Introducing a new temporal partitioning scheme to AUTOSAR OS

29th Oct., 2015

Hiroaki TAKADA

Professor, Inst. of Innovation for Future Society, Nagoya Univ.
Executive Director and Professor,
Center for Embedded Computing Systems, Nagoya Univ.
Chairman, TOPPERS Project

Email: hiro@ertl.jp  URL: http://www.ertl.jp/~hiro/
Self Introduction – Hiroaki Takada

Current Positions

▶ Professor, Institute of Innovation for Future Society, Nagoya University
▶ Executive Director, Center for Embedded Computing Systems (NCES), Nagoya University
▶ Chairman, TOPPERS Project
▶ Chairman and CTO, APTJ Co., Ltd.
▶ Academic Member, JASPAR

and many others

Major Research Topics

▶ Real-time operating systems for embedded systems
▶ Real-time scheduling and analysis
▶ Electronic system-level design of embedded systems
▶ Automotive embedded systems, functional safety
AGENDA

Software Partitioning

- Necessity
- Concept
- Partitioning Shared Resources

Temporal Partitioning Schemes

- Temporal Partitioning in AUTOSAR
- Timing Protection Functions of AUTOSAR OS SC2
- Temporal Protection Scheme in ARINC 653
- Proposed Temporal Protection Scheme

Our Activities

- AP Consortium and TOPPERS Project
- Other Joint Activities

Conclusion
Software Partitioning – Necessity

Freedom from Interference (FFI)

- Defined to be “absence of cascading failures between two or more elements that could lead to the violation of a safety requirement” (ISO 26262-1)
- Required when SW-Cs with different ASILs coexist in an MCU
- A necessary condition of independence, which is required between ASIL-decomposed SW-Cs

Software Partitioning

- Avoidance of cascading failures by the separation between SW-Cs
  - Propagation of a failure can be stopped by separating SW-Cs.
- One of the realization approaches of FFI
**Software Partitioning – Concept**

**Paths of Failure Propagation**
- A failure of a SW-C is propagated to other SW-Cs through:
  - communication channel
  - shared resources
- Software partitioning can stop the propagation of failures through shared resources.

**Possible Shared Resources**
- Processor
- Main memory
- Peripheral devices
  - I/O port
  - Network controller
  - Non-volatile memory
  *and others*
Partitioning Shared Resources

Processor
- Processor is shared by SW-Cs with time-division scheme (in a uniprocessor system).
- How to achieve temporal partitioning? [today’s topic]

Main Memory
- Main memory is shared by SW-Cs with space-division scheme (without dynamic memory management).
- Memory partitioning can be achieved with memory protection functions of AUTOSAR OS SC3.

Peripheral Devices
- Peripheral devices are usually managed by BSW modules (services layer and/or MCAL).
- Those BSW modules should be designed to achieve the partitioning of peripheral devices. [future work]
Temporal Partitioning in AUTOSAR

No Temporal Partitioning Functions in Current AUTOSAR

▶ In “Overview of Functional Safety Measures in AUTOSAR”, the term “temporal/time/timing partitioning” is not used.
  ▶ The term “memory partitioning” is used.
  ▶ “Timing Monitoring” is discussed in the document, instead.

Timing Protection Functions of AUTOSAR OS SC2

▶ With the SC2 timing protection functions, temporal partitioning can be achieved.
  ▶ The schedulability of the whole system is analyzed with the rate monotonic analysis (RMA) technique.
  ▶ Timing parameters of each task/ISR2 used in the analysis are enforced with AUTOSAR OS SC2.
▶ This approach has some problems (described in the next slide) and is not efficient.
Problems of SC2 Timing Protection Functions

Too Small Unit for Timing Protection
- Unit of protection should be partition, rather than tasks and ISRs.
- This causes following two problems.

Pessimistic Schedulability Analysis
- The max. execution time of OS critical sections should be added to the max. exec. time of each task/ISR.

Lack of Mode Change Mechanism
- This problem is serious when a partition is terminated or restarted (how to schedule the restart task?).

Timing Protection Violation within a Trusted Function

Complicated Specification and Implementation
- e.g.) DisableAllInterrupts does not disable all interrupts.
Motivations of New Temporal Partitioning Scheme

Increasing Necessity of Software Partitioning

- For efficient support for functional safety, software partitioning is important for saving software development and verification cost.
- A key technology for ECU integration

Lack of Good Partitioning Standard

- Timing protection functions of AUTOSAR OS SC2 are not suitable for software partitioning.
- ARINC 653 (a standard for avionics systems) approach is too restrictive for automotive systems.

Necessity of a Standard

- We need a common partitioning scheme applicable to different RTOS.
Requirements on Partitioning Functions

More Strict Partitioning for Functional Safety

- Supporting the “freedom from interference between software elements” requirement of ISO 26262.
- Similar requirement is [SRS_Os_11008].

Partition Termination and Restart

- A faulty partition can be safety terminated or restarted [SRS_Os_11022] [SRS_Os_11023].

Resource Sharing and Protection

- When multiple partitions share a resource/device (eg. network controller), a fault in a partition should not propagate to others through the shared resource/device.

Reasonable Overhead
**Temporal Protection Scheme in ARINC 653**

**What is ARINC 653?**
- “Avionics Application Standard Software Interface”
- A software specification for space and time partitioning in safety-critical avionics RTOS, allowing the hosting of multiple applications of different levels on the same hardware.

**Overview of the ARINC 653 Scheme**
- The system cycle is divided into several time windows.
- Each time window is assigned to a partition. The partition is executed within the assigned time windows.

![Diagram showing the system cycle divided into time windows](image-url)
Hierarchical Scheduling

- Within a time window, tasks belonging to the partition are executed in priority order.

Advantages

- No temporal influence between partitions
  - Each SW-C can be verified independently.
- Simple and easy to understand
  - It is easy to demonstrate the FFI between SW-Cs.

Drawback

- Interrupt latency becomes long because an interrupt is accepted only within the time window assigned to the partition it belongs to.
  - We think this scheme is too restrictive for automotive systems.
Proposed Temporal Partitioning Scheme

Overview of the Proposed Scheme

- An extension of the ARINC 653 scheme
- The idle window is placed at the last of the system cycle and is not assigned to a partition.
- Interrupts are handled in system-level and are accepted/executed regardless of the time window.
  - ISRs must be implemented in highest ASIL.
Our Activities

Detailed Specification and Implementation

- Detailed specification and implementation were developed by the AP Consortium.
  - The extended OS specification with the proposed temporal protection scheme was defined.
  - TOPPERS/ATK2, an open implementation of AUTOSAR OS, was extended to support the scheme.

Distribution Status

- The version without memory protection functions (ATK2-SC1-TP) are distributed from the TOPPERS Project.
  - http://www.toppers.jp/atk2-download.html
- The version with memory protection functions (ATK2-SC3-TP) will be distributed soon.
AP (Automotive Platform) Consortium

Organization of the Project

▶ A collaborative research consortium led by Center for Embedded Computing Systems (NCES), Nagoya Univ.
  ▶ 26 member companies (incl. observer members)
  ▶ Member company provide resources (engineers and/or expense) to the consortium.

Project Overview

▶ Develop a high-quality AUTOSAR-based software platform
  ▶ We focus only on key components, incl. OS, CAN communication stack, watchdog stack, and RTE generator.
  ▶ The developed software will be opened.
    ▶ Anyone can adapt/optimize it to his own requirements.
    ! Some of the development results are not opened.
Objectives of the Project

▶ To develop various open-source software for embedded systems including RTOS and to promote their use

*Building a widely used open-source OS as Linux in the area of embedded systems!*

Main Activities of the Project

▶ Building a definitive μITRON-conformant RTOS
▶ Developing a next generation RTOS technology
▶ Developing software development technology and tools for embedded systems
▶ Fostering Embedded System Engineers

Project Members

▶ ~100 companies, ~80 individual engineers, ~20 university/college laboratories and non-profit organizations
Application Example of TOPPERS OS

- Kizashi (SUZUKI)
- Skyline Hybrid (NISSAN)
- H-IIB (JAXA)
- ASTRO-H (JAXA) *under development*
- OSP-P300 (Okuma)
- SoftBank 945SH (SHARP)
- IPSiO GX e3300 (Ricoh)
- UA-101 (Roland)
- PM-A970 (EPSON)

Provided by: JAXA, Illustration: Akihiko Kojima
Other Related Activities

WITZ Corporation

- WITZ implemented the proposed temporal partitioning scheme in ParOS (TOPPERS/PARK). ParOS gets concept report on functional safety standard from TUV SUD.

OTSL Inc.

- OTSL, a Japanese distributor of SYSGO PikeOS, implemented the proposed temporal partitioning scheme in PikeOS with SYSGO.
  - *They call it “Automotive Time Partitioning Scheduler”.*

TOPPERS Project

- An implementation of the proposed temporal partitioning scheme to an ITRON-based RTOS is also in progress.
Conclusion

Future Work

▶ Partitioning of peripheral devices
  ▶ Investigation should be conducted for each (potentially shared) peripheral device or for each BSW module.
  ▶ Application of the proposed temporal partitioning scheme to actual ECU development.

Proposal to AUTOSAR

▶ If AUTOSAR partners are interested in the proposed temporal partitioning scheme, we are willing to propose it to AUTOSAR.