parMERASA

Multi-Core Execution of Parallelised Hard Real-time Applications Supporting Analysability

D4.10
Final parMERASA Multi-core System Software

Nature:	**P - Prototype**
Dissemination Level:	**PU - Public**
Due date of deliverable:	**30/09/2014**
Actual submission:	**30/09/2014**
Responsible beneficiary:	**UAU**
Responsible person:	**Theo Ungerer**

Grant Agreement number:	FP7-287519
Project acronym:	parMERASA
Project title:	Multi-Core Execution of Parallelised Hard Real-Time Applications Supporting Analysability
Project website address:	http://www.parmerasa.eu
Funding Scheme:	STREP SEVENTH FRAMEWORK PROGRAMME THEME ICT – 2011.3.4 Computing Systems
Date of latest version of Annex I against which the assessment will be made:	June 20, 2012
Project start:	October 1, 2011
Duration:	36 month

Project coordinator name, title and organisation: Prof. Dr. Theo Ungerer, University of Augsburg

Tel: + 49-821-598-2350 Fax: + 49-821-598-2359 Email: ungerer@informatik.uni-augsburg.de

Release Approval

Name	Role	Date
Christian Bradatsch	Author	2014-09-15
Theo Ungerer	WP leader	2014-09-15
Theo Ungerer	Coordinator	2014-09-15

Deliverable Summary

Deliverable 4.10 "Final parMERASA Multi-core System Software" concerns the prototype implementation of the final parMERASA system software, which supports the integration of the three different domain RTEs (runtime environments), namely automotive, avionic, and construction machinery.

This deliverable comprises the multi-core Kernel Library implementation providing basic common real-time operating system (RTOS) kernel functions. Furthermore, this document covers the implementations of the tiny automotive RTE, the tiny avionic RTE, and the construction machinery middleware layer.

Deliverable 4.10 summarises the work done in task 4.8.

Task description of T4.8 (m31 :: 6m):
Support of Evaluations of Pilot Studies and Open Source SW (see DoW, sect. 1.3.3, p. 48)

- Continuous support, debugging and fine tuning with input from experience of evaluations.
- All evaluation results are reported in WP2 deliverable D2.6.
- The final parMERASA system software will be made publicly available under an Open Source license at the end of parMERASA project.

Conclusion of task 4.8:

All targets of task 4.8 have been reached.

Table of Contents

1 Introduction

Work package 4 developed a common system architecture for a many-core processor suitable for applications of the automotive, avionic, and construction machinery domain. Furthermore, the system architecture supports the execution of parallelised hard real-time applications and their WCET analysability.

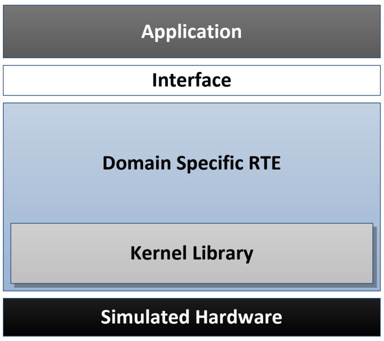

Figure 1: General overview of the parMERASA system architecture

Figure 1 gives a general overview of the parMERASA system architecture. The *simulated hardware* provides the execution platform for all domains. The *application* resides on top. In between, the domain specific system software is placed. It consists of the *domain specific runtime environment*, *interface*, and the *Kernel Library*.

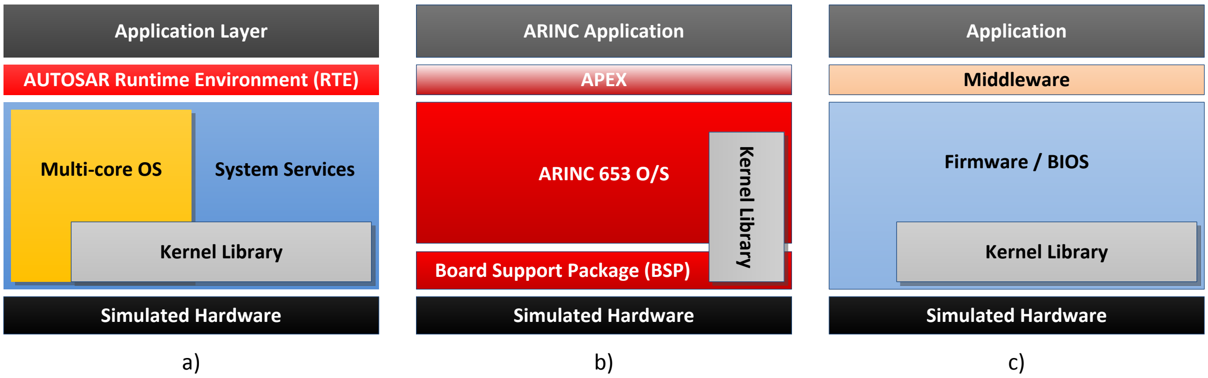

Figure 2: Comparison of the a) automotive, b) avionic, and c) construction machinery domain architectures integrating the Kernel Library

Figure 2 compares the system architectures of the a) automotive, b) avionic, and c) construction machinery domain. The basic structure of each domain architecture is similar to the general architecture shown in Figure 1.). For each application domain, a separated RTE is implemented to provide a reasonable subset of the application programming interface (API) defined in the domain specific standards (e.g. AUTOSAR, ARINC 653). In the automotive domain we call the domain specific

RTE tiny automotive RTE and in the avionic domain tiny avionic RTE. The Kernel Library is common to all RTEs and provides basic functionalities shared among the RTE implementations. In the automotive RTE, the Kernel Library further abstracts the hardware from the system services and the multi-core OS. The OS additionally has direct access to the hardware for some services. Likewise, in the avionic RTE the Kernel Library abstracts the hardware from the ARINC 653 O/S. The board support package provides access to specific hardware peripherals, which are not covered by the Kernel Library. The same accounts for the construction machinery RTE. The Kernel Library abstracts large chunks of the hardware. Only for some hardware peripherals direct access is needed from the Firmware. The complete specification of the Kernel Library can be found in [1].

2 Multi-core System Software

The main business in task 4.8 was the continuous support and fine tuning of the multi-core system software including the three different domain runtime environments. The following sections give a short introduction on installing the Kernel Library together with the tiny automotive RTE on top of the parMERASA simulator. An example program is also provided and explained to demonstrate the supplied functionalities.

2.1 Installation

To install the Multi-core system software and example program the corresponding parMERASA simulator has to be installed (see deliverable D5.7). Extract the content of ***kernel_lib.zip*** file to folder ***path_to_parMERASA_simulator/sim/parMERASA_sim***. Open a terminal console and switch to folder ***path_to_parMERASA_simulator/sim/parMERASA_sim***. Be sure that all environments variables have been set correctly. This can be tested by typing ***echo $SOCLIB*** in the console. The output shall display the path to your soclib installation directory. Afterwards the package has to be compiled by typing ***make all*** in the console and starting the example program with ***./execute_sim param_file***.

2.2 Sample Application

The example application demonstrates the usage of the buffer mechanism provided by the tiny automotive RTE. The DNDE application uses the buffers to exploit the time triggered execution model. The syntax of the buffer functions is similar to the implicit read and write of the API of AUTOSAR Specification of RTE [1]. *Rte_IRead* and *Rte_IWrite* are extended by the parameter `<publication time>`. For each data value written to the buffer a corresponding, user selectable publication time stamp is saved. When reading a data value from the buffer, the passed publication time indicates which data value has to be read.

The sample application is executed on eight cores. Core zero is responsible for interrupt handling. Core one and core two exchange a primitive data type value using the buffer mechanism. Depending on the publication time and the buffer size, the values read at core two differ. The same accounts for the buffers used on core four/five and core six/seven. Core four and five demonstrate the buffer mechanism when using structure type value, and core six and seven when using a data array.

Deliverable 2.6 comprises how the buffers are exactly integrated in the engine management system application.

List of Figures/Tables

List of References

[1] C. Bradatsch und F. Kluge, „parMERASA Multi-core RTOS Kernel," University of Augsburg, Augsburg, Germany, 2013.

[2] AUTOSAR, „Specification of RTE, Document Version 3.5.0," 2014.