Title: Configuration and Scheduling tools for TSP systems based on XtratuM.

Authors: I. Ripoll, M. Masmano, V. Brocal, S. Peiró, P. Balbastre, A. Crespo

Affiliation: Instituto de Informática Industrial, Universidad Politécnica de Valencia, Spain

Authors: P. Arberet, J.J. Metge

Affiliation: CNES, France

Publication: DAta Systems In Aerospace DASIA 2010. Budapest.

CONFIGURATION AND SCHEDULING TOOLS FOR TSP SYSTEMS BASED ON XTRATUM

I. Ripoll[1], M. Masmano[1], V. Brocal[1], S. Peiró[1], P. Balbastre[1], A. Crespo[1], P. Arberet[2], and J.J. Metge[2]

[1]*Instituto de Informática Industrial, Universidad Politécnica de Valencia, Spain*
[2]*CNES, France*

ABSTRACT

Hypervisor is a promising technology to build partitioned systems. However, it has to be adapted and customized to the requirements of the target application. Hypervisors are small software layers which can be designed to meet real-time and security properties. Its correctness can be sufficient to ensure the security of the system as a whole or, at least, the security of a set of trusted partitions.

Hypervisor technology provides execution environments to build partitions which contain the applications. Several aspects arise from partitioned systems: the new roles and functions of the different teams involved in the development and the enviroments to develop the applications. In this paper we discuss the roles and functions when using XtratuM as virtualisation platform and the guest operating systems available to develop the applications.

On the other hand, a critical point is the cost of the virtualisation in terms of overhead, memory and complexity to build a scheduling plan. This paper analysises these costs for partitioned systems built on XtratuM.

Key words: Partitioning systems, hypervisors,microkernels.

1. INTRODUCTION

The availability of new processors for embedded applications has raised new possibilities for these applications. Embedded applications now have more functionalities and, as a consequence, are more complex. There is growing interest in enabling multiple applications to share a single processor and memory. To facilitate such a model, the execution time and memory space of each application must be protected from other applications in the system.

Partitioned software architectures represent the future of secure systems. They have evolved to fulfill security and avionics requirements where predictability is extremely important. The separation kernel proposed in[8] established a combination of hardware and software to enable the performance of multiple functions on a common set of physical resources without interference. The MILS (Multiple Independent Levels of Security and Safety) initiative is a joint research effort between academia, industry, and government to develop and implement a high-assurance, real-time architecture for embedded systems. The ARINC-653 [1] standard uses these principles to define a baseline operating environment for application software used within Integrated Modular Avionics (IMA) and based on partitioned architecture. a separation kernel.

The Integrated Modular Avionics (IMA) was the solution that allowed the Aeronautic Industry to integrate new functionalities maintaining the level of complexity and efficiency. The main goal was to define an architecture that captures and handles faults at the different levels and permits the parallel application development. Currently, the ESA (European Space Agency) has initiated a program for the development of Time and Space Partitioning (TSP) solutions in space avionics.

Some of the potential benefits identified of TSP are [9]:

- Provide a clear model of the development process by enforcing the integrator and developer roles.
- Facilitate the integrator activities providing a set of assistance tools.
- Provide a common target for developing applications.
- Increase the efficiency of the software validation and qualification process.

Three roles are clearly identified in the TSP development process: the integrator, the partition supplier and the platform provider. In the TSP development process, the integrator plays and fundamental role. The main functions can be seen as:

- Platform specification: selection of the appropriated target.
- System definition: specification of the partitions and the resources associated to each one.
- Software platform configuration and deployment: the integrator should produce an execution environment to be used by the partition suppliers.

- Design of the scheduling plan: it implies the consideration of the temporal constraints of the partition's activities and its dependencies.
- Integration of the partitions in the platform: the integrator receives the partition source code or binaries and the hardware platform and integrates all the software in the final platform.
- System validation: the integrator validates the TSP development.

Most of these aspects have been considered in the design of XtratuM and its associated tools. In this paper we present the principles and relevant tools designed around XtratuM to assist the integrator in the different phases of the development.

2. XTRATUM OVERVIEW

XtratuM [**?**] is a type 1 hypervisor that uses para-virtualization. The para-virtualized operations are as close to the hardware as possible. Therefore porting an operating system that already works on the native system is a simple task: replace some parts of the operating system HAL (Hardware Abstraction Layer) with the corresponding hypercalls.

Currently, version 2.2 is being used by CNES as a TSP-based solution for building highly generic and reusable on-board payload software for space applications [2]. TSP (time and space partitioning) based architecture has been identified as the best solution to ease and secure reuse. It enables a major decoupling of the generic features that are being developed, validated, and maintained in mission-specific data processing [3].

In the framework of the ESA Project AO5829 "Securely Partitioning Spacecraft Computing Resources", XtratuM has been adapted to LEON3 processors with MMU. It corresponds to the version 3.1.

In a hypervisor, and in particular in XtratuM, a partition is a *virtual computer* rather than a group of strongly isolated processes. When multi-threading (or tasking) support is needed in a partition, then an operating system or a run-time support library has to provide it. In fact, it is possible to run a different operating system on each XtratuM partition.

XtratuM was designed to meet safety critical real-time requirements. The most relevant features are:

- Bare hypervisor.
- Employs para-virtualisation techniques.
- An hypervisor designed for embedded systems: some devices can be directly managed by a designated partition.
- Strong temporal isolation: fixed cyclic scheduler.
- Strong spatial isolation: all partitions are executed in processor user mode, and do not share memory.
- Fine grain hardware resource allocation via a configuration file.
- Robust communication mechanisms (XtratuM sampling and queuing ports).

3. EXECUTION ENVIRONMENTS

A partition is an execution environment managed by the hypervisor which uses the virtualised services. Each partition consists of one or more concurrent processes (implemented by the operating system of each partition), sharing access to processor resources based upon the requirements of the application.

In order to develop partition several development environments are supported. Next figure shows the relation between XtratuM and these development environments.

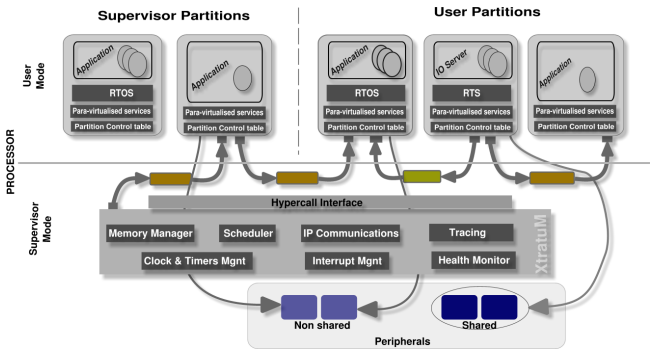

Figure 1. XtratuM and its development environments.

3.1. XAL partition development environment

The XAL (XtratuM Abstraction Layer) is a minimal partition development support for the development of C programs directly on top of XtratuM. This abstraction layer provides the basic and minimal services to setup a basic "C" execution environment. XAL is only useful for those partitions that are mono-thread and do not need an operating systems. All services provided the XtratuM hypercalls are available to the application.

The XAL development environment is integrated by a library with the services and the minimal runtime to execute the partition and handle the virtualised interrupts.

The services provided by XAL are:

3.2. RTEMS personality RTOS

RTEMS is a free open source real-time operating system (RTOS) designed for embedded systems and adopted by the ESA for space applications and space missions. RTEMS (4.8.1) has been ported on top of XtratuM.

This development environment permit to develop RTEMS/XM2 applications in a partitioned system. This execution environment facilitates legacy code execution of RTEMS applications. Any RTEMS application could be executed without modification in this environment.

Group of services	Hypercalls	Partition type
Clock management	get clock; define timers	Normal
IRQ Management	enable / disable IRQs, mask / unmaks IRQs	Normal
IP Communication	create ports; read / receive / write / send messages	Normal
IO management	read/write IO	Normal
Partition management	mode change, halt / reset / resume / suspend / shutdown partitions (system)	System
Health monitoring management	read / seek / status HM events	System
Audit facilities	read / status	System

Table 1. XAL services.

The porting defines a new BSP leon2xm or leon3xm depending on the XtratuM version. This BSP para-virtualises the RTEMS services. It includes the interrupt management, clock and timers management.

It permits to execute legacy code based on RTEMS without modifications on top of XtratuM. One of the most critical aspects when executiing RTEMS on top of a partitioned system is the clock management. RTEMS assumes a periodic interrupt (clock tick) to deal with clock and timers management. The RTEMS runtime is based on this clock tick. In a partitioned system, there are some time intervals in which the RTEMS partition is not under execution and, as consequence, clock ticks are not updated. When the partition is scheduled again, it has to consider this behavior. It involves the accummulation of ticks at the begin of the slot.

3.3. LithOS personality RTOS

LithOS is a para-virtualised guest operating system which uses the services provided by XtratuM to offer the complete ARINC-653 APEX to the applications.

The LithOS architecture is shown in the next figure:

It implements the services defined in the standard to handle processes, sampling and queuing ports, blackboards, buffers, semaphores, events and health monitoring.. The ARINC-653 services are provided directly or indirectly by LithOS. Directly means that services are implemented by LithOS. Indirectly refers to the adaptation of services provided by XtratuM to the standard.

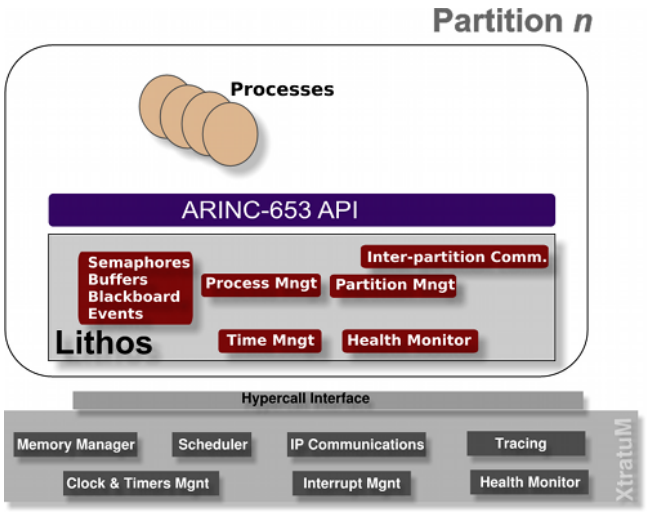

Figure 2. LithOS architecture.

Service	Provided by	Comments
Partition management	XtratuM	Adapted by LithOS
Interpartition Comm.	XtratuM	Propagated by LithOS
Time management	XtratuM	Adapted by LithOS
Health monitor	XtratuM & LithOS	LithOS implements the partition HM
Process management	LithOS	Fully implemented by LithOS
Intra-partition communication	LithOS	Fully implemented by LithOS

Table 2. List of groups of hypercalls.

3.4. Performance evaluation

In order to compare the performance and memory footprint of the three execution environments, an application that defines a background thread (increasing a counter) which is preempted by a more priority thread (an interrupt handler in XAL environment) has been defined. The number of counts in a second under different number of interrupts permits to evaluate the impact of the thread context switch.

Next picture shows the performance loss depending on the period of the interrupt. Note that RTEMS is configured with a clock tick of 10 milliseconds and, as consequence, there is not possible to execute it at intervals lower than the value.

As it can be seen in the figure, the performance lost when the slot duration is 10 milliseconds is 0.8%, 1.6% and 4.0% in the XAL, LithOS and RTEMS environments, respectively.

With respect to the footprint. Next table shows the foot-

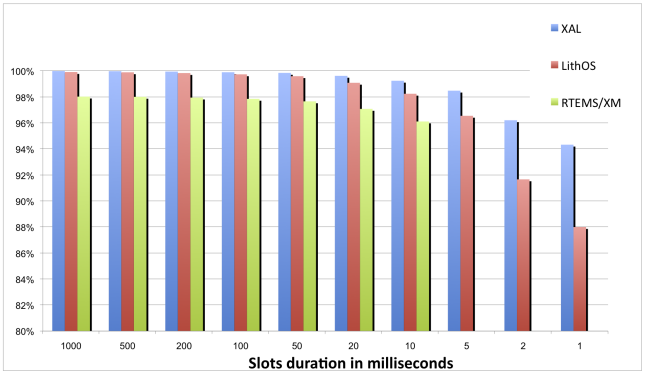

Figure 3. Performance evaluation

prints of XtratuM, XAL, LithOS and RTEMS.

Component	text	data	bss
XtratuM	61.4K	6.5K	68.2K
XAL	22.7K	8.0K	0.7K
LithOS	43.1K	8.2K	46.0K
RTEMS	109.3K	3.2K	2.9K

Table 3. Memory footprint.

4. DEPLOYMENT TOOLS

Deployment tools are related to the set of tools available for the integrator to build a distributable development environment for the partition's developers. The activities of the integrator are:

- Configuration of XtratuM for the appropriated target.
- Compilation and generation of a binary distribution of XtratuM to be distributed to the partition developers.
- Definition of the configuration data with the resource allocation for all partitions
- Integration of the partitions in the platform: the integrator receives the partition source code or binaries and the hardware platform and integrates all the software in the final platform.
- System validation: the integrator validates the TSP development.

To deal with these activities, the following tools are provided:

- XtratuM configuration: Source code is configured to the appropriated target.
- xmcparser: Processes a XML configuration file and generate a configuration file in binary format
- xmpack: Generates a container including xmcore, configuration files and partitions
- rswbuild: Takes a container generated by xmpack and includes the resident software
- distro-build: Builds a deployment of the XtratuM binaries and libraries (previously compiled for a specific target) in a . It includes the development enviroments to develop the applications using any of the available (XAL, RTEMS, Lithos).

Once the binary distribution is generated and distributed to the partition developers, they can install the binary distribution obtaining a development environment ready to develop partitions based on any of the execution environments.

5. CONFIGURATION AND SCHEDULING TOOL

Hierarchical scheduling has recently been used to provide temporal isolation to partitioned-based systems. Although has been a topic of research interest in recent years, hierarchical scheduling is not a mature technique. Some important aspects have to be considered: two levels of scheduling coexist. Each level is scheduled with its own scheduling policy. The worst case response scenario does not occur in a synchronized system, which makes the schedulability analysis quite complex. Recent works ([6, 4]) give an exact value of the worst case response time of a task scheduled under fixed-priority policy in both levels, but it is still an open issue to provide exact schedulability analysis for other scheduling policies.

Xoncrete is an analysis tool that performs the schedulability analysis of a system based on XtratuM. It also allows to tune and optimise the scheduling plan taking into account the temporal constraints of the partition's activities and its dependencies.

The user provides the temporal system model that it is the input of Xoncrete tool. The classical scheduling model make up of tasks (with period, deadline and execution time attributes) does not capture properly the operation of partitioned complex systems. A new model has to be considered. The concept of end-to-end flow [**?**] fits with the main characteristics of a partitioned-based system. This way, the temporal model considered has the following elements:

- End to End Flow (ETEF): It is a description of the temporal behaviour of the workload. It consists of a sequence of tasks with temporal attributes and it is the element that has periodic behaviour and denes the temporal restrictions. The tasks that are part of an ETEF can belong to different partitions that is, an end-to-end-ow is not linked with an specic partition.
- Partition: A container where the tasks are executed. Task are directly scheduled by the partition itself. Partitions are used to dene spatial isolation, and where the tasks are executed/scheduled.

- Task: The elemental execution unit. A task is executed by a partition and belongs to it.

5.1. Scheduling analysis

Once the system parameters and timing requirements hasve been introduced and validated, the system can be analysed and the scheduling plan can be generated.

The analysis and plan generation is divided in three steps:

1. Tune ETEF parameters. Compute the MAF, compute the periods, and select the offsets. Figure **??** shows the analysis window, previously to the MAF calculation. The user is allowed to modify the ranges of the periods, while the tool computes the MAF in an iterative process.

   The result from this step is an exact, i.e. fixed periods and fixed offsets, workload (ETEFs) specification which can be used to generate the plan in the next step.

2. Generate schedule. The plan is generated and statistical information jointly with a graphical representation of the plan is presented to the user (figure **??**).

   The plan is generated using well known scheduling techniques:

   - EDF as the base scheduling policy [7].
   - Minor modifications to the EDF criteria in order to reduce the number of partition context switches.
   - SRP to access mutual exclusion resources [5].

   The goal is to generate a short cyclic plan (with less slots) which can be used as an ARINC 653 partition plan. The tool also checks if the tasks of each partition can be scheduled by the local scheduler. In the case that the local scheduler is not able to schedule it (because there is not a valid priority assignment) then the plan that makes it schedulable is provided.

3. Tune schedule. The user can make minor local changes to the generated plan.

   This step is still under development.

The results of a first prototype of this tool are shown in Figures 4 and 5.

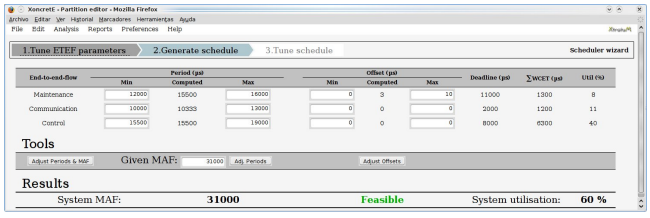

Figure 4. Xoncrete configuration window

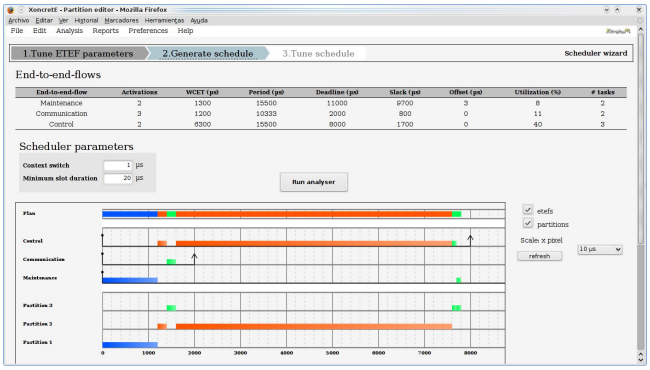

Figure 5. Xoncrete analysis window

6. CONCLUSIONS

XtratuM was the first implementation of an hypervisor for the LEON2 processor. The version of XtratuM 2.2.2 is currently used by CNES as a TSP-based solution for building highly generic and reusable on-board payload software for space applications.

In this paper we detailed the tools that have been designed to assist the integrator in his procedures to define the configuration of the system, design a scheduling plan and generate a developing environment to be distributed to the partition suppliers.

An important point is the scheduling of a partitioned system. Rather than implementing a complex tool (been able to use a wide variety of scheduling policies and complex systems models), Xoncrete is a tool designed to assist the integrator to specify a scheduling plan for a partitioned system.

REFERENCES

[1] *Avionics Application Software Standard Interface (ARINC-653)*, March 1996. Airlines Electronic Eng. Committee.

[2] P. Arberet, J.-J. Metge, O. Gras, and A. Crespo. TSP-based generic payload on-board software. In *DASIA 2009. DAta Systems In Aerospace.*, May. Istanbul 2009.

[3] P. Arberet and J. Miro. IMA for space : status and considerations. In *ERTS 2008. Embedded Real-Time Software.*, Jannuary. Toulouse. France 2008.

[4] Patricia Balbastre, Ismael Ripoll, and Alfons Crespo. Exact response time analysis of hierarchical fixed-priority scheduling. In *Proceedings of 15th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, August 2009.

[5] Sanjoy K. Baruah. Resource sharing in edf-scheduled systems: A closer look. In *RTSS '06: Proceedings of the 27th IEEE International Real-Time Systems Symposium*, pages 379–387, Washington, DC, USA, 2006. IEEE Computer Society.

[6] R. I. Davis and A. Burns. Hierarchical fixed priority pre-emptive scheduling. In *Proceedings of 26th IEEE International Real-Time Systems Symposium*, pages 389–398, 2005.

[7] E. W. Giering, III and T. P. Baker. A tool for the deterministic scheduling of real-time programs implemented as periodic ada tasks. In *SETA2: Proceedings of the second international symposium on Environments and tools for Ada*, pages 54–73, New York, NY, USA, 1994. ACM.

[8] John Rushby. Design and verification of secure systems. volume 15, pages 12–21, Pacific Grove, California, Dec 1981.

[9] J. Windsor and K. Hjortnaes. Time and space partitioning in spacecraft avionics. In *IEEE Conference on Space Mission Challenges for Information Technology*, July 19-23. Pasadena (USA) 2009.