Technische Universität Braunschweig

INSTITUTE OF COMPUTER AND NETWORK ENGINEERING

From Rosetta to Current Developments Using FPGAs for Scientific Space Missions

Tobias Lange, Björn Fiethe, Holger Michel, Harald Michalik

SEFUW | September 2014

Outline

- FPGA Usage on the ROSINA Data Processing Unit
- The PHI Instrument on Solar Orbiter
- Heritage of SRAM-based FPGAs
- Data Processing Flow of the PHI DPU
- Architecture of the PHI DPU
- FPGA Failure Tolerance
- Package Assembly Qualification
- Controlling Concurrent Change
- Conclusion

In the Past: FPGA Usage on the ROSINA DPU

- Data Processing Unit (DPU) of the Rosetta Orbiter Spectrometer for Ion and Neutral Analysis (ROSINA) instrument
  - Developed in 1998, launched in 2004
- Data acquired from two mass spectrometers & one pressure sensor to analyze the composition of cometary neutral gases and ions:
  - Allocated data rate of **2 kbit/s** (lossless compressed)
- DPU includes:
  - 32-bit processor system, SRAM memory
  - 6x Actel RT14100A devices (w/o red.)
- The 6 FPGAs provide a total capacity of about 8000 logic modules with **fixed** functionality

Why Adaptable Processing Platform is Needed? − Present

Currently operational space computing platforms:

- **Weak computing performance**
- **Low functional integration** due to resource limitation
- **Limited ability** for changes and adaptation of system functionality
- *Example − payload data handling:*
  - ➢ *relying on simple data compression (lossless)*
  - ➢ *performed by dedicated devices (e.g. ASIC)*

➢**Increasing demands** on data rate and flexibility at limited resources (power consumption, mass, volume, cost)

- *Example − final physical quantities have to be extracted already on board the spacecraft*

Why Adaptable Processing Platform is Needed? − Future

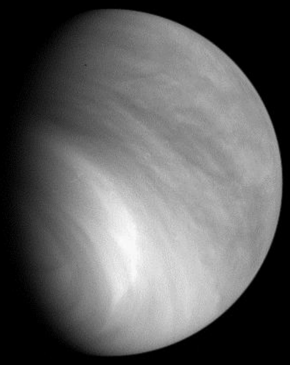
Space computing platforms:

- Fixed, ground verified configurations → **reconfigurable systems**
- Powerful on-board processing under challenging constraints of space missions

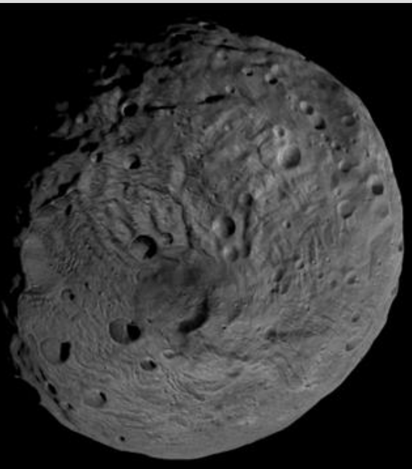
- **SW and HW adaptation:**
  - **In-flight** with high grade of system reliability, availability and safety
  - Even **dynamically** during run-time
  - ➢ Maximize use of resource limited HW platforms
  - ➢ **SRAM based FPGAs** are well suited

- *Example − final physical quantities are already extracted on board by an autonomous and reliable application, adapting itself to changing needs*

Heritage of SRAM-based FPGAs in Scientific Instruments



Venus Monitoring Camera (VMC)

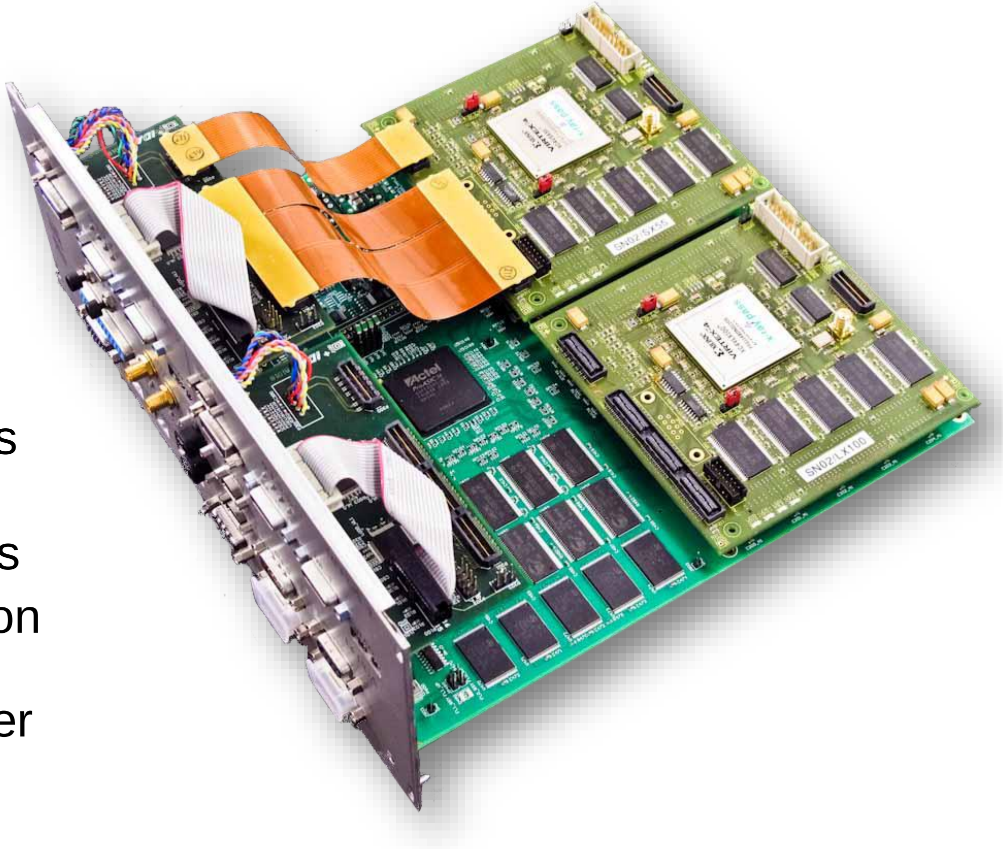


Dawn Framing Camera (DawnFC)

- Based on Xilinx SRAM FPGAs
- First European scientific System-on-Chip computer in Space
- VMC operational since 8 years with only a few predicted reboots due to SEE
  - ➢Availability 0.999995
- Proven solution for scientific space instruments if appropriate mitigation techniques against radiation effects are applied
- **But: Dynamic in-flight reconfiguration was not used yet**

Dynamically Reconfigurable Processing Module (DRPM)

- **Demonstrator** for Dynamic Reconfigurable Processing Module (DRPM) developed together with Astrium Ltd. under ESA contract
- Demonstrating feasibility of reconfigurable FPGA technology for flight programs
- **Modular concept:**
  - ✓Processing capacity scalable
  - ✓Redundancy by addition of modules
  - ✓Based on devices and I/F also available in space qualified versions
- SpaceWire based NoC communication architecture **SoCWire** to isolate modules physically and logically under high real-time, availability and dependability constraints



Solar Orbiter PHI Instrument Data Processing Unit 1

- PHI will probe the solar interior and provide maps of the solar photosphere

- One Active Pixel Sensor (APS)
  - ➢ 2048×2048 pixels

Constraints:

- Limited telemetry rate: **3.2 Gbit → 100 Mbit**
  - ➢ Classical ground processing steps need to be performed already on board
- **40 GFlops** for on-board processing
- Available space proven GPP: 10..100 MFlops
  - ➢ Only dedicated HW feasible (ASIC, FPGA)
  - ➢ But: Algorithms need to be adaptable during long-term mission

Source: ESA

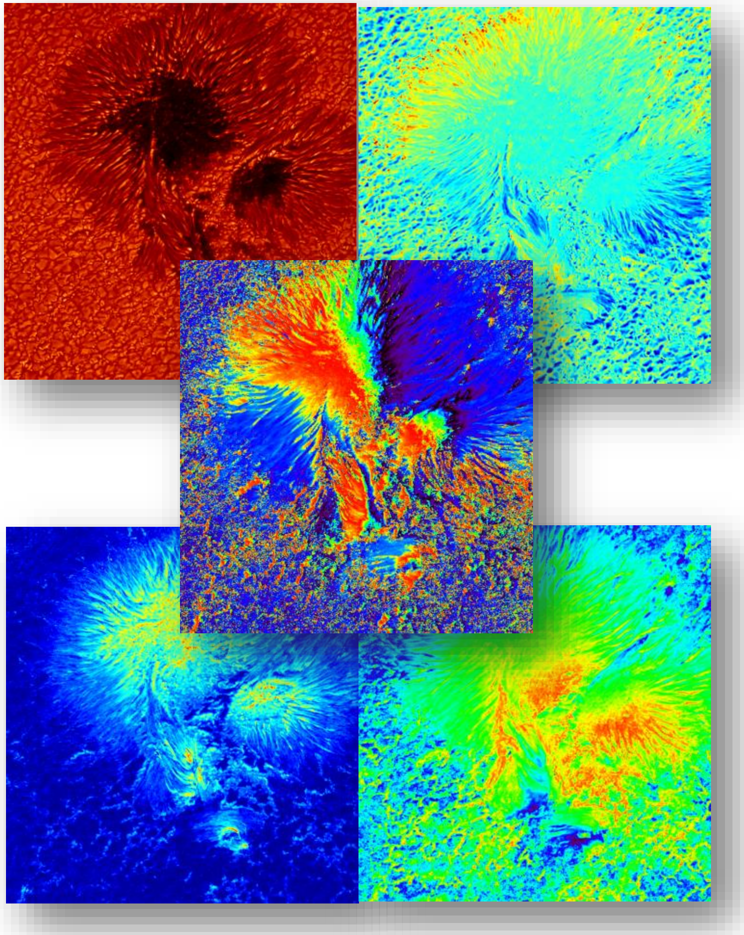
Max-Planck-Institut für Sonnensystemforschung

Solar Orbiter PHI Instrument Data Processing Unit 2

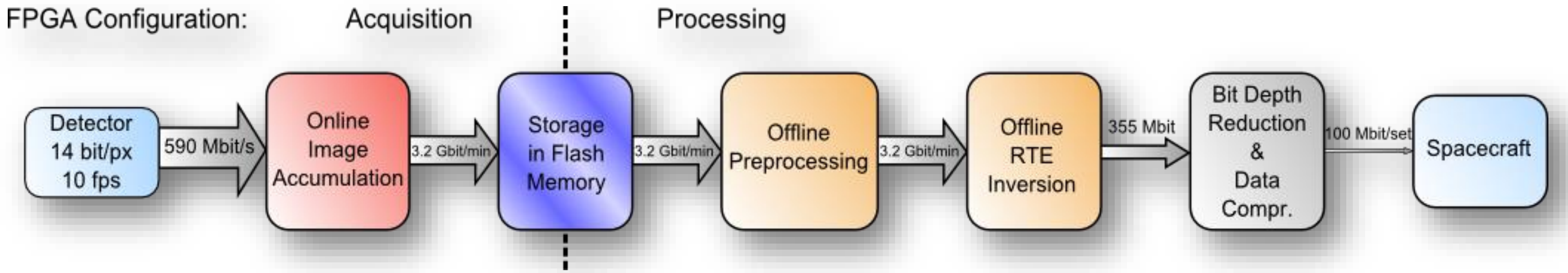
Solution:

- In-flight adaptive system using FPGAs
- Extraction of scientific parameters by inversion of the Radiative Transfer Equation (RTE)
- Reconfiguration of HW and SW allows multi-functional use of available resources
- Acquired data will be stored in memory until FPGAs are reconfigured
- Enhances system with adaptive functionality on demand and operational flexibility

➢ **Architecture based on DRPM**



Data Flow of the PHI Data Processing Unit



FPGA Configuration: Acquisition | Processing

Detector 14 bit/px 10 fps → 590 Mbit/s → Online Image Accumulation → 3.2 Gbit/min → Storage in Flash Memory → 3.2 Gbit/min → Offline Preprocessing → 3.2 Gbit/min → Offline RTE Inversion → 355 Mbit → Bit Depth Reduction & Data Compr. → 100 Mbit/set → Spacecraft

~9 hours

~30 days

Four processing tasks can be mapped on two FPGAs and two modes of operation:

- **Data acquisition mode:**
  - Image Stabilization System (ISS)
  - Capturing and consecutive accumulation of image data

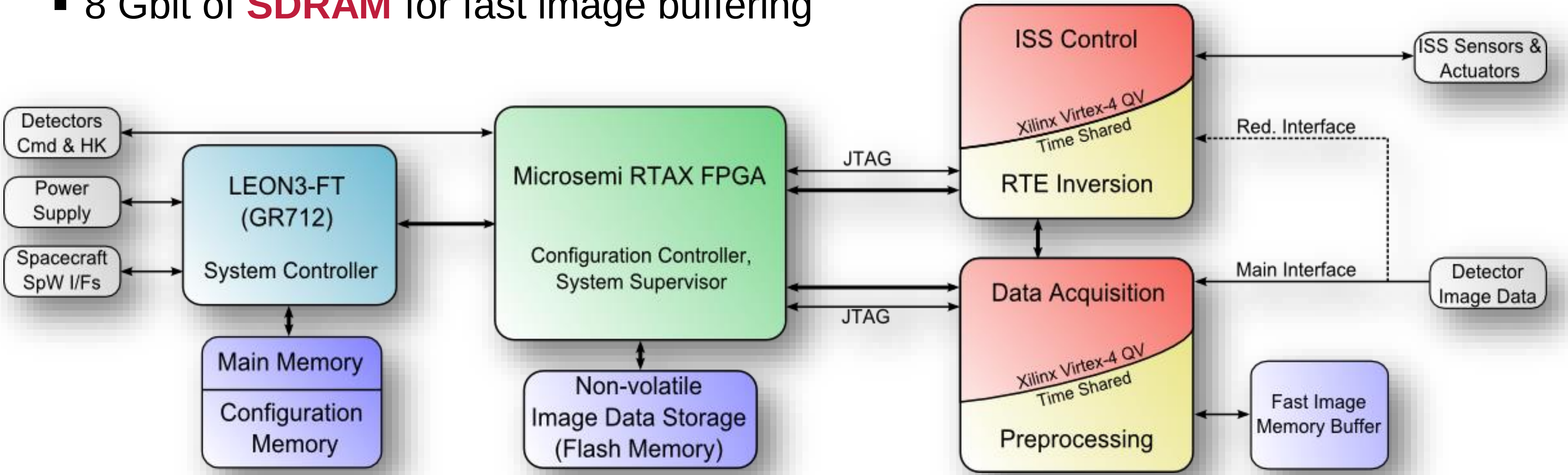
Save acquired data to NAND-Flash

- **Processing mode:**
  - Data preprocessing
  - RTE inversion

Recall stored data from NAND-Flash

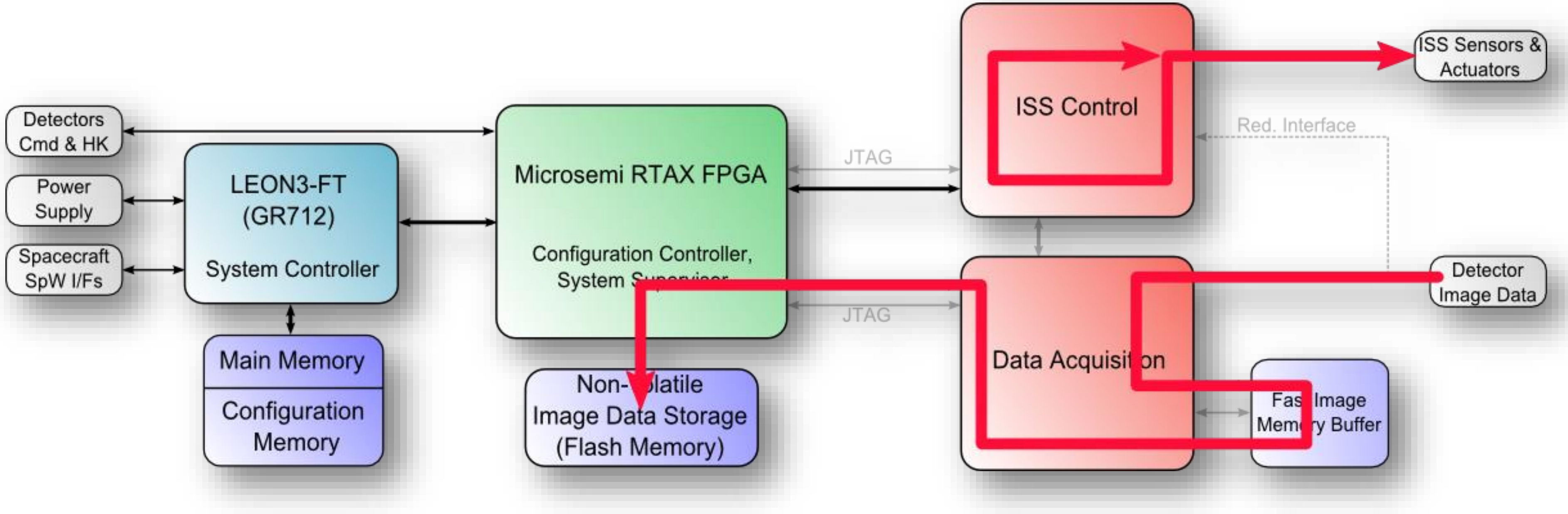
Architecture of the PHI DPU 1

- Aeroflex Gaisler **GR712** dual-core LEON3-FT processor
  - 2 Gbit **SDRAM** with EDAC
  - 1 Gbit **NOR-Flash** for software and FPGA configurations
- TMR by design, rad-hard Microsemi **RTAX2000** FPGA as system supervisor
- 4 Tbit of non-volatile **NAND-Flash** memory
- **Two** SRAM-based, reconfigurable Xilinx **Virtex-4 QV** FPGAs
- 8 Gbit of **SDRAM** for fast image buffering



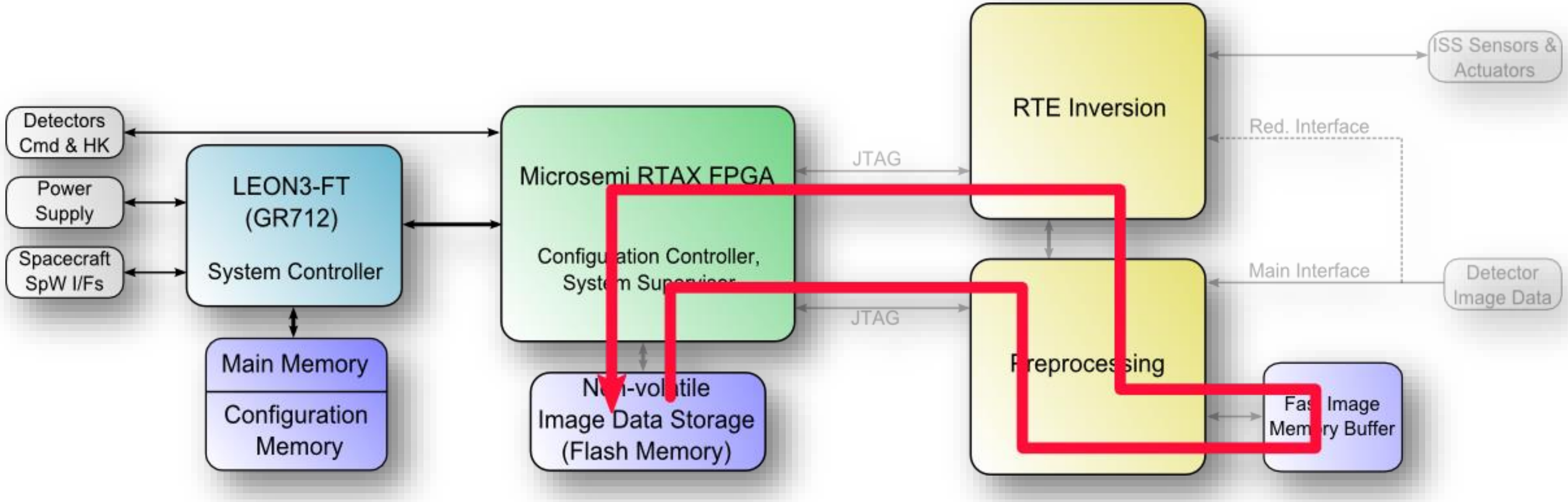
Architecture of the PHI DPU 2

Configuration 1: Data Acquisition



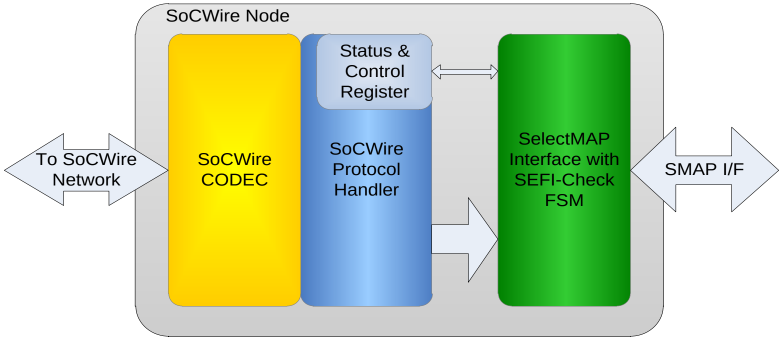
Architecture of the PHI DPU 3

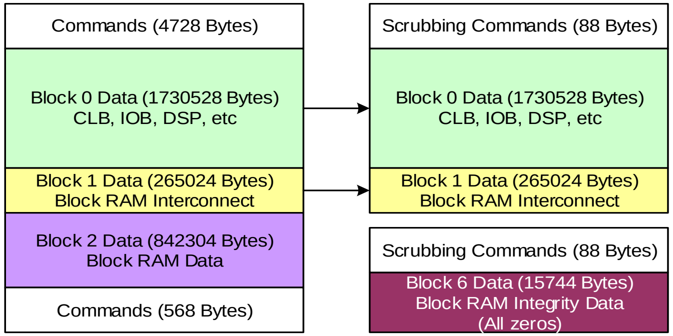
Configuration 2: Processing



Single Event Upset (SEU) Mitigation and Scrubbing

- Scrubbing by external configuration controller:
  - ➢ Provides much higher reliability than Xilinx internal ICAP interface (prone to SEU errors)





Commands (4728 Bytes)

Block 0 Data (1730528 Bytes)
CLB, IOB, DSP, etc

Block 1 Data (265024 Bytes)
Block RAM Interconnect

Block 2 Data (842304 Bytes)
Block RAM Data

Commands (568 Bytes)

Scrubbing Commands (88 Bytes)

Block 0 Data (1730528 Bytes)
CLB, IOB, DSP, etc

Block 1 Data (265024 Bytes)
Block RAM Interconnect

Scrubbing Commands (88 Bytes)

Block 6 Data (15744 Bytes)
Block RAM Integrity Data
(All zeros)

- Internal network: Node for SelectMAP interface with functional check available

- Configuration data file needs to be adapted for scrubbing

- **DRPM study** uses SelectMAP interface for configuration scrubbing:

➢ **Advantages**

+ Frame based blind scrubbing
+ Partial reconfiguration
+ Merging of partial bitstreams
+ Fault injection
+ Detection of functional errors in SelectMap interface

➢ **Disadvantages**

− Use of the same network for configuration and payload data
− Requires 12 I/Os per configuration interface
− Blind scrubbing may cause internal contention in case of rare errors during the writing process
− SEUs are overwritten and can not be detected

Simple Read Back Scrubbing via JTAG

- PHI DPU on Solar Orbiter will use read back scrubbing via **JTAG**:
  1. **Read back** single or full configuration frames
  2. **Compare** the read back data to a **golden copy**
  3. **Only affected** configuration frames are scrubbed

➢ **Advantages:**

+ Only 4 I/Os per interface
+ Fault injection can also be performed
+ Setting `GLUTMASK` option to avoid corruption of SRL16 and distributed RAM
  - ➢ In turn SEUs in SRL16 and distributed RAM have to be mitigated in the user design
+ I/O checking via boundary scan could be added
+ Completely external solution (implemented in radiation hard devices)

➢ **Disadvantages:**

− Mask file required

− Data transfer of complete read back file to processor required (if done by SW)

Optimized Read Back Scrubbing via JTAG

- Simple read back scrubbing **requires complete data transfer** to the processor and **comparison** with regard to the mask file
- Comparison can be done by **dedicated HW** using generated **check sums**

➢No processor support would be required

For each read back frame a checksum could be calculated, however the mask file has to be considered

- Mask file indicates bits that need to be disregarded
- The mask file is design dependent
  ➢ No simple algorithm to calculate it based on frame type or address
- Setting the `GLUTMASK` option results in some masked read back bits being always zero, however this does not apply to all bits

➢Therefore such a process is desirable, but detailed information needed

- However the Xilinx `CAPTURE_VIRTEX4` core seems to accomplish this

Single Event Upset (SEU) Mitigation

- Scrubbing **does not prevent a propagation** of SEU error through the system
  - ➢ **Mitigation** for critical datapath and control modules **needed**

- Typically, no full TMR or error correction for data processing
  - A single pixel error in the image data is not necessarily critical
  - ➢ Goal: **Achieve optimum availability for given constraints**
    - e.g. cost, performance, mass
  - Example: Data processing decoupled by intermediate data storage
    - ➢ No mitigation needed
      1. Regular test with predefined test vectors
      2. Correct FPGA configuration
      3. Re-process again

- Prepared also for TMR on IO level, e.g. for critical memory control signals

General Design Issues and Tools

- ✓ Ability to update FPGA configurations

- **But keep in mind:**
  - ➢ Solar Orbiter will end the nominal mission in 2028

- **Cope with changing staff and technology**
  - ➢ Availability of vendor-specific software / hardware tools
    - ○ e.g. Xilinx ISE, TMR-Tool, EGSE
  - ➢ Design is implemented in plain VHDL code
  - ➢ No additional tools used, e.g. Synplify
  - ➢ Essential:
    - ✓ Clean coding style
    - ✓ Sufficient comments in the source code
    - ✓ Good documentation

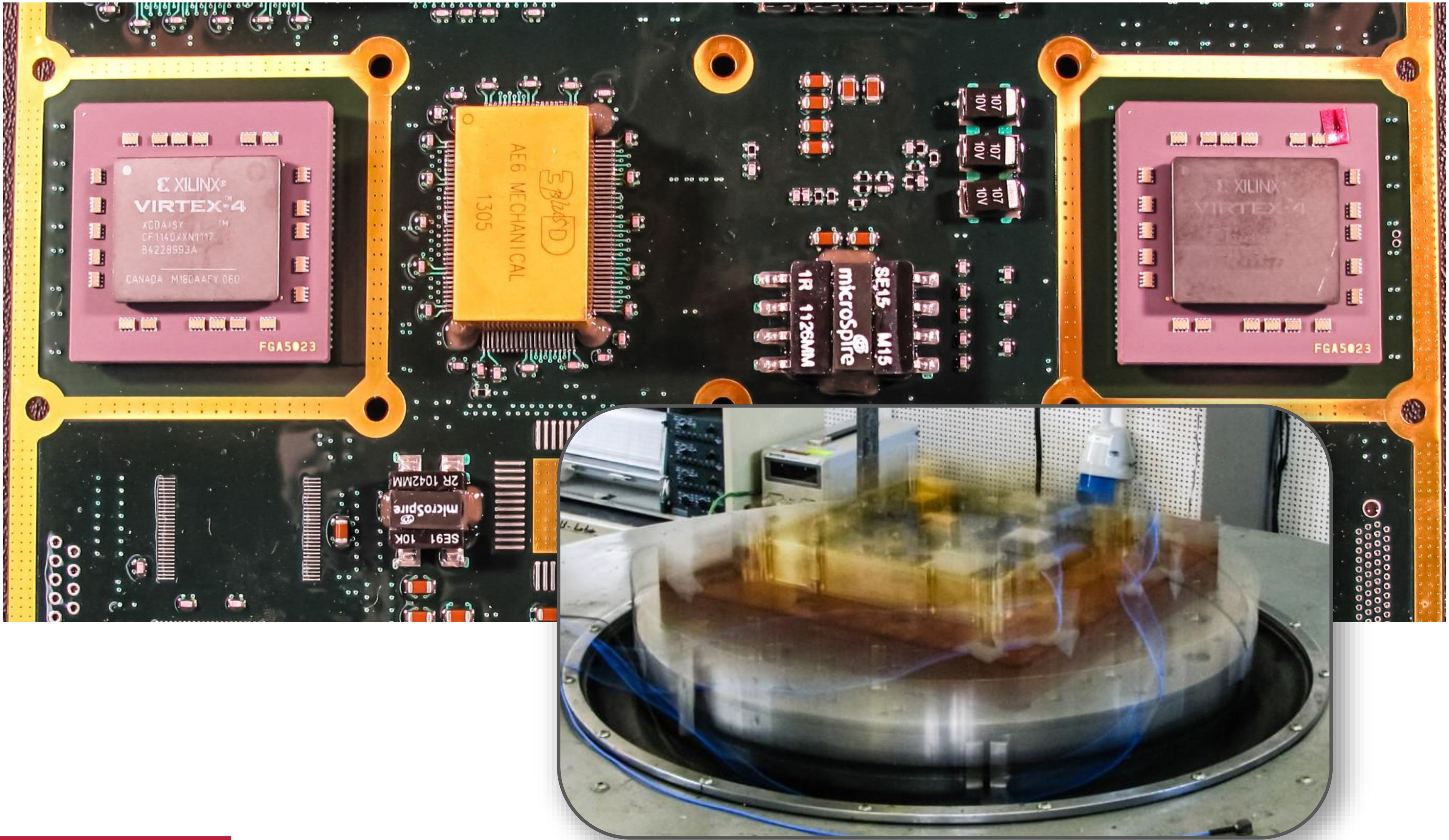
Virtex-4 FPGA CF1140 Package Assembly Qualification 1

No qualified process manufacturer available in Europe (2013)

➢ **Mission specific assembly qualification**

- Based on general requirements according to:
  - ECSS-Q-ST-70-38C
  - TEC-QT/2009/1059/CV
- Manufacturing and assembly of completely representative PCBs and mounting frames
- Daisy-chain packages for probing of all connections
- Detailed qualification plan available
- Thermal cycling test as last step ongoing

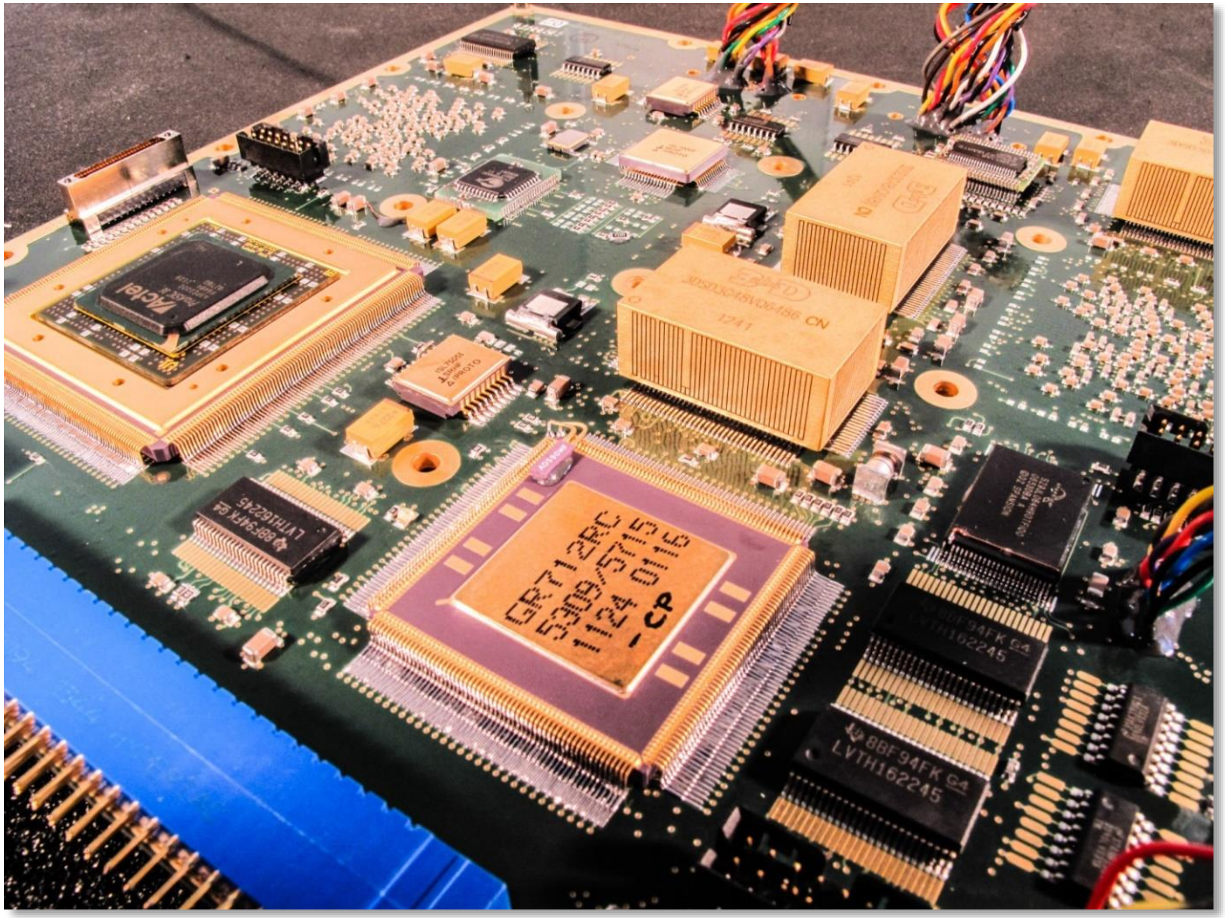
Virtex-4 FPGA CF1140 Package Assembly Qualification 2



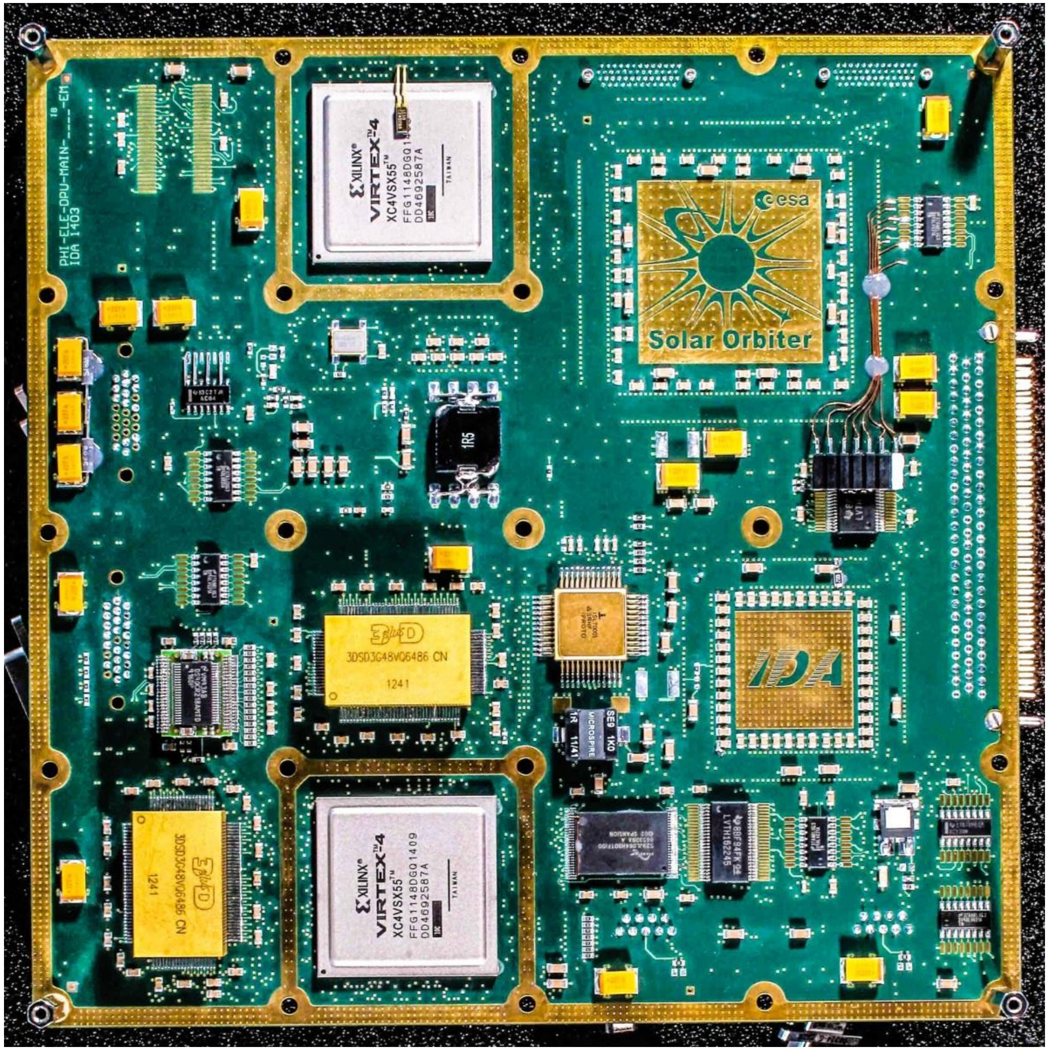
Engineering Model Available, QM Ongoing

- Multi-layer PCB
- Assembled on both sides
- Weight: 550g
- Size: 380 cm^2

➢ **See demo session**



Engineering Model Available, QM Ongoing



Comparison: One and a half Decade ago and Nowadays

	Rosina DPU	Solar Obiter PHI DPU	Factor
Processor system	32-bit TSC21020F DSP	GR-712RC dual-core Leon-3	
Processor memory	11 Mbytes SRAM	256 Mbytes SDRAM	**23**
Processing speed	10 MIPS	50 MIPS/core	5-10
Program size	0.5 Mbytes	<16 Mbytes	<32
FPGAs used	6x Actel RT14100A	RTAX2000 2x XQR4VSX55	
Logic functionality	8k fixed modules	32k fixed modules + RAM 110k reconfigurable + RAM	**17**
Power	5 W primary	15 W secondary	~4
PCB Size	600 cm^2	380 cm^2	0.6
Mass (w/o box)	620 g	550 g	0.9
Error correction	8-bit EDAC	16-bit EDAC	

Future: Controlled Change in Space Applications

Computing platforms are migrating from fixed, ground verified configurations into reconfigurable platforms:

- Configuration changes compete for available platform resources – e.g. timing, memory usage, energy, ... leads to **dependencies** between **changes**
- ➢ **Changes** after deployment must be **controlled** under high grade of system reliability, dependability, availability and safety constraints of space applications
- Space applications typically rely on a high level of functional, performance and environmental verification, often still achieved by intensive testing before use
- ➢ The **contract based change approach** of the CCC project is a missing link which allows to achieve system level dependability, availability and safety using a scalable computing platform architecture

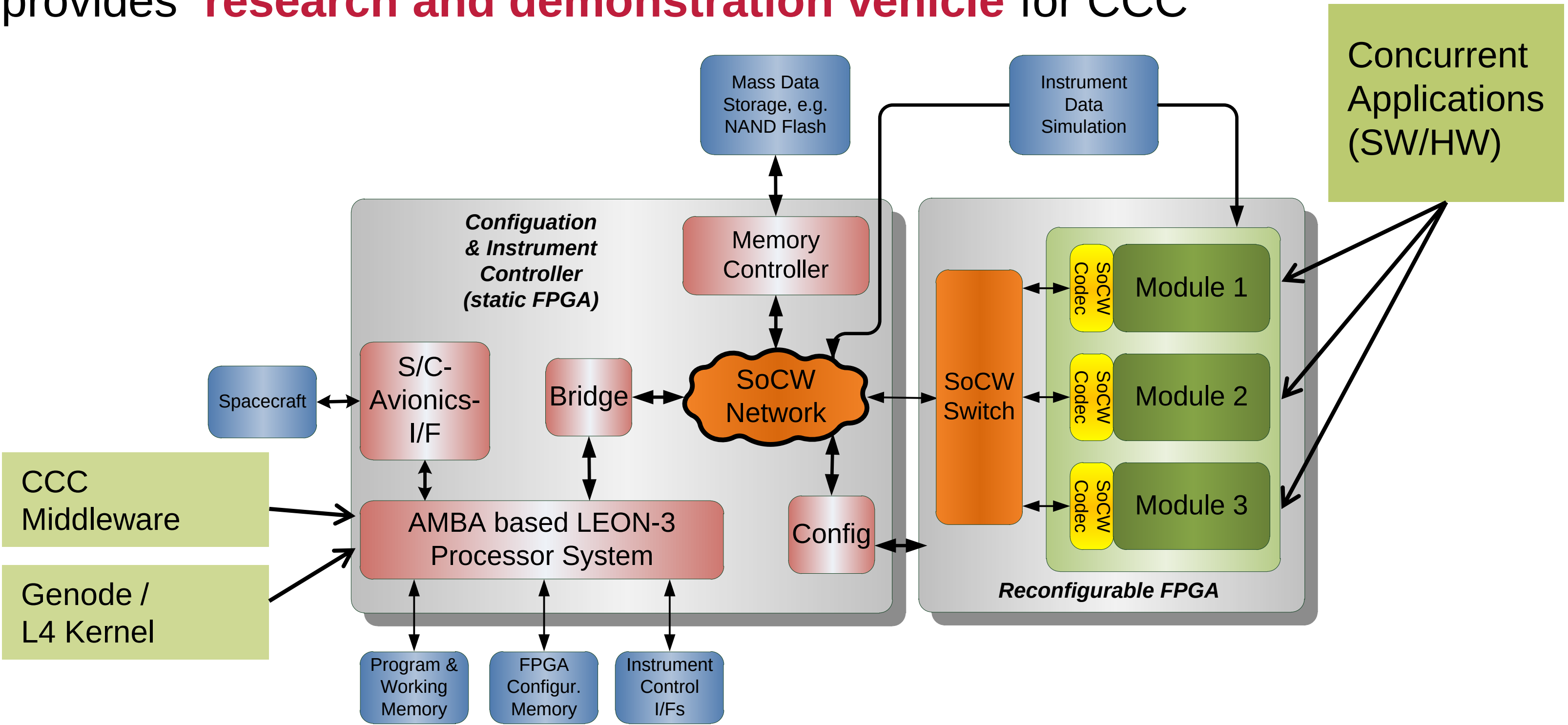
⇒ ***Controlling Concurrent Change - CCC***

Research unit at TU Braunschweig, funded by DFG (Deutsche Forschungsgemeinschaft)

Extension for CCC

Existing applications and platforms will be extended for the controlled change environment to demonstrate usability and capabilities of the CCC approach under the safety, reliability and availability requirements of a typical space application and use of resource limited HW:

➢ provides **research and demonstration vehicle** for CCC



Conclusion

- Increasing demands on data rate and flexibility

- Dynamically reconfigurable FPGAs offer an efficient platform:
  - ➢Efficient use of power and mass
  - ➢Ideal for computation intensive tasks
  - ➢Flexibility to adapt mission specific changes

- PHI first scientific instrument using in-flight reconfigurable FPGAs

- SEUs have to be corrected in the FPGA configuration
  - Scrubbing of the two Virtex-4 FPGAs through JTAG interface
- Error propagation has to be handled in the application itself

- Assembly qualification ongoing / EFM available

Thank you for your attention!

tlange@ida.ing.tu-bs.de

Questions?