Robust Partitioning and Composability in ARINC 653 Conformant Real-Time Operating Systems

José Rufino
LASIGE - FCUL*
ruf@di.fc.ul.pt

João Craveiro
LASIGE - FCUL
jcraveiro@lasige.di.fc.ul.pt

1 Introduction and Motivation

The ARINC 653 standard [2] has its origin in the civil aviation world and it aims to provide a standardized interface between a given Real-Time Operating System (RTOS) and the corresponding application software as well as a set of functionalities aimed to improve the safety and certification process of a safety-critical system. The ARINC 653 standard specification [2] and its concept of partitioning (spatial and temporal) are gaining increased importance and acceptance also in the realm of aerospace applications [9], as it happens within the scope of the European Space Agency (ESA) Technology Harmonization effort for space on-board software [7].

The adoption of the ARINC 653 concept in space on-board software will provide the application developers with an environment that is standard and independent from any RTOS and the integrators with an easier integration environment together with portable applications [3]. The partitioning concept makes it adequate to software with different degrees of criticality [7].

The technological interest of ESA in the ARINC 653 standard was expressed in [10]. The ESA support to the development of a proof of concept [3, 8] and demonstration of feasibility of use is being provided within the scope of the ESA innovation triangular initiative - AIR project.

2 ARINC 653 Fundamental Concepts

The ARINC 653 specification is an important block from the Integrated Modular Avionics (IMA) definition [1], where the partitioning concept emerges for protection and functional separation between applications, usually for fault containment and ease of verification, validation and certification [2, 9].

*LASIGE - Laboratório de Sistemas Informáticos de Grande Escala - Faculdade de Ciências da Universidade de Lisboa, Campo Grande - Bloco C8, 1749-016 Lisboa, Portugal. Tel: +351-21-7500254 - Fax: +351-21-7500084. This work was partially supported by FCT through the Multiannual Funding Programme. This work was partially supported by ESA (European Space Agency) through the ITI program, ESTEC Contract 21217/07/NL/CB - AIR project (http://air.di.fc.ul.pt).

2.1 ARINC 653 System Architecture

The architecture of a standard ARINC 653 system is sketched in Figure 1. At the application software layer, each application is executed in a confined context, dubbed partition in ARINC 653 terminology [2]. The application software layer may include system partitions intended to manage interactions with specific hardware devices.

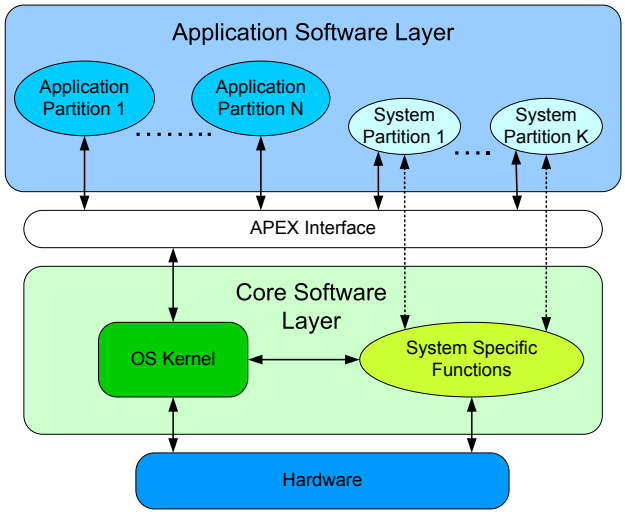

Figure 1. Standard ARINC 653 Architecture

Application partitions consist in general of one or more processes and can only use the services provided by a logical application executive (APEX) interface, as defined in the ARINC 653 specification [2]. System partition may use also specific functions provided by the core software layer (e.g. hardware interfacing and device drivers), being allowed to bypass the standard APEX interface.

The execution environment provided by the OS kernel module must furnish a relevant set of operating system services, such as process scheduling and management, time and clock management, and inter-process synchronization and communication.

2.2 Spatial and Temporal Partitioning

Spatial partitioning ensures that it is not possible to an application to access the memory space (both code and data) of another application running on a different partition. Temporal partitioning ensures that the activities in one partition do not affect the timing of the activities in other partition. In ARINC 653, this is supported by a fixed cycle based scheduling, where a major time frame of fixed duration is periodically repeated throughout runtime operation.

2.3 Health Monitoring

The Health Monitoring (HM) functions consist in a set of mechanisms to monitor system resources and application components. The HM helps to isolate faults and to prevent failures from propagating. Within the scope of the ARINC 653 standard specification the HM functions are defined for process, partition and system levels [2].

2.4 ARINC 653 Service Interface

The ARINC 653 service requests define the application executive APEX interface layer (Figure 1) provided to the application software developer and the facilities the core executive shall supply. A set of services is mandatory for strict compliance with the ARINC 653 standard [2]. Those services are grouped in the following major categories: partition and process management, time management, intra and inter-partition communications, and health monitoring.

3 ARINC 653 Interface in Real-Time Operating Systems

Currently available ARINC 653 implementations are commercial and very expensive solutions provided by major companies of the aeronautic market. The AIR innovation initiative represents a first but significant step toward the usage of off-the-shelf open-source RTOS kernels in the definition and design of ARINC 653 based systems.

This section describes the fundamental ideas on how a RTOS kernel (e.g. the Real-Time Executive for Multiprocessor Systems - RTEMS [5]) can be adapted to offer the application interface and the functionality required by the ARINC 653 specification [2, 8].

3.1 AIR System Architecture

A simple solution for providing the ARINC 653 functionality missing in off-the-shelf RTOS kernels, such as RTEMS, implies to encapsulate those functions in components with a well-defined interface and add them to the bare operating system architecture.

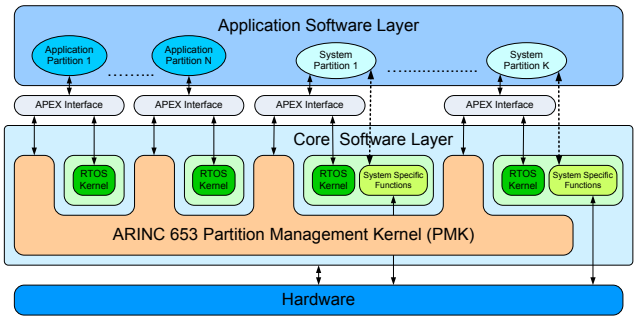

Figure 2. AIR System Architecture

The design of the AIR architecture in essence preserves the hardware and RTOS independence defined within the scope of the ARINC 653 specification [2, 3, 8]. A specific module (cf. Figure 2) that needs to be added to the RTOS kernel (e.g. RTEMS) is the **AIR Partition Management Kernel (PMK)** which includes the following functions:

- **AIR partition scheduler**, selecting at given times which partition owns system resources, namely the processing infrastructure. It secures temporal segregation using a single fixed cyclic scheduler.
- **AIR partition dispatcher**, which has the responsibility of saving the execution context of the running partition and of restoring the execution context for the heir partition. It secures the management of all provisions required to guarantee spatial segregation.
- **AIR inter-partition communication module**, allowing the exchange of information between different partitions without violating spatial segregation constraints.

Another fundamental component concerns the **ARINC 653 application executive (APEX) interface**, defining for each partition in the system a set of services in strict conformity with the ARINC 653 standard. It is designed as much as possible by mapping the ARINC 653 services into the native and/or POSIX primitives of the RTOS [2, 5, 4].

3.2 AIR Robust Partitioning and Composability

Robust partitioning comprises the protection of each partition's memory addressing space, to be provided by specific memory protection mechanisms usually implemented in a hardware memory management unit (MMU). It requires also a functional protection concerning the management of privilege levels and restrictions to the execution of privileged instructions. Though there is room for enhancements,

a basic set of such mechanisms do exist in the Intel IA-32 architecture and, to a given extent, in the SPARC LEON processor core.

The ARINC 653 standard specification [2] restricts the processing time assigned to each partition, in conformity with given configuration parameters. The scheduling of partitions defined by the ARINC 653 standard is strictly deterministic over time. Each partition has a fixed temporal window in which it has control over the computational platform. Each partition is scheduled on a fixed, cyclic basis.

In the AIR architecture, temporal segregation is ensured by the AIR partition scheduler. This opens room for the temporal composability of applications.

To ensure flexibility and modularity, instead of modifying the RTOS scheduler to extend it to the partitioning concept, the approach followed in the AIR architecture uses one instance of the native RTOS scheduler for process scheduling inside each partition. No fundamental modification is needed to the functionality of the RTOS process scheduler for its integration in the AIR system. Such a two-level hierarchical scheduler approach secures partition and process scheduler decoupling, thus allowing the use of different operating systems in different partitions (e.g. RTEMS [5], eCos [4],...).

4 Concluding Remarks

The AIR architecture aims to provide the developers and the integrators of space on-board software with an environment that is standard and in strict conformity with the ARINC 653 specification [2].

The AIR solution is hardware and operating system independent and it exploits the usage of conventional off-the-shelf open-source RTOS kernels, such as RTEMS [5], a real-time multitasking kernel qualified for use in space on-board software developments.

We have discussed the fundamental aspects of the definition and design of the AIR architecture and how its bare components can be integrated in a multi-executive core layer structure, which uses a RTOS kernel instance per partition. Partitions are the units of protection and functional separation of applications in both spatial and temporal domains. The AIR architecture enforces the concept of partitioning and provides the ARINC 653 services and functionality without making significant changes to the partition RTOS kernel.

The RTEMS kernel is being used in the engineering of an AIR proof of concept prototype applied to Intel IA-32 processors and to a SPARC LEON processor core (synthetic target).

Further developments on the use of synthetic and simulation environments have defined the architecture of a multi-platform and modular ARINC 653 simulator that emulates an execution environment for ARINC 653 space applications, thus providing an allowing environment to develop space applications and verify their behaviour without having access to the final target platform and without the need for a real ARINC 653 RTOS [6].

Open Research Issues

A set of open research issues have been identified within the current context of the AIR project, as follows:

- partition and process schedulability analysis;
- application composability;
- segregation, protection and management of the input/output addressing space;
- impact of input/output on schedulability analysis;
- timeliness of synthetic processor target environments.

References

[1] Airlines electronic engineering committee (AEEC), design guidance for integrated modular avionics (ARINC specification 651). ARINC, Inc., 1991.

[2] Airlines electronic engineering committee (AEEC), avionics application software standard interface (ARINC specification 653-1). ARINC, Inc., 2003.

[3] N. Diniz and J. Rufino. ARINC 653 in space. In *Proceedings of the DASIA 2005 "DAta Systems In Aerospace" Conference*, Edinburgh, Scotland, June 2005. EUROSPACE.

[4] A. Massa. *Embedded Software Development with eCos*. Prentice-Hall, 2002. ISBN 0130354732.

[5] OAR - On-Line Applications Research Corporation. *RTEMS C Users Guide*, Feb. 2008. Edition 4.8, for RTEMS 4.8 edition.

[6] E. Pascoal, J. Rufino, T. Schoofs, and J. Windsor. AMOBA - ARINC 653 simulator for modular based space applications. In *Proceedings of the DASIA 2008 "DAta Systems In Aerospace" Conference*, Palma de Majorca, Spain, May 2008. EUROSPACE.

[7] P. Plancke and P. David. Technical note on on-board computer and data systems. European Space Technology Harmonisation, Technical Dossier on Mapping, TOS-ES/651.03/PP, ESA, Feb. 2003.

[8] J. Rufino, S. Filipe, M. Coutinho, S. Santos, and J. Windsor. ARINC 653 interface in RTEMS. In *Proceedings of the DASIA 2007 "DAta Systems In Aerospace" Conference*, Naples, Italy, June 2007. EUROSPACE.

[9] J. Rushby. Partitioning in avionics architectures: Requirements, mechanisms and assurance. Technical Report NASA CR-1999-209347, SRI International, California, USA, June 1999.

[10] J.-L. Terraillon and K. Hjortnaes. Technical note on on-board software. European Space Technology Harmonisation, Technical Dossier on Mapping, TOSE-2-DOS-1, ESA, Feb. 2003.