Reprogrammable FPGAs at Astrium

ESA Workshop on Fault-Injection and Fault-Tolerance tools for Reprogrammable FPGAs

Tim Pike & Chris Topping

11 September 2009

All the space you need

EADS astrium

Contents

- Application of Reprogrammable FPGAs at Astrium
- Radiation and Single Event Functional Interrupts (SEFIs)
- Determination of SEFI probability
- Some techniques to mitigate SEFIs

- DRPM: FPGA Based Generic Module and Dynamic Reconfigurator
  - ESA contract: Astrium (UK) + IDA (Germany)
  - Successful negotiation in May 2009 with KO in June 2009

Introduction

- Reprogrammable FPGAs are attractive:
  - Flexibility to change programming (algorithm) during development/flight
  - Offer potentially high performance
- At present reprogrammable FPGAs are not used in Astrium flight equipment; instead standard processors (SW), anti-fuse FPGAs and/or custom ASICs are used.
  Reprogrammable FPGAs only used at Astrium in DM and some EMs.
  ATF280 from Atmel presently being evaluated.
- In the space community reprogrammable FPGAs have generally only been used in non-critical payloads where some data corruption or data loss is accepted by the customer.
- Issues affecting use of reprogrammable FPGAs:
  - Performance: convincing equipment reliability and availability analysis
  - FPGA design and tool set visibility and validation
  - Understanding radiation effects on performance including FPGA state machine.

Survey: NASA (JPL) Recommendations

- «Assessing and Mitigating Radiation Effects in Xilinx FPGAs» JPL Publication 08-9 2/08, NEPP Program – very good overview!
- Assessment of mitigation needs:
  - ***None: if rate is acceptable and application is NOT critical;***
  - ***Detection only: reconfigure upon an upset;***
  - ***Full mitigation: design-level Triple Modular Redundancy (TMR) and configuration scrubbing.***
- Mitigation Techniques:
  - ***Internal: still requires, at least, an external watchdog timer;***
  - ***External: upset-hardened application-specific integrated circuit (ASIC) or one-time programmable (OTP) FPGA.***
- Highly recommended that actual flight designs be subject to radiation testing (TMR (tool) implementation, dynamic effects, ...)

SEFIs in reprogrammable FPGAs

- SEFI = Single Event Functional Interrupt
  - FPGA device state machine stops or
  - FPGA device state machine continues but output is corrupted.

1. Characterise basic radiation performance
   - Static Radiation Characterisation: sensitivity of FPGA architectural elements to heavy ion and proton radiation.
   - Application and Dynamic effects assumed negligible (?)
2. Establish relation between SEU and SEFIs
   - Are only SEUs in «used» configuration cells significant?
   - Are SEUs in «unused» configuration cells negligible?
   - Are SEUs in Registers, Block-RAM, User Flip-Flops etc. negligible?
   - What about radiation effects on FPGA device state machine?

SEFI probability derivation I

- SEFI probability by test (preferred)
  - Perform dynamic heavy ion & proton radiation tests on real application under real operational conditions (including mitigation strategies, e.g. TMR) to determine SEFI probability

    BUT In most cases this approach is compatible with neither the project cost envelope nor the schedule!

SEFI probability derivation II

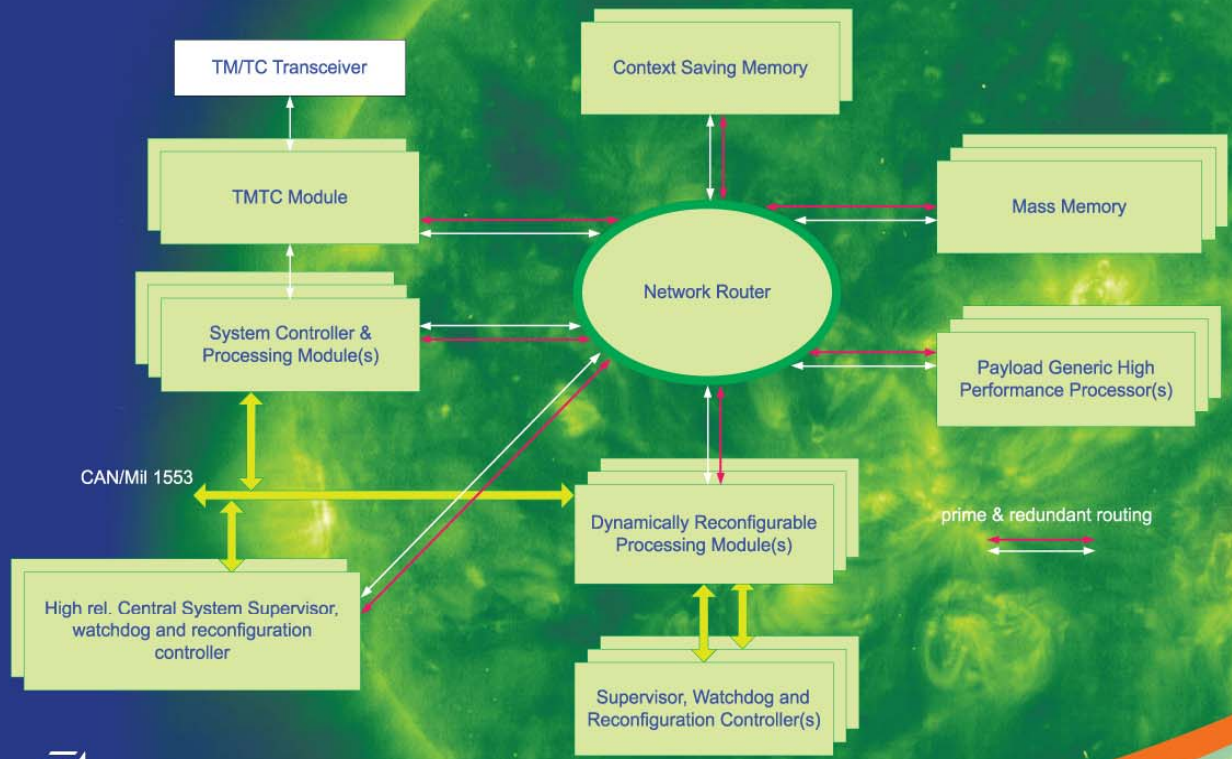
- SEFI probability by analysis (alternative)
  - If radiaton tests not possible, then introduce an appropriate margin (e.g. factor 10) on static radiation data to account for dynamic (temperature?) effects;
  - Derive the SEFI probability as a function of SEU through fault injection (including multiple bit flips) in the real application under real operational dynamic conditions using an appropriate tool;
  - For FPGA elements where fault injection is not feasible then either SEFI originating from these elements must be shown to be negligible (analysis) or an appropriate margin must be taken.
- Watchdog and Test Pattern Insertion
  - During operation, SEFI should be monitored by watch dog and regular insertion of a test pattern with high coverage of the design.

Some Techniques to Mitigate SEFIs

- Closed loop refresh (scrubbing) of configuration layer?
  - Difficult as dynamic elements must be masked out;
  - Most bit errors in configuration layer will not cause a SEFI;
  - ➢ Open loop refresh as much as possible & as often as possible (necessary);
  - ➢ Insert Test Pattern with high coverage in data stream.
- Triple Modular Redundancy (TMR)?
  - Yes, but TMR implementation (tool) must be validated;
  - Dual path (TMR?) at component level;
  - Buffer data with FPGA refresh & retry if SEFI.

FPGA Based Generic Module and Dynamic Reconfigurator

ITT AO/1-5969/08/NL/LvH



All the space you need

EADS astrium

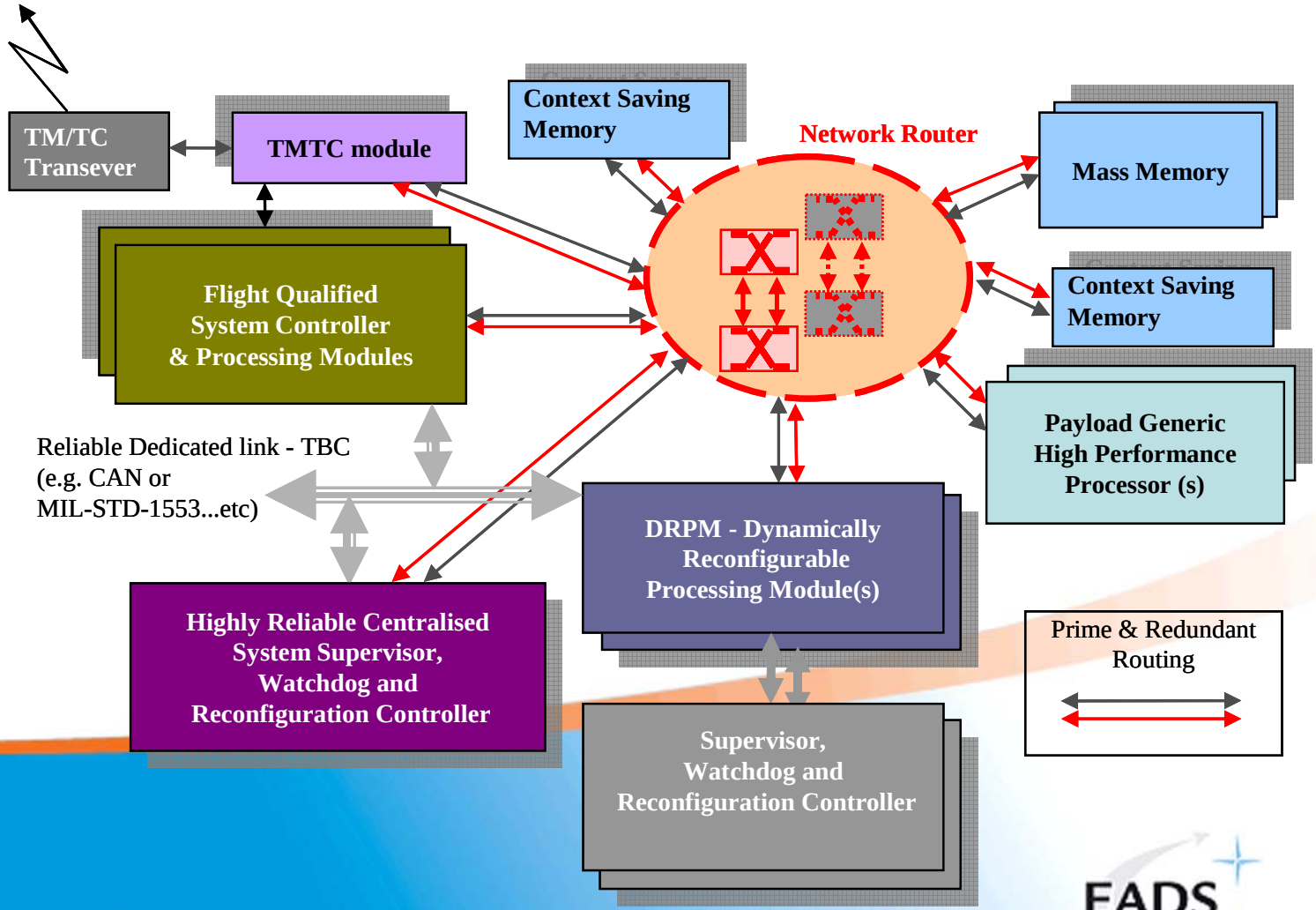
DRPM Project Objectives

- Project aims to develop a demonstrator of an in-flight reconfigurable processing platform for primary use in missions demanding hardware re-usability and design flexibility.
- Deliver a development flow and validation methodology for application design and deployment.

Aim to satisfy the following broad requirements:

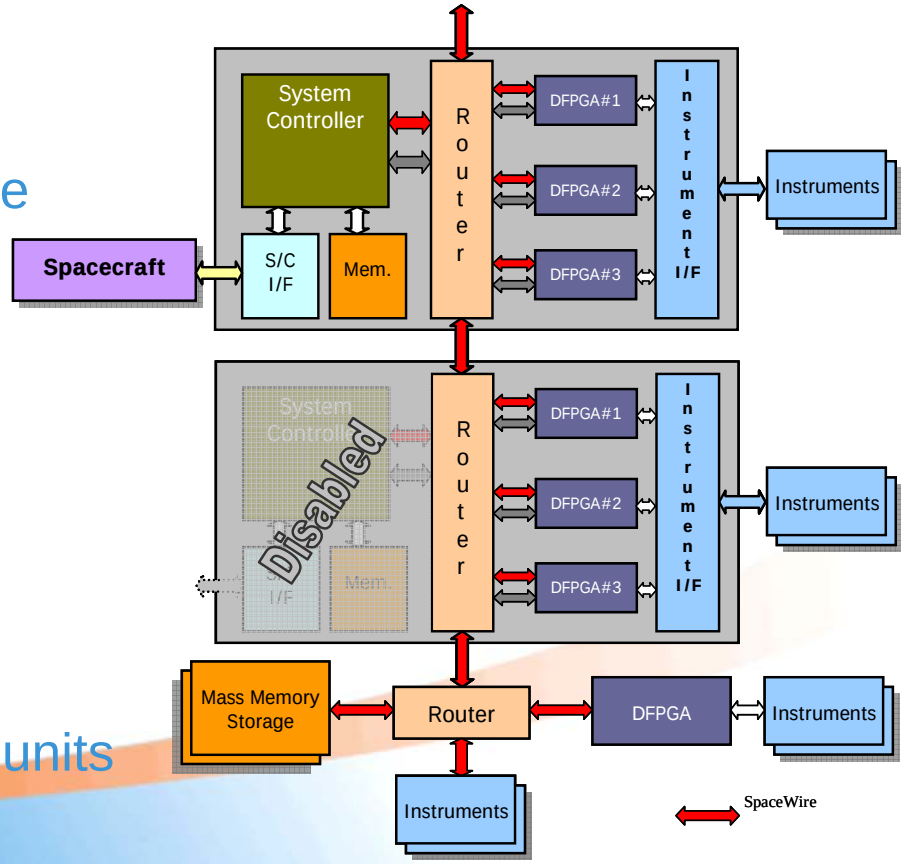
- **Versatile processing and interfacing** catering for multi-instrument payloads
  - ✓ Modular and Scalable solution
- Provide a **reusable processor** for cost effectiveness
  - ✓ Mission and/or In-flight reprogramming
- Improve processing module **reliability** despite sensitivities of reconfigurable technology in radiation environments
- Provide roadmap to **flight solution**, enabled by appropriate technology selection and **application development methodology**

DRPM Concept



DRPM Modularity and Scalability

- Router provides backbone for interconnection between DRPMs
- System controller can handle more than one DRPM
- DFPGA modules provide in-flight reconfiguration and ultimately application programming
- Network interfaces can provide system controller and DFPGAs accessibility to a number of Instruments and memory storage units



Demonstrator Design Drivers

- Technology Drivers
  - Reconfigurable Core
    - ability to reconfigure the devices and provide as much flexibility, reliability and efficiency in achieving this
    - the provision of enough reconfigurable resources for handling the processing requirements
    - Technology supported by tools, enabling application development
  - Reconfigurable Core Supervisor and Controller
    - Device capable of supporting required processing requirements (e.g. size, maximum interface speed)
    - Technology selected based on reliability offered by space qualified counterpart
  - System Controller
    - sufficient computing power
    - software driven for solution flexibility
    - high radiation tolerance and overall reliability
  - Interfaces
    - Limitations of IO rates and electrical requirements of aforementioned units, instrumentation and control interfaces

Application Development Environment (1)

- The aim is to reduce the difficulty in managing dynamically reconfigured applications and to provide a reliable implementation, by providing tools and associated methodologies addressing the following issues:
  - Automatic or manual **partitioning** of a conventional design
  - Specification of the dynamic **constraints**
  - **Verification** of the dynamic implementation through dynamic simulations at key steps of the design flow
  - **Development of configuration controller** core
  - Dynamic **floorplanning management** and guidelines for modular back-end implementation if not supported easily by vendor tools.

Application Development Environment (2)

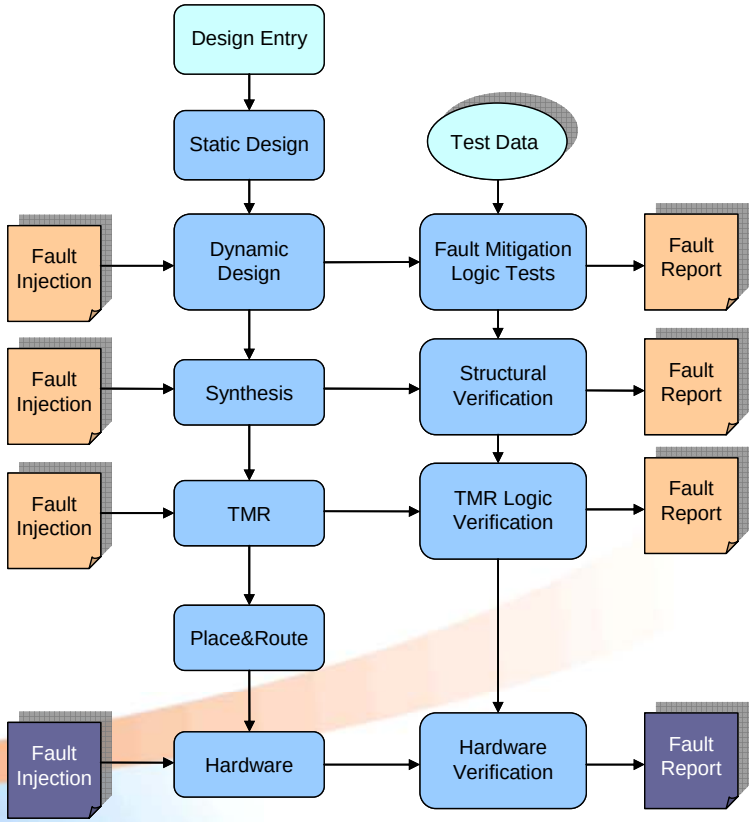
- In support of the application development and validation, it is necessary to provide the user with a design tool kit and methodology

- Considerations are :
  - the **technology of the reconfigurable elements** used within the DRPM;
  - a design containing microprocessors, thus requiring a **software and hardware development thread**; i.e. co-development issues;
  - the **validation strategy**, which must allow for the incremental testing of applications as part of the wider DRPM processor.

Fault Testing

Fault Injection for testing effectiveness of:

- Triple Modular Redundancy;
- configuration memory scrubbing;
- applying EDAC or CRCs where necessary (e.g. memory data, data path processing etc.);
- using partial reconfiguration for correcting faulty configuration or user memory data;
- automatic or semi-automatic switching out of faulty units or system elements (e.g. system controllers, DFPGA modules, DRPM modules, interfaces etc.);
- re-distributing application functionality from faulty reconfigurable cores in the event of partial failure of reconfigurable core fabric;



Dates of Delivery and Progress Milestones