ARINC 653 INTERFACE IN RTEMS

José Rufino[1], Sérgio Filipe[2], Manuel Coutinho[1], Sérgio Santos[2], and James Windsor[3]

[1]*Faculdade de Ciências da Universidade de Lisboa, Campo Grande, 1749-016 Lisboa, Portugal, E-mail: ruf@di.fc.ul.pt*
[2]*Skysoft Portugal, S.A. - Av. D. João II, Lote 1.17.02, 7, 1998-025 Lisboa, Portugal, E-mail: sergio.filipe@skysoft.pt*
[3]*European Space Agency, ESA/ESTEC, 2200 AG Noordwijk, The Netherlands, E-mail: James.Windsor@esa.int*

ABSTRACT

The ARINC 653 specification is assuming a key role in the provision of a standard operating system interface for safety-critical applications in the aeronautic market and it is foreseen to acquire a similar status on the space market. The ARINC 653 application interface is independent from the underlying hardware and from a given operating system implementation. This paper describes how RTEMS, the Real-Time Executive for Multiprocessor Systems, can be adapted to offer the application interface and the functionality required by the ARINC 653 standard. The use of RTEMS is highly relevant given its qualification for on-board software of unmanned space programs.

Key words: ARINC 653; Real-Time Kernels; RTEMS; Safety-critical embedded systems.

1. INTRODUCTION

The ARINC 653 standard [1] 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 specification [1] and its concept of partitioning (spatial and temporal) are gaining increased importance and acceptance in the realm of aeronautics and space applications [2], as it happens within the scope of the European Space Agency (ESA) Technology Harmonization effort for space on-board software [3].

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 [4]. The partitioning concept makes it adequate to software with different degrees of criticality [3].

The technological interest of ESA in the ARINC 653 standard was expressed in [5]. This paper discusses the main results achieved under the scope of AIR, an innovation initiative sponsored by ESA aiming the definition and design of an ARINC 653 compliant system architecture based on RTEMS, the Real-Time Executive for Multiprocessor Systems [6], a highly modular multitasking kernel qualified for critical on-board software of unmanned space programs [7].

2. ARINC 653 FUNDAMENTAL CONCEPTS

The ARINC 653 specification is one of the most important blocks from the Integrated Modular Avionics definition [8], where the partitioning concept emerges as a way to ensure protection and functional separation between applications, usually for fault containment and ease of verification, validation and certification [1, 2]. Similar concerns do exist for space applications [3, 5, 4].

2.1. ARINC 653 System Architecture

The architecture of a standard ARINC 653 system is illustrated in Figure 1. It comprises the application software layer, with each application running in a confined context, dubbed partition in ARINC 653 terminology [1]. The application software layer may include also a set of optional system partitions intended to manage the interactions with specific hardware devices. Appropriate support from the core software layer (e.g. hardware interfacing and device drivers) is required.

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 [1]. However, a system partition may use also specific functions provided by the core software layer, thus 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 as well as inter-process synchronization and communication. Containment of pos-

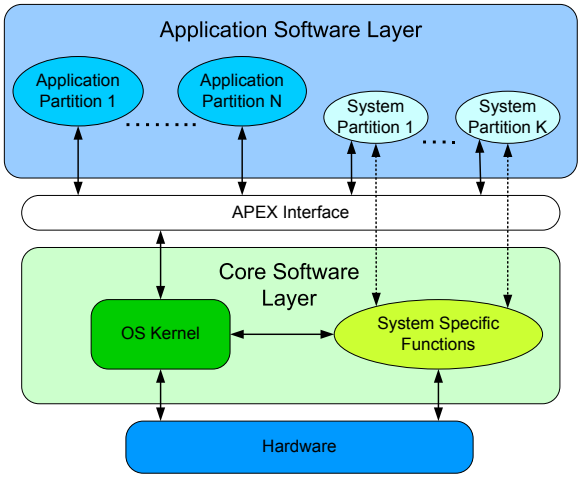

Figure 1. Standard ARINC 653 System Architecture

sible faults inside the domain of each partition must be ensured by the core software layer [1].

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 [1]. The fundamental issue regarding the migration of the HM services from the aeronautic to the space environment is the impossibility of providing human assistance on space devices. As such the HM mechanisms on space must provide much stronger recovery mechanisms as to assure a failure will not produce permanent losses.

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 required set of services is mandatory to claim strict compliance with the ARINC 653 standard [1]. Those services are grouped in the following major categories: partition and process management, time management, intra and inter-partition communications, and health monitoring.

2.5. ARINC 653 Implementations

At the present time the currently available ARINC 653 implementations are commercial and very expensive solutions provided by major companies of the aeronautic market.

The most relevant example is the Thales MACS2 OS currently installed on the new AIRBUS A380. Other examples are the LynxOS-178 ARINC 653 compliant that shall be used as the RTOS for some of the Galileo ground segment elements and Wind River Platform for Safety Critical ARINC 653 which is already in use in some unmanned aerial vehicle projects like the US Air Force test aircraft X47B Pegasus.

The AIR innovation initiative represents a first but significant step toward the usage of off-the-shelf license-free open-source RTOS kernels in the definition and design of ARINC 653 based systems. In this context, a solution integrating RTEMS is particularly interesting given its qualification for use in space on-board software developments [7].

3. THE RTEMS KERNEL

The RTEMS (Real-Time Executive for Multiprocessor Systems) is a well-known, real-time multitasking kernel, with a modular architecture, offering interesting characteristics to support the development of real-time embedded applications.

The RTEMS is designed to support applications with real-time requirements while maintaining a compatible interface with open standards. The set of features offered by RTEMS includes:

- multitasking capabilities;
- event-driven, priority-based, preemptive scheduling;
- comprehensive mechanisms for inter-task communication and synchronization;
- high degree of user configurability;
- support to homogeneous and heterogeneous multiprocessor system architectures.

Though not providing all the required mechanisms, the RTEMS functionality is in conformity with the deployment of the ARINC 653 specification, in the sense the standard specifically requires, among other features, priority-based, preemptive task scheduling and a set of

inter-task communication and synchronization capabilities.

The executive interface presented to the application is formed by grouping directives into logical sets called in RTEMS terminology, resource managers (Figure 2). These managers provide a fundamental basis for the implementation of the ARINC 653 services.

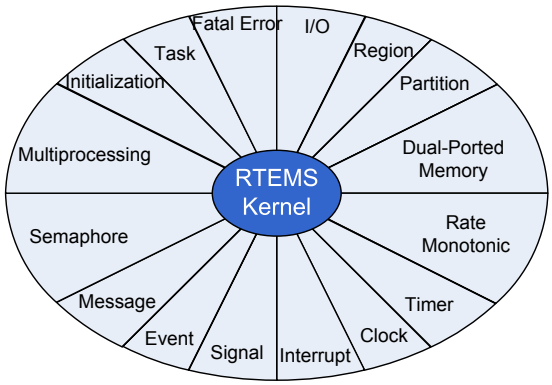

Figure 2. The RTEMS resource managers

From the point-of-view of its internal architecture, RTEMS can be viewed as a set of layered components that provide a set of services to real-time applications.

The RTEMS is considered a robust multitasking operating system kernel, supporting a wide range of processors through the encapsulation of hardware dependent features in an adaptation layer, known as board support package (BSP). This includes processor architectures as diverse as: the Intel IA-32 family [9]; SPARC architectures, such as the radiation-hardened fault-tolerant LEON and ERC32 processor cores [10].

4. AIR - ARINC 653 INTERFACE IN RTEMS

This section describes the fundamental ideas on how the Real-Time Executive for Multiprocessor Systems (RTEMS) [6] can be adapted in order to offer the application interface and the functionality required by the ARINC 653 specification [1].

4.1. 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.

The design of the AIR architecture in essence preserves the hardware and RTOS independence defined within the scope of the ARINC 653 specification [1, 4]. The AIR specific modules (cf. Figure 3) that need to be added to the RTOS kernel (e.g. RTEMS) do concern both spatial and temporal partitioning and do include the provision of 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.
- **ARINC 653 application executive interface (APEX)**, for each partition in the system, defining the services in strict conformity with the ARINC 653 standard and designed as much as possible by mapping the ARINC 653 service primitives into the native and/or POSIX primitives of RTEMS [1, 6, 11].

4.2. AIR Partitioning Support Mechanisms

This section discusses the design and the attributes of the specific mechanisms introduced in the AIR architecture to secure ARINC 653 functionality.

Design Principles

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 (as provided by the RTEMS kernel, in the example illustrated in Figure 3) for process priority-based preemptive scheduling inside each partition. This is in conformity with the ARINC 653 specification [1]. No fundamental modification is needed to the functionality of the RTOS process scheduler for its integration in the AIR system. In fact, this 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 [6], eCos [12],...).

Dependability Attributes

The use of a RTOS kernel instance per partition is also useful to guarantee a set of safety and security-related attributes, given it restricts code, data, configuration and execution context references to the confined and protected scope of a partition (cf. Figure 3).

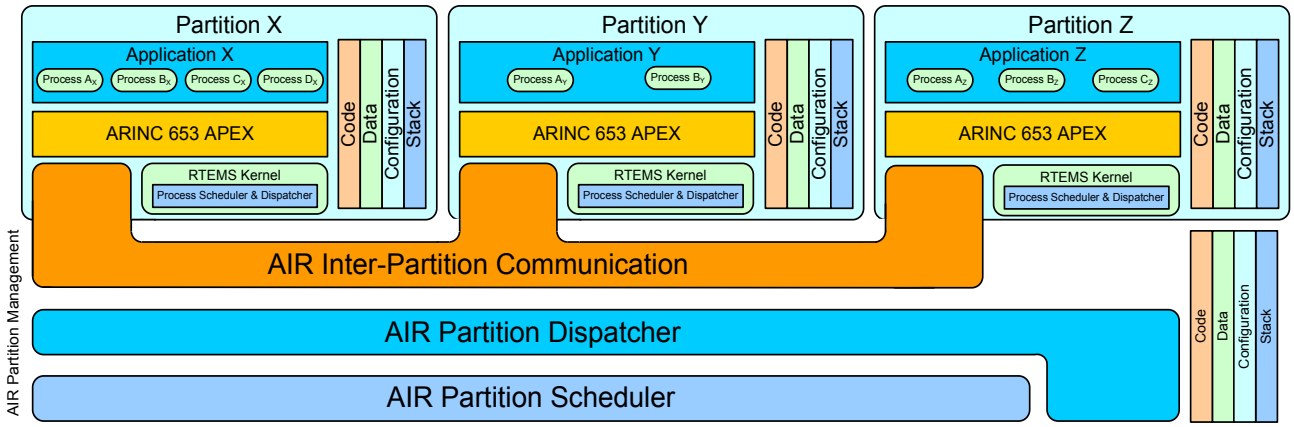

Figure 3. Overview of the AIR System Architecture

Securing Spatial Segregation

An effective support to ARINC 653 spatial partitioning requires the use of specific memory protection mechanisms usually implemented in a hardware memory management unit (MMU). This comprises not only the protection of partition memory addressing spaces but 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, in a given extent, in the SPARC LEON and ERC32 processor cores. Inside each partition, i.e. at the process level, a flat memory addressing scheme fully compliant with the ARINC 653 requirements is specified for the AIR architecture [1, 4].

Securing Temporal Segregation

The ARINC 653 standard specification [1] restricts the processing time assigned to each partition, in conformity with given configuration parameters. This means a single partition cannot monopolize the usage of the processor infrastructure thus preventing the applications in other partitions of being executed.

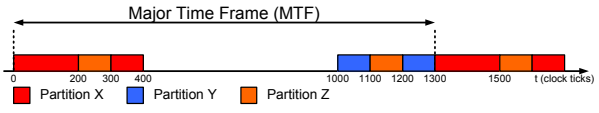

Figure 4. Example of a AIR partition scheduling

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. A Major Time Frame (MTF) of fixed duration, defined off-line, is periodically repeated throughout runtime operation [1]. An example of temporal partitioning is provided in Figure 4.

In the AIR architecture, temporal segregation is ensured by the AIR partition scheduler.

Additional Dependability Mechanisms

The AIR architecture also includes a set of advanced control mechanisms related with both spatial and temporal partitioning, such as: interrupt management and control of interrupt servicing on a per partition basis; optimal servicing of global system clock interrupts, allowing the use of precise timing references. In addition, the AIR architecture may include a set of extensions to the ARINC 653 standard intended to enhance system dependability and determinism. For example, the utilization of specific timeliness control mechanisms, enforcing bounded processing times: in the servicing of external/internal asynchronous events (e.g. interrupts) [13]; in the management of ARINC 653 services.

4.3. AIR APEX Interface

In strict conformity with the ARINC 653 specification [1], the AIR architecture furnishes at the application executive (APEX) interface the following set of services:

- partition management;
- process management;
- time management services;
- inter-partition communication services;
- intra-partition communication services;
- health monitoring.

Most of these services can be implemented by mapping the ARINC 653 functionality into the native and/or POSIX primitives of RTEMS [6, 11]. The exceptions do concern:

- some details of partition management (e.g. partition status), which must be furnished by the AIR Partition scheduling/dispatcher components;
- inter-partition communication services, which are almost directly mapped into the functionality provided by the AIR inter-partition communication module.

In respect to health monitoring functions and in contrast with the avionics requirements where maintenance is provided when the aircraft is on ground and is mostly done off-line, in the space environment maintenance must be provided during system operation, defining a major constraint to system requirements. Hence, on the space environment, the health monitoring services should have major adaptations and added complexity as human assistance is probably not available.

5. 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 [1]. The AIR solution is hardware and operating system independent and it exploits the usage of conventional off-the-shelf license-free open-source RTOS kernels, such as RTEMS [6], a real-time multitasking kernel qualified for use in space on-board software developments [7].

The paper discusses the definition and design of the AIR architecture and how its fundamental 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 all the ARINC 653 services and functionally without making significant changes to the bare RTOS kernel. RTEMS is being used in the engineering of an AIR proof of concept prototype applied to Intel IA-32 processors and (possibly) to a SPARC LEON/ERC32 processor core (synthetic target).

REFERENCES

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

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

[3] 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, February 2003.

[4] 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.

[5] 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, February 2003.

[6] OAR - On-Line Applications Research Corporation. *RTEMS C Users Guide*, August 2003. Edition 4.6.6, for RTEMS 4.6.6 edition.

[7] J. Seronie-Vivien and C. Cantenot. RTEMS operating system qualification. In *Proceedings of the DASIA 2005 "DAta Systems In Aerospace" Conference*, Edinburgh, Scotland, June 2005. EUROSPACE.

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

[9] OAR - On-Line Applications Research Corporation. *RTEMS Intel i386 Applications Supplement*, August 2003. Edition 4.6.6, for RTEMS 4.6.6 edition.

[10] OAR - On-Line Applications Research Corporation. *RTEMS SPARC Applications Supplement*, August 2003. Edition 4.6.6, for RTEMS 4.6.6 edition.

[11] OAR - On-Line Applications Research Corporation. *RTEMS POSIX API Users Guide*, August 2003. Edition 4.6.6, for RTEMS 4.6.6 edition.

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

[13] M. Coutinho, J. Rufino, and C. Almeida. Control of event handling timeliness in RTEMS. In *Proceedings of the 17th IASTED International Conference on Paralel and Distributed Computing Systems - PDCS 2005*, Phoenix, Arizona, USA, November 2005. IASTED.

[14] IEEE Std 1003.1 - standard for information technology portable operating system interface (POSIX) system interfaces, 2004.

[15] IEEE Std 1003.13 - standard for information technology - standardized application environment profile (AEP) - POSIX real-time and embedded application support, 2003.