AIR: Technology Innovation for Future Spacecraft Onboard Computing Systems

J. Rufino, J. Craveiro
Universidade de Lisboa, Faculdade de Ciências, LaSIGE
Lisbon, Portugal
Email: ruf@di.fc.ul.pt, jcraveiro@lasige.di.fc.ul.pt

T. Schoofs, J. Cristóvão, S. Santos, C. Tatibana
GMV Portugal
Lisbon, Portugal
Email: {tobias.schoofs, joao.cristovao,
sergio.santos, cassia.tatibana}@gmv.com

I. Introduction and Motivation

Aerospace systems have strict dependability and real-time requirements, as well as a need for flexible resource reallocation and reduced size, weight and power consumption. To cope with these issues, while still maintaining safety and fault containment properties, temporal and spatial partitioning (TSP) principles are employed [4]. In a TSP system, the various onboard functions (avionics, payload) are integrated in a shared computing platform, however being logically separated into partitions. Robust temporal and spatial partitioning means that partitions do not mutually interfere in terms of fulfilment of real-time and addressing space encapsulation requirements.

II. The AIR architecture

The AIR architecture is the result of a partnership between academia and industry partners (Fig. 1), motivated by the interest of the European Space Agency (ESA) in applying TSP concepts to space missions. AIR foresees the use of different operating systems among the partitions, either real-time or generic non-real-time ones (Fig. 2). Partitions are scheduled cyclically under a fixed schedule, and processes are scheduled by the native scheduler of the operating system of the partition in which they are executing (Fig. 3) [3].

A. Portable Application Executive (APEX) interface

The APEX Interface implements the services defined in the ARINC 653 standard [1]. The AIR APEX consists of two components: the APEX Core Layer and the APEX Layer Interface (Fig. 4). The APEX core Layer implements the advanced notion of *Portable APEX* intended to ensure portability between the different OS supported by AIR. The APEX also coordinates when required the interactions with the AIR Health Monitor (e. g., upon the detection of an error) [2].

B. Health Monitor

The AIR Health Monitor is responsible for handling hardware and software errors (like deadlines missed, memory protection violations, exceptions or hardware failures). As much as possible, it will isolate the error propagation within its domain of occurrence (process, partition, system).

C. Temporal robustness and adaptability

1) Mode-based partition schedules: The AIR advanced design addresses configuration flexibility and fault tolerance issues by introducing support for multiple mode-based partition schedules (Fig. 3). Partition scheduling can thus adapt to different modes/phases (initialization, operation, etc.) or accommodate component failures. Switching between the different partition scheduling tables (PST) can be requested by authorized partitions, as a response to either an order from a ground station or to sensed environmental conditions.

2) Process deadline violation monitoring: During the execution of the system, it may be the case that a process exceeds its deadline; this can be caused by a malfunction or because that process's execution time was underestimated at system configuration and integration time. Deadline information is updated by the appropriate APEX primitives, and verification of the minimum required earliest deadlines is performed at each system clock tick to allow timely detection of the fault.

III. Proof-of-concept prototype

To demonstrate the innovative design and its advanced timeliness control features, we developed prototype implementations of an AIR-based system. Each partition executes an RTEMS-based (Real-Time Executive for Multiprocessor Systems) mockup application representative of typical functions present in a satellite system. The demonstrations were implemented for Intel IA-32 (Fig. 5) and SPARC LEON (Fig. 6) targets.

This work was partially developed within the scope of the European Space Agency Innovation Triangle Initiative program, through ESTEC Contract 21217/07/NL/CB, Project AIR-II (ARINC 653 in Space RTOS — Industrial Initiative, http://air.di.fc.ul.pt). This work was partially supported by FCT, through the Multiannual Funding and CMU-Portugal Programs and the Individual Doctoral Grant SFRH/BD/60193/2009.

References

[1] AEEC, "Avionics application software standard interface, part 1 - required services," Aeronautical Radio, Inc., ARINC Specification 653P1-2, Mar. 2006.

[2] J. Rufino, J. Craveiro, T. Schoofs, C. Tatibana, and J. Windsor, "AIR Technology: a step towards ARINC 653 in space," in *Proc. DASIA 2009 "DAta Systems In Aerospace" Conf.*, Istanbul, Turkey, May 2009.

[3] J. Rufino, J. Craveiro, and P. Verissimo, "Architecting robustness and timeliness in a new generation of aerospace systems," in *Architecting Dependable Systems VII*, ser. Lecture Notes in Computer Science, A. Casimiro, R. de Lemos, and C. Gacek, Eds. Springer Berlin / Heidelberg, 2010, vol. 6420, pp. 146–170.

[4] J. Rushby, "Partitioning in avionics architectures: Requirements, mechanisms and assurance," SRI International, California, USA, NASA Contractor Report CR-1999-209347, Jun. 1999.

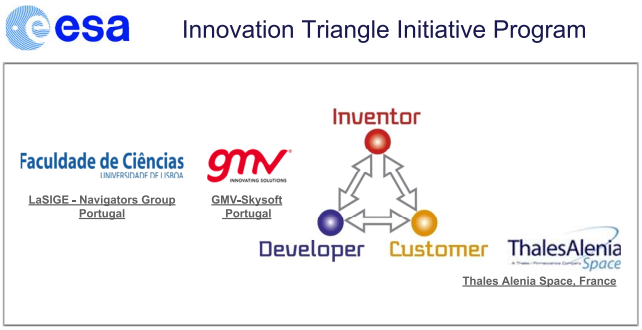

Fig. 1. Research program and consortium

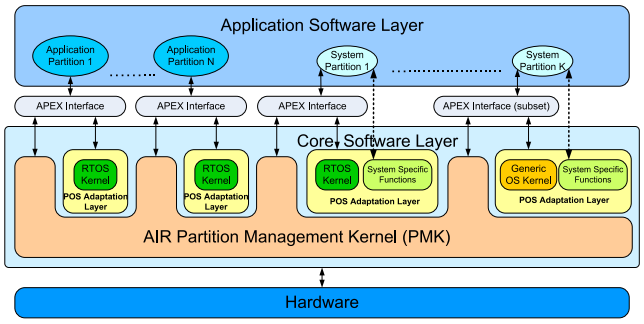

Fig. 2. AIR time- and space-partitioned architecture, enabling safe integration of operating systems and applications

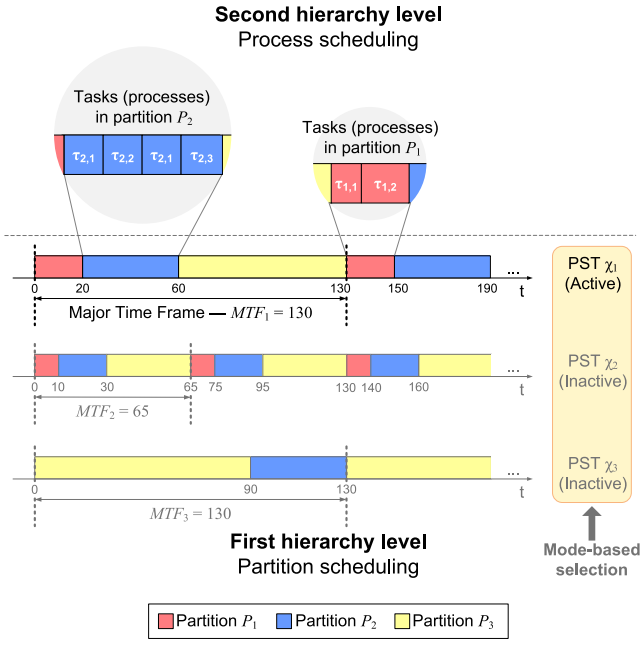

Fig. 3. Flexible and adaptive two-level hierarchical scheduling, with mode-based partition schedules

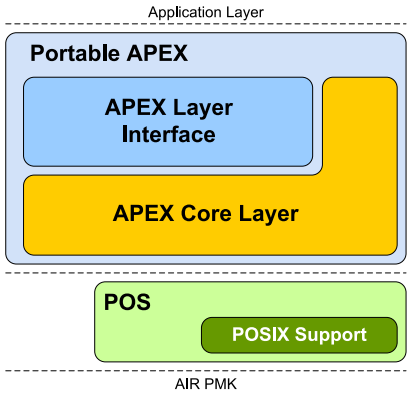

Fig. 4. Flexible and portable application programming APEX interface, conformant with the ARINC 653 specification

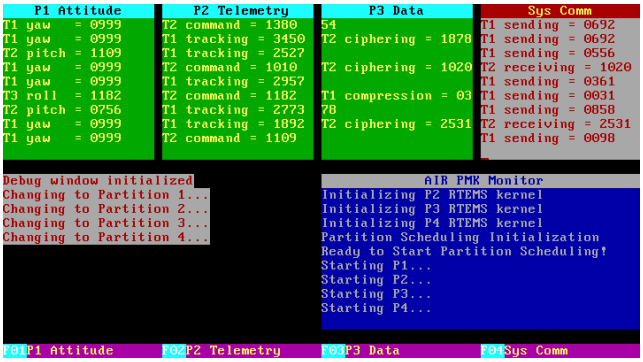

Fig. 5. Proof-of-concept IA-32 prototype, running on the QEMU emulator

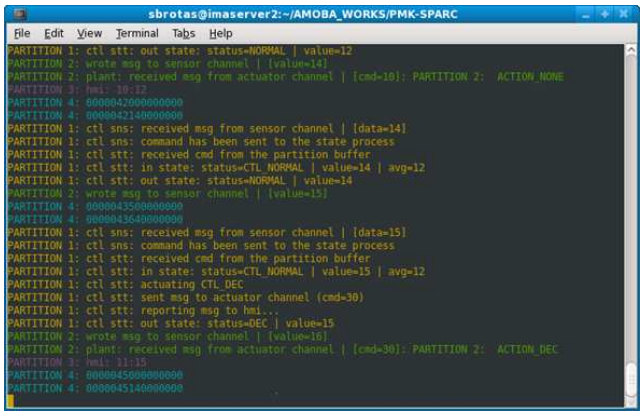

Fig. 6. Industrial-grade SPARC LEON prototype demonstrator, running on the Gaisler TSIM simulator