Modular Architecture for Robust Computation

Session: SpaceWire Onboard Equipment and Software - Short Paper

Dr. W.Gasti,

European Space Agency, Postbus 299, NL-2200 AG Noordwijk, The Netherlands

A.Senior

SEA, Building 660, Bristol Business Park, Coldharbour Lane, Bristol, BS16 1EJ, United Kingdom

Dr. O. Emam, T. Jorden and R.Knowelden

EADS Astrium, Gunnels Wood Road Stevenage, Hertfordshire SG1 2AS, United Kingdom

S. Fowell

SciSys UK, Methuen Park, Chippenham Wiltshire, SN14 0GB, United Kingdom

E-mail: wahida.gasti@esa.int, alan.senior@sea.co.uk, omar.emam@astrium.eads.net, tony.jorden@astrium.eads.net, richard.knowelden@astrium.eads.net,stuart.fowell@scisys.co.uk,

1. Introduction

The classical On-Board Computing System (OBCS) based on one-processor and related multi-drop bus architectures is too limiting for future space missions. The ESA ROSETTA mission has already faced this problem and it was necessary to enhance the network architecture by upgrading the OBCS with an additional processor and incorporate high speed serial links (based on IEEE-1355 Std) to respectively cope with the AOCS and payload needs. Indeed, nowadays the satellite core functions; e.g. AOCS, communication system, power system and the Payloads demand algorithms/techniques requiring the incorporation of Intelligent Units (IUs). IUs are more demanding in terms of power and computation resources and communication speeds are exceeding the capabilities of the large classical OBCS.

ESA anticipated these needs by initiating the SpW Network Technology Program several years ago. In this scope, MARC can be considered as a significant development that will provide to the space user community a robust SpaceWire based OBCS architecture that may be implemented as a Flight Qualifiable design.

2. MARC Requirements

The principal requirements are that MARC shall provide a **Robust and Fault Tolerant Computing System** based on:

- A **Modular and Scalable** network and system architecture
- A **SpaceWire network** capable of handling the future demands of both spacecraft data handling and payload data processing.
- A **hierarchical Fault Detection Isolation and Recovery** (FDIR) principles
- The **CCSDS SOIS layered software architecture** [RF3]
- Flight Qualifiable technologies such that the design can be considered as "**One Step from a Flight Model**".
- The key SpW components and standards [RF1, RF2] developed by ESA: **SpW links, SpW-10X router, RMAP.**

3. MARC FT-Design

3.1. MARC FT-Design Principles

MARC is based on a 5 layer hierarchical FDIR architecture. Each layer either handles any fault in that layer itself, or passes the fault up the hierarchy to the next layer. The layers from the highest to lowest are:

1.Ground
2.TMTC system
3.Hardware Reconfiguration Controller (HRC)
4.Master Core Computing Module
5.General Computing Module (based on COTS processor) and/or Module(s) level FDIR.

The MARC Fault Tolerant (FT) design is based on the following principles:

- No single point of failure:
- No single point of repair: If MARC experiences a single failure; it must be able to continue the handling of key functions without interruption during the repair process by provision of two

identical instances of the same SpW-module and switching (via SpW-Routers) to one of the remaining instances in case of a failure (hot redundancy failover).

- Fault isolation to the failing component, module;
- Fault containment to prevent propagation of the failure;
- Availability of autonomous safe computing system mode and reversion to operational modes on external command;

3.2. MARC FT-Design

To fulfil the MARC requirements and the FT-design principles, MARC has been built starting from a set of HW and SW Building Blocks (BBs). These BBs have been designed to insure the robustness of MARC and are summarised in the following subsections.

3.2.1. MARC Hardware Building Blocks

MARC Highly Reliable HW Components

ESA initiated and funded a set of ASSPs (Application Specific Standard Product) which are devices developed in the same way as an ASIC with the intention to be widely used and covering multi-mission needs. MARC modules are based on 2 ASSPs they are the LEON2FT based processor (AT697F) and SpW_10X router i.e. (AT7910).

MARC Cluster & HW Architecture

The MARC SpaceWire network architecture is based on a building block called a High Flexibility Cluster (HFC) which comprises 2 routers and 4 Modules. The modules may be any function (processing, memory, I/O etc) that requires a network interface. The two 8 port routers provide redundancy and each module interfaces to both routers to allow for failure of one router within the cluster.

The HFC has 4 spare ports, each port comprising a redundant SpW link. These spare ports may be used to connect clusters together or to connect to other system components such as the TMTC system or EGSE.

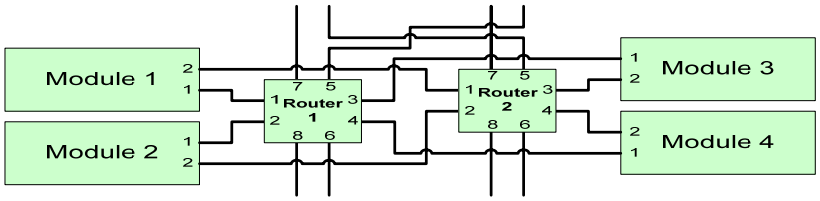

Figure 1: **MARC SpW High Flexibility Cluster (SpW-HFC)**

The HFC is in effect a 4 port router with failure tolerance and thus multiple HFCs may be connected together in any network topology. The simplest HFC based topology is to stack the clusters vertically as shown in Figure 2. If an increased bus bandwidth is required then the number of 8 port routers within a cluster may be increased to 4 or more as shown in Figure 3.

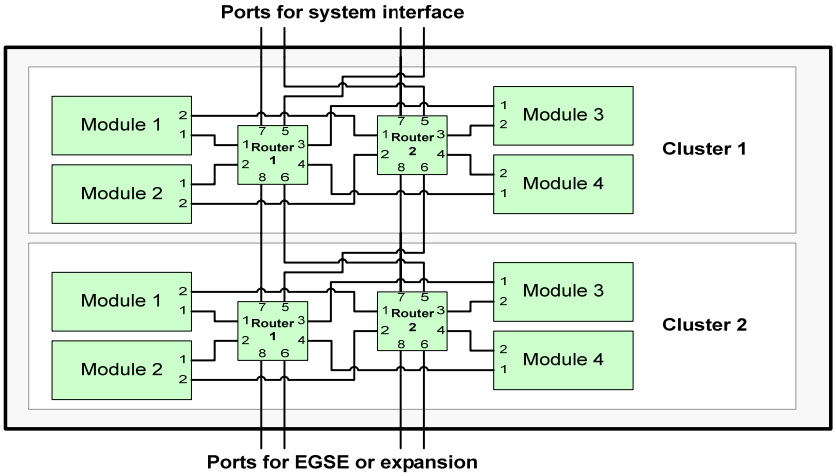

Figure 2: **SpW-HFCs Vertical Assembly**

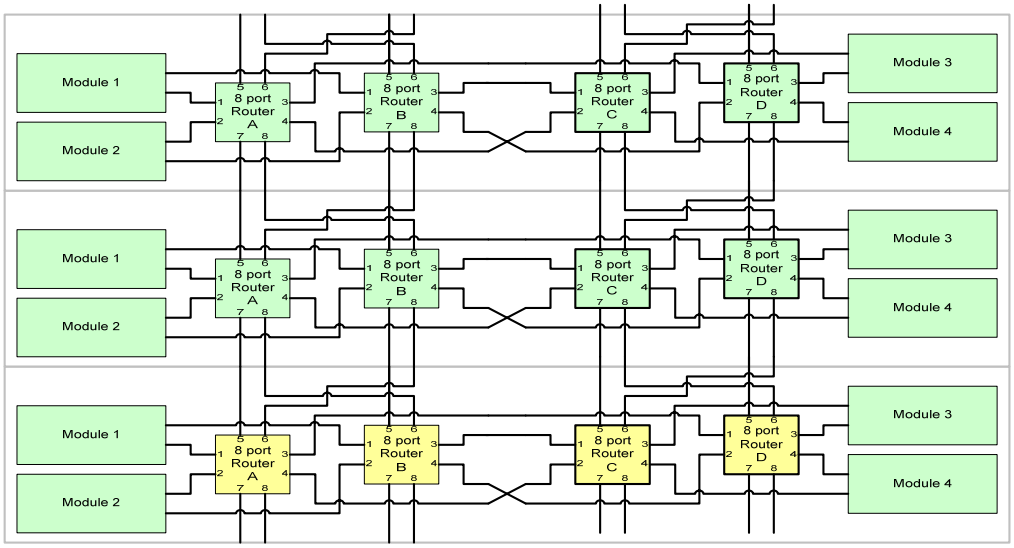

Figure 3: **SpW-HFCs Lateral Assembly**

To fulfil basic Platform and Payload computing needs, MARC can typically be built out of five types of SpW-modules that are:

- Telemetry/Telecommand Module (TTM),
- Core Computing Module (CCM)
- Memory Module (MM)
- I/Os Module (IOM)
- General Computing Module (CGM).

The related SpW-Network is configured and monitored by the master CCM. The SpW-Network is implemented using 8 port SpW-routers that are part of the backplane.

MARC network monitoring will make use of 2 types of heartbeat function as follows:

- SpW-Private Heartbeat Function (SpW-PHF) for each module which monitors its health and status in the cluster. Its report (which can be tailored down to a minimum “I am alive”) is under the responsibility of the module and it shall be send regularly to the MARC master CCM.
- SpW-Network Heartbeat Function (SpW-NHF) that is provided by the MARC backplane Hardware Reconfiguration Controller (HRC) which is responsible for ensuring that critical Modules are powered (according to a configuration table), monitoring the health and status of each CCM and the distribution of SpW time codes.

The management of failures will be handled in a hierarchical manner, such that resolution is sought from the lowest possible level (Module) to the highest local system level as follows:

1. Module level (local FDIR e.g. watchdog timer): Module takes internal action (e.g. reboots);
2. SpW-PHF from Module to CCM: CCM takes action if heartbeat is missing;
3. SpW-NHF from CCM to HRC: HRC takes action if heartbeat is missing

RMAP for SpW-Module memory access

All MARC modules have at least two SpW interfaces for redundancy and these incorporate RMAP functionality (via ESA-RMAP-IP-Core). So, for example, it is possible for the CCM to access memory banks housed in any module as an extension of its own memory. The CCM can access software and data resources from at least 3 sources, i.e. local, another CCM or the Mass Memory. Although simple, this memory access capability is very efficient in coping with the Prime CCM failure and permits context saving memory access by the redundant CCM. The access is possible without any system reconfiguration. At configuration and the system initialisation phase, it is possible to distribute boot SW, context saving and data mission storage to the available memory banks of MARC. Nevertheless, each CCM has its own SW boot capability in case MARC is used only for platform command/control application.

3.2.2. MARC SW Building Blocks

MARC SW Architecture: GenFas

GenFas is based on the CCSDS-SOIS layered SW architecture [RF3] using the LEONFT2 processor of the CCM. GenFas is relying on SpW Real-Time Communication Protocol (SpW-RTCP). SpW-RTCP communication model implying:

- a virtual point-to-point connections across SpW network: Virtual points address either source or destination and are represented by channel buffers. A user application (highest SOIS Layer) writes

into one of the source channel buffers available and related data packet is then transferred across the SpW network and becomes available in the corresponding destination channel buffer.

- a scheduled communication: Transmission of data packets is synchronous with each source channel being assigned one or more time-slots when it is allowed to transmit packet(s). Timeliness of delivery is controlled by a schedule table used to specify which source channel can send packet(s) in which time-slot. This provides deterministic delivery.

MARC SpW-RTCP provides 3 QoS according to CCSDS standard definition that are: Basic, Best Effort, Assured.

MARC FDIR

MARC implements a hierarchical FDIR management architecture detecting, isolating and determining the recovery strategy of failed MARC modules, SpW-Routers and SpW-links with failover strategies or switching to a safe computing mode. The hierarchy in terms of computing capability shall consist of 1st class reliable computing modules, i.e. the CCMs, that have the role of MARC system supervisor as well as their own FDIR capabilities, and 2nd class less-reliable computing modules, i.e. the GCMs that are limited to only managing their own FDIR capabilities and only act as a servant to the system supervisor. This hierarchical FDIR strategy is providing Fault Containment Regions (FCRs) possibly confined to modules and/or clusters, with faults autonomously handled or escalated at each level in the hierarchy (with module and system as the minimum defined levels).

MARC FDIR Manager is GenFas embedded software as advocated by SOIS standard. FDIR Manager uses tables which define the initial system configuration and the alternative configurations to be used in response to one or more identified failures of a SpaceWire link, router or system module. The FDIR tables (FDIR-Tabs) are generated by an analysis tool which is conceived to run off-line during the architectural design phase of MARC system. The FDIR Manager Strategy is achieved this by performing its four principal functions that are:

- *Health Check Monitoring*: Health check monitoring is a regular and continuous activity performed through the exchange of Health Status Messages (HSM) between the FDIR manager and every component making up the system modules, routers and links. HSM can be both solicited (pulled) or unsolicited (pushed). Each component will have a programmable time period (T*hs*) associated with it. HSM system is the basis by which the FDIR Manager is able to detect the occurrence of fault within MARC.
- *Fault Detection*: Fault detection is determined by both the arrival time of expected HSM (both solicited and unsolicited) and by the content of the messages themselves. Faults are registered for a specific component if an expected HSM from that component is either early or late or does not arrive at all, or if the content of a message that does arrive on time flags an 'I am not OK' status. Using the expected T*hs*, and three further programmable time constants T0, T1 and T2 (where T0<T1<T2), the FDIR Manager can work out a time range within which T*hs* of the expected HSM can be classified as **Valid**, **Missing, Late** or **Lost**.
- *Fault Diagnosis and Identification:* Following detection of a fault, the FDIR Manager will log the event in its Registered Fault Log. The Fault Diagnosis Procedure is then initiated and during this process, statuses are assigned by the FDIR Manager to the fault events in order to record the seriousness of the fault and the action that should be taken in the event of a recurrence. There are three fault event statuses: **Transient, Intermittent** and **Permanent.**
- *Fault Recovery:* fault recovery is based on 3 procedures that are:

a) *Initial Recovery Procedure*: That is triggered by the FDIR Manager in the event that the Diagnosis Procedure has identified and localised a fault within a component. It represents a final attempt to recover functionality in the component before the use of redundant components and rerouting is performed.

a) Initial Recovery Procedure: That is triggered by the FDIR Manager in the event that the Diagnosis Procedure has identified and localised a fault within a component. It represents a final attempt to recover functionality in the component before the use of redundant components and rerouting is performed.

b) *Router Recovery Procedure*: That is initiated by the FDIR Manager following the Initial Recovery Procedure to recover a faulty router. The Procedure uses the stored FDIR-Tabs to reconfigure and restore the network and varies depending on if the network has cross strapping. Once the new router or routers have been activated, new routing address tables are set and the FDIR Manager sends notifications to all affected nodes that the new routers are available and must now be used.

c) *Node Recovery Procedure*: That is initiated by the FDIR Manager following the Initial Recovery Procedure to recover a faulty module. The procedure is linear, and uses the stored FDIR-Tabs to initiate redundant module(s) and set up new routing address tables. Firstly, the faulty module is

switched off and the selected redundant node (identified using the FDIR-Tabs) switched on. The new module is then configured, using the Configuration Manager, and a new routing address table set up. Finally the FDIR Manager sends notifications to all affected modules that the new module is available and must now be used.

4. MARC Demonstrator

In a first stage, to validate MARC conceptual architecture a prototype called MARC Demonstrator will be developed. As presented in figure 4, it is based on 2 clusters and will encompass the following features:

- Backplane (i.e. SpW_10X routers and HRC)
- Core Computing Module (based on AT697
- Mass Memory Module including file system
- General Computing Module: (based on COTS Power-PC I/Fs)
- Emulation of I/O and TMTC module via PCs and specific SpW EGSEs.
- GenFas software including FDIR Manager

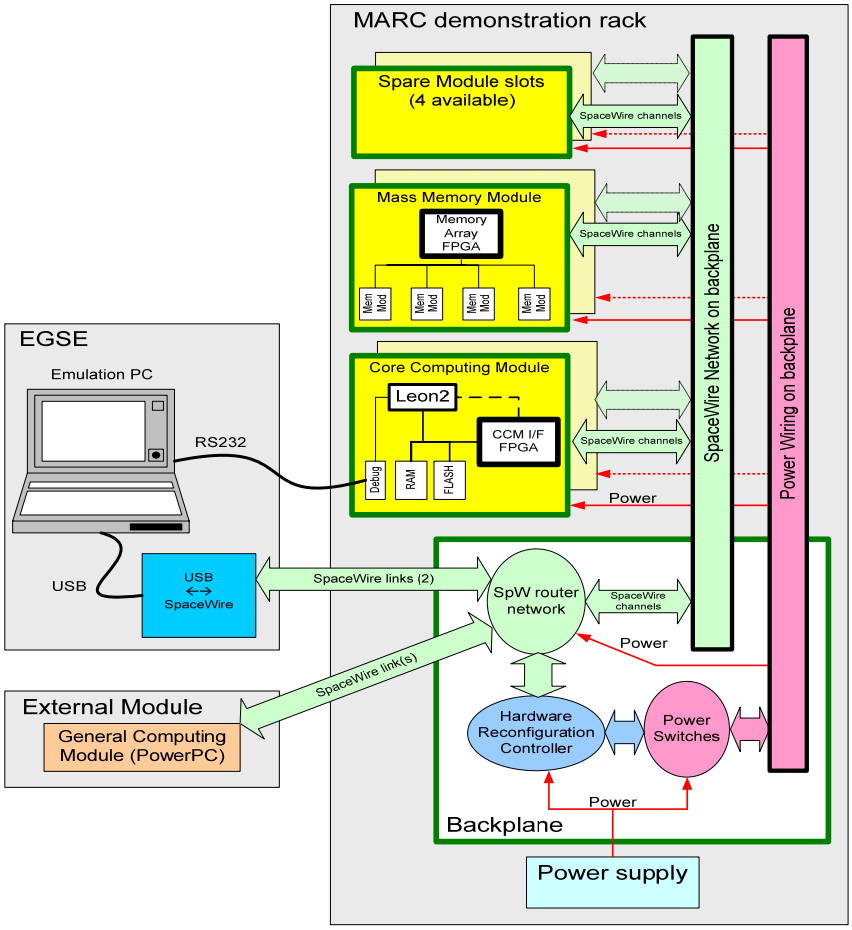

Figure 4: **MARC Demonstrator and Test Bench**

5. Conclusion

As presented in this paper, MARC design is relying on BBs not only ensuring its robustness but also its high flexibility. The MARC development is based on two phases: phase one for preliminary design and feasibility aspect, phase two for detailed design, manufacturing and tests/verification. The first phase has been completed and indicates that:

- The requirements and the FT-Design Principles as defined are clear and feasible,
- MARC preliminary design is consistent with previous requirements.
- This preliminary design of MARC is fulfilling many future ESA mission OBCS needs.
- The time and the cost needed to adapt MARC Demonstrator system to new mission appears to be significantly shorter than a development of a new system.

As a general requirement, it was also required to assess MARC overall performances compared to existing solutions. To this end, MARC Demonstrator will be the first MARC System prototype

allowing performance assessment. MARC Demonstrator will be available at the level of Elegant-Bread Board (as defined by ECSS-E-10-02A).

6. Reference Documents

RF1: ECSS-E-50-12A**,** SpaceWire - Links, nodes, routers and networks, January 2003
RF2: ECSS-E-50-11 Draft version09, Remote Memory Access Protocol (RMAP), June 2008
RF3: CCSDS Spacecraft On-Board Interfaces Services (SOIS) Informational Report, Green Book, 850.0-G-1.1, Jan 2008 and related documents (i.e. SOIS Sub-Network Packet Service, SOIS Sub-Network Memory Access Service, SOIS Sub-Network Time Distribution Service)