without ws 40MHz
- 50MHz clock freq. with 1 ws

**SRAM 32 MiB**
- Max clock freq. 72MHz
- Max clock freq. without ws 29MHz
- 50MHz clock freq. with 2 ws

**SDRAM 256 MiB**
- Max clock freq. 44MHz

Data obtained from Hyperlinx simulations on prototype PCB → worst case, *conservative*
The common Basic Software
A short introduction to each part of the Boot SW follows.

2.3.1 Processor Module Initialisation
The processor module initialisation sequence is triggered by reset condition. It is responsible for initialising and self-testing DPU functions, writing the boot report and to setup a C runtime environment.

Processor module initialisation has a linear execution path, meaning that no branch decisions which depend on system external factors are made. Initialisation sequence is directly followed by the standby mode.

2.3.2 Standby Mode
Standby mode software communicates with ground relayed via the OBC using the PUS protocol, [ECSS-E7041] over an inter-processor communication link (SpaceWire, link 0). In this mode, Boot SW is a PUS packet terminal and any actions taken by the standby mode are caused by PUS service commands. In addition to this, housekeeping reports are continuously sent to the OBC to indicate DPU status, and local time is maintained using a SpaceWire time synchronization protocol [SPW-TIME].

The standby mode also features uploading of new application software to the DPU for execution. This allows for updating or patching DPU software in flight. Standby mode is always activated by Boot SW, with the option to automatically exit and resume with ASW loading on a timeout condition.

2.3.3 ASW load
The ASW load part is responsible for validating, loading and executing an ASW image residing in application storage memory (ASM) or RAM. ASW load will eventually end up with ASW execution.

- SUN SPARC and Linux kernel coding style conventions
- GNU Make automatic build
- Statically allocated RAM
- No OS or externally developed SW
- Drivers SW common to the HW Drivers SW library
- Minimal CPU resource utilisation
  - [no FPU, interrupts, IU mul/div]
- GR712RC internal device addresses hardcoded
  - [no AMBA PnP]
- Unused GR712RC resources are disabled
  - [clock gated]
- Instrument-specific parameters configurable at compile-time
Operating System Abstraction Layer (OSAL)
Low level and callback functions

Interfaces supported:
- SPI
- UARTs
- SpaceWire
- Timers
- AHB status
- Watchdog
- Clock gating
- Memory controller

RTEMS 4.10 used for the HW drivers SW validation
Basic Software validation

**SW Unit Tests**
- GRMON2 and TSIM2
- Instruction code decision coverage on object files from TSIM2
- Unit test each component separately
- Fully automated test suite
- Code coverage also available with GRMON2

→ 98% code coverage on unit-tested code
→ 15-20% of the code tested by validation instead

**SW Validation Tests**
- GRMON2 and GRESB
- TCL scripting and GRMON2 scripting
- Unit test each component separately
- Fully automated test suite
- Test suite also ported to S/C Interface Simulator (SIS)

→ Black box tests checking TM
→ White box tests checking internal registers or traces with GRMON2
Lessons learned and conclusions
Results

- Despite the initial skepticism, **7/8 out of 12 JUICE units** use the provided DPU design and/or Basic SW
- **Test suite** developed by ESA/Airbus for the DPU/Boot SW is used as basis for the instruments **Full Functional Tests (FFT)**
- **Uniform, efficient and effective support** to the instruments thanks to the common design and SW
- **Uniform OBC-instrument TM/TC interface** for basic PUS services
- Resources in the **PI teams focused** on SW **science and performance** requirements
- Instruments **EM deliveries** so far have only **minor delays**
- SW bugs discovered by a PI team are reported to all the teams and solution distributed to all at once using a shared GITLAB environment
- **Tools and SW test suites** developed for one instrument can be **shared with other teams** with small adaptation effort

<table>
<thead>
<tr>
<th>Instr</th>
<th>DPU</th>
<th>Boot SW</th>
</tr>
</thead>
<tbody>
<tr>
<td>GALA</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>JANUS</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>JMAG</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>MAJIS</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>PEP-Lo</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>PEP-Hi</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>RIME</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>RPWI</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>SWI</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>UVS</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>RADEM</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>HAA</td>
<td>NO</td>
<td>NO</td>
</tr>
</tbody>
</table>

- In comparison, PEP-Lo instrument **not** using the common DPU design is facing important cost and schedule overruns
Conclusions

• While in JUICE the common DPU and Basic SW were introduced after instruments selection (end of phase B1) and only recommended to the PIs for adoption, **future missions with multiple instruments should require the use of common DPU and Basic SW** since the early study phase.

• System engineering early effort can prevent the use of custom FPGA in favor of GR712RC built-in interfaces

• Developing HW and SW “products” in advance is key to safeguard instruments schedule

• Other instruments elements with commonalities could be studied for a similar common development, e.g.:
  • Power Supply Unit
  • ASICs for recurrent functions
  • Common IP cores for FPGAs
  • Common SW libraries