Timing Verification of Application Software in Multi-Core Systems

A Methodology to Verify Timing Precisely in Multi-Core Systems

Master's thesis in Computer Systems and Networks

Aravindan Anbarasu

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2019

Master's thesis 2019

Timing Verification of Application Software in Multi-Core Systems

A Methodology to Verify Timing Precisely in Multi-Core Systems

Aravindan Anbarasu

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2019

Timing Verification of Application Software in Multi-Core Systems
A Methodology to Verify Timing Precisely in Multi-Core Systems
Aravindan Anbarasu

Supervisor: Jan-Philipp Steghöfer, Computer Science and Engineering Department
Advisor: Henrik Lönn, Volvo Group Trucks Technology
Examiner: Erland Jonsson, Computer Science and Engineering Department

Master's Thesis 2019
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2019

Timing Verification of Application Software in Multi-Core Systems
A Methodology to Verify Timing Precisely in Multi-Core Systems
Aravindan Anbarasu
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Vehicles of today are becoming more autonomous with advanced technologies to obtain real-time information regarding the road and traffic situations. Computational processing must keep pace with this growth in order to meet the timing requirements. Timing constraints are crucial for a safety-critical automotive embedded system, as the consequence of ignoring timing could scale from loss of comfort to life threatening situations. Unfortunately, an Electronic Control Unit (ECU) with a single-core will not have enough computational capacity to perform heavy data processing in real-time to meet the timing constraints. So the automotive industry are replacing their traditional single-core ECUs with multicore ECUs which can perform parallel data processing to meet the timing constraints whenever necessary. The increasing functionality of automotive systems requires not only the use of complex hardware, but it is also very important to identify and prototype methods to capture and verify timing requirements for software components on such multi-core systems, as it impacts their safety as well as their perceived customer value.

The aim of this project is to verify timing constraints for software components on a multi-core platform. The thesis shows how architectural models (EAST-ADL/AUTOSAR) and models capable of precise timing analysis (AMALTHEA) shall be integrated/related for meticulous timing verification. This thesis automates the transformation of an EAST-ADL/AUTOSAR model to an AMALTHEA model, with the intention to retain the EAST-ADL/AUTOSAR models which are the standardized software architecture for automotive ECUs and use the AMALTHEA model only for precise timing verification. A comparison of the different tools that are capable of simulating AMALTHEA models is also presented. The results of this thesis work are a methodology and prototype tooling for precise timing verification in multi-core systems.

Keywords: Vehicles, Timing constraints, ECU, Multi-core, EAST-ADL, AUTOSAR, AMALTHEA

Acknowledgements

First and foremost, I would like to thank my industrial advisor Henrik Lönn and my academic supervisor Jan-Philipp Steghöfer for their support, advice, and criticism during this entire thesis work. Needless to say, without their constant support and valuable ideas, this thesis work would not have been possible. Furthermore, I would like to thank my examiner Erland Jonsson for his support and time throughout this work. And last but not least, I am thankful to all my unnamed friends at Volvo for their help and support.

Aravindan Anbarasu, Gothenburg, June 2019

Contents

List of Figures

List of Tables

Listings

Acronyms

ECU Electronic Control Unit

AUTOSAR AUTomotive Open System ARchitecture

FAA Functional Analysis Architecture

FDA Functional Design Architecture

HDA Hardware Design Architecture

ADL Architecture Description Language

RTE Runtime Environment

BSW Basic Software

ISR Interrupt Service Routine

AREAToP AUTOSAR EAST-ADL Tool Platform

EPL Eclipse Public License

Artop AUTOSAR Tool Platform

EATOP EAST-ADL Tool Platform

TA Timing Architects

SLX Silexica

XML eXtensible Markup Language

XSLT eXtensible Stylesheet Language Transformations

RCM Rubus Component Model

BBW	Brake-by-wire
ABS	Anti-lock Braking System

1
Introduction

Background

Nowadays, autonomous vehicles are evolving with advanced technologies to support autonomous driving and safety features, which in turn requires enormous computing capabilities. Unfortunately, the computing capacity of a single-core will not be enough to meet these requirements. So the automotive industry has made a move from single-core to multi-core systems that can perform heavy data processing in real-time. This is one of the biggest challenges in the automotive industry, because the existing applications can not realize instant benefit because they were not designed initially to run on such multi-core architectures [5]. Also, most of the systems and applications have been migrated into AUTOSAR compatible architectures. Both trends imply the need for a new development environment that can serve these requirements. Model-based approaches have been widely used in the automotive industry for managing such modelling complexities. Handling timing requirements is one of the most important things to be taken care in a modelling approach. This is because timing requirements like execution and response times are crucial for a safety-critical single and multi-core automotive systems, and neglecting timing could range from loss of comfort to life-threatening situations. Timing related issues have always been a big concern while modelling automotive embedded systems. In particular, it is very hard to verify the time in multi-core systems with the prevailing standard modelling languages which have only limited support for multi-core systems.

Therefore, it is very important to identify and prototype methods to capture and verify timing requirements for software components on multi-core systems. Based on the current state of the art research and industry requirements, modelling languages like EAST-ADL and AUTOSAR provide support for modelling multi-core systems. However, the level of details provided by them in terms of multi-core resource utilization will not be sufficient for a precise timing verification in multi-core systems. So a modeling language capable of precise timing analysis like AMALTHEA can be deployed.

Aim

For single core automotive systems, some domain specific modeling languages like EAST-ADL/AUTOSAR exist already. But these model-based methodologies only have limited support for multi-core systems. So precise timing analysis cannot be captured. To tackle this problem, a modeling language like AMALTHEA can be utilized, which provides extended support for multi-core platform. For example, AMALTHEA has additional support for software design constraints like Runnable Affinity Constraints that are used to define the mapping of runnable objects to specific processing cores or scheduling units [4][5].

Rather than creating a new AMALTHEA system model from scratch for each software component, the idea is to transform the existing EAST-ADL/AUTOSAR model into an AMALTHEA model via automation scripts to verify timing requirements on the multi-core platform. The decision to transform is taken because the intention is to retain AUTOSAR (standard implementation level model) and EAST-ADL (complements AUTOSAR with descriptions at a higher level of abstractions) and use the AMALTHEA model mainly for precise timing analysis. Also, the thesis aims to find a tool that is best suited for simulating the AMALTHEA model.

Problem Statement

The thesis focuses on the following questions listed below:

- How can one verify timing more precisely on a multi-core platform?
- How can different timing modelling approaches be mapped to achieve meticulous timing verification for multi-core systems?
- How can one reduce the manual effort done during the model transformation process?
- Which is the best tool to simulate the model that is capable of precise timing verification for multi-core systems?

Intended Project Result

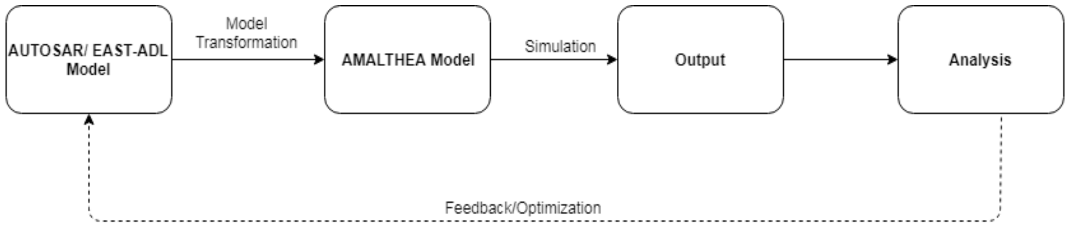


Figure 1.1: Engineering Loop

The results/outcome of this thesis work will be a process/methodology and prototype tooling which will be beneficial for the embedded engineers during the engineering workflow as shown in Figure 1.1.

Limitations

The idea of this thesis is not to invent new ways to do timing analysis but to show how architecture models and multicore deployment models shall be integrated/related. This will be helpful for the automotive industry in finding ways to relate the architectural models with models capable of analysis.

2
Technical Background

This chapter provides some technical background required to give the reader sufficient information about the concepts discussed in this project. A background on EAST-ADL, AUTOSAR and AMALTHEA models are presented with in depth information regarding the relationship between them. The chapter also covers the information regarding the tools and transformation language used in the thesis.

2.1 EAST-ADL

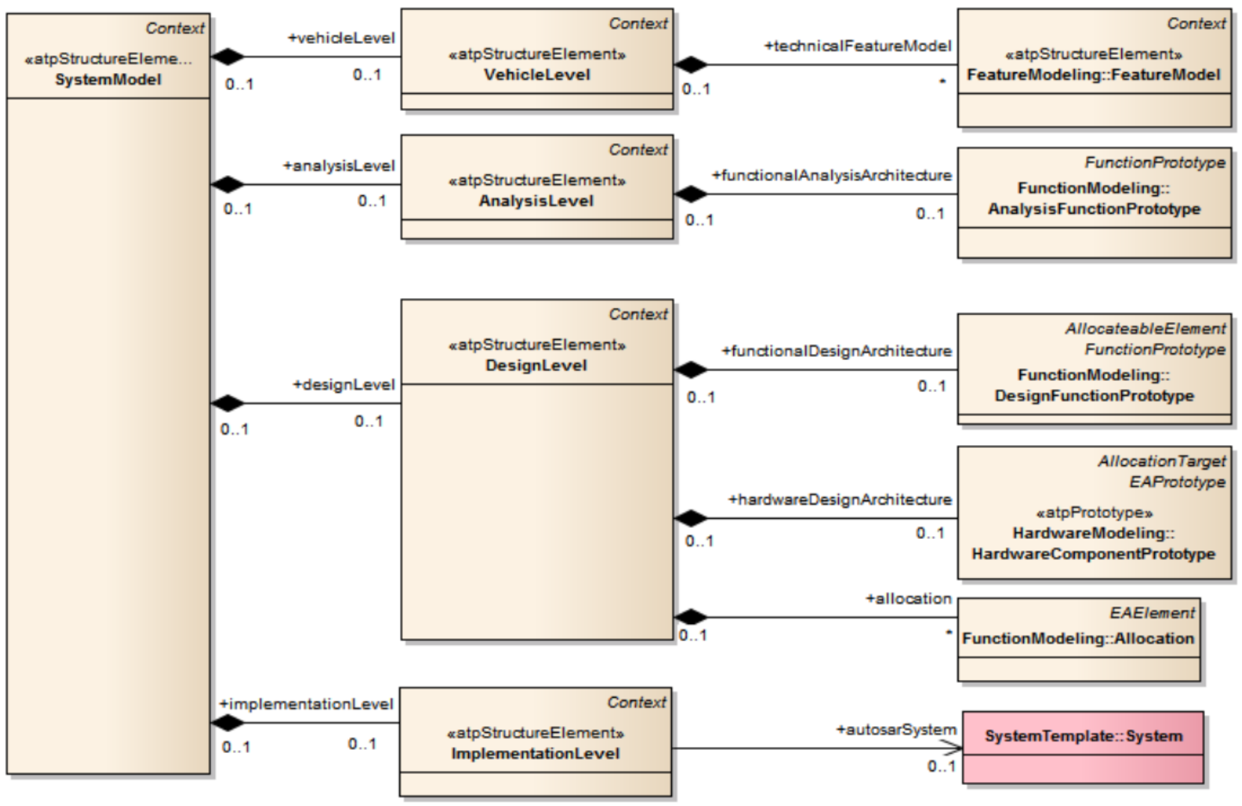


Figure 2.1: EAST-ADL System Model [6]

EAST-ADL is an Architecture Description Language (ADL) for automotive embedded systems initially defined to complement AUTOSAR with descriptions at a higher level of abstraction [6]. The EAST-ADL model is structured in several abstraction levels, that contain vehicle features, requirements, software components, hardware components, functions, variability and communication details [7]. The top-level container of the EAST-ADL model is the System Model as shown in Figure 2.1, which represents the vehicle's electrical/electronic system details and its related concepts [6]. The EAST-ADL meta-model is organized into the following four abstraction levels, namely Vehicle level, Analysis level, Design level and Implementation level.

Vehicle level

The Vehicle level provides support for configuration and definition of product lines at the highest abstraction level in terms of features [6]. A feature could be any characteristic property that may or may not be present in the individual variant of the vehicle. The vehicle level contains an arbitrary set of feature models to represent the intended functionality. The feature model contains the vehicle features that reflect the vehicle configurations [6]. In general, there will be a core technical feature model and one or more product feature models. The core technical feature model defines the overall features of the complete system on vehicle level, and the product feature models provide an orthogonal view on the core technical feature model [6].

Analysis level

The Analysis level contains the abstract functional definition of the electrical/electronic system in the vehicle with some important internal and external interfaces. The Analysis level includes the Functional Analysis Architecture (FAA), that represents the functional structure and realizes the vehicle features to capture analysis support of what the system shall do [6]. On Analysis level, Functional Analysis Architecture is the root component of the function compositional hierarchy. In FAA, Functional component modeling is the main modeling concept used, where two functions interact with each other via ports that are connected by connectors [6].

Design level

The Design level represents the abstract design definition of the electrical/electronic system in the vehicle [6]. It also contains the non-transparent infrastructure functionality details like error handling and mode change to estimate the application's behaviour [6]. It includes the Functional Design Architecture (FDA), the Hardware Design Architecture (HDA).

Functional Design Architecture (FDA)

The Functional Design Architecture represents a decomposition of functionalities mentioned in the Analysis level, including behavioral description but excluding software implementation constraints [6]. FDA is supposed to implement all the functionalities at the vehicle level, if no FAA has been defined during the modelling process [6]. With FDA, it possible to meet constraints regarding non-functional properties such as allocation, efficiency, or reuse. Entities in the FDA and entities in the FAA are related by an n-to-m mapping (Realization Relationships) [6]. The main modeling concept applied in FDA is functional modeling, where each function communicates with another function via ports that are connected by connectors owned by the composing function [6].

Hardware Design Architecture (HDA)

The HDA represents the hardware architecture of the embedded system. The HDA models ECUs, sensors, actuators and communication links [6]. The HDA also acts as an allocation target for the functions of the FDA [6].

Implementation level

The Implementation level represents the software and hardware architecture of the system in the vehicle. It refers to the system element in an AUTOSAR model and does not have any EAST-ADL specific standards [6].

2.2 AUTOSAR

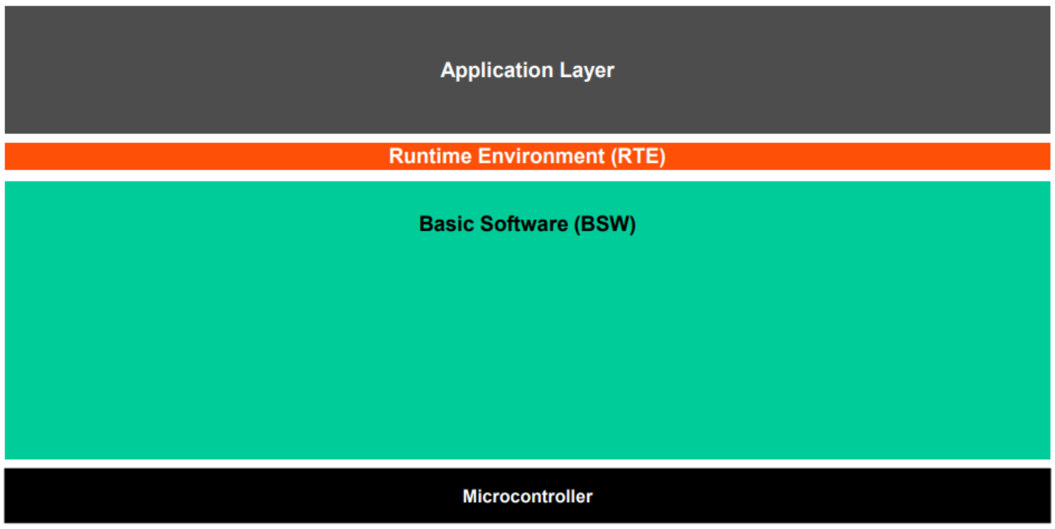


Figure 2.2: Overview of AUTOSAR Software Layers [17]

AUTomotive Open System ARchitecture (AUTOSAR) is an open and standardized software, jointly developed by the vehicle manufacturers, suppliers and service providers [16]. AUTOSAR aims to enhance complexity management of integrated Electrical/Electronic architectures through increased reuse and exchangeability of software modules between original equipment manufacturers and suppliers. AUTOSAR standardizes two software platforms, namely Classic and Adaptive. This thesis focuses only on classic AUTOSAR.

Software Architecture

AUTOSAR uses a three-layered architecture, namely Application layer, Runtime Environment (RTE) layer and Basic Software (BSW) layer which run on a micro-controller as shown in Figure 2.2. The Application layer includes various application software components (AUTOSAR software components and/or AUTOSAR sensor/actuator components) that are designed to execute a specific set of tasks, as per the use-case [20]. The RTE is at the heart of the AUTOSAR ECU architecture and provides communication services to the application software components. The main purpose of having the RTE layer is to make AUTOSAR software components completely ECU independent by abstracting the information exchange between the application software components and the Basic Software [17]. The BSW layer includes the standardized software modules that offer services necessary to run the functional part of the upper software layers.

Component Types

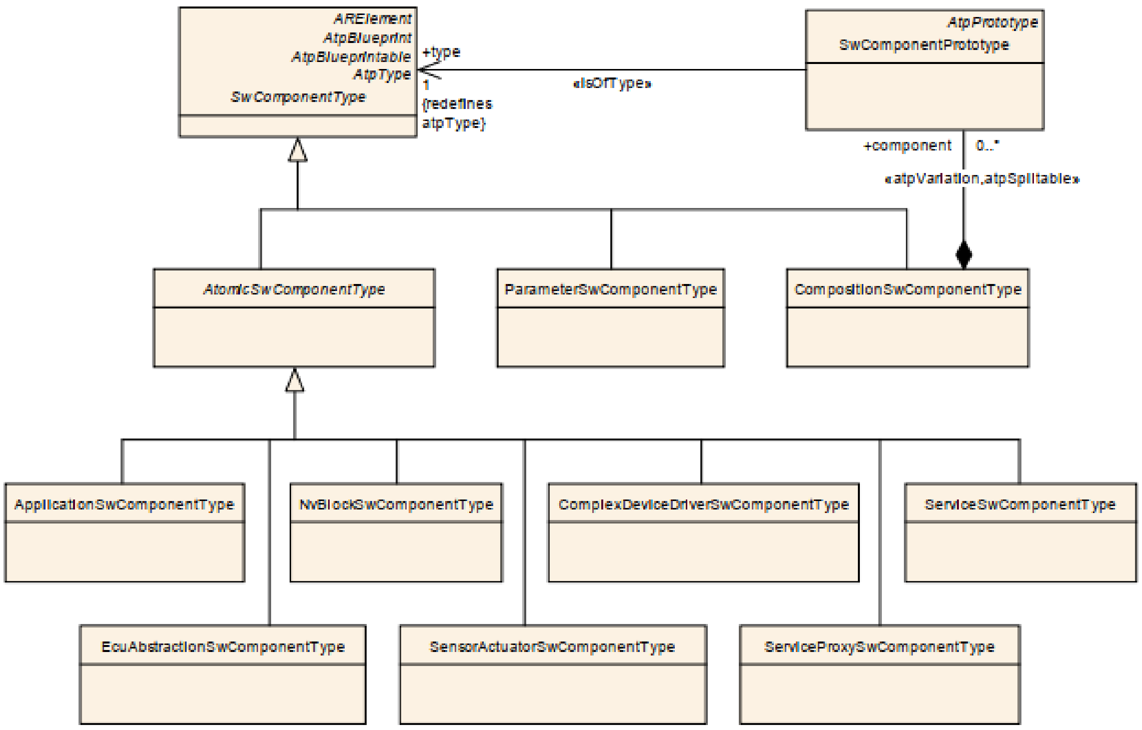


Figure 2.3: Overview of Component Types [15]

A general outline of the component types in AUTOSAR is shown in Figure 2.3. Some of the component types that are more relevant to this thesis are described below,

SwComponentType

SwComponentType is the base class for all AUTOSAR software components [15]. The abstract SwComponentType cannot be instantiated, but can be referenced by one or many SwComponentPrototypes.

SwComponentPrototype

SwComponentPrototype implements the usage of a SwComponentType in a specific role, i.e., they are used to instantiate the SwComponentType [15]. In general, arbitrary numbers of SwComponentPrototypes can be instantiated from the same SwComponentType. All the SwComponentPrototypes instantiated from the same SwComponentType will have the same properties of that corresponding SwComponentType. For example, if SwComponentType is defined with 3 Provider PortPrototypes and 4 Receiver PortPrototypes then all the SwComponentPrototypes pointing to that SwComponentType will have the same number of Receiver and Provider PortPrototypes.

CompositionSwComponentType

Encapsulation of specific functionality by aggregating existing software components is the main purpose of a CompositionSwComponentType [15]. CompositionSwComponentType aggregates SwComponentPrototypes which in turn are typed by a SwComponentType. It also aggregates SwConnectors for primarily connecting SwComponentPrototypes [15]. Because CompositionSwComponentType is also a SwComponentType, it may be aggregated again into further CompositionSwComponentTypes [15]. CompositionSwComponentTypes do not add any new functionality to the software components they aggregate. Representing the application software of an entire vehicle in a single CompositionSwComponentType is one implication of the concept of CompositionSwComponentType [15].

AtomicSwComponentType

AtomicSwComponentType is derived from SwComponentType and it encapsulates the implementation of the functionality and behavior of the software components [15]. An atomic software component is atomic in the sense that it cannot be further disintegrated into software components. Only AtomicSwComponentType can have RunnableEntitys, which are the smallest code-fragment that are executed under control of the RTE [15].

ApplicationSwComponentType

The ApplicationSwComponentType is a specialization of AtomicSwComponentType that uses all AUTOSAR communication mechanisms and services for representing hardware-independent application software [15]. The SwcInternalBehavior of an ApplicationSwComponentType contains the RTE information regarding the software component, i.e., the RunnableEntities and the RTE Events they respond to [15].

Port Interfaces

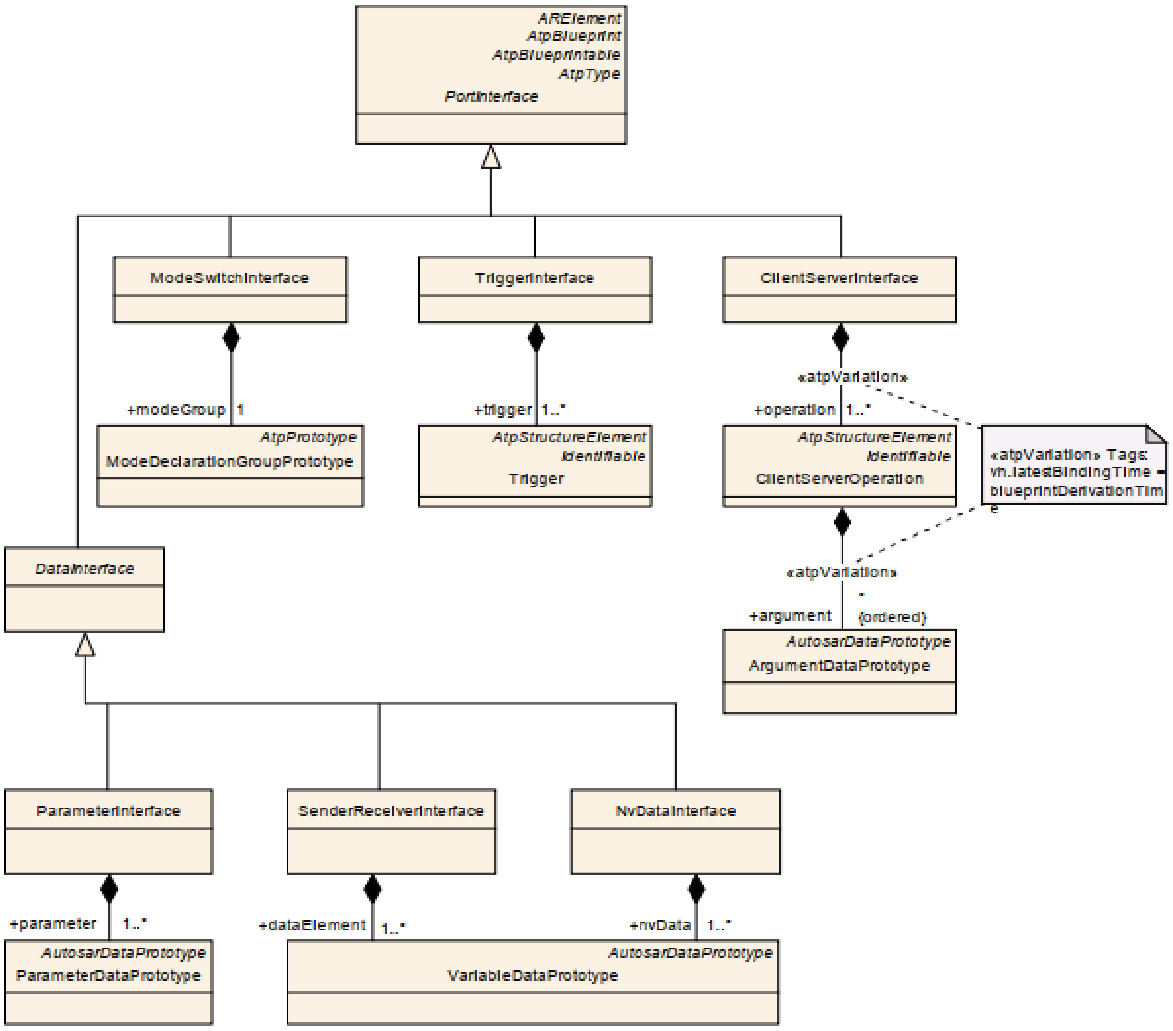


Figure 2.4: Port Interfaces in the AUTOSAR meta-model [15]

A Port Interface is attached to every port on a software component, that describes the data or operations that are provided or required by the port of that Software Component [21]. A Port Interface can be either a Parameter Interface, Trigger Interface, Mode Switch Interface or Non Volatile Data Interface [21] as shown in Figure 2.4. This thesis focuses only on Client-Server Interface and Sender-Receiver Interface.

Sender-Receiver Interface

The sender-receiver interface is a special kind of port-interface that are used for sender-receiver communication. The sender-receiver interface defines the data-elements that are sent by a sending component or received by a receiving component [21]. Sender-Receiver interfaces are composed of variable data prototypes that used for data exchange between software components and also specifies which data is transferred from the sender to the receiver [21].

Client-Server Interface

The client-server interface is a special kind of port-interface that are used for client-server communication. The client-server interface defines the operations that are implemented by the server component and that can be used by the client component [21]. In the client-server interface, a client is allowed to call an operation at a server, which in turn provides the result to the client [21].

Interested readers can refer to [15] for more detailed information about the AUTOSAR model.

2.3 AMALTHEA/APP4MC

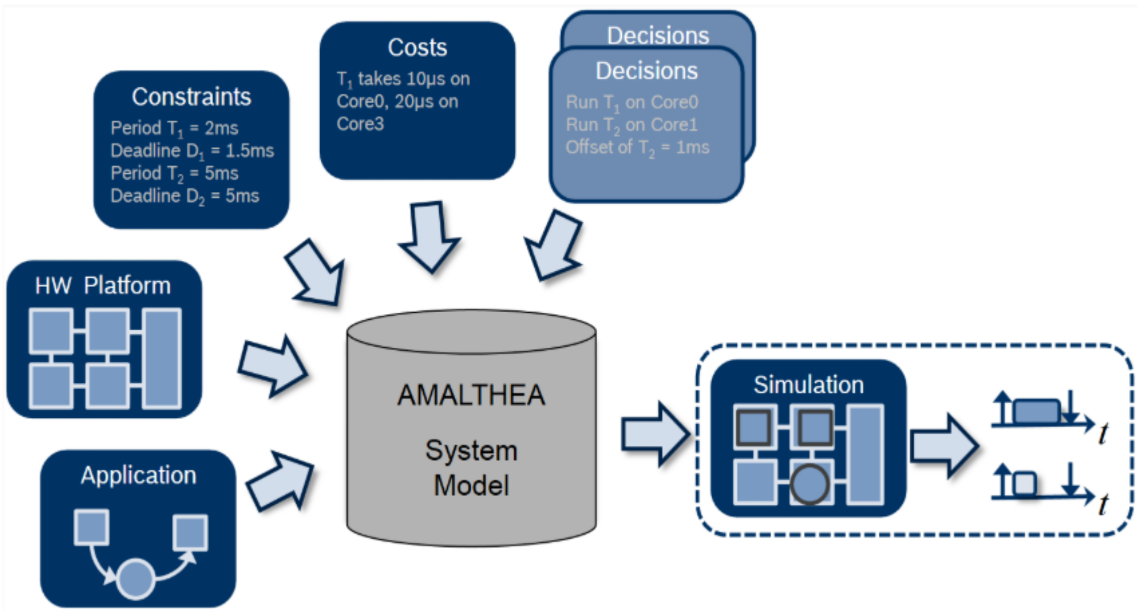


Figure 2.5: Overview of the contents of AMALTHEA meta-model [4]

AMALTHEA is an open source project started in 2011 for engineering embedded multi and many-core software systems, because of the lack of multi-core support in the development tools at that time [4]. In later years, AMALTHEA platform became an official Eclipse project called APP4MC, which is an open source Eclipse tool platform that provides an AUTOSAR compliant system model called AMALTHEA for optimizing embedded multi-core systems [2]. The AMALTHEA system model covers engineering activities such as system/software modelling, partitioning, mapping, and tracing. The AMALTHEA model is sub-divided into eleven models, namely Components Model, Configuration Model, Constraints Model, Event Model, Hardware Model, Mapping Model, Measurement Model, OS Model, PropertyConstraints Model, Stimuli Model and Software Model. Each of these models covers a particular aspect of the system under development as in Figure 2.5.

The current version, Eclipse APP4MC v0.9.3 does not have support for simulation tools, visualization tools and graphical editors [5]. APP4MC typically addresses automotive domain but it is also applicable to telecommunication and generic real-time systems. Interested readers can also refer to [5] for more information about the AMALTHEA model. Some of the sub-models that are relevant to this thesis are explained below,

Hardware Model

The AMALTHEA hardware model is used to describe the hardware architecture of the embedded system. It includes details related to ECU, micro controllers, processing cores, memories, connections, additional peripherals, etc [5].

Mapping Model

The mapping model provides information about the mappings and allocations for different entities. The mapping details include, description of memory address location where the physical memory section is allocated, mapping of various elements to a specific memory, etc [5]. The allocation details include, Scheduler Allocation (associate a scheduler with a processing unit), Runnable Allocation (associate a runnable with a scheduler), Task Allocation (associate a task with a scheduler) and Interrupt Service Routine (ISR) Allocation (associate an ISR with an interrupt controller) [5].

Operating System Model

The Operating System model describes the functionality of an operating system. It includes details related to task scheduler (used to manage and distribute tasks using a scheduling algorithm), interrupt controller (used to indicate an event that needs immediate attention), semaphores (used to control access to a common resource by multiple processes in a concurrent system such as a multitasking operating system), etc., thus providing a way to specify how access is given to certain system resources [5].

Stimuli Model

The Stimuli model provides information about the stimulus and clock objects. A stimulus is responsible to activate processes in a different manner. For example, Event Stimulus (activation triggered by an event), Periodic Stimulus (periodic activation based on an offset, a recurrence and a jitter), Inter Process Stimulus (activation based on an explicit inter-process trigger), etc. The clock objects are used to describe the progress of time for one or more variables in relation to global time [5]. The time of task activation can be different, if two equal stimuli have a different time base [5].

Software Model

The Software model provides information about the entire functional behaviour of the software. This includes details regarding labels (data element located in memory), runnables (run-time entity with instruction count) , tasks (collection of runnables), call graph (define how a task or ISR behaves during execution), process prototypes (define runnable order precedence), process chains (list of tasks representing an end-to-end data processing path), etc [5].

Constraints Model

The Constraints model provides information about the different kind of constraints that has to be satisfied during execution. This includes Runnable Affinity Constraints (mapping of runnable objects to processing cores or scheduling units restriction), Event Chain Latency Constraint (how long after a stimulus a corresponding response must occur), Data Affinity Constraints (mapping of label objects to memory units restriction), Process Requirements for tasks(response-time restriction for tasks), Runnable Sequencing Constraints (restriction on execution orders of runnables), etc [5].

Event Model

The Event model provides information about different event entities that can be used for the modeling of event chains and for some timing constraints. This includes Process Event, Process Chain Event, Stimulus Event, Runnable Event, Semaphore Event, etc [5]. Each of these event entities has an event type that represents different state-transitions. Some of the event types are, activate (denotes that the entity is activated by a stimulus), start (denotes that the entity starts to execute for the first time), preempt (denotes that the entity is stopped by the scheduler), resume (denotes that the entity continues execution on the same or other core) and terminate (denotes that the entity has finished execution) [5].

2.4 Tool Platform

Design Tools

The thesis intends to use the AUTOSAR EAST-ADL Tool Platform (AREAToP) Technology Demonstrator tool developed by AB Volvo for designing EAST-ADL and AUTOSAR models.

AREAToP Technology Demonstrator

AREAToP Technology Demonstrator is a design tool developed by AB Volvo with an intention to design and develop both EAST-ADL and AUTOSAR models under a single tool platform. It is based on the model driven approach as basic engineering methodology. AREAToP is an implementation of the common base functionality of

AUTOSAR Tool Platform (Artop) and EAST-ADL Tool Platform (EATOP). However, the tool has several of its own implementation of functionalities in the form of software plugins. One of the plugins used in this thesis is the Graph Modelling plugin, which enables the user to view a graphical representation of a selected element along with its ports and communication link as shown in 6.1. Interested readers can refer [18] and [19] to learn more about the functionalities of EATOP and Artop tool platforms respectively.

Simulation Tools

Since the AMALTHEA research project was started in 2011, only a few commercial tools provide support for simulating AMALTHEA models. This thesis focuses on two such tools, namely Silexica and Timing Architects (TA) tool suite.

SILEXICA

Silexica (SLX) is a multicore development tool that provides software execution awareness in the hardware and software inter-dependencies. The tool provides a precise and full analysis of software inter-dependencies thereby helping to fully understand the application behavior on a multi-core system [10]. With SLX, the interaction between runnables and tasks, data dependencies and the program flow can be revealed [10]. Additionally, information for parallelization such as core migration policies and scheduler configuration are also revealed. Based on these information, runnable to task mapping as well as task to core mapping can be optimized [10]. The tool is proprietary and a license cost has to be paid inorder to use it. A limited period demo version is also available upon request.

SLX + AMALTHEA

The current version, SLX-2018.10 sp-1 of the tool platform comes with AMALTHEA support. With SLX the user can import an ALMATHEA model into a new project and generate schedules. Once an AMALTHEA model is imported, the first step is the Analyze phase, where the model is analyzed to get a clear understanding of the inter-task and inter-runnable dependencies [10]. Next step is the optimize phase, where SLX allows the user to parallelize selected tasks by creating a mapping of tasks to processing cores for the schedule creation. Finally, the tool allows the user to run the scheduler which computes schedules and the results of the simulation are displayed through gantt charts [10]. The tool also allows the user to export the modified AMALTHEA model after configuring the scheduler configuration.

TIMING ARCHITECTS TOOL SUITE

TA tool suite provides user-friendly tools for the design, simulation, verification and optimization of embedded multi-core systems [11]. The TA tool suite is capable of providing an in-depth analysis of the timing behaviour and supports mapping of application software on different cores in a multi-core platform, which helps in increasing the efficiency of multi-core real-time systems [11]. For each step in the software project life-cycle, TA tool Suite provides a dedicated software product. This includes TA-Design, TA-Simulation, TA-Optimization and TA-Inspection [11].

TA-Design provides support for defining timing requirements of the application software through an interactive dynamic visualization [11]. TA-Simulation provides support for model-based simulation of ECU timing behavior. It also aids in analyzing the system behavior through graphical and table based evaluations of timing metrics [11]. TA-Optimization provides support for the distribution and mapping of application software on different cores in a multi-core platform [11]. TA-Inspection provides support for verification of the timing behavior of application software in terms of deadlines, response times, utilization, and other metrics [11]. The tool is proprietary to the company (Vector) and a license cost has to be paid in-order to use it. A limited period demo version is also available upon request.

TA SUITE + AMALTHEA

The current version, 19.1 of the TA tool suite comes with AMALTHEA support. The TA tool suite allows the user to import an existing AMALTHEA model as a new project. However, using TA-Design a new AMALTHEA model can also be created from scratch. After the AMALTHEA model is imported/created, the TA tool suite analyzes the model and indicates the user for any inconsistencies in the model. For example, the periodic tasks might be missing the recurrence time value in the model. These inconsistencies can be fixed by editing the AMALTHEA model with the help of TA-Design. Once the model has become error free, the TA tool suite allows the user to perform the simulation with the help of TA-Simulation. The simulation results (scheduling of tasks) are visualized via different views like gantt chart view, histogram view, etc [11]. The simulation results are analyzed via different metric tables like the process table, requirements table, hardware resource table, event chain requirement table, etc. In case, if some of the timing requirements are not fulfilled, the user can use TA-Design to edit the AMALTHEA model accordingly and perform a new simulation.

2.5 eXtensible Stylesheet Language Transformations

eXtensible Stylesheet Language Transformations (XSLT) is the best known XML transformation language [12]. XSLT version 1.0 has been widely used for transforming XML documents into other XML documents since 1999 [12]. The XSLT processor takes an XML source document and an XSLT style sheet as inputs, and processes them to produce an output document as shown in Figure 2.6. In this process, the source document is unaltered and the resultant output document will be a new document based on the content of the source document. Xalan is a known open source XSLT 1.0 processor from the Apache Software Foundation available for Java and C++ [13].

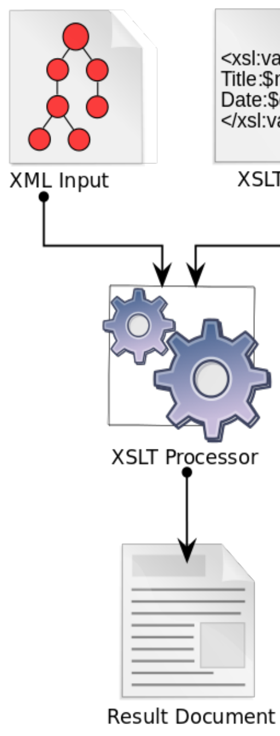


Figure 2.6: Process Flow in XSLT [13]

The XSLT processor implements a fixed algorithm. Initially, the processor reads the XSLT style sheet and builds a source tree from the input XML document. Then the processor starts processing the source tree from the root node. The processor tries to find the best-matching template for each node in the XSLT style sheet and evaluates the template's contents. In general, the contents in each template instructs the processor to either process more nodes in the source tree or to create new nodes in the result tree [13]. XML Path Language, shortly called XPath is a major element in the XSLT standard that contains over 200 built-in functions to navigate through the subsets of the source document tree and perform calculations. XSLT 1.0 uses XPath 1.0 to navigate through elements and attributes in an XML document. Interested readers can also refer to [14] for more information about the XSLT syntax.

3
Literature Review

3.1 Relevant Literature from the Problem Domain

Bucaioni et al. [1] propose a methodology for early verification of non-functional properties like timing requirements during the design level within EAST-ADL, which could possibly reduce the expensive modifications later in the implementation level (Rubus Component Model) [1]. In general, the translation from the design to implementation level is performed manually, pushing the timing verification to implementation phase [1]. The paper demonstrates a proof of concept to automate the integration (through model transformations) between EAST-ADL and Rubus Component Model (RCM) on single/multi core platforms thereby enabling the early verification during the design phase itself [1]. For multi-core systems the authors propose to use RCM as the implementation level for EAST-ADL because at the application software level, AUTOSAR does not differentiate between the control and the data flows, which is fundamental for providing early timing verification [1]. Hence this paper [1] intends to substitute AUTOSAR with RCM at the implementation level within EAST-ADL as it does not have extended support for multi-core systems.

Mubeen et al. [22] propose an approach to represent the end-to-end timing models. At a higher abstraction level, the approach provides timing information on the system models that are created with the EAST-ADL language utilizing the TIMing MOdel (TIMMO) methodology, annotated with timing information using Timing Augmented Description Language (TADL2) [22]. At the lower level, the approach uses RCM to represent the timing information. The paper proposes to choose RCM instead of AUTOSAR for two reasons. Though the current AUTOSAR specification provides a timing model, it fails to specify a few low-level details which are needed to perform the end-to-end timing analysis like the control flow between functionalities, etc [22]. The other reason is that the implementations built with RCM have relatively smaller runtime footprints, i.e., memory overheads and timing [22].

From the literature [1] and [22], one can clearly see the deficits of the AUTOSAR over RCM model. Since RCM is a proprietary modeling language developed by Arcticus Systems, this thesis aims to experiment with some open source modeling language like AMALTHEA.

3.2 Relevant Literature on Potential Solution Approaches

APP4MC is an open source application platform project designed to utilize multi/-many core systems managing timing through simulation and continuous design [2]. APP4MC uses AUTOSAR compliant data model called AMALTHEA for performing timing and behavioural analysis and also provides basic parallelization features, model-based editing and visualization and many more [2]. Since AMALTHEA is AUTOSAR compliant it can be easily adapted into the automotive industry when it comes to multi-core platform [3]. AMALTHEA system model is subdivided into ten models, each covering a specific aspect of the system under development [4]. Information in these models are basically similar to AUTOSAR but the level of detail can be higher in AMALTHEA [4].

From the literature [2], [3] and [4] one can clearly see that AMALTHEA addresses some of the deficits of AUTOSAR, especially within the scope of multi-core.

3.3 Previous Work within the Project

"EAST-ADL architecture description language describes a system on a higher level of abstraction than AUTOSAR. Starting with a description of the features that have to be developed, the model is refined until the level of abstraction covered by AUTOSAR" [4]. The aim of this thesis is to reuse this already existing EAST-ADL/AUTOSAR models for the software components, and then use model transformation to transform them into AMALTHEA for precise timing analysis.

4 Evaluation Criteria

After conducting the literature review, an evaluation criteria (list of capabilities) that the outcome of this thesis should address are framed and are as follows,

- Successful transformation of source architectural model to AMALTHEA model.
- Ability to verify response time for each task of the software component in a multi-core system.
- Ability to verify response time for each event-chain of the software component in a multi-core system.
- Ability to measure the load distribution of tasks and runnables on the processing cores.

During the experimentation process, the results obtained from each iteration will be compared against the above mentioned list of capabilities to ensure if it satisfies all/some of the criteria.

5
Methods

This thesis follows Action Research methodology as in Figure 5.1. Several iterations of the research loop are carried out until an optimized plan/design choice is achieved, that is when the plan satisfies all/most of the evaluation criteria.

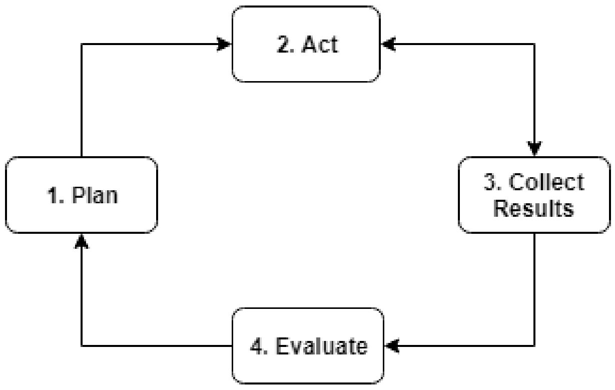


Figure 5.1: Action Research Loop

The steps followed in each iteration of the Action Research Loop are listed below,

Step 1 (Plan):

Input: Thesis description (background, aim, problem statement, literature review and limitations), knowledge on AUTOSAR/EAST-ADL/AMALTHEA data models, available model transformation tools/languages at AB Volvo and supported tool suite for simulating/analyzing AMALTHEA models. Possibly the output of step 4 in each iteration will be an extra input to step 1 in the following iterations.

Activity: A preliminary plan on which architectural model type between EAST-ADL and AUTOSAR to be used (both EAST-ADL and AUTOSAR are not replacements for one another and each serves a different purpose. AUTOSAR is the standard implementation level model and EAST-ADL complements AUTOSAR with descriptions at a higher level of abstraction for support regarding variability, requirements, safety, etc.,), which example application to be used (Selection is based on the available data rich application models within vehicle dynamics control department at AB Volvo), which model transformation tool to be used (the thesis intends to use a tool that is developed by AB Volvo), which model transformation language to be used (Selection is based on whether the transformation language is supported by the selected transformation tool), which tool suite to be used for simulation and analysis (Selection is based on whether the tool is open-source/commercial, level of support to AMALTHEA data model, etc.,) are decided.

Output: A preliminary plan including an architectural model type, example application model, model transformation tool, transformation language, simulation and analysis tool suite.

Step 2 (Act):

Input: Architectural model type to be used, example application model to be used, tool suite to be used for model transformation, model transformation language to be used, tool suite to be used for simulation and analysis.

Activity: The second step is to get ready all the necessary aspects such as example application, model transformation language and tool suites required for the engineering workflow as shown in Figure 1.1. The possible activities would be to manually add details to the chosen example application model if it lacks some of the details which it supports and that can be mapped to AMALTHEA model during model transformation. The guidelines for mapping will be framed and implemented in the model transformation language after a careful analysis of the meta models of both the source and target models. The activity also includes the preparation of model transformation file.

Output: A working setup including an example application model, model transformation tool, model transformation language, simulation and analysis tool.

Step 3 (Collect Results):

Input: A working setup including an example application model, model transformation language, model transformation tool, simulation and analysis tool.

Activity: The third step is to perform model transformation and simulation by taking the role of an engineer as shown in Figure 1, and to collect results/outcomes.

Output: Results from the simulation and analysis activities.

Step 4 (Evaluate):

Input: Evaluation criteria, results from the simulation and analysis activities.

Activity: In the final step, the results collected from the step 3 are compared against the evaluation criteria (framed after conducting the literature review) to ensure if it satisfies all/some of the criteria. Based on the evaluation, a new plan with possible changes for improvement is created and the same cycle of steps is repeated as shown in Figure 5.1.

Output: Decision on whether to stop the iteration or to start the next iteration with a new plan. The output of this step is then an extra input to step 1 in the next iteration.

The iteration continues until step 4 concludes that an optimized plan/design choice is achieved, that is when the plan satisfies all/most of the evaluation criteria.

6
Results

In this chapter, the results of each iteration of the action research loop are presented.

6.1 Results of each Iteration

6.1.1 Iteration 1:

Goal

- Successful transformation of EAST-ADL model to AMALTHEA model.
- Successful simulation and analysis of the resultant AMALTHEA model with the help of a tool platform.

Step 1 (Plan)

Architectural Model Type

Since this thesis intends to experiment only with EAST-ADL and AUTOSAR architectural models, during this iteration we select EAST-ADL architectural model for experimentation. Both EAST-ADL and AUTOSAR are not replacements for one another, as each serves a different purpose.

Sample Application Model

The thesis intends to use the BBW EAST-ADL application model, that pre-exist within vehicle dynamics control department at AB Volvo. The model is available in the URL: `http://www.east-adl.info/Specification/V2.1.12/EAST-ADL_V2.1.12.zip`.

Model Transformation Tool

The thesis intends to use the AREAToP Technology Demonstrator tool developed by AB Volvo for model transformation.

XML Transformation Language

The thesis intends to use the XSLT Version 1.0 as the XML transformation language. This is because the AREAToP tool by AB Volvo has a built-in XSLT processor

(Xalan-Java Version 2.7.1) that supports only XSLT Version 1.0 for transforming XML documents.

Simulation and Analysis Tool

This thesis focuses on two known tools that support AMALTHEA, namely Silexica and Timing Architects. During this iteration, we pick Silexica tool platform for simulation and analysis. A trial version (SLX-2018.10 sp-1) of the tool suite is used for the experiment.

Step 2 (Act)

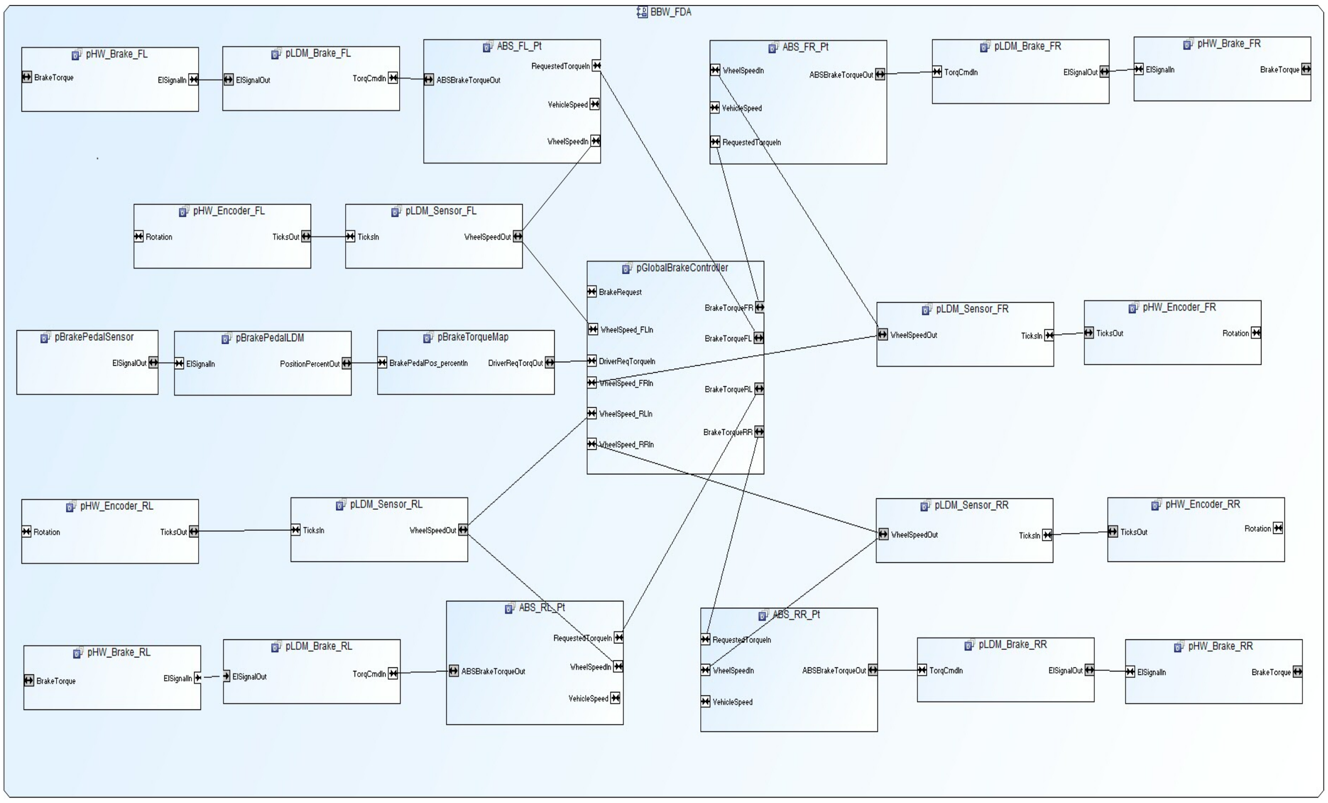


Figure 6.1: Functional Design Architecture of BBW component in AREAToP

The FDA of the Brake-by-wire component is analyzed, to check if it lacks some of the details which it supports and that can be mapped to AMALTHEA model during model transformation. The thesis intends to focus only on the design level abstraction layer of the EAST-ADL model. This is because the design level contains the FDA, representing a full decomposition of functionalities denoted in the analysis level with behavioral description. The Functional Design Architecture diagram of the Brake-by-wire component is shown in Figure 6.1.

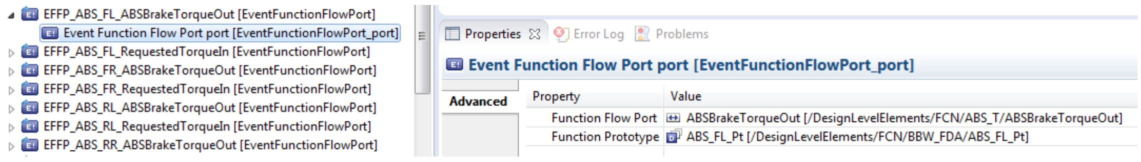
Table 6.1: Mapping between EAST-ADL and AMALTHEA elements during Model Transformation

EAST-ADL	**AMALTHEA**	**Remarks**
FunctionModeling :: DesignFunctionPrototype	Task (Software Model)	To avoid false positives during model transformation, DesignFunctionPrototype elements that satisfy the hierarchical structure "SYSTEM-MODEL/DESIGN-LEVEL/FUNCTIONAL-DESIGN-ARCHITECTURE" and with TYPE-TREF attribute value matching either DESIGN-FUNCTION-TYPE or LOCAL-DEVICE-MANAGER are alone considered.
FunctionModeling :: DesignFunctionType	Runnable (Software Model)	To avoid false positives during model transformation, DesignFunctionType elements with property IS-ELEMENTARY = true are alone considered.
TimingConstraint :: PeriodicConstraint	Periodic Stimulus (Stimuli Model)	Nil
TimingConstraint :: PrecedenceConstraint	OrderPrecedenceSpec (Software Model :: Process Prototype)	Nil
TimingConstraint :: ExecutionTimeConstraint	Instructions Constant (Software Model :: Runnable)	Nil
HardwareModeling :: HardwareComponent Prototype	Processing Unit (Hardware Model)	To avoid false positives during model transformation, HardwareComponentPrototype elements with execution rate are alone considered. .
Events :: EventFunction	Process Event (Events Model)	Nil
Timing :: EventChain	Event Chain (Constraints Model)	Nil
TimingConstraints :: AgeConstraint	Event Chain Latency Constraint (Constraints Model)	Nil

After a careful analysis of the EAST-ADL and AMALTHEA models, the guidelines for mapping is framed as shown in Table 6.1. The guidelines for mapping are implemented in the XSLT as shown in Listing A.1, and will take effect during the model transformation process. Moreover, the Brake-by-wire EAST-ADL model lacks some details which it supports and that can be mapped to AMALTHEA model. This thesis uses AREAToP tool to edit the Brake-by-wire EAST-ADL model. The additional details added manually to the Brake-by-wire EAST-ADL model before the transformation process are listed below,

Event Function Flow Port

Event Function Flow Port refers to the time when data is sent or received at the Function Flow Port of a Function Prototype [6]. A function Flow Port refers to either IN/OUT port of a Function Prototype. Figure 6.2 shows an Event Function Flow Port being added for theABS module in AREAToP.



EFFP_ABS_FL_ABSBrakeTorqueOut [EventFunctionFlowPort]
Event Function Flow Port port [EventFunctionFlowPort_port]
EFFP_ABS_FL_RequestedTorqueIn [EventFunctionFlowPort]
EFFP_ABS_FR_ABSBrakeTorqueOut [EventFunctionFlowPort]
EFFP_ABS_FR_RequestedTorqueIn [EventFunctionFlowPort]
EFFP_ABS_RL_ABSBrakeTorqueOut [EventFunctionFlowPort]
EFFP_ABS_RL_RequestedTorqueIn [EventFunctionFlowPort]
EFFP_ABS_RR_ABSBrakeTorqueOut [EventFunctionFlowPort]

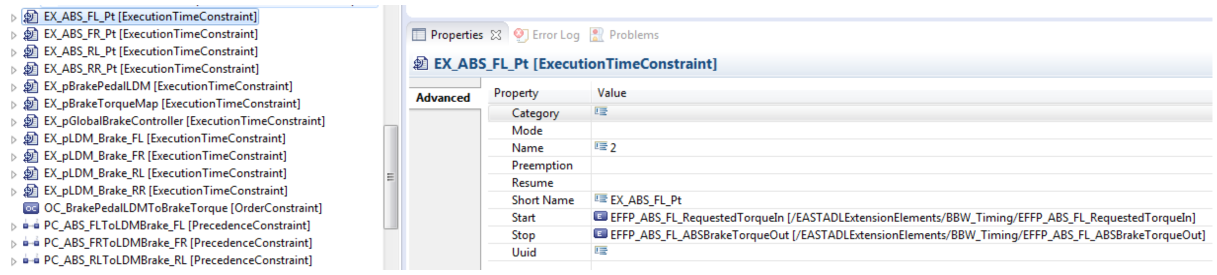
Event Function Flow Port port [EventFunctionFlowPort_port]

Property	Value
Function Flow Port	ABSBrakeTorqueOut [/DesignLevelElements/FCN/ABS_T/ABSBrakeTorqueOut]
Function Prototype	ABS_FL_Pt [/DesignLevelElements/FCN/BBW_FDA/ABS_FL_Pt]

Figure 6.2: Event Function Flow Port of ABS in AREAToP

Execution Time Constraint

An Execution Time Constraint is a timing constraint that limits the time between the start and stop point of an executable function [6]. The start and stop point refer to the corresponding Event Function Flow Port of the executable function. The Execution Time Constraint does not count the intervals when the execution of such an executable has been interrupted [6]. Figure 6.3 shows an Execution Time Constraint being added for the ABS module in AREAToP.



EX_ABS_FL_Pt [ExecutionTimeConstraint]
EX_ABS_FR_Pt [ExecutionTimeConstraint]
EX_ABS_RL_Pt [ExecutionTimeConstraint]
EX_ABS_RR_Pt [ExecutionTimeConstraint]
EX_pBrakePedalLDM [ExecutionTimeConstraint]
EX_pBrakeTorqueMap [ExecutionTimeConstraint]
EX_pGlobalBrakeController [ExecutionTimeConstraint]
EX_pLDM_Brake_FL [ExecutionTimeConstraint]
EX_pLDM_Brake_FR [ExecutionTimeConstraint]
EX_pLDM_Brake_RL [ExecutionTimeConstraint]
EX_pLDM_Brake_RR [ExecutionTimeConstraint]
OC_BrakePedalLDMToBrakeTorque [OrderConstraint]
PC_ABS_FLToLDMBrake_FL [PrecedenceConstraint]
PC_ABS_FRToLDMBrake_FR [PrecedenceConstraint]
PC_ABS_RLToLDMBrake_RL [PrecedenceConstraint]

EX_ABS_FL_Pt [ExecutionTimeConstraint]

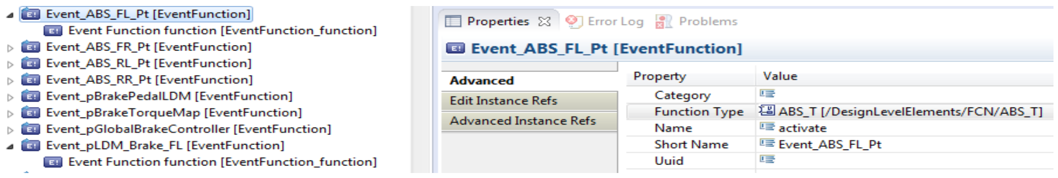
Property	Value
Category	
Mode	
Name	2
Preemption	
Resume	
Short Name	EX_ABS_FL_Pt
Start	EFFP_ABS_FL_RequestedTorqueIn [/EASTADLExtensionElements/BBW_Timing/EFFP_ABS_FL_RequestedTorqueIn]
Stop	EFFP_ABS_FL_ABSBrakeTorqueOut [/EASTADLExtensionElements/BBW_Timing/EFFP_ABS_FL_ABSBrakeTorqueOut]
Uuid	

Figure 6.3: Execution Time Constraint for ABS in AREAToP

Since the Execution Time Constraint elements in EAST-ADL model lack the 'value' property due to a flaw in AREAToP tool, the instruction count values are entered in the 'Name' property as a workaround.

Event Function

The Event Function refers to the invocation event of the function (when data is consumed and execution starts) and may be data or time-related depending of trigger. The target function of an Event Function will be either a Function Type or a Function Prototype [6]. Figure 6.4 shows an Event Function being added for the ABS module in AREAToP.


Event_ABS_FL_Pt [EventFunction]
Event Function function [EventFunction_function]
Event_ABS_FR_Pt [EventFunction]
Event_ABS_RL_Pt [EventFunction]
Event_ABS_RR_Pt [EventFunction]
Event_pBrakePedalLDM [EventFunction]
Event_pBrakeTorqueMap [EventFunction]
Event_pGlobalBrakeController [EventFunction]
Event_pLDM_Brake_FL [EventFunction]
Event Function function [EventFunction_function]

Properties | Error Log | Problems

Event_ABS_FL_Pt [EventFunction]

Advanced
Edit Instance Refs
Advanced Instance Refs

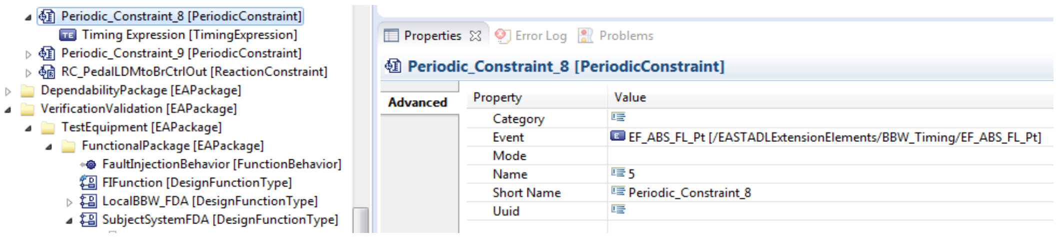
Property	Value
Category	
Function Type	ABS_T [/DesignLevelElements/FCN/ABS_T]
Name	activate
Short Name	Event_ABS_FL_Pt
Uuid	

Figure 6.4: Event Function for ABS in AREAToP

Since the Event Function elements in EAST-ADL model lack the 'Event Type' property due to a flaw in AREAToP tool, the event type options are entered in the 'Name' property as a workaround.

Periodic Constraint

A Periodic Constraint is a timing constraint that describes a periodically occurring event [6]. Figure 6.5 shows a Periodic Constraint being added for the ABS module in AREAToP.


Periodic_Constraint_8 [PeriodicConstraint]
Timing Expression [TimingExpression]
Periodic_Constraint_9 [PeriodicConstraint]
RC_PedalLDMtoBrCtrlOut [ReactionConstraint]
DependabilityPackage [EAPackage]
VerificationValidation [EAPackage]
TestEquipment [EAPackage]
FunctionalPackage [EAPackage]
FaultInjectionBehavior [FunctionBehavior]
FIFunction [DesignFunctionType]
LocalBBW_FDA [DesignFunctionType]
SubjectSystemFDA [DesignFunctionType]

Properties | Error Log | Problems

Periodic_Constraint_8 [PeriodicConstraint]

Advanced

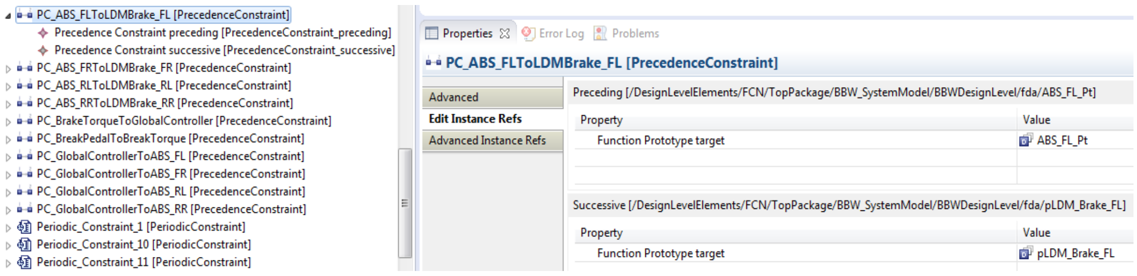
Property	Value
Category	
Event	EF_ABS_FL_Pt [/EASTADLExtensionElements/BBW_Timing/EF_ABS_FL_Pt]
Mode	
Name	5
Short Name	Periodic_Constraint_8
Uuid	

Figure 6.5: Periodic Constraint for ABS in AREAToP

Since the Periodic Constraint elements in EAST-ADL model lack the 'value' property due to a flaw in AREAToP tool, the periodicity values are entered in the 'Name' property as a workaround.

Precedence Constraint

The Precedence Constraint represents a constraint applied on the execution sequence of functional entities. For uni-directional functions without a precedence constraint, the functions will be executed according to their data dependencies. However, for bidirectional functions precedence constraint is mandatory [6]. Figure 6.6 shows a Precedence Constraint being added for the ABS and Brake Actuator module in AREAToP.


PC_ABS_FLToLDMBrake_FL [PrecedenceConstraint]
Precedence Constraint preceding [PrecedenceConstraint_preceding]
Precedence Constraint successive [PrecedenceConstraint_successive]
PC_ABS_FRToLDMBrake_FR [PrecedenceConstraint]
PC_ABS_RLToLDMBrake_RL [PrecedenceConstraint]
PC_ABS_RRToLDMBrake_RR [PrecedenceConstraint]
PC_BrakeTorqueToGlobalController [PrecedenceConstraint]
PC_BreakPedalToBreakTorque [PrecedenceConstraint]
PC_GlobalControllerToABS_FL [PrecedenceConstraint]
PC_GlobalControllerToABS_FR [PrecedenceConstraint]
PC_GlobalControllerToABS_RL [PrecedenceConstraint]
PC_GlobalControllerToABS_RR [PrecedenceConstraint]
Periodic_Constraint_1 [PeriodicConstraint]
Periodic_Constraint_10 [PeriodicConstraint]
Periodic_Constraint_11 [PeriodicConstraint]

Properties | Error Log | Problems

PC_ABS_FLToLDMBrake_FL [PrecedenceConstraint]

Advanced
Edit Instance Refs
Advanced Instance Refs

Preceding [/DesignLevelElements/FCN/TopPackage/BBW_SystemModel/BBWDesignLevel/fda/ABS_FL_Pt]

Property	Value
Function Prototype target	ABS_FL_Pt

Successive [/DesignLevelElements/FCN/TopPackage/BBW_SystemModel/BBWDesignLevel/fda/pLDM_Brake_FL]

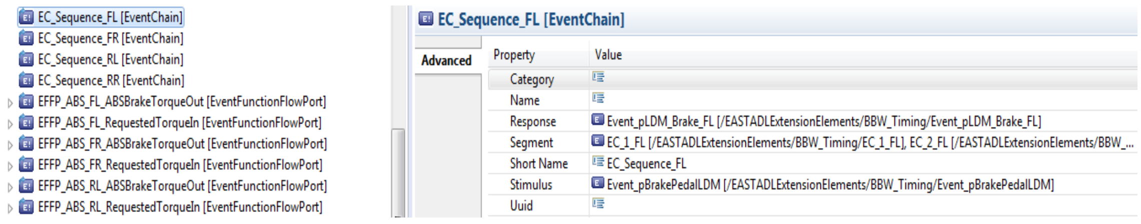
Property	Value
Function Prototype target	pLDM_Brake_FL

Figure 6.6: Precedence Constraint for ABS and Brake Actuator in AREAToP

In each Precedence Constraint element of the EAST-ADL model, a successive entity is an immediate successor of the preceding entity. The preceding and successive entities of each Precedence Constraint point to Design Function Prototype elements rather Design Function Type elements.

Event Chain

An Event Chain is a container for two events, a stimulus event and a response event that must be causally related [6]. Figure 6.7 shows an Event Chain being added for ABS module in AREAToP.



EC_Sequence_FL [EventChain]
EC_Sequence_FR [EventChain]
EC_Sequence_RL [EventChain]
EC_Sequence_RR [EventChain]
EFFP_ABS_FL_ABSBrakeTorqueOut [EventFunctionFlowPort]
EFFP_ABS_FL_RequestedTorqueIn [EventFunctionFlowPort]
EFFP_ABS_FR_ABSBrakeTorqueOut [EventFunctionFlowPort]
EFFP_ABS_FR_RequestedTorqueIn [EventFunctionFlowPort]
EFFP_ABS_RL_ABSBrakeTorqueOut [EventFunctionFlowPort]
EFFP_ABS_RL_RequestedTorqueIn [EventFunctionFlowPort]

EC_Sequence_FL [EventChain]

Advanced

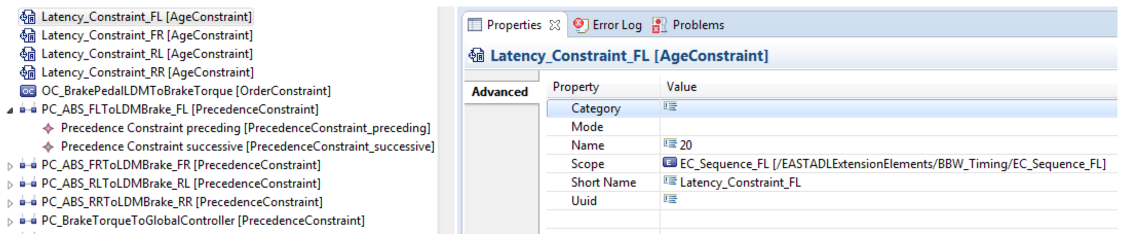
Property	Value
Category	
Name	
Response	Event_pLDM_Brake_FL [/EASTADLExtensionElements/BBW_Timing/Event_pLDM_Brake_FL]
Segment	EC_1_FL [/EASTADLExtensionElements/BBW_Timing/EC_1_FL], EC_2_FL [/EASTADLExtensionElements/BBW_...
Short Name	EC_Sequence_FL
Stimulus	Event_pBrakePedalLDM [/EASTADLExtensionElements/BBW_Timing/Event_pBrakePedalLDM]
Uuid	

Figure 6.7: Event Chain for BBW in AREAToP

An event chain can have several sub event chains that are added as segments to the parent in sequence as in Figure 6.7.

Age Constraint

The Age Constraint defines how long before each response a corresponding stimulus must have occurred. The Age Constraint is an alternative to the normal Delay Constraint for situations where the causal relation between event occurrences must be considered [6]. Figure 6.8 shows an Age constraint being added for an Event Chain in the BBW model.



Latency_Constraint_FL [AgeConstraint]
Latency_Constraint_FR [AgeConstraint]
Latency_Constraint_RL [AgeConstraint]
Latency_Constraint_RR [AgeConstraint]
OC_BrakePedalLDMToBrakeTorque [OrderConstraint]
PC_ABS_FLToLDMBrake_FL [PrecedenceConstraint]
Precedence Constraint preceding [PrecedenceConstraint_preceding]
Precedence Constraint successive [PrecedenceConstraint_successive]
PC_ABS_FRToLDMBrake_FR [PrecedenceConstraint]
PC_ABS_RLToLDMBrake_RL [PrecedenceConstraint]
PC_ABS_RRToLDMBrake_RR [PrecedenceConstraint]
PC_BrakeTorqueToGlobalController [PrecedenceConstraint]

Latency_Constraint_FL [AgeConstraint]

Advanced

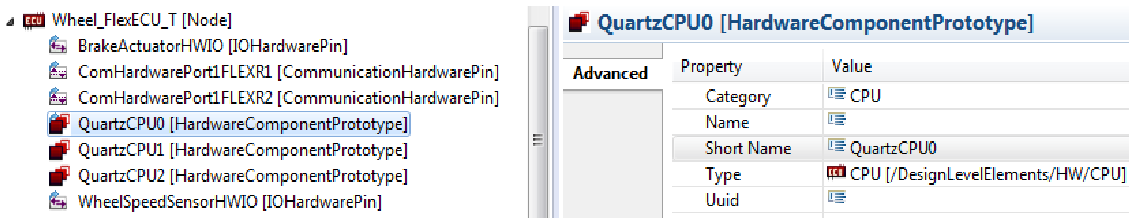
Property	Value
Category	
Mode	
Name	20
Scope	EC_Sequence_FL [/EASTADLExtensionElements/BBW_Timing/EC_Sequence_FL]
Short Name	Latency_Constraint_FL
Uuid	

Figure 6.8: Age Constraint for BBW in AREAToP

Since the Age Constraint elements in EAST-ADL model lack the 'value' property due to a flaw in AREAToP tool, the age values are entered in the 'Name' property as a workaround.

Hardware Component Prototype

The Hardware Component Prototype represents an occurrence of a hardware element based on the type of the Hardware Component Prototype [6]. Figure 6.9 shows Hardware Component Prototypes being added for ABS module in AREAToP.



- Wheel_FlexECU_T [Node]
  - BrakeActuatorHWIO [IOHardwarePin]
  - ComHardwarePort1FLEXR1 [CommunicationHardwarePin]
  - ComHardwarePort1FLEXR2 [CommunicationHardwarePin]
  - QuartzCPU0 [HardwareComponentPrototype]
  - QuartzCPU1 [HardwareComponentPrototype]
  - QuartzCPU2 [HardwareComponentPrototype]
  - WheelSpeedSensorHWIO [IOHardwarePin]

QuartzCPU0 [HardwareComponentPrototype]

Advanced

Property	Value
Category	CPU
Name	
Short Name	QuartzCPU0
Type	CPU [/DesignLevelElements/HW/CPU]
Uuid	

Figure 6.9: Hardware Component Prototypes of BBW in AREAToP

Preparing the XSLT file

Before starting the model transformation process, the transformation file (XSLT file) is made ready by utilizing the available details supported by EAST-ADL model which can be mapped to the AMALTHEA model. The details regarding the mapping are explained in Table 6.1 and implemented in the XSLT file as in Listing A.1.

Some additional details that are not supported by EAST-ADL model, but required by AMALTHEA model for simulation are also added in the XSLT file as shown in Listing A.1. During model transformation process these details get added to the resultant AMALTHEA model. The additional details are as follows,

1. Mapping Model
   - Details regarding the scheduler allocation, i.e., scheduler to processor core mapping.
   - Details regarding the task allocation, i.e., task to scheduler mapping.
   - Details regarding the runnable allocation, i.e., runnable to scheduler mapping.

2. OS Model
   - Defining an operating system that can be either a generic operating system or vendor specific operating system.
   - Defining a number of task schedulers within the defined operating system.
   - Defining a scheduling algorithm for each task schedulers defined.

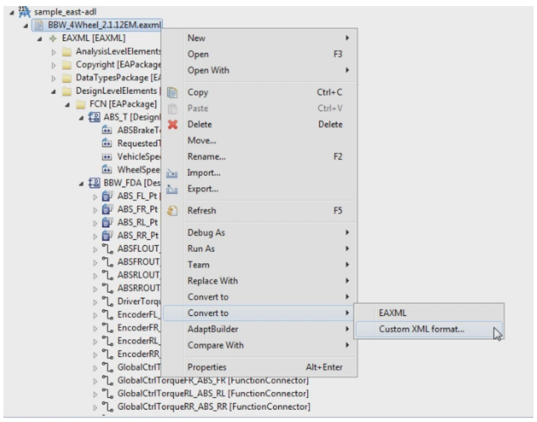
3. Hardware Model
   - Defining a Hardware System with Instructions Per Cycle details.
   - Defining several ECUs within the defined hardware system.
   - Defining several micro-controller units for each ECU.

Step 3 (Collect Results)

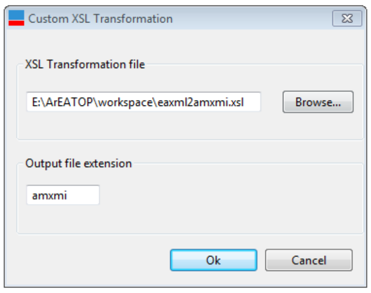
In this step, model transformation and simulation are performed.

Model Transformation Process

Once the transformation file is ready, the model transformation is carried out with the help of AREAToP tool as shown in Figure 6.10. The XSLT code used in this transformation process is shown in Listing A.1. The outcome of this model transformation process would be an AMALTHEA model for Brake-by-wire component with a file extension "amxmi". This file will be saved in the same directory as of the XSLT file.



(a) EAXML File Selection



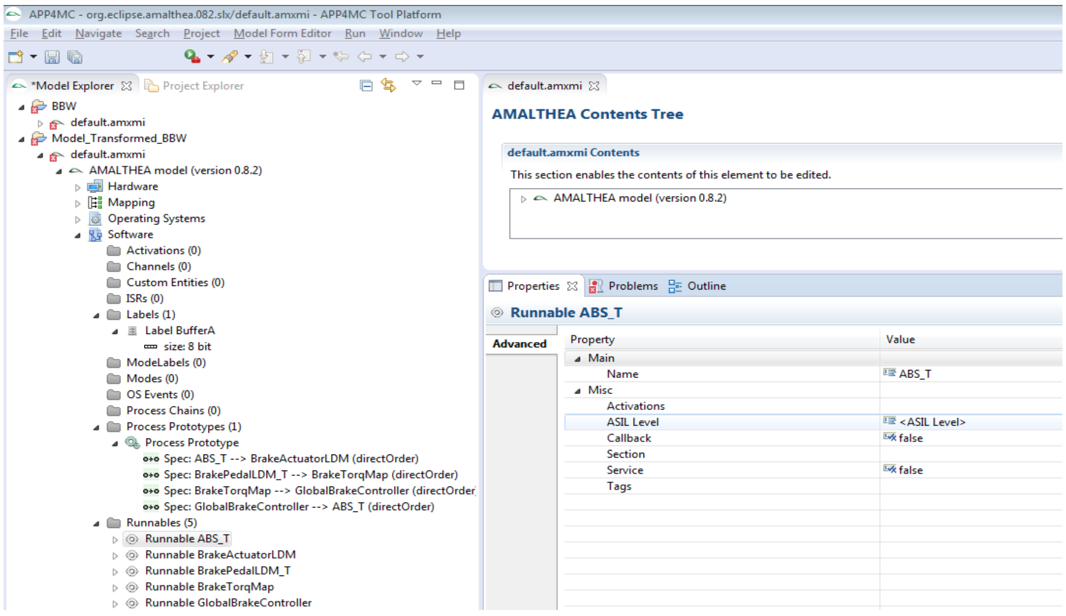
XSL Transformation file: E:\ArEATOP\workspace\eaxml2amxmi.xsl

Output file extension: amxmi

(b) XSLT File Selection

Figure 6.10: Model Transformation of EAST-ADL model in AREAToP

Editing AMALTHEA Model (Optional)


APP4MC - org.eclipse.amalthea.082.slx/default.amxmi - APP4MC Tool Platform
File Edit Navigate Search Project Model Form Editor Run Window Help
*Model Explorer
Project Explorer
default.amxmi
Properties
Problems
Outline
Advanced

- BBW
  - default.amxmi
- Model_Transformed_BBW
  - default.amxmi
    - AMALTHEA model (version 0.8.2)
      - Hardware
      - Mapping
      - Operating Systems
      - Software
        - Activations (0)
        - Channels (0)
        - Custom Entities (0)
        - ISRs (0)
        - Labels (1)
          - Label BufferA
            - size: 8 bit
        - ModelLabels (0)
        - Modes (0)
        - OS Events (0)
        - Process Chains (0)
        - Process Prototypes (1)
          - Process Prototype
            - Spec: ABS_T --> BrakeActuatorLDM (directOrder)
            - Spec: BrakePedalLDM_T --> BrakeTorqMap (directOrder)
            - Spec: BrakeTorqMap --> GlobalBrakeController (directOrder
            - Spec: GlobalBrakeController --> ABS_T (directOrder)
        - Runnables (5)
          - Runnable ABS_T
          - Runnable BrakeActuatorLDM
          - Runnable BrakePedalLDM_T
          - Runnable BrakeTorqMap
          - Runnable GlobalBrakeController

AMALTHEA Contents Tree

default.amxmi Contents

This section enables the contents of this element to be edited.

AMALTHEA model (version 0.8.2)

Runnable ABS_T

Property	Value
Main	
Name	ABS_T
Misc	
Activations	
ASIL Level	<ASIL Level>
Callback	false
Section	
Service	false
Tags	

Figure 6.11: Editing AMALTHEA model in APP4MC

If necessary, before simulation the resultant AMALTHEA model can be edited graphically by importing into the APP4MC tool as shown in Figure 6.11.

Simulation and Analysis

The resultant AMALTHEA model is fed into the SLX tool platform for simulation and analysis. Once the model is imported SLX will convert the given AMALTHEA input system model to SLX internal representation followed by the generation of task graph as shown in Figure 6.12.

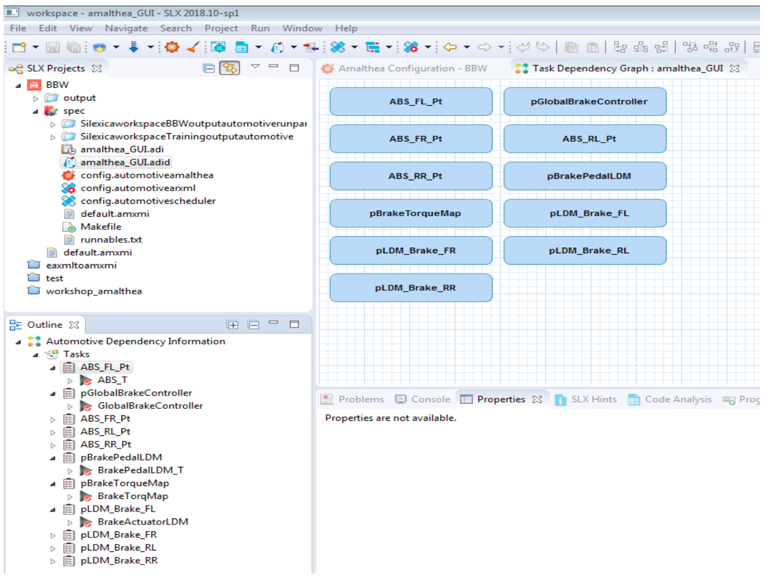
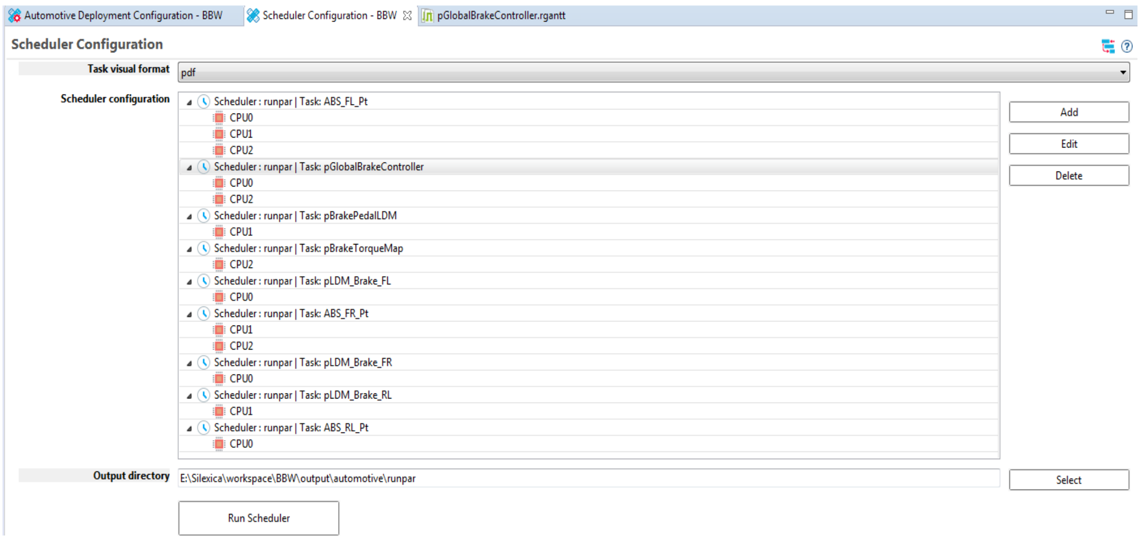


Figure 6.12: Task Graph for imported BBW in Silexica

The task graph displays all the tasks of the imported BBW model along with its properties like linked runnables, etc. Generating multi-core schedules is one of the key features of the SLX tool chain. For each specified tasks in the system model, a multi-core schedule can be generated which allows the tasks to be distributed to multiple processors. This task splitting mechanism can be applied to all or only a subset of processors of the target platform [10].


Automotive Deployment Configuration - BBW | Scheduler Configuration - BBW | pGlobalBrakeController.rgantt

Scheduler Configuration

Task visual format: pdf

Scheduler configuration:

- Scheduler : runpar | Task: ABS_FL_Pt
  - CPU0
  - CPU1
  - CPU2
- Scheduler : runpar | Task: pGlobalBrakeController
  - CPU0
  - CPU2
- Scheduler : runpar | Task: pBrakePedalLDM
  - CPU1
- Scheduler : runpar | Task: pBrakeTorqueMap
  - CPU2
- Scheduler : runpar | Task: pLDM_Brake_FL
  - CPU0
- Scheduler : runpar | Task: ABS_FR_Pt
  - CPU1
  - CPU2
- Scheduler : runpar | Task: pLDM_Brake_FR
  - CPU0
- Scheduler : runpar | Task: pLDM_Brake_RL
  - CPU1
- Scheduler : runpar | Task: ABS_RL_Pt
  - CPU0

Add | Edit | Delete

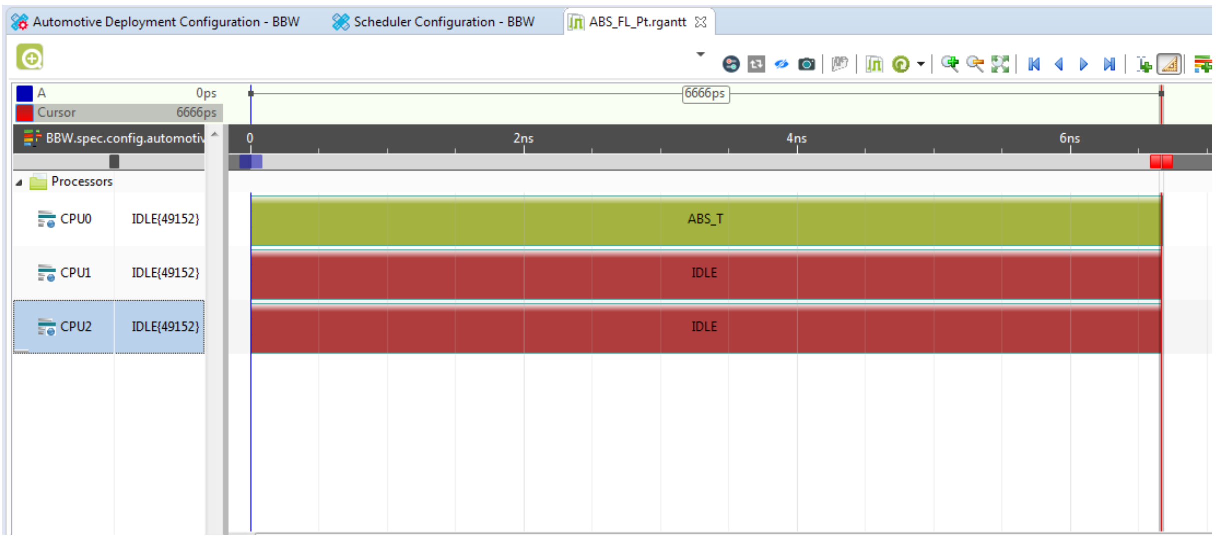
Output directory: E:\Silexica\workspace\BBW\output\automotive\runpar

Select

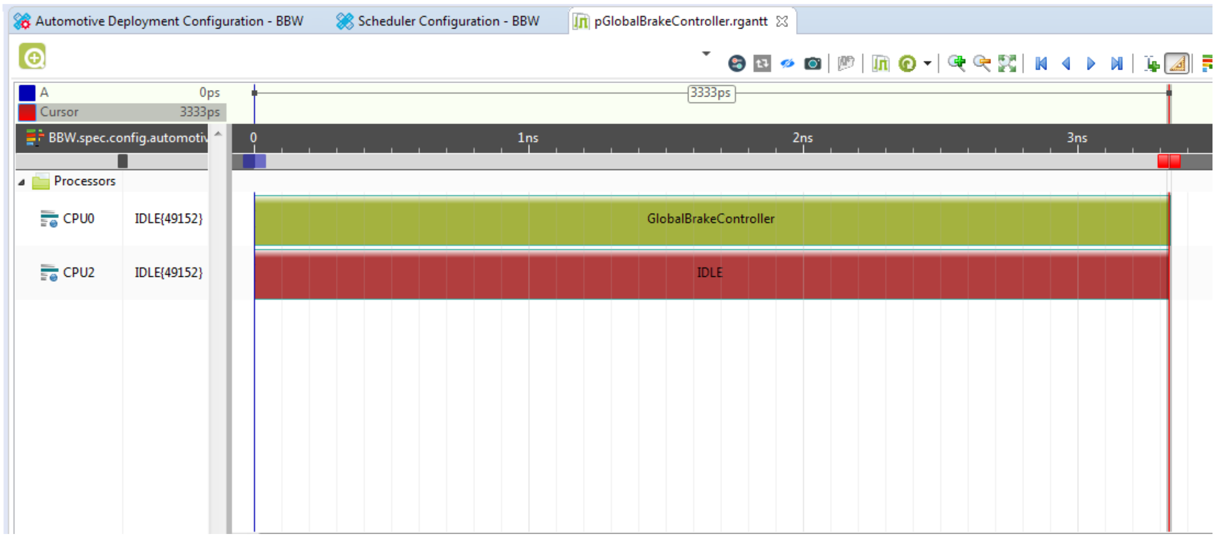
Run Scheduler

Figure 6.13: Scheduler Configuration Editor in SLX

The Scheduler Configuration editor used to configure the creation of such schedules is shown in Figure 6.13. Multi-core schedules of the runnables in each task are provided via corresponding task specific Gantt charts as shown in Figure 6.14. The green and red area in Figure 6.14 represents “running” and “idle” states of the tasks respectively. SLX is not capable of generating a combined schedule for all tasks in the system. Also, SLX does not provide any means to further analyze the simulation results.



(a) ABS



(b) Global Brake Controller

Figure 6.14: Schedule for Tasks in SLX

Step 4 (Evaluate)

In this step, the results collected from step 3 are compared against the evaluation criteria.

Criterion 1: Successful transformation of source architectural model to AMALTHEA model

This criterion is satisfied by the current plan/iteration because the XSLT code in Listing A.1 successfully transformed the BBW EAST-ADL model to BBW AMALTHEA model.

Criterion 2: Ability to verify response time for each task of the software component in a multi-core system

This criterion is not satisfied by the current plan/iteration because the SLX tool does not support Stimuli Model in AMALTHEA, that is responsible to activate processes in a defined manner. Because of this several instances of the task cannot be scheduled, which is one of the key factors to be considered while measuring response time.

Criterion 3: Ability to verify response time for each event-chain of the software component in a multi-core system

This criterion is not satisfied by the current plan/iteration because the SLX tool does not support Event Model in AMALTHEA, that provide information about different event entities that can be used for the modeling of event chains and for some timing constraints.

Criterion 4: Ability to measure the load distribution of tasks and runnables on the processing cores

This criterion is not satisfied by the current plan/iteration because the SLX tool does not have the support to analyze load distribution of tasks and runnables on the processing cores.

Thus it is clear from the evaluation results that the current plan/iteration has succeeded in transforming the source architectural model to AMALTHEA model. However, the SLX tool suite used in this iteration has only limited support in utilizing AMALTHEA model during simulation. Hence a new tool will be used for simulation in the next iteration.

6.1.2 Iteration 2:

Goal

- Effective utilization of AMALTHEA model during simulation.

Step 1 (Plan)

Architectural Model Type

During this iteration we select EAST-ADL architectural model for experimentation.

Sample Application Model

The thesis intends to use the BBW EAST-ADL application model, that pre-exist within vehicle dynamics control department at AB Volvo. The model is available in the URL: `http://www.east-adl.info/Specification/V2.1.12/EAST-ADL_V2.1.12.zip`.

Model Transformation Tool

The thesis intends to use the AREAToP Technology Demonstrator tool developed by AB Volvo for model transformation.

XML Transformation Language

During this iteration, we continue to use XSLT Version 1.0 as the XML transformation language.

Simulation and Analysis Tool

During this iteration, we pick Timing Architects tool platform for simulation and analysis as the Silexica tool in the previous iteration failed to effectively utilize the AMALTHEA model during simulation. A trial version (version 19.1) of the TA tool suite is used for the experiment.

Step 2 (Act)

No activity is performed in this step, i.e., the modified data rich Brake-by-wire EAST-ADL model and XSLT file from the previous iteration (Iteration 1) will be directly used in this iteration.

Step 3 (Collect Results)

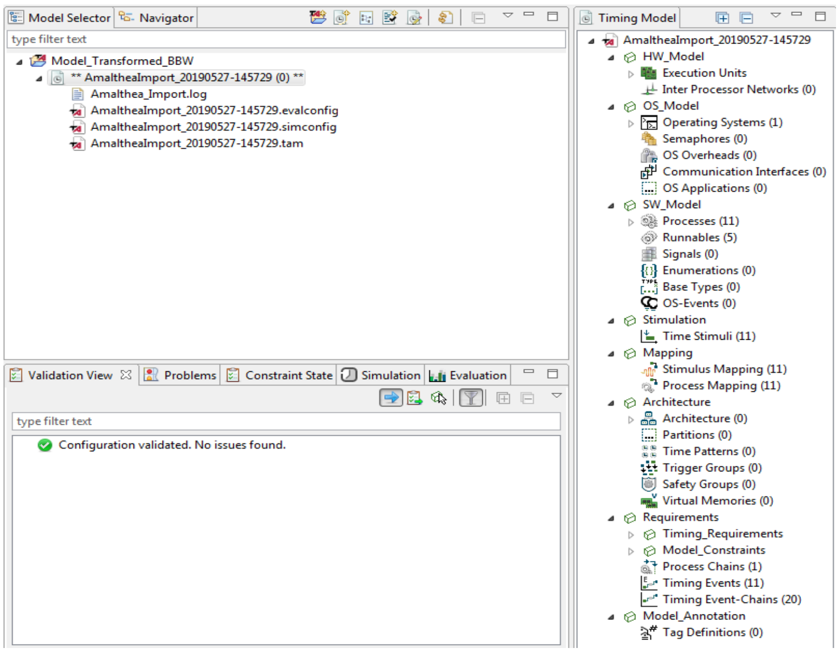
In this step, model transformation and simulation are performed.

Model Transformation Process

No activity is done for performing model transformation, i.e., the resultant AMALTHEA model from the previous iteration (Iteration 1) will be directly used in this iteration.

Simulation and Analysis

The resultant BBW AMALTHEA model is fed into the TA tool suite platform for simulation and analysis. Once the AMALTHEA model is imported, the TA tool suite will analyze the model for any inconsistencies in the model through *Validation View* as in Figure 6.15. For example, the periodic tasks might be missing the recurrence time value in the model.



```
Model_Transformed_BBW
  ** AmaltheaImport_20190527-145729 (0) **
    Amalthea_Import.log
    AmaltheaImport_20190527-145729.evalconfig
    AmaltheaImport_20190527-145729.simconfig
    AmaltheaImport_20190527-145729.tam
```

```
Configuration validated. No issues found.
```

```
AmaltheaImport_20190527-145729
  HW_Model
    Execution Units
    Inter Processor Networks (0)
  OS_Model
    Operating Systems (1)
    Semaphores (0)
    OS Overheads (0)
    Communication Interfaces (0)
    OS Applications (0)
  SW_Model
    Processes (11)
    Runnables (5)
    Signals (0)
    Enumerations (0)
    Base Types (0)
    OS-Events (0)
  Stimulation
    Time Stimuli (11)
  Mapping
    Stimulus Mapping (11)
    Process Mapping (11)
  Architecture
    Architecture (0)
    Partitions (0)
    Time Patterns (0)
    Trigger Groups (0)
    Safety Groups (0)
    Virtual Memories (0)
  Requirements
    Timing_Requirements
    Model_Constraints
    Process Chains (1)
    Timing Events (11)
    Timing Event-Chains (20)
  Model_Annotation
    Tag Definitions (0)
```

Figure 6.15: Validating AMALTHEA model in TA tool suite

Once the model is error free, simulation is started. The simulation results (scheduling of tasks) of the BBW model on a multi-core (triple-core) platform are visualized via gantt chart as shown in Figure 6.16. The grey and dark-green areas in Figure 6.16 represents "activated" and "running" states of the tasks respectively. The blue, pink, red and green connections in Figure 6.16 represent different event chains in the BBW model.

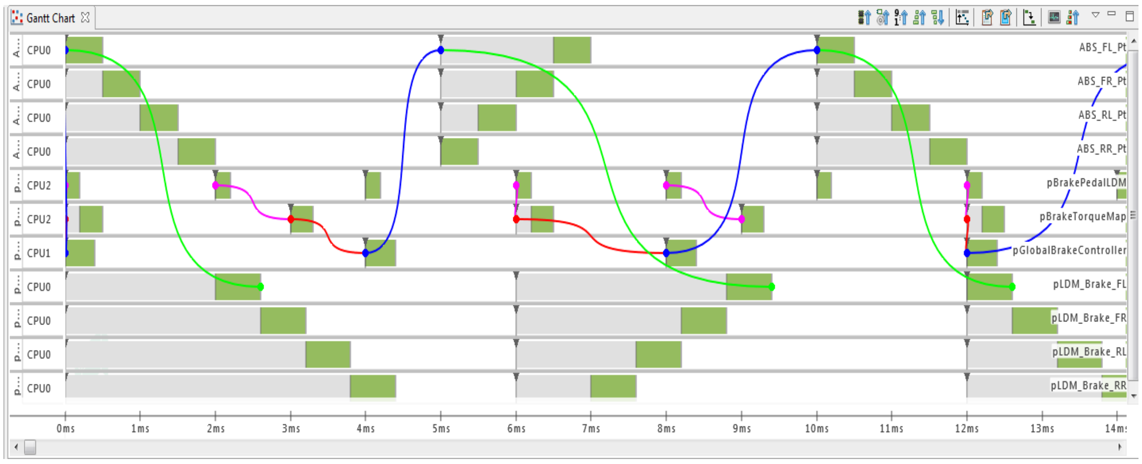
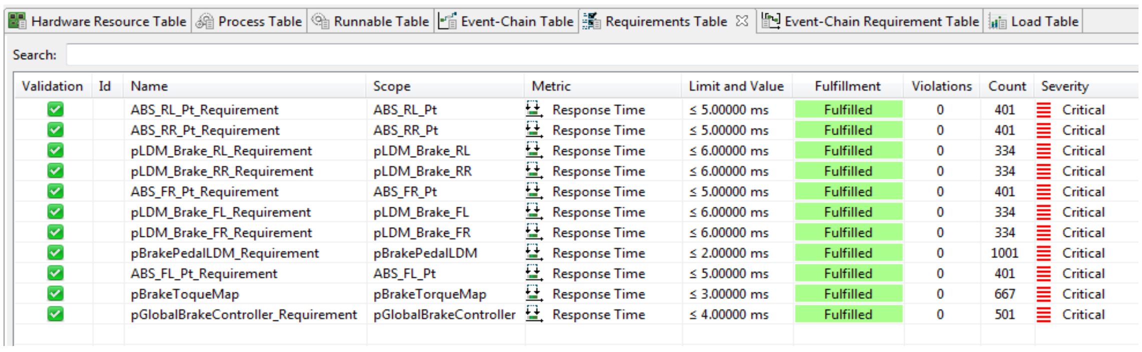


Figure 6.16: Simulation of BBW model in TA tool suite

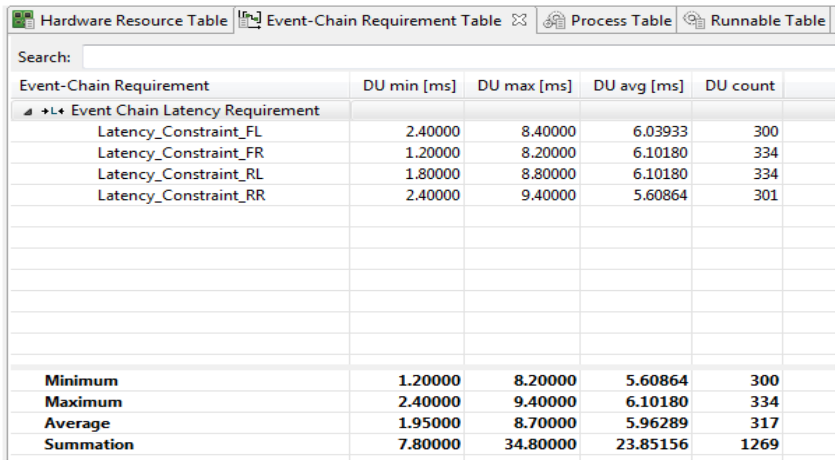
The TA tool suite also provides means to analyze the simulation results via different metric tables. Figure 6.17 shows the response time requirements table for different tasks in the BBW model. This requirements table contains details regarding requirement name, task name, response time value, fulfillment status, task occurrence count and task severity level.



Validation	Id	Name	Scope	Metric	Limit and Value	Fulfillment	Violations	Count	Severity
✓		ABS_RL_Pt_Requirement	ABS_RL_Pt	Response Time	≤ 5.00000 ms	Fulfilled	0	401	Critical
✓		ABS_RR_Pt_Requirement	ABS_RR_Pt	Response Time	≤ 5.00000 ms	Fulfilled	0	401	Critical
✓		pLDM_Brake_RL_Requirement	pLDM_Brake_RL	Response Time	≤ 6.00000 ms	Fulfilled	0	334	Critical
✓		pLDM_Brake_RR_Requirement	pLDM_Brake_RR	Response Time	≤ 6.00000 ms	Fulfilled	0	334	Critical
✓		ABS_FR_Pt_Requirement	ABS_FR_Pt	Response Time	≤ 5.00000 ms	Fulfilled	0	401	Critical
✓		pLDM_Brake_FL_Requirement	pLDM_Brake_FL	Response Time	≤ 6.00000 ms	Fulfilled	0	334	Critical
✓		pLDM_Brake_FR_Requirement	pLDM_Brake_FR	Response Time	≤ 6.00000 ms	Fulfilled	0	334	Critical
✓		pBrakePedalLDM_Requirement	pBrakePedalLDM	Response Time	≤ 2.00000 ms	Fulfilled	0	1001	Critical
✓		ABS_FL_Pt_Requirement	ABS_FL_Pt	Response Time	≤ 5.00000 ms	Fulfilled	0	401	Critical
✓		pBrakeToqueMap	pBrakeTorqueMap	Response Time	≤ 3.00000 ms	Fulfilled	0	667	Critical
✓		pGlobalBrakeController_Requirement	pGlobalBrakeController	Response Time	≤ 4.00000 ms	Fulfilled	0	501	Critical

Figure 6.17: Requirements table showing Response Time requirements for BBW model in TA tool suite

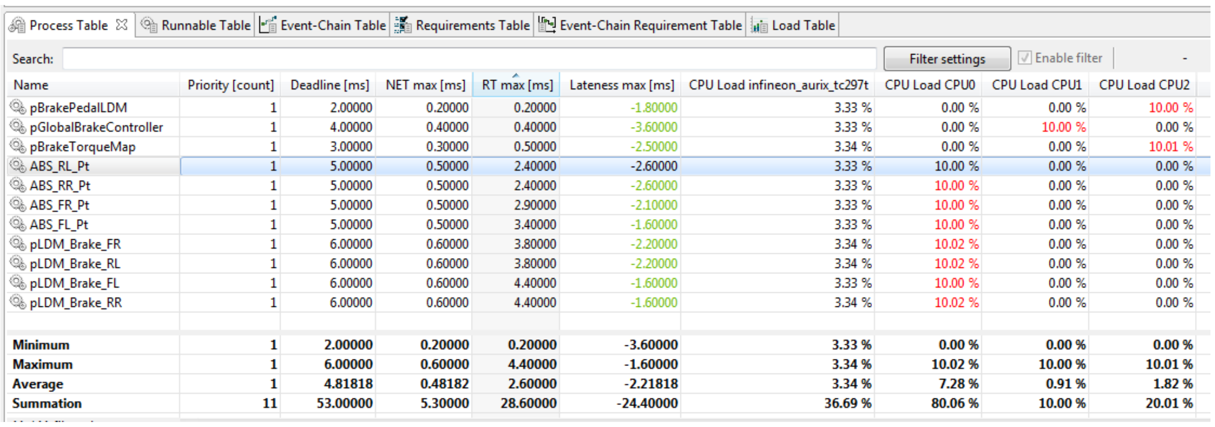
Figure 6.18 shows the event chain requirement table for the BBW model. This requirement table contains details regarding minimum duration, maximum duration, average duration and occurrence count of all event chains in the BBW model. However, there is no information regarding the fulfillment status of the Event Chain Latency Requirement. Hence the user has to manually compare the latency requirement values against maximum duration occurred to know the fulfillment status of all event chains in the BBW model.



Event-Chain Requirement	DU min [ms]	DU max [ms]	DU avg [ms]	DU count
Event Chain Latency Requirement				
Latency_Constraint_FL	2.40000	8.40000	6.03933	300
Latency_Constraint_FR	1.20000	8.20000	6.10180	334
Latency_Constraint_RL	1.80000	8.80000	6.10180	334
Latency_Constraint_RR	2.40000	9.40000	5.60864	301
Minimum	**1.20000**	**8.20000**	**5.60864**	**300**
Maximum	**2.40000**	**9.40000**	**6.10180**	**334**
Average	**1.95000**	**8.70000**	**5.96289**	**317**
Summation	**7.80000**	**34.80000**	**23.85156**	**1269**

Figure 6.18: Event Chain Requirement table for BBW model in TA tool suite

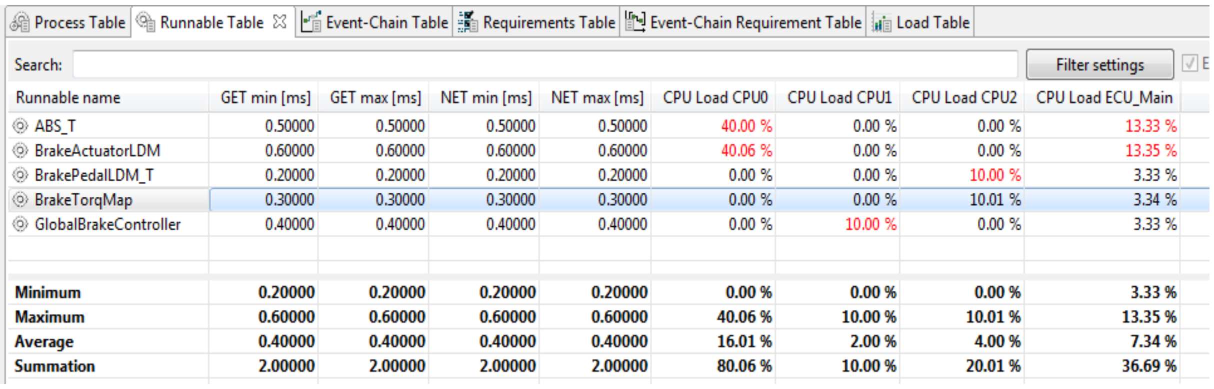
Figure 6.19 shows the process table with task metrics for BBW model. The process table contains details regarding task priority, deadline for tasks, net execution time of tasks (time from the moment of start to termination of a task), maximum response time of tasks, maximum delay/lateness and CPU load (utilization) of each task on different cores.



Name	Priority [count]	Deadline [ms]	NET max [ms]	RT max [ms]	Lateness max [ms]	CPU Load infineon_aurix_tc297t	CPU Load CPU0	CPU Load CPU1	CPU Load CPU2
pBrakePedalLDM	1	2.00000	0.20000	0.20000	-1.80000	3.33 %	0.00 %	0.00 %	10.00 %
pGlobalBrakeController	1	4.00000	0.40000	0.40000	-3.60000	3.33 %	0.00 %	10.00 %	0.00 %
pBrakeTorqueMap	1	3.00000	0.30000	0.50000	-2.50000	3.34 %	0.00 %	0.00 %	10.01 %
ABS_RL_Pt	1	5.00000	0.50000	2.40000	-2.60000	3.33 %	10.00 %	0.00 %	0.00 %
ABS_RR_Pt	1	5.00000	0.50000	2.40000	-2.60000	3.33 %	10.00 %	0.00 %	0.00 %
ABS_FR_Pt	1	5.00000	0.50000	2.90000	-2.10000	3.33 %	10.00 %	0.00 %	0.00 %
ABS_FL_Pt	1	5.00000	0.50000	3.40000	-1.60000	3.33 %	10.00 %	0.00 %	0.00 %
pLDM_Brake_FR	1	6.00000	0.60000	3.80000	-2.20000	3.34 %	10.02 %	0.00 %	0.00 %
pLDM_Brake_RL	1	6.00000	0.60000	3.80000	-2.20000	3.34 %	10.02 %	0.00 %	0.00 %
pLDM_Brake_FL	1	6.00000	0.60000	4.40000	-1.60000	3.33 %	10.00 %	0.00 %	0.00 %
pLDM_Brake_RR	1	6.00000	0.60000	4.40000	-1.60000	3.34 %	10.02 %	0.00 %	0.00 %
Minimum	**1**	**2.00000**	**0.20000**	**0.20000**	**-3.60000**	**3.33 %**	**0.00 %**	**0.00 %**	**0.00 %**
Maximum	**1**	**6.00000**	**0.60000**	**4.40000**	**-1.60000**	**3.34 %**	**10.02 %**	**10.00 %**	**10.01 %**
Average	**1**	**4.81818**	**0.48182**	**2.60000**	**-2.21818**	**3.34 %**	**7.28 %**	**0.91 %**	**1.82 %**
Summation	**11**	**53.00000**	**5.30000**	**28.60000**	**-24.40000**	**36.69 %**	**80.06 %**	**10.00 %**	**20.01 %**

Figure 6.19: Process table showing task metrics for BBW model in TA tool suite

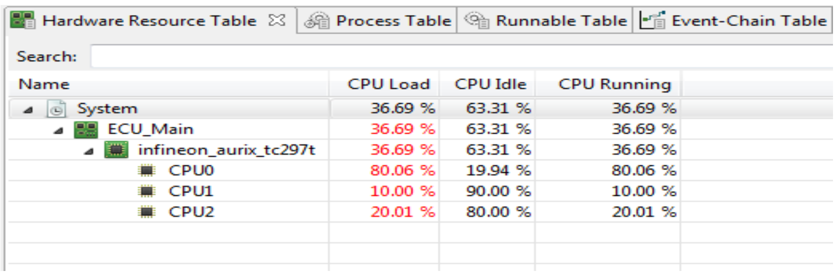
Figure 6.20 shows the runnable table with runnable metrics for BBW model. The runnable table contains details regarding minimum and maximum gross execution time (time from the moment of activation to termination of a runnable), minimum and maximum net execution time (time from the moment of start to termination of a runnable) and CPU load (utilization) of each runnable on different cores.



Runnable name	GET min [ms]	GET max [ms]	NET min [ms]	NET max [ms]	CPU Load CPU0	CPU Load CPU1	CPU Load CPU2	CPU Load ECU_Main
ABS_T	0.50000	0.50000	0.50000	0.50000	40.00 %	0.00 %	0.00 %	13.33 %
BrakeActuatorLDM	0.60000	0.60000	0.60000	0.60000	40.06 %	0.00 %	0.00 %	13.35 %
BrakePedalLDM_T	0.20000	0.20000	0.20000	0.20000	0.00 %	0.00 %	10.00 %	3.33 %
BrakeTorqMap	0.30000	0.30000	0.30000	0.30000	0.00 %	0.00 %	10.01 %	3.34 %
GlobalBrakeController	0.40000	0.40000	0.40000	0.40000	0.00 %	10.00 %	0.00 %	3.33 %
Minimum	**0.20000**	**0.20000**	**0.20000**	**0.20000**	**0.00 %**	**0.00 %**	**0.00 %**	**3.33 %**
Maximum	**0.60000**	**0.60000**	**0.60000**	**0.60000**	**40.06 %**	**10.00 %**	**10.01 %**	**13.35 %**
Average	**0.40000**	**0.40000**	**0.40000**	**0.40000**	**16.01 %**	**2.00 %**	**4.00 %**	**7.34 %**
Summation	**2.00000**	**2.00000**	**2.00000**	**2.00000**	**80.06 %**	**10.00 %**	**20.01 %**	**36.69 %**

Figure 6.20: Runnable table showing runnable metrics for BBW model in TA tool suite

Figure 6.21 shows the hardware resource table with CPU metrics for BBW model. The hardware resource table contains details regarding CPU running and Idle percentage for different cores.



Name	CPU Load	CPU Idle	CPU Running
System	36.69 %	63.31 %	36.69 %
ECU_Main	36.69 %	63.31 %	36.69 %
infineon_aurix_tc297t	36.69 %	63.31 %	36.69 %
CPU0	80.06 %	19.94 %	80.06 %
CPU1	10.00 %	90.00 %	10.00 %
CPU2	20.01 %	80.00 %	20.01 %

Figure 6.21: Hardware Resource table showing CPU metrics for BBW model in TA tool suite

Step 4 (Evaluate)

In this step, the results collected from step 3 are compared against the evaluation criteria.

Criterion 1: Successful transformation of source architectural model to AMALTHEA model

This criterion is satisfied by the current plan/iteration because the XSLT code in Listing A.1 successfully transformed the BBW EAST-ADL model to BBW AMALTHEA model.

Criterion 2: Ability to verify response time for each task of the software component in a multi-core system

This criterion is satisfied by the current plan/iteration, which is evident from the Figure 6.17.

Criterion 3: Ability to verify response time for each event-chain of the software component in a multi-core system

This criterion is satisfied by the current plan/iteration, which is evident from the Figure 6.18.

Criterion 4: Ability to measure the load distribution of tasks and runnables on the processing cores

This criterion is satisfied by the current plan/iteration, which is evident from the Figures 6.19 and 6.20.

Thus it is clear from the evaluation results that TA tool suite has effectively utilized AMALTHEA model during simulation. Hence it is concluded that an optimized plan is achieved since the plan satisfies all of the evaluation criteria. However, the thesis also intends to experiment with AUTOSAR model in the next iteration.

6.1.3 Iteration 3:

Goal

- Successful transformation of AUTOSAR model to AMALTHEA model.

Step 1 (Plan)

Architectural Model Type

During this iteration we select AUTOSAR architectural model for experimentation.

Sample Application Model

The thesis intends to use the BBW AUTOSAR application model that will be created during the “Act” phase of this iteration and does not pre-exist within AB Volvo. The created model is available in Listing A.3.

Model Transformation Tool

The thesis intends to use the AREAToP Technology Demonstrator tool developed by AB Volvo for model transformation.

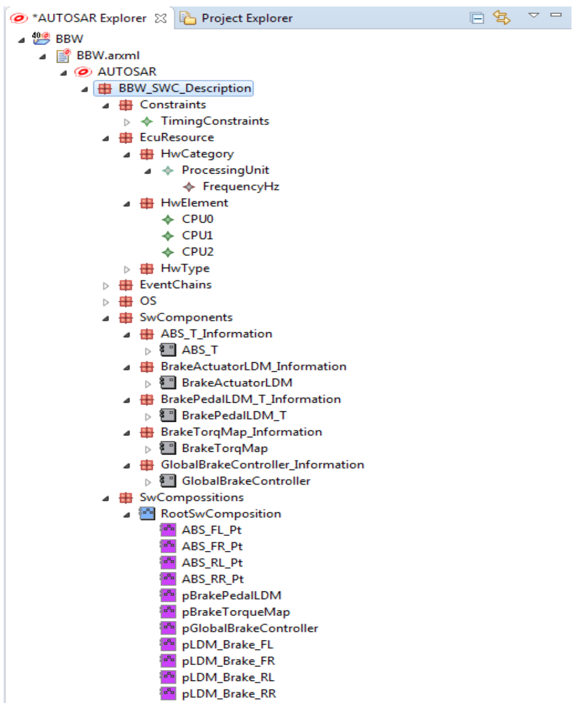
XML Transformation Language

During this iteration, we continue to use XSLT Version 1.0 as the XML transformation language.

Simulation and Analysis Tool

During this iteration, we continue to use the Timing Architects tool platform since it effectively utilized the AMALTHEA model during simulation.

Step 2 (Act)



```
*AUTOSAR Explorer    Project Explorer
BBW
  BBW.arxml
    AUTOSAR
      BBW_SWC_Description
        Constraints
          TimingConstraints
        EcuResource
          HwCategory
            ProcessingUnit
              FrequencyHz
          HwElement
            CPU0
            CPU1
            CPU2
          HwType
        EventChains
        OS
        SwComponents
          ABS_T_Information
            ABS_T
          BrakeActuatorLDM_Information
            BrakeActuatorLDM
          BrakePedalLDM_T_Information
            BrakePedalLDM_T
          BrakeTorqMap_Information
            BrakeTorqMap
          GlobalBrakeController_Information
            GlobalBrakeController
        SwCompossitions
          RootSwComposition
            ABS_FL_Pt
            ABS_FR_Pt
            ABS_RL_Pt
            ABS_RR_Pt
            pBrakePedalLDM
            pBrakeTorqueMap
            pGlobalBrakeController
            pLDM_Brake_FL
            pLDM_Brake_FR
            pLDM_Brake_RL
            pLDM_Brake_RR
```

Figure 6.22: AUTOSAR model of BBW in AREAToP

In this step, the AUTOSAR model for BBW component is developed from scratch. The thesis intends to use the AREAToP Technology Demonstrator tool developed by AB Volvo for creating the AUTOSAR model. The complete model is available in

Listing A.3, which contains details regarding runnables, tasks, stimulus, execution order, execution time, processing cores, events, event chains and timing constraints as in Figure 6.22.

After a careful analysis of the AUTOSAR and AMALTHEA models, the guidelines for mapping is framed as shown in Table 6.2. The guidelines for mapping are implemented in the XSLT as shown in Listing A.2, and will take effect during the model transformation process. Since the BBW AUTOSAR model is developed from scratch, all the details that it supports and that can be mapped to AMALTHEA model are added during the development process itself.

Table 6.2: Mapping between AUTOSAR and AMALTHEA elements during Model Transformation

AUTOSAR	**AMALTHEA**
SwComponentPrototype	Task (Software Model)
AtomicSwComponentType :: RunnableEntity	Runnable (Software Model)
Event TriggeringConstraint :: PeriodicEventTriggering	Periodic Stimulus (Stimuli Model)
TimingConstraint :: ExecutionOrderConstraint	OrderPrecedenceSpec (Software Model :: Process Prototype)
TimingConstraint :: ExecutionTimeConstraint	Instructions Constant (Software Model :: Runnable)
HwDescriptionEntity :: HwElement	Processing Unit (Hardware Model)
TimingDescriptionEvent :: TDEventSwcInternalBehavior	Process Event (Events Model)
TimingDescription :: TimingDescriptionEventChain	Event Chain (Constraints Model)
TimingConstraint :: LatencyTimingConstraint	Event Chain Latency Constraint (Constraints Model)

Preparing the XSLT file

Before starting the model transformation process, the transformation file (XSLT file) is made ready by utilizing the available details supported by AUTOSAR model which can be mapped to the AMALTHEA model. The details regarding the mapping are explained in Table 6.2 and implemented in the XSLT file as in Listing A.2.

Some additional details that are not supported by AUTOSAR model, but required by AMALTHEA model for simulation are also added in the XSLT file as shown in Listing A.2. During model transformation process these details get added to the resultant AMALTHEA model. The additional details are as follows,

1. OS Model
   - Defining a number of task schedulers within the defined operating system.
   - Defining a scheduling algorithm for each task schedulers defined.

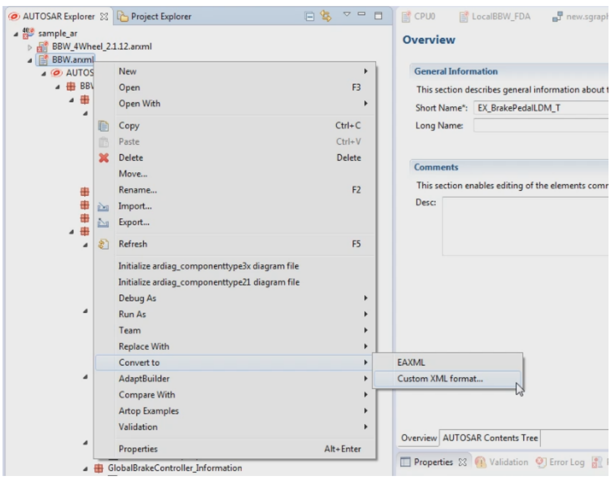
2. Mapping Model
   - Details regarding the scheduler allocation, i.e., scheduler to processor core mapping.
   - Details regarding the task allocation, i.e., task to scheduler mapping.
   - Details regarding the runnable allocation, i.e., runnable to scheduler mapping.

Step 3 (Collect Results)

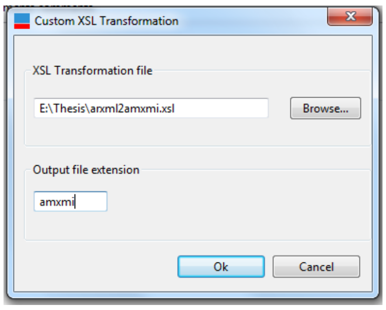
In this step, model transformation and simulation are performed.

Model Transformation Process

Once the transformation file is ready, the model transformation is carried out with the help of AREAToP tool as shown in Figure 6.23. The XSLT code used in this transformation process is shown in Listing A.2. The outcome of this model transformation process would be an AMALTHEA model for Brake-by-wire component with a file extension “amxmi”. This file will be saved in the same directory as of the XSLT file.



(a) ARXML File Selection



XSL Transformation file: E:\Thesis\arxml2amxmi.xsl

Output file extension: amxmi

(b) XSLT File Selection

Figure 6.23: Model Transformation of AUTOSAR model in AREAToP

Simulation and Analysis

No activity is done for simulation and analysis because for simplicity the thesis intends to have a single AMALTHEA model for BBW component that can be simulated and analyzed, i.e., the AMALTHEA model obtained by model transforming BBW EAST-ADL model will be an exact copy of the AMALTHEA model obtained by model transforming BBW AUTOSAR model. In general, this would not be the case. Thus the simulation and analysis results will be the same as of previous iteration (Iteration 2).

Step 4 (Evaluate)

In this step, the results collected from step 3 are compared against the evaluation criteria.

Criterion 1: Successful transformation of source architectural model to AMALTHEA model

This criterion is satisfied by the current plan/iteration because the XSLT code in Listing A.2 successfully transformed the BBW AUTOSAR model to BBW AMALTHEA model.

Criterion 2: Ability to verify response time for each task of the software component in a multi-core system

This criterion is satisfied by the current plan/iteration, which is evident from the Figure 6.17.

Criterion 3: Ability to verify response time for each event-chain of the software component in a multi-core system

This criterion is satisfied by the current plan/iteration, which is evident from the Figure 6.18.

Criterion 4: Ability to measure the load distribution of tasks and runnables on the processing cores

This criterion is satisfied by the current plan/iteration, which is evident from the Figures 6.19 and 6.20.

Thus it is clear from the evaluation results that the AUTOSAR model has been successfully model transformed to AMALTHEA model and the TA tool suite has effectively utilized the resultant AMALTHEA model during simulation. With no further plans in experimenting with architectural models apart from EAST-ADL and AUTOSAR, the iteration is stopped.

7 Discussion

7.1 Comparison of the effort to extend the EAST-ADL and AUTOSAR models to AMALTHEA

Table 7.1: Comparison of model elements covered by AUTOSAR, AMALTHEA and EAST-ADL

Concept	**AUTOSAR**	**AMALTHEA**	**EAST-ADL**
Hardware Model • ECUs, cores, memories, peripherals	YES	YES	PARTLY
Operating System Model • Abstract OS description • Task schedulers • Scheduling algorithm	PARTLY	YES	NO
Dynamic Software Architecture • Stimuli (Activation pattern) • Function runtimes • Complex call graph	PARTLY	YES	PARTLY
Mapping • Scheduler to processor core mapping • Task to scheduler mapping • Runnable to scheduler mapping	PARTLY	YES	NO
Timing Requirements • Task deadlines • Response Time • Events and Event Chain requirements • Periodicity	YES	YES	YES
Software Design Constraints • Execution Order Constraint • Data Age Constraint • Data Coherency Groups • Affinity Constraints	YES	YES	PARTLY
Static Software Architecture • Software component description • Function and data definitions • Communication interfaces	YES	YES	YES

In this thesis, comparison towards the effort made to extend the EAST-ADL and AUTOSAR model into AMALTHEA is very basic, i.e., the comparison will only be on the model elements that are mandatory for simulating an AMALTHEA model. Based on Table 6.1, 6.2 and Table 7.1, it is clear that AUTOSAR has a close resemblance to AMALTHEA compared to EAST-ADL. Thus it is concluded that the effort needed to extend the EAST-ADL model is more than the effort to extend the AUTOSAR model

7.2 Engineering Work Flow

Optimized Plan

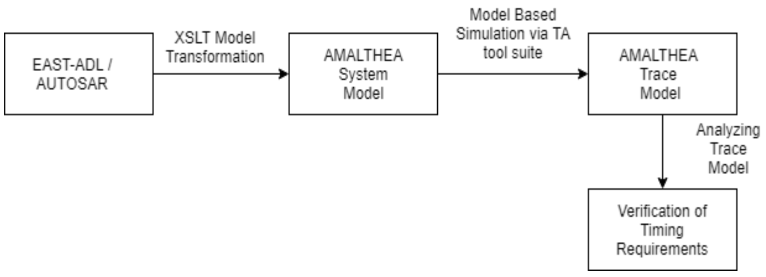

EAST-ADL / AUTOSAR
XSLT Model Transformation
AMALTHEA System Model
Model Based Simulation via TA tool suite
AMALTHEA Trace Model
Analyzing Trace Model
Verification of Timing Requirements

Figure 7.1: Optimized Plan

From the results of each iteration of the action research loop in the previous chapter, it was clear that an optimized plan/design choice as in Figure 7.1 was reached in Iteration 2, as the plan satisfied all of the evaluation criteria. This optimized plan can be integrated into a work flow as shown in Figure 7.2.

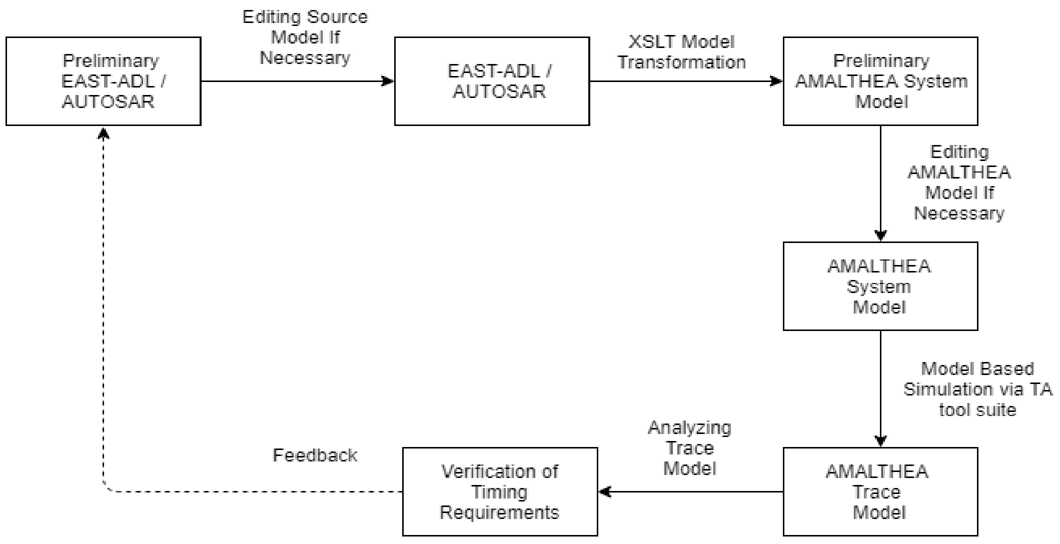

Preliminary EAST-ADL / AUTOSAR
Editing Source Model If Necessary
EAST-ADL / AUTOSAR
XSLT Model Transformation
Preliminary AMALTHEA System Model
Editing AMALTHEA Model If Necessary
AMALTHEA System Model
Model Based Simulation via TA tool suite
AMALTHEA Trace Model
Analyzing Trace Model
Verification of Timing Requirements
Feedback

Figure 7.2: Engineering Work Flow Loop

This work flow will be beneficial for embedded engineers in taking timing-aware design decisions during the software development process. The engineers can perform several iterations of the engineering work flow loop with possible changes in timing properties, scheduling design, etc., based on the feedback obtained from every iteration, until all the timing requirements are met.

Merits and Demerits

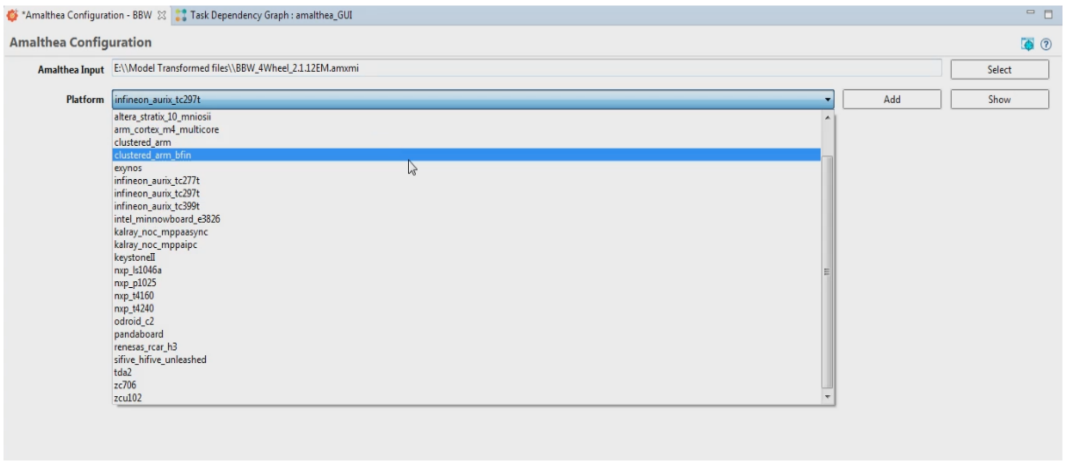
- This approach of transforming the existing architectural models to AMALTHEA model will save plenty of development time spent on creating a new AMALTHEA model from scratch, thus increasing the productivity. Incorporating automated model transformation via XSLT scripts rather manual model transformation also aids in reducing the development time.

- One disadvantage with the model transformation script used in this thesis is that it can generate only a basic AMALTHEA model comprising of only the mandatory model elements necessary for simulation. In order to add additional details, the developer has to manually add the details to the AMALTHEA model using APP4MC tool as shown in Subsection 6.1.1.

7.3 Limitations in Simulation/Analysis Tool Platform

The tool platforms used in this thesis for simulating/analyzing AMALTHEA models have certain limitations as discussed below,

Limitations in Silexica Tool Platform


*Amalthea Configuration - BBW
Task Dependency Graph : amalthea_GUI
Select
Add
Show

Amalthea Configuration

Amalthea Input: E:\\Model Transformed files\\BBW_4Wheel_2.1.12EM.amxmi

Platform: infineon_aurix_tc297t

altera_stratix_10_mniosii
arm_cortex_m4_multicore
clustered_arm
clustered_arm_bfin
exynos
infineon_aurix_tc277t
infineon_aurix_tc297t
infineon_aurix_tc399t
intel_minnowboard_e3826
kalray_noc_mppaasync
kalray_noc_mppaipc
keystoneII
nxp_ls1046a
nxp_p1025
nxp_t4160
nxp_t4240
odroid_c2
pandaboard
renesas_rcar_h3
sifive_hifive_unleashed
tda2
zc706
zcu102

Figure 7.3: Platform Models Supported by Silexica Tool [10]

- The current version, SLX-2018.10 sp-1 of the tool supports only AMALTHEA xml-schema-definition version 0.8.2. A single Software Model that will contain all the task, runnable and variable information is required for importing

[10]. Meanwhile, current xml-schema-definition supported by AMALTHEA is version 0.9.4.

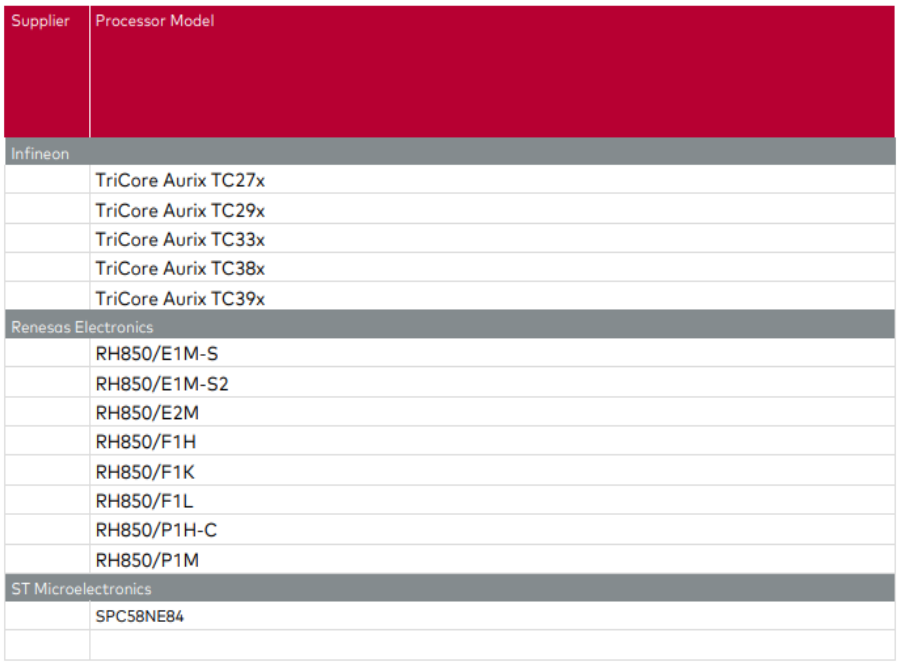
- When importing an AMALTHEA model, the tool only supports a model with Hardware Model that matches one of the many platform models supported by the SLX tool as shown in Figure 7.3. The criteria for a matching model are that the number of cores and the core names match those given in an SLX platform model.

- The current version, SLX-2018.10 sp-1 of the tool does not support Stimuli Model in AMALTHEA, that contains stimulus and clock objects. A stimulus is responsible to activate a task/process.

- The current version, SLX-2018.10 sp-1 of the tool does not support Constraints Model in AMALTHEA, that provide information about the different kind of constraints that has to be satisfied during execution.

- The current version, SLX-2018.10 sp-1 of the tool does not support Event Model in AMALTHEA, that provide information about different event entities that can be used for the modeling of event chains and for some timing constraints.

Limitations in TA Tool Platform



Supplier	Processor Model
Infineon	
	TriCore Aurix TC27x
	TriCore Aurix TC29x
	TriCore Aurix TC33x
	TriCore Aurix TC38x
	TriCore Aurix TC39x
Renesas Electronics	
	RH850/E1M-S
	RH850/E1M-S2
	RH850/E2M
	RH850/F1H
	RH850/F1K
	RH850/F1L
	RH850/P1H-C
	RH850/P1M
ST Microelectronics	
	SPC58NE84

Figure 7.4: Supported Processor Models for TA Tool Suite [11]

- Currently available processor models in TA tool suite are shown in Figure 7.4, but the tool suite also allows the integration of supplier specific processor models depending upon the software license issued [11].

- Currently supported operating system models by TA tool suite are MICROSA-OS, Tresos AutoCore and Tresos Safety OS [11].

- Current version of TA tool suite (version 19.1.0) supports only AMALTHEA xml-schema-definition version 0.9.2. Meanwhile current xml-schema-definition supported by AMALTHEA is version 0.9.4.

7.4 Possible Future Work

- The AMALTHEA model, modelled in this thesis is primary, which contains the basic information necessary for simulation. In future, a complete AMALTHEA model utilizing all the features that it supports can be modelled (by extending the model transformation script) and simulated.

- The thesis focused only on the verification of the timing behavior of application software. Further research into optimization of software distribution on processing cores can be conducted.

8
Conclusion

The work presented in this thesis describes a timing-aware model-driven methodology for software development, with a special focus on embedded multi-core systems. The thesis gives an overview of three modelling approaches, namely EAST-ADL, AUTOSAR and AMALTHEA which are commonly used within the scope of multi-core automotive software development. By going into detail and comparing these models with respect to their meta models, methodologies and implementations, the thesis was able to identify similarities and differences between these models. The thesis also proposed and succeeded in model transforming the existing EAST-ADL/AUTOSAR model to AMALTHEA model, which provides extended support for multi-core timing verification. Several iterations of the methodology are carried out with possible changes in the design plan, and an optimized design is picked based on evaluation criteria. Considering the implementation alternatives, a model transformation would be harder without automated support. The thesis solved this problem by introducing an automated model transformation via XSLT. Finally, the thesis leverages the generated AMALTHEA model for model-based precise timing verification with the help of supported simulation/analysis tools. Though AMALTHEA is recent to the automotive industry, the development support from corporates and its open source license makes it an aspiring approach for modelling the future complex multi-core ECU architectures. The outcome of this thesis will be beneficial for embedded engineers in taking timing-aware design decisions during the development cycle.

Bibliography

[1] Bucaioni, A., Addazi, L., Cicchetti, A., Ciccozzi, F., Eramo, R., Mubeen, S. and Sjödin, M., 2018. MoVES: A Model-Driven Methodology for Vehicular Embedded Systems. IEEE Access, 6, pp.6424-6445.

[2] Höttger, R., Mackamul, H., Sailer, A., Steghöfer, J.P. and Tessmer, J., 2017. APP4MC: Application platform project for multi-and many-core systems. it-Information Technology, 59(5), pp.243-251.

[3] Höttger, R., Lauschner, U., Närdemann, P., Heisig, P., Wolff, C., Kamsties, E. and Igel, B., Teaching distributed and parallel systems with APP4MC. In International Symposium on Embedded Systems and Trends in Teaching Engineering (pp. 126-134).

[4] Sailer, A., Schmidhuber, S., Hempe, M., Deubzer, M. and Mottok, J., 2016, April. Distributed Multi-Core Development in the Automotive Domain-A Practical Comparison of ASAM MDX vs. AUTOSAR vs. AMALTHEA. In ARCS 2016; 29th International Conference on Architecture of Computing Systems (pp. 1-8). VDE.

[5] AMALTHEA4public consortium, "APP4MC help documentation", 2019. [Online]. Available: `https://www.eclipse.org/app4mc/documentation/`.

[6] EAST-ADL Domain Model Specification version V2.1.12. [Online]. Available: `http://www.east-adl.info/Specification/V2.1.12/EAST-ADL-Specification_V2.1.12.pdf`

[7] Cuenot, P., Frey, P., Johansson, R., Lönn, H., Papadopoulos, Y., Reiser, M.O., Sandberg, A., Servat, D., Kolagari, R.T., Törngren, M. and Weber, M., 2007, November. 11 the east-adl architecture description language for automotive embedded software. In Dagstuhl Workshop on Model-Based Engineering of Embedded Real-Time Systems (pp. 297-307). Springer, Berlin, Heidelberg.

[8] Qureshi, T.N., Chen, D., Lönn, H. and Törngren, M., 2011, September. From EAST-ADL to AUTOSAR software architecture: a mapping scheme. In European Conference on Software Architecture (pp. 328-335). Springer, Berlin, Heidelberg.

[9] Wikipedia contributors. (2018, October 7). EAST-ADL. In Wikipedia, The Free Encyclopedia. Retrieved 20:31, May 1, 2019, from `https://en.wikipedia.org/w/index.php?title=EAST-ADL&oldid=862836615`

[10] SILEXICA, [Online]: `https://www.silexica.com/`

[11] VECTOR, "TA Tool Suite" [Online]: `https://www.vector.com/int/en/products/products-a-z/software/ta-tool-suite/`

[12] Wikipedia contributors. (2019, March 25). XML transformation language. In Wikipedia, The Free Encyclopedia. Retrieved 14:02, May 20, 2019, from `https://en.wikipedia.org/w/index.php?title=XML_transformation_language&oldid=889470678`

[13] Wikipedia contributors. (2019, May 15). XSLT. In Wikipedia, The Free Encyclopedia. Retrieved 15:04, May 20, 2019, from `https://en.wikipedia.org/w/index.php?title=XSLT&oldid=897247948`

[14] W3schools, [Online]: `https://www.w3schools.com/xml/xsl_intro.asp`

[15] AUTOSAR, Software Component Template, [Online]: `https://www.autosar.org/fileadmin/user_upload/standards/classic/4-3/AUTOSAR_TPS_SoftwareComponentTemplate.pdf`

[16] The AUTOSAR Consortium, [Online]: `https://www.autosar.org/`

[17] AUTOSAR, Layered Software Architecture, [Online]: `https://www.autosar.org/fileadmin/user_upload/standards/classic/4-3/AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf`

[18] EATOP Project, [Online]: `https://www.eclipse.org/eatop/`

[19] Artop Project, [Online]: `https://www.artop.org/`

[20] embitel, [Online]: `https://www.embitel.com/blog/embedded-blog/decoding-the-component-concept-of-the-application-layer-in-autosar`

[21] SCHEID Automotive, [Online]: `https://automotive.wiki/index.php/Interface`

[22] Mubeen, S., Nolte, T., Sjödin, M., Lundbäck, J. and Lundbäck, K.L., 2019. Supporting timing analysis of vehicular embedded systems through the refinement of timing constraints. Software Systems Modeling, 18(1), pp.39-69.

A

Appendix 1

A.1 Code Snippet

Listing A.1: XSLT code for transforming EAST-ADL to AMALTHEA model

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- xsl stylesheet declaration -->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:ea="http://east-adl.info/2.1.12" xmlns:exslt="http://exslt.org/common" version="1.0" extension-element-prefixes="exslt">
  <xsl:output method="xml" indent="yes" />
  <xsl:key name="orderPrecedence" match="orderPrecedenceSpec" use="@origin" />
  <xsl:key name="instruction_count" match="ea:EXECUTION-TIME-CONSTRAINT" use="ea:NAME" />
  <!-- This template is used to extract a sub-string within a string after the last occurence of the delimiter '/' -->
  <xsl:template name="substring-after-last">
    <xsl:param name="string" />
    <xsl:param name="delimiter" />
    <xsl:choose>
      <xsl:when test="contains($string, $delimiter)">
        <xsl:call-template name="substring-after-last">
          <xsl:with-param name="string" select="substring-after($string, $delimiter)" />
          <xsl:with-param name="delimiter" select="$delimiter" />
        </xsl:call-template>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="$string" />
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>
  <!-- This template is used to get the processPrototypes -->
  <xsl:template name="get_processPrototypes">
    <!-- Here we are loading some nodes to a variable to get a Result Tree Fragment -->
    <xsl:variable name="RTF">
      <xsl:for-each select="//ea:PRECEDENCE-CONSTRAINT">
        <xsl:variable name="preceding_task">
          <xsl:call-template name="substring-after-last">
            <xsl:with-param name="string" select="./ea:PRECEDING-IREF/ea:FUNCTION-PROTOTYPE-TARGET-REF" />
            <xsl:with-param name="delimiter" select="'/'" />
          </xsl:call-template>
        </xsl:variable>
        <xsl:variable name="successive_task">
          <xsl:call-template name="substring-after-last">
            <xsl:with-param name="string" select="./ea:SUCCESSIVE-IREF/ea:FUNCTION-PROTOTYPE-TARGET-REF" />
            <xsl:with-param name="delimiter" select="'/'" />
          </xsl:call-template>
        </xsl:variable>
        <xsl:variable name="preceding_runnable">
          <xsl:call-template name="get-preceding-successive-runnables">
```

```
            <xsl:with-param name="functionPrototype" select="$preceding_task" />
          </xsl:call-template>
        </xsl:variable>
        <xsl:variable name="successive_runnable">
          <xsl:call-template name="get-preceding-successive-runnables">
            <xsl:with-param name="functionPrototype" select="$successive_task" />
          </xsl:call-template>
        </xsl:variable>
        <orderPrecedenceSpec origin="{concat($preceding_runnable, '?type=Runnable')
    }" target="{concat($successive_runnable, '?type=Runnable')}" orderType="
    directOrder" />
      </xsl:for-each>
    </xsl:variable>
    <!-- Now we take the RTF and convert it to a node-set so we can process it yet
    again! -->
    <xsl:variable name="set_RTF" select="exslt:node-set($RTF)" />
    <processPrototypes name="">
      <!-- Go and play with the new set -->
      <xsl:for-each select="$set_RTF/orderPrecedenceSpec[generate-id(.) =
    generate-id(key('orderPrecedence', @origin)[1])]">
        <xsl:copy-of select="." />
      </xsl:for-each>
    </processPrototypes>
  </xsl:template>
  <!-- This template is used to get the stimuli name -->
  <xsl:template name="get-stimuli-name">
    <xsl:param name="taskName" />
    <xsl:for-each select="//ea:EVENT-FUNCTION/ea:FUNCTION-IREF/
    ea:FUNCTION-PROTOTYPE-TARGET-REF">
      <xsl:variable name="EventFunctionName" select="../../ea:SHORT-NAME" />
      <xsl:variable name="Var2">
        <xsl:call-template name="substring-after-last">
          <xsl:with-param name="string" select="current()" />
          <xsl:with-param name="delimiter" select="'/'" />
        </xsl:call-template>
      </xsl:variable>
      <!-- <varoutput>
        <xsl:value-of select="$Var2"/>
      </varoutput> -->
      <xsl:if test="$taskName = $Var2">
        <xsl:for-each select="//ea:PERIODIC-CONSTRAINT/ea:EVENT-REF">
          <xsl:variable name="Var3">
            <xsl:call-template name="substring-after-last">
              <xsl:with-param name="string" select="current()" />
              <xsl:with-param name="delimiter" select="'/'" />
            </xsl:call-template>
          </xsl:variable>
          <xsl:if test="$Var3 = $EventFunctionName">
            <xsl:value-of select="concat(../ea:SHORT-NAME, '?type=PeriodicStimulus
    ')" />
          </xsl:if>
        </xsl:for-each>
      </xsl:if>
    </xsl:for-each>
  </xsl:template>
  <!-- This template is used to get the instruction count for each runnable -->
  <xsl:template name="get-instruction-count">
    <xsl:param name="runnableName" />
    <xsl:for-each select="//ea:DESIGN-FUNCTION-PROTOTYPE/ea:TYPE-TREF[@TYPE='
    DESIGN-FUNCTION-TYPE'] | //ea:DESIGN-FUNCTION-PROTOTYPE/ea:TYPE-TREF[@TYPE='
    LOCAL-DEVICE-MANAGER']">
      <xsl:variable name="Var4">
        <xsl:call-template name="substring-after-last">
          <xsl:with-param name="string" select="current()" />
          <xsl:with-param name="delimiter" select="'/'" />
        </xsl:call-template>
      </xsl:variable>
      <xsl:if test="$runnableName = $Var4">
        <xsl:variable name="prototypeName" select="../ea:SHORT-NAME" />
        <xsl:for-each select="//ea:EVENT-FUNCTION-FLOW-PORT/ea:PORT-IREF/
    ea:FUNCTION-PROTOTYPE-REF">
          <xsl:variable name="Var5">
            <xsl:call-template name="substring-after-last">
              <xsl:with-param name="string" select="current()" />
              <xsl:with-param name="delimiter" select="'/'" />
            </xsl:call-template>
          </xsl:variable>
          <xsl:if test="$prototypeName = $Var5">
            <xsl:variable name="functionFlowPortName" select="../../ea:SHORT-NAME" />
            <xsl:for-each select="//ea:EXECUTION-TIME-CONSTRAINT[generate-id() = generate-id(key('instruction_count',ea:NAME)[1])]">
              <xsl:variable name="Var6">
                <xsl:call-template name="substring-after-last">
                  <xsl:with-param name="string" select="ea:START-REF" />
                  <xsl:with-param name="delimiter" select="'/'" />
                </xsl:call-template>
              </xsl:variable>
              <xsl:if test="$functionFlowPortName = $Var6">
                <xsl:value-of select="ea:NAME" />
              </xsl:if>
            </xsl:for-each>
          </xsl:if>
        </xsl:for-each>
      </xsl:if>
    </xsl:for-each>
  </xsl:template>
  <!-- This template is used to resolve the precedence contraints -->
  <xsl:template name="get-preceding-successive-runnables">
    <xsl:param name="functionPrototype" />
    <xsl:for-each select="//ea:SYSTEM-MODEL[1]/ea:DESIGN-LEVEL[1]/ea:FUNCTIONAL-DESIGN-ARCHITECTURE[1]/ea:TYPE-TREF[1][@TYPE='DESIGN-FUNCTION-TYPE']">
      <xsl:variable name="myVar">
        <xsl:call-template name="substring-after-last">
          <xsl:with-param name="string" select="current()" />
          <xsl:with-param name="delimiter" select="'/'" />
        </xsl:call-template>
      </xsl:variable>
      <xsl:for-each select="//ea:DESIGN-FUNCTION-PROTOTYPE[ancestor::ea:DESIGN-FUNCTION-TYPE[ea:SHORT-NAME= $myVar]]/ea:TYPE-TREF[@TYPE='DESIGN-FUNCTION-TYPE'] | //ea:DESIGN-FUNCTION-PROTOTYPE[ancestor::ea:DESIGN-FUNCTION-TYPE[ea:SHORT-NAME=$myVar]]/ea:TYPE-TREF[@TYPE='LOCAL-DEVICE-MANAGER']">
        <xsl:variable name="taskName" select="../ea:SHORT-NAME" />
        <xsl:if test="$functionPrototype = $taskName">
          <xsl:variable name="Var7">
            <xsl:call-template name="substring-after-last">
              <xsl:with-param name="string" select="current()" />
              <xsl:with-param name="delimiter" select="'/'" />
            </xsl:call-template>
          </xsl:variable>
          <varoutput>
            <xsl:value-of select="$Var7" />
          </varoutput>
        </xsl:if>
      </xsl:for-each>
    </xsl:for-each>
  </xsl:template>
  <!-- xsl template declaration:
template tells the xlst processor about which section of xml
document to be formatted. It takes an XPath expression.
In our case, it is matching  root element -->
  <xsl:template match="/">
    <!-- New Line -->
    <xsl:text />
    <am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.2" xmlns:xmi="http://www.omg.org/XMI" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:version="2.0">
      <swModel>
        <xsl:for-each select="//ea:SYSTEM-MODEL[1]/ea:DESIGN-LEVEL[1]/ea:FUNCTIONAL-DESIGN-ARCHITECTURE[1]/ea:TYPE-TREF[1][@TYPE='DESIGN-FUNCTION-TYPE']">
          <!-- <varoutput>
        <xsl:value-of select="$myVar"/>
     </varoutput> -->
          <xsl:variable name="myVar">
            <xsl:call-template name="substring-after-last">
              <xsl:with-param name="string" select="current()" />
              <xsl:with-param name="delimiter" select="'/'" />
            </xsl:call-template>
          </xsl:variable>
          <xsl:for-each select="//ea:DESIGN-FUNCTION-PROTOTYPE[ancestor::ea:DESIGN-FUNCTION-TYPE[ea:SHORT-NAME= $myVar]]/ea:TYPE-TREF[@TYPE='DESIGN-FUNCTION-TYPE'] | //ea:DESIGN-FUNCTION-PROTOTYPE[ancestor::ea:DESIGN-FUNCTION-TYPE[ea:SHORT-NAME=$myVar]]/ea:TYPE-TREF[@TYPE='LOCAL-DEVICE-MANAGER']">
            <xsl:variable name="taskName" select="../ea:SHORT-NAME" />
            <xsl:variable name="Var1">
              <xsl:call-template name="substring-after-last">
                <xsl:with-param name="string" select="current()" />
                <xsl:with-param name="delimiter" select="'/'" />
              </xsl:call-template>
            </xsl:variable>
            <xsl:variable name="ref1">
              <xsl:value-of select="concat($Var1, '?type=Runnable')" />
            </xsl:variable>
            <xsl:variable name="call_seq_name">
              <xsl:value-of select="concat('CS_',$taskName)" />
            </xsl:variable>
            <xsl:variable name="ref2">
              <xsl:call-template name="get-stimuli-name">
                <xsl:with-param name="taskName" select="$taskName" />
              </xsl:call-template>
            </xsl:variable>
            <tasks name="{$taskName}" stimuli="{$ref2}" preemption="preemptive" multipleTaskActivationLimit="1">
              <callGraph>
                <graphEntries xsi:type="am:CallSequence" name="{$call_seq_name}">
                  <calls xsi:type="am:TaskRunnableCall" runnable="{$ref1}" />
                </graphEntries>
              </callGraph>
            </tasks>
          </xsl:for-each>
        </xsl:for-each>
        <xsl:for-each select="//ea:DESIGN-FUNCTION-TYPE[ea:IS-ELEMENTARY = 'true'] | //ea:LOCAL-DEVICE-MANAGER[ea:IS-ELEMENTARY = 'true']">
          <xsl:variable name="runnableName" select="ea:SHORT-NAME" />
          <xsl:variable name="ref3">
            <xsl:call-template name="get-instruction-count">
              <xsl:with-param name="runnableName" select="$runnableName" />
            </xsl:call-template>
          </xsl:variable>
          <runnables name="{$runnableName}" callback="false" service="false">
            <runnableItems xsi:type="am:ExecutionNeed">
              <default key="Instructions">
                <value xsi:type="am:NeedConstant" value="{$ref3}" />
              </default>
            </runnableItems>
          </runnables>
        </xsl:for-each>
        <xsl:call-template name="get_processPrototypes" />
      </swModel>
      <hwModel>
        <definitions xsi:type="am:ProcessingUnitDefinition" name="CPU0_type" puType="CPU" features="Instructions/IPC_1.0?type=HwFeature" />
        <definitions xsi:type="am:ProcessingUnitDefinition" name="CPU1_type" puType="CPU" features="Instructions/IPC_1.0?type=HwFeature" />
        <definitions xsi:type="am:ProcessingUnitDefinition" name="CPU2_type" puType="CPU" features="Instructions/IPC_1.0?type=HwFeature" />
```

```
      <featureCategories name="Instructions" featureType="performance">
        <features name="IPC_1.0" value="1.0" />
      </featureCategories>
      <structures name="infineon_aurix_tc297t_model" structureType="System">
        <structures name="ECU_Main" structureType="ECU">
          <structures name="infineon_aurix_tc297t" structureType="Microcontroller">
            <modules xsi:type="am:ProcessingUnit" name="CPU0" frequencyDomain="QuartzCPU0?type=FrequencyDomain" definition="CPU0_type?type=ProcessingUnitDefinition" />
            <modules xsi:type="am:ProcessingUnit" name="CPU1" frequencyDomain="QuartzCPU1?type=FrequencyDomain" definition="CPU1_type?type=ProcessingUnitDefinition" />
            <modules xsi:type="am:ProcessingUnit" name="CPU2" frequencyDomain="QuartzCPU2?type=FrequencyDomain" definition="CPU2_type?type=ProcessingUnitDefinition" />
          </structures>
        </structures>
      </structures>
      <xsl:for-each select="//ea:HARDWARE-COMPONENT-PROTOTYPE">
        <xsl:variable name="name" select="ea:SHORT-NAME" />
        <xsl:variable name="HW_Type">
          <xsl:call-template name="substring-after-last">
            <xsl:with-param name="string" select="ea:TYPE-TREF" />
            <xsl:with-param name="delimiter" select="'/'" />
          </xsl:call-template>
        </xsl:variable>
        <xsl:for-each select="//ea:NODE">
          <xsl:if test="ea:SHORT-NAME = $HW_Type">
            <xsl:variable name="EX_Rate" select="ea:EXECUTION-RATE" />
            <xsl:if test="$EX_Rate != ''">
              <xsl:variable name="clock_freq" select="ea:EXECUTION-RATE" />
              <domains xsi:type="am:FrequencyDomain" name="{$name}" clockGating="false">
                <defaultValue value="{$clock_freq}" unit="Hz" />
              </domains>
            </xsl:if>
          </xsl:if>
        </xsl:for-each>
      </xsl:for-each>
    </hwModel>
    <osModel>
      <operatingSystems name="MICROSAR">
        <taskSchedulers name="MICROSAR_task_scheduler_CPU0">
          <schedulingAlgorithm xsi:type="am:OSEK" />
        </taskSchedulers>
        <taskSchedulers name="MICROSAR_task_scheduler_CPU1">
          <schedulingAlgorithm xsi:type="am:OSEK" />
        </taskSchedulers>
        <taskSchedulers name="MICROSAR_task_scheduler_CPU2">
          <schedulingAlgorithm xsi:type="am:OSEK" />
        </taskSchedulers>
      </operatingSystems>
    </osModel>
    <stimuliModel>
      <xsl:for-each select="//ea:PERIODIC-CONSTRAINT">
        <stimuli xsi:type="am:PeriodicStimulus" name="{ea:SHORT-NAME}">
          <offset value="0" unit="ms" />
          <recurrence value="{ea:NAME}" unit="ms" />
        </stimuli>
      </xsl:for-each>
    </stimuliModel>
    <eventModel>
      <xsl:for-each select="//ea:EVENT-FUNCTION">
        <xsl:variable name="eventName" select="ea:SHORT-NAME" />
        <xsl:variable name="eventType" select="ea:NAME" />
        <xsl:variable name="entity_name">
          <xsl:call-template name="substring-after-last">
            <xsl:with-param name="string" select="ea:FUNCTION-IREF/ea:FUNCTION-PROTOTYPE-TARGET-REF" />
            <xsl:with-param name="delimiter" select="'/'" />
          </xsl:call-template>
        </xsl:variable>
        <xsl:variable name="entity">
          <xsl:value-of select="concat($entity_name, '?type=Task')" />
        </xsl:variable>
        <events xsi:type="am:ProcessEvent" name="{$eventName}" eventType="{$eventType}" entity="{$entity}" />
      </xsl:for-each>
    </eventModel>
    <constraintsModel>
      <xsl:for-each select="//ea:EVENT-CHAIN">
        <xsl:variable name="eventChainName" select="ea:SHORT-NAME" />
        <xsl:variable name="segment_refs" select="ea:SEGMENT-REFS" />
        <xsl:if test="$segment_refs != ''">
          <xsl:variable name="stimulus">
            <xsl:call-template name="substring-after-last">
              <xsl:with-param name="string" select="ea:STIMULUS-REF" />
              <xsl:with-param name="delimiter" select="'/'" />
            </xsl:call-template>
          </xsl:variable>
          <xsl:variable name="concat_stimulus">
            <xsl:value-of select="concat($stimulus, '?type=ProcessEvent')" />
          </xsl:variable>
          <xsl:variable name="response">
            <xsl:call-template name="substring-after-last">
              <xsl:with-param name="string" select="ea:RESPONSE-REF" />
              <xsl:with-param name="delimiter" select="'/'" />
            </xsl:call-template>
          </xsl:variable>
          <xsl:variable name="concat_response">
            <xsl:value-of select="concat($response, '?type=ProcessEvent')" />
          </xsl:variable>
          <eventChains name="{$eventChainName}" stimulus="{$concat_stimulus}" response="{$concat_response}">
            <xsl:for-each select="ea:SEGMENT-REFS/ea:SEGMENT-REF">
              <xsl:variable name="segments_name">
                <xsl:call-template name="substring-after-last">
                  <xsl:with-param name="string" select="current()" />
                  <xsl:with-param name="delimiter" select="'/'" />
                </xsl:call-template>
              </xsl:variable>
              <xsl:for-each select="//ea:EVENT-CHAIN">
                <xsl:if test="$segments_name = ea:SHORT-NAME">
                  <xsl:variable name="stimulus1">
                    <xsl:call-template name="substring-after-last">
                      <xsl:with-param name="string" select="ea:STIMULUS-REF" />
                      <xsl:with-param name="delimiter" select="'/'" />
                    </xsl:call-template>
                  </xsl:variable>
                  <xsl:variable name="concat_stimulus1">
                    <xsl:value-of select="concat($stimulus1, '?type=ProcessEvent')" />
                  </xsl:variable>
                  <xsl:variable name="response1">
                    <xsl:call-template name="substring-after-last">
                      <xsl:with-param name="string" select="ea:RESPONSE-REF" />
                      <xsl:with-param name="delimiter" select="'/'" />
                    </xsl:call-template>
                  </xsl:variable>
                  <xsl:variable name="concat_response1">
                    <xsl:value-of select="concat($response1, '?type=ProcessEvent')" />
                  </xsl:variable>
                  <segments xsi:type="am:EventChainContainer">
                    <eventChain name="{$segments_name}" stimulus="{$concat_stimulus1}" response="{$concat_response1}" />
                  </segments>
                </xsl:if>
              </xsl:for-each>
            </xsl:for-each>
          </eventChains>
        </xsl:if>
      </xsl:for-each>
      <xsl:for-each select="//ea:AGE-CONSTRAINT">
        <xsl:variable name="ageConstraint_name" select="ea:SHORT-NAME" />
        <xsl:variable name="ageValue" select="ea:NAME" />
        <xsl:variable name="scope">
          <xsl:call-template name="substring-after-last">
            <xsl:with-param name="string" select="ea:SCOPE-REF" />
            <xsl:with-param name="delimiter" select="'/'" />
          </xsl:call-template>
        </xsl:variable>
        <xsl:variable name="concat_scope">
          <xsl:value-of select="concat($scope, '?type=EventChain')" />
        </xsl:variable>
        <timingConstraints xsi:type="am:EventChainLatencyConstraint" name="{$ageConstraint_name}" scope="{$concat_scope}" type="Reaction">
          <minimum value="{$ageValue}" unit="ms" />
          <maximum value="{$ageValue}" unit="ms" />
        </timingConstraints>
      </xsl:for-each>
      <xsl:for-each select="//ea:PERIODIC-CONSTRAINT">
        <xsl:variable name="period" select="ea:NAME" />
        <xsl:variable name="event_name">
          <xsl:call-template name="substring-after-last">
            <xsl:with-param name="string" select="ea:EVENT-REF" />
            <xsl:with-param name="delimiter" select="'/'" />
          </xsl:call-template>
        </xsl:variable>
        <xsl:for-each select="//ea:EVENT-FUNCTION">
          <xsl:if test="$event_name = ea:SHORT-NAME">
            <xsl:variable name="task_name">
              <xsl:call-template name="substring-after-last">
                <xsl:with-param name="string" select="ea:FUNCTION-IREF/ea:FUNCTION-PROTOTYPE-TARGET-REF" />
                <xsl:with-param name="delimiter" select="'/'" />
              </xsl:call-template>
            </xsl:variable>
            <xsl:variable name="process_concat">
              <xsl:value-of select="concat($task_name, '?type=Task')" />
            </xsl:variable>
            <requirements xsi:type="am:ProcessRequirement" name="{$task_name}" severity="Critical" process="{$process_concat}">
              <limit xsi:type="am:TimeRequirementLimit" limitType="UpperLimit" metric="ResponseTime">
                <limitValue value="{$period}" unit="ms" />
              </limit>
            </requirements>
          </xsl:if>
        </xsl:for-each>
      </xsl:for-each>
    </constraintsModel>
    <mappingModel>
      <schedulerAllocation scheduler="MICROSAR_task_scheduler_CPU0?type=TaskScheduler" responsibility="CPU0?type=ProcessingUnit" executingPU="CPU0?type=ProcessingUnit" />
      <schedulerAllocation scheduler="MICROSAR_task_scheduler_CPU1?type=TaskScheduler" responsibility="CPU1?type=ProcessingUnit" executingPU="CPU1?type=ProcessingUnit" />
      <schedulerAllocation scheduler="MICROSAR_task_scheduler_CPU2?type=TaskScheduler" responsibility="CPU2?type=ProcessingUnit" executingPU="CPU2?type=ProcessingUnit" />
      <runnableAllocation scheduler="MICROSAR_task_scheduler_CPU0?type=TaskScheduler" entity="ABS_T?type=Runnable" />
      <runnableAllocation scheduler="MICROSAR_task_scheduler_CPU1?type=TaskScheduler" entity="GlobalBrakeController?type=Runnable" />
      <runnableAllocation scheduler="MICROSAR_task_scheduler_CPU1?type=TaskScheduler" entity="BrakePedalLDM_T?type=Runnable" />
      <runnableAllocation scheduler="MICROSAR_task_scheduler_CPU2?type=
TaskScheduler" entity="BrakeTorqMap?type=Runnable" />
        <runnableAllocation scheduler="MICROSAR_task_scheduler_CPU0?type=
    TaskScheduler" entity="BrakeActuatorLDM?type=Runnable" />
        <taskAllocation task="ABS_FL_Pt?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="ABS_FR_Pt?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="ABS_RL_Pt?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="ABS_RR_Pt?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pGlobalBrakeController?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU1?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pLDM_Brake_FL?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pLDM_Brake_FR?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pLDM_Brake_RL?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pLDM_Brake_RR?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pBrakeTorqueMap?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU2?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pBrakePedalLDM?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU2?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
      </mappingModel>
    </am:Amalthea>
  </xsl:template>
</xsl:stylesheet>
```

Listing A.2: XSLT code for transforming AUTOSAR to AMALTHEA model

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- xsl stylesheet declaration -->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:ar="http://
    autosar.org/schema/r4.0" xmlns:exslt="http://exslt.org/common" version="1.0"
    extension-element-prefixes="exslt">
  <xsl:output method="xml" indent="yes" />
  <!-- This template is used to extract a sub-string within a string after the last
      occurence of the delimiter '/' -->
  <xsl:template name="substring-after-last">
    <xsl:param name="string" />
    <xsl:param name="delimiter" />
    <xsl:choose>
      <xsl:when test="contains($string, $delimiter)">
        <xsl:call-template name="substring-after-last">
          <xsl:with-param name="string" select="substring-after($string, $delimiter
    )" />
          <xsl:with-param name="delimiter" select="$delimiter" />
        </xsl:call-template>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="$string" />
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>
  <!-- This template is used to get the stimuli name -->
  <xsl:template name="get-stimuli-name">
```

```
    <xsl:param name="taskName" />
    <xsl:for-each select="//ar:TD-EVENT-SWC-INTERNAL-BEHAVIOR/ar:COMPONENT-IREF/
    ar:TARGET-COMPONENT-REF">
      <xsl:variable name="EventFunctionName" select="../../ar:SHORT-NAME" />
      <xsl:variable name="Var2">
        <xsl:call-template name="substring-after-last">
          <xsl:with-param name="string" select="current()" />
          <xsl:with-param name="delimiter" select="'/'" />
        </xsl:call-template>
      </xsl:variable>
      <!-- <varoutput>
        <xsl:value-of select="$Var2"/>
      </varoutput> -->
      <xsl:if test="$taskName = $Var2">
        <xsl:for-each select="//ar:PERIODIC-EVENT-TRIGGERING/ar:EVENT-REF">
          <xsl:variable name="Var3">
            <xsl:call-template name="substring-after-last">
              <xsl:with-param name="string" select="current()" />
              <xsl:with-param name="delimiter" select="'/'" />
            </xsl:call-template>
          </xsl:variable>
          <xsl:if test="$Var3 = $EventFunctionName">
            <xsl:value-of select="concat(../ar:SHORT-NAME, '?type=PeriodicStimulus
    ')" />
          </xsl:if>
        </xsl:for-each>
      </xsl:if>
    </xsl:for-each>
  </xsl:template>
  <!-- This template is used to get the instruction count for each runnable -->
  <xsl:template name="get-instruction-count">
    <xsl:param name="runnableName" />
    <xsl:for-each select="//ar:EXECUTION-TIME-CONSTRAINT">
      <xsl:variable name="runnable_entity">
        <xsl:call-template name="substring-after-last">
          <xsl:with-param name="string" select="ar:EXECUTABLE-REF" />
          <xsl:with-param name="delimiter" select="'/'" />
        </xsl:call-template>
      </xsl:variable>
      <xsl:if test="$runnableName = $runnable_entity">
        <xsl:variable name="instruction_count" select="ar:MAXIMUM/ar:CSE-CODE" />
        <xsl:value-of select="$instruction_count" />
      </xsl:if>
    </xsl:for-each>
  </xsl:template>
  <!-- This template is used to get the processPrototypes -->
  <xsl:template name="get_processPrototypes">
    <processPrototypes name="">
      <xsl:for-each select="//ar:EXECUTION-ORDER-CONSTRAINT/ar:ORDERED-ELEMENTS/
    ar:EOC-EXECUTABLE-ENTITY-REF">
        <xsl:variable name="direct_successor" select="ar:DIRECT-SUCCESSOR-REFS" />
        <xsl:if test="$direct_successor != ''">
          <xsl:variable name="origin">
            <xsl:call-template name="substring-after-last">
              <xsl:with-param name="string" select="ar:EXECUTABLE-REF" />
              <xsl:with-param name="delimiter" select="'/'" />
            </xsl:call-template>
          </xsl:variable>
          <xsl:variable name="concat_origin">
            <xsl:value-of select="concat($origin, '?type=Runnable')" />
          </xsl:variable>
          <xsl:variable name="eo_successor_name">
            <xsl:call-template name="substring-after-last">
              <xsl:with-param name="string" select="ar:DIRECT-SUCCESSOR-REFS/
    ar:DIRECT-SUCCESSOR-REF" />
              <xsl:with-param name="delimiter" select="'/'" />
            </xsl:call-template>
          </xsl:variable>
          <xsl:for-each select="//ar:EOC-EXECUTABLE-ENTITY-REF">
            <xsl:if test="$eo_successor_name = ar:SHORT-NAME">
```

```
              <xsl:variable name="target">
                <xsl:call-template name="substring-after-last">
                  <xsl:with-param name="string" select="ar:EXECUTABLE-REF" />
                  <xsl:with-param name="delimiter" select="'/'" />
                </xsl:call-template>
              </xsl:variable>
              <xsl:variable name="concat_target">
                <xsl:value-of select="concat($target, '?type=Runnable')" />
              </xsl:variable>
              <orderPrecedenceSpec origin="{$concat_origin}" target="{$
    concat_target}" orderType="directOrder" />
            </xsl:if>
          </xsl:for-each>
        </xsl:if>
      </xsl:for-each>
    </processPrototypes>
  </xsl:template>
  <!-- xsl template declaration:
template tells the xlst processor about which section of xml
document to be formatted. It takes an XPath expression.
In our case, it is matching  root element -->
  <xsl:template match="/">
    <!-- New Line -->
    <xsl:text />
    <am:Amalthea xmlns:am="http://app4mc.eclipse.org/amalthea/0.9.2" xmlns:xmi="
    http://www.omg.org/XMI" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmi:version="2.0">
      <swModel>
        <xsl:for-each select="//ar:SW-COMPONENT-PROTOTYPE">
          <xsl:variable name="taskName" select="ar:SHORT-NAME" />
          <xsl:variable name="Var1">
            <xsl:call-template name="substring-after-last">
              <xsl:with-param name="string" select="ar:TYPE-TREF" />
              <xsl:with-param name="delimiter" select="'/'" />
            </xsl:call-template>
          </xsl:variable>
          <xsl:variable name="ref1">
            <xsl:value-of select="concat($Var1, '?type=Runnable')" />
          </xsl:variable>
          <xsl:variable name="call_seq_name">
            <xsl:value-of select="concat('CS_',$taskName)" />
          </xsl:variable>
          <xsl:variable name="stimuli_name">
            <xsl:call-template name="get-stimuli-name">
              <xsl:with-param name="taskName" select="$taskName" />
            </xsl:call-template>
          </xsl:variable>
          <tasks name="{$taskName}" stimuli="{$stimuli_name}" preemption="
    preemptive" multipleTaskActivationLimit="1">
            <callGraph>
              <graphEntries xsi:type="am:CallSequence" name="{$call_seq_name}">
                <calls xsi:type="am:TaskRunnableCall" runnable="{$ref1}" />
              </graphEntries>
            </callGraph>
          </tasks>
        </xsl:for-each>
        <xsl:for-each select="//ar:RUNNABLE-ENTITY">
          <xsl:variable name="runnableName" select="ar:SHORT-NAME" />
          <xsl:variable name="ref3">
            <xsl:call-template name="get-instruction-count">
              <xsl:with-param name="runnableName" select="$runnableName" />
            </xsl:call-template>
          </xsl:variable>
          <runnables name="{$runnableName}" callback="false" service="false">
            <runnableItems xsi:type="am:ExecutionNeed">
              <default key="Instructions">
                <value xsi:type="am:NeedConstant" value="{$ref3}" />
              </default>
            </runnableItems>
          </runnables>
        </xsl:for-each>
        <xsl:call-template name="get_processPrototypes" />
      </swModel>
      <hwModel>
        <definitions xsi:type="am:ProcessingUnitDefinition" name="CPU0_type" puType
    ="CPU" features="Instructions/IPC_1.0?type=HwFeature" />
        <definitions xsi:type="am:ProcessingUnitDefinition" name="CPU1_type" puType
    ="CPU" features="Instructions/IPC_1.0?type=HwFeature" />
        <definitions xsi:type="am:ProcessingUnitDefinition" name="CPU2_type" puType
    ="CPU" features="Instructions/IPC_1.0?type=HwFeature" />
        <featureCategories name="Instructions" featureType="performance">
          <features name="IPC_1.0" value="1.0" />
        </featureCategories>
        <structures name="infineon_aurix_tc297t_model" structureType="System">
          <structures name="ECU_Main" structureType="ECU">
            <structures name="infineon_aurix_tc297t" structureType="Microcontroller
    ">
              <modules xsi:type="am:ProcessingUnit" name="CPU0" frequencyDomain="
    QuartzCPU0?type=FrequencyDomain" definition="CPU0_type?type=
    ProcessingUnitDefinition" />
              <modules xsi:type="am:ProcessingUnit" name="CPU1" frequencyDomain="
    QuartzCPU1?type=FrequencyDomain" definition="CPU1_type?type=
    ProcessingUnitDefinition" />
              <modules xsi:type="am:ProcessingUnit" name="CPU2" frequencyDomain="
    QuartzCPU2?type=FrequencyDomain" definition="CPU2_type?type=
    ProcessingUnitDefinition" />
            </structures>
          </structures>
        </structures>
        <xsl:for-each select="//ar:HW-ELEMENT">
          <xsl:variable name="name" select="ar:SHORT-NAME" />
          <xsl:if test="ar:CATEGORY = 'CPU'">
            <xsl:variable name="HW_Type">
              <xsl:call-template name="substring-after-last">
                <xsl:with-param name="string" select="ar:HW-TYPE-REF" />
                <xsl:with-param name="delimiter" select="'/'" />
              </xsl:call-template>
            </xsl:variable>
            <xsl:for-each select="//ar:HW-TYPE">
              <xsl:if test="ar:SHORT-NAME = $HW_Type">
                <xsl:variable name="clock_freq" select="ar:HW-ATTRIBUTE-VALUES/
    ar:HW-ATTRIBUTE-VALUE/ar:V" />
                <domains xsi:type="am:FrequencyDomain" name="{$name}" clockGating="
    false">
                  <defaultValue value="{$clock_freq}" unit="Hz" />
                </domains>
              </xsl:if>
            </xsl:for-each>
          </xsl:if>
        </xsl:for-each>
      </hwModel>
      <osModel>
        <xsl:for-each select="//ar:SYSTEM">
          <xsl:variable name="oSName" select="ar:SHORT-NAME" />
          <xsl:variable name="categoryName" select="ar:CATEGORY" />
          <xsl:if test="$categoryName = 'Operating System'">
            <operatingSystems name="{$oSName}">
              <taskSchedulers name="MICROSAR_task_scheduler_CPU0">
                <schedulingAlgorithm xsi:type="am:OSEK" />
              </taskSchedulers>
              <taskSchedulers name="MICROSAR_task_scheduler_CPU1">
                <schedulingAlgorithm xsi:type="am:OSEK" />
              </taskSchedulers>
              <taskSchedulers name="MICROSAR_task_scheduler_CPU2">
                <schedulingAlgorithm xsi:type="am:OSEK" />
              </taskSchedulers>
            </operatingSystems>
          </xsl:if>
        </xsl:for-each>
      </osModel>
```

```
      <stimuliModel>
        <xsl:for-each select="//ar:PERIODIC-EVENT-TRIGGERING">
          <stimuli xsi:type="am:PeriodicStimulus" name="{ar:SHORT-NAME}">
            <offset value="0" unit="ms" />
            <recurrence value="{ar:PERIOD/ar:CSE-CODE}" unit="ms" />
          </stimuli>
        </xsl:for-each>
      </stimuliModel>
      <eventModel>
        <xsl:for-each select="//ar:TD-EVENT-SWC-INTERNAL-BEHAVIOR">
          <xsl:variable name="eventName" select="ar:SHORT-NAME" />
          <xsl:variable name="eventType" select="
    ar:TD-EVENT-SWC-INTERNAL-BEHAVIOR-TYPE" />
          <xsl:variable name="entity_name">
            <xsl:call-template name="substring-after-last">
              <xsl:with-param name="string" select="ar:COMPONENT-IREF/
    ar:TARGET-COMPONENT-REF" />
              <xsl:with-param name="delimiter" select="'/'" />
            </xsl:call-template>
          </xsl:variable>
          <xsl:variable name="entity">
            <xsl:value-of select="concat($entity_name, '?type=Task')" />
          </xsl:variable>
          <xsl:choose>
            <xsl:when test="$eventType = 'RUNNABLE-ENTITY-ACTIVATED'">
              <events xsi:type="am:ProcessEvent" name="{$eventName}" eventType="
    activate" entity="{$entity}" />
            </xsl:when>
            <xsl:otherwise>
              <events xsi:type="am:ProcessEvent" name="{$eventName}" eventType="
    terminate" entity="{$entity}" />
            </xsl:otherwise>
          </xsl:choose>
        </xsl:for-each>
      </eventModel>
      <constraintsModel>
        <xsl:for-each select="//ar:TIMING-DESCRIPTION-EVENT-CHAIN">
          <xsl:variable name="eventChainName" select="ar:SHORT-NAME" />
          <xsl:variable name="segment_refs" select="ar:SEGMENT-REFS" />
          <xsl:if test="$segment_refs != ''">
            <xsl:variable name="stimulus">
              <xsl:call-template name="substring-after-last">
                <xsl:with-param name="string" select="ar:STIMULUS-REF" />
                <xsl:with-param name="delimiter" select="'/'" />
              </xsl:call-template>
            </xsl:variable>
            <xsl:variable name="concat_stimulus">
              <xsl:value-of select="concat($stimulus, '?type=ProcessEvent')" />
            </xsl:variable>
            <xsl:variable name="response">
              <xsl:call-template name="substring-after-last">
                <xsl:with-param name="string" select="ar:RESPONSE-REF" />
                <xsl:with-param name="delimiter" select="'/'" />
              </xsl:call-template>
            </xsl:variable>
            <xsl:variable name="concat_response">
              <xsl:value-of select="concat($response, '?type=ProcessEvent')" />
            </xsl:variable>
            <eventChains name="{$eventChainName}" stimulus="{$concat_stimulus}"
    response="{$concat_response}">
              <xsl:for-each select="ar:SEGMENT-REFS/ar:SEGMENT-REF">
                <xsl:variable name="segments_name">
                  <xsl:call-template name="substring-after-last">
                    <xsl:with-param name="string" select="current()" />
                    <xsl:with-param name="delimiter" select="'/'" />
                  </xsl:call-template>
                </xsl:variable>
                <xsl:for-each select="//ar:TIMING-DESCRIPTION-EVENT-CHAIN">
                  <xsl:if test="$segments_name = ar:SHORT-NAME">
                    <xsl:variable name="stimulus1">
                      <xsl:call-template name="substring-after-last">
                        <xsl:with-param name="string" select="ar:STIMULUS-REF" />
                        <xsl:with-param name="delimiter" select="'/'" />
                      </xsl:call-template>
                    </xsl:variable>
                    <xsl:variable name="concat_stimulus1">
                      <xsl:value-of select="concat($stimulus1, '?type=ProcessEvent')" />
                    </xsl:variable>
                    <xsl:variable name="response1">
                      <xsl:call-template name="substring-after-last">
                        <xsl:with-param name="string" select="ar:RESPONSE-REF" />
                        <xsl:with-param name="delimiter" select="'/'" />
                      </xsl:call-template>
                    </xsl:variable>
                    <xsl:variable name="concat_response1">
                      <xsl:value-of select="concat($response1, '?type=ProcessEvent')" />
                    </xsl:variable>
                    <segments xsi:type="am:EventChainContainer">
                      <eventChain name="{$segments_name}" stimulus="{$concat_stimulus1}" response="{$concat_response1}" />
                    </segments>
                  </xsl:if>
                </xsl:for-each>
              </xsl:for-each>
            </eventChains>
          </xsl:if>
        </xsl:for-each>
        <xsl:for-each select="//ar:LATENCY-TIMING-CONSTRAINT">
          <xsl:variable name="latencyConstraint_name" select="ar:SHORT-NAME" />
          <xsl:variable name="latencyValue" select="ar:MAXIMUM/ar:CSE-CODE" />
          <xsl:variable name="scope">
            <xsl:call-template name="substring-after-last">
              <xsl:with-param name="string" select="ar:SCOPE-REF" />
              <xsl:with-param name="delimiter" select="'/'" />
            </xsl:call-template>
          </xsl:variable>
          <xsl:variable name="concat_scope">
            <xsl:value-of select="concat($scope, '?type=EventChain')" />
          </xsl:variable>
          <timingConstraints xsi:type="am:EventChainLatencyConstraint" name="{$latencyConstraint_name}" scope="{$concat_scope}" type="Reaction">
            <minimum value="{$latencyValue}" unit="ms" />
            <maximum value="{$latencyValue}" unit="ms" />
          </timingConstraints>
        </xsl:for-each>
        <xsl:for-each select="//ar:PERIODIC-EVENT-TRIGGERING">
          <xsl:variable name="period" select="ar:PERIOD/ar:CSE-CODE" />
          <xsl:variable name="event_name">
            <xsl:call-template name="substring-after-last">
              <xsl:with-param name="string" select="ar:EVENT-REF" />
              <xsl:with-param name="delimiter" select="'/'" />
            </xsl:call-template>
          </xsl:variable>
          <xsl:for-each select="//ar:TD-EVENT-SWC-INTERNAL-BEHAVIOR">
            <xsl:if test="$event_name = ar:SHORT-NAME">
              <xsl:variable name="task_name">
                <xsl:call-template name="substring-after-last">
                  <xsl:with-param name="string" select="ar:COMPONENT-IREF/ar:TARGET-COMPONENT-REF" />
                  <xsl:with-param name="delimiter" select="'/'" />
                </xsl:call-template>
              </xsl:variable>
              <xsl:variable name="process_concat">
                <xsl:value-of select="concat($task_name, '?type=Task')" />
              </xsl:variable>
              <requirements xsi:type="am:ProcessRequirement" name="{$task_name}" severity="Critical" process="{$process_concat}">
                <limit xsi:type="am:TimeRequirementLimit" limitType="UpperLimit"
    metric="ResponseTime">
                  <limitValue value="{$period}" unit="ms" />
                </limit>
              </requirements>
            </xsl:if>
          </xsl:for-each>
        </xsl:for-each>
      </constraintsModel>
      <mappingModel>
        <schedulerAllocation scheduler="MICROSAR_task_scheduler_CPU0?type=
    TaskScheduler" responsibility="CPU0?type=ProcessingUnit" executingPU="CPU0?type
    =ProcessingUnit" />
        <schedulerAllocation scheduler="MICROSAR_task_scheduler_CPU1?type=
    TaskScheduler" responsibility="CPU1?type=ProcessingUnit" executingPU="CPU1?type
    =ProcessingUnit" />
        <schedulerAllocation scheduler="MICROSAR_task_scheduler_CPU2?type=
    TaskScheduler" responsibility="CPU2?type=ProcessingUnit" executingPU="CPU2?type
    =ProcessingUnit" />
        <runnableAllocation scheduler="MICROSAR_task_scheduler_CPU0?type=
    TaskScheduler" entity="ABS_T?type=Runnable" />
        <runnableAllocation scheduler="MICROSAR_task_scheduler_CPU1?type=
    TaskScheduler" entity="GlobalBrakeController?type=Runnable" />
        <runnableAllocation scheduler="MICROSAR_task_scheduler_CPU1?type=
    TaskScheduler" entity="BrakePedalLDM_T?type=Runnable" />
        <runnableAllocation scheduler="MICROSAR_task_scheduler_CPU2?type=
    TaskScheduler" entity="BrakeTorqMap?type=Runnable" />
        <runnableAllocation scheduler="MICROSAR_task_scheduler_CPU0?type=
    TaskScheduler" entity="BrakeActuatorLDM?type=Runnable" />
        <taskAllocation task="ABS_FL_Pt?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="ABS_FR_Pt?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="ABS_RL_Pt?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="ABS_RR_Pt?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pGlobalBrakeController?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU1?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pLDM_Brake_FL?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pLDM_Brake_FR?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pLDM_Brake_RL?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pLDM_Brake_RR?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU0?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pBrakeTorqueMap?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU2?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
        <taskAllocation task="pBrakePedalLDM?type=Task" scheduler="
    MICROSAR_task_scheduler_CPU2?type=TaskScheduler" affinity="CPU2?type=
    ProcessingUnit CPU1?type=ProcessingUnit CPU0?type=ProcessingUnit" />
      </mappingModel>
    </am:Amalthea>
  </xsl:template>
</xsl:stylesheet>
```

Listing A.3: AUTOSAR model for BBW component

```

<?xml version="1.0" encoding="UTF-8"?>
```

```xml
<AUTOSAR xmlns="http://autosar.org/schema/r4.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_4-2-2.xsd">
  <AR-PACKAGES>
    <AR-PACKAGE>
      <SHORT-NAME>BBW_SWC_Description</SHORT-NAME>
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>Constraints</SHORT-NAME>
          <ELEMENTS>
            <SWC-TIMING>
              <SHORT-NAME>TimingConstraints</SHORT-NAME>
              <TIMING-GUARANTEES>
                <EXECUTION-TIME-CONSTRAINT UUID="">
                  <SHORT-NAME>EX_ABS_T</SHORT-NAME>
                  <EXECUTABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/ABS_T_Information/ABS_T//ABS_T</EXECUTABLE-REF>
                  <EXECUTION-TIME-TYPE>NET</EXECUTION-TIME-TYPE>
                  <MAXIMUM>
                    <CSE-CODE>150000</CSE-CODE>
                  </MAXIMUM>
                </EXECUTION-TIME-CONSTRAINT>
                <EXECUTION-TIME-CONSTRAINT UUID="">
                  <SHORT-NAME>EX_BrakePedalLDM_T</SHORT-NAME>
                  <EXECUTABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/BrakePedalLDM_T_Information/BrakePedalLDM_T//BrakePedalLDM_T</EXECUTABLE-REF>
                  <EXECUTION-TIME-TYPE>NET</EXECUTION-TIME-TYPE>
                  <MAXIMUM>
                    <CSE-CODE>60000</CSE-CODE>
                  </MAXIMUM>
                </EXECUTION-TIME-CONSTRAINT>
                <EXECUTION-TIME-CONSTRAINT UUID="">
                  <SHORT-NAME>EX_BrakeTorqMap</SHORT-NAME>
                  <EXECUTABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/BrakeTorqMap_Information/BrakeTorqMap//BrakeTorqMap</EXECUTABLE-REF>
                  <EXECUTION-TIME-TYPE>NET</EXECUTION-TIME-TYPE>
                  <MAXIMUM>
                    <CSE-CODE>90000</CSE-CODE>
                  </MAXIMUM>
                </EXECUTION-TIME-CONSTRAINT>
                <EXECUTION-TIME-CONSTRAINT UUID="">
                  <SHORT-NAME>EX_GlobalBrakeController</SHORT-NAME>
                  <EXECUTABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/GlobalBrakeController_Information/GlobalBrakeController//GlobalBrakeController</EXECUTABLE-REF>
                  <EXECUTION-TIME-TYPE>NET</EXECUTION-TIME-TYPE>
                  <MAXIMUM>
                    <CSE-CODE>120000</CSE-CODE>
                  </MAXIMUM>
                </EXECUTION-TIME-CONSTRAINT>
                <LATENCY-TIMING-CONSTRAINT UUID="">
                  <SHORT-NAME>LatencyConstraint_FL</SHORT-NAME>
                  <CATEGORY />
                  <LATENCY-CONSTRAINT-TYPE>REACTION</LATENCY-CONSTRAINT-TYPE>
                  <SCOPE-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/BBW_SWC_Description/EventChains//EC_Sequence_FL</SCOPE-REF>
                  <MAXIMUM>
                    <CSE-CODE>20</CSE-CODE>
                  </MAXIMUM>
                </LATENCY-TIMING-CONSTRAINT>
                <LATENCY-TIMING-CONSTRAINT UUID="">
                  <SHORT-NAME>LatencyConstraint_FR</SHORT-NAME>
                  <LATENCY-CONSTRAINT-TYPE>REACTION</LATENCY-CONSTRAINT-TYPE>
                  <SCOPE-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/BBW_SWC_Description/EventChains//EC_Sequence_FR</SCOPE-REF>
                  <MAXIMUM>
                    <CSE-CODE>20</CSE-CODE>
                  </MAXIMUM>
                </LATENCY-TIMING-CONSTRAINT>
                <LATENCY-TIMING-CONSTRAINT UUID="">
                  <SHORT-NAME>LatencyConstraint_RL</SHORT-NAME>
                  <LATENCY-CONSTRAINT-TYPE>REACTION</LATENCY-CONSTRAINT-TYPE>
                  <SCOPE-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
    BBW_SWC_Description/EventChains//EC_Sequence_RL</SCOPE-REF>
                  <MAXIMUM>
                    <CSE-CODE>20</CSE-CODE>
                  </MAXIMUM>
                </LATENCY-TIMING-CONSTRAINT>
                <PERIODIC-EVENT-TRIGGERING UUID="">
                  <SHORT-NAME>Periodic_Constraint_1</SHORT-NAME>
                  <EVENT-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
    BBW_SWC_Description/EventChains/Events/Event_BrakePedalLDM</EVENT-REF>
                  <PERIOD>
                    <CSE-CODE>2</CSE-CODE>
                  </PERIOD>
                </PERIODIC-EVENT-TRIGGERING>
                <PERIODIC-EVENT-TRIGGERING S="" UUID="">
                  <SHORT-NAME>Periodic_Constraint_10</SHORT-NAME>
                  <EVENT-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
    BBW_SWC_Description/EventChains/Events/Event_ABS_RL_Pt</EVENT-REF>
                  <PERIOD>
                    <CSE-CODE>5</CSE-CODE>
                  </PERIOD>
                </PERIODIC-EVENT-TRIGGERING>
                <PERIODIC-EVENT-TRIGGERING UUID="">
                  <SHORT-NAME>Periodic_Constraint_2</SHORT-NAME>
                  <EVENT-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
    BBW_SWC_Description/EventChains/Events/EventBrakeTorqueMap</EVENT-REF>
                  <PERIOD>
                    <CSE-CODE>3</CSE-CODE>
                  </PERIOD>
                </PERIODIC-EVENT-TRIGGERING>
                <PERIODIC-EVENT-TRIGGERING UUID="">
                  <SHORT-NAME>Periodic_Constraint_3</SHORT-NAME>
                  <EVENT-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
    BBW_SWC_Description/EventChains/Events/Event_GlobalBrakeController</EVENT-REF>
                  <PERIOD>
                    <CSE-CODE>4</CSE-CODE>
                  </PERIOD>
                </PERIODIC-EVENT-TRIGGERING>
                <PERIODIC-EVENT-TRIGGERING UUID="">
                  <SHORT-NAME>Periodic_Constraint_4</SHORT-NAME>
                  <EVENT-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
    BBW_SWC_Description/EventChains/Events/Event_pLDM_Brake_FL</EVENT-REF>
                  <PERIOD S="">
                    <CSE-CODE>6</CSE-CODE>
                  </PERIOD>
                </PERIODIC-EVENT-TRIGGERING>
                <PERIODIC-EVENT-TRIGGERING UUID="">
                  <SHORT-NAME>Periodic_Constraint_5</SHORT-NAME>
                  <EVENT-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
    BBW_SWC_Description/EventChains/Events/Event_pLDM_Brake_FR</EVENT-REF>
                  <PERIOD>
                    <CSE-CODE>6</CSE-CODE>
                  </PERIOD>
                </PERIODIC-EVENT-TRIGGERING>
                <PERIODIC-EVENT-TRIGGERING UUID="">
                  <SHORT-NAME>Periodic_Constraint_6</SHORT-NAME>
                  <EVENT-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
    BBW_SWC_Description/EventChains/Events/Event_pLDM_Brake_RL</EVENT-REF>
                  <PERIOD>
                    <CSE-CODE>6</CSE-CODE>
                  </PERIOD>
                </PERIODIC-EVENT-TRIGGERING>
                <PERIODIC-EVENT-TRIGGERING UUID="">
                  <SHORT-NAME>Periodic_Constraint_7</SHORT-NAME>
                  <EVENT-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
    BBW_SWC_Description/EventChains/Events/Event_pLDM_Brake_RR</EVENT-REF>
```

```
                <PERIOD>
                  <CSE-CODE>6</CSE-CODE>
                </PERIOD>
              </PERIODIC-EVENT-TRIGGERING>
              <PERIODIC-EVENT-TRIGGERING UUID="">
                <SHORT-NAME>Periodic_Constraint_8</SHORT-NAME>
                <EVENT-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_ABS_FL_Pt</EVENT-REF>
                <PERIOD>
                  <CSE-CODE>5</CSE-CODE>
                </PERIOD>
              </PERIODIC-EVENT-TRIGGERING>
              <PERIODIC-EVENT-TRIGGERING UUID="">
                <SHORT-NAME>Periodic_Constraint_9</SHORT-NAME>
                <EVENT-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_ABS_FR_Pt</EVENT-REF>
                <PERIOD>
                  <CSE-CODE>5</CSE-CODE>
                </PERIOD>
              </PERIODIC-EVENT-TRIGGERING>
            </TIMING-GUARANTEES>
            <TIMING-REQUIREMENTS>
              <EXECUTION-TIME-CONSTRAINT UUID="">
                <SHORT-NAME>EX_BrakeActuatorLDM</SHORT-NAME>
                <EXECUTABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/BrakeActuatorLDM_Information/BrakeActuatorLDM//BrakeActuatorLDM</EXECUTABLE-REF>
                <EXECUTION-TIME-TYPE>NET</EXECUTION-TIME-TYPE>
                <MAXIMUM>
                  <CSE-CODE>180000</CSE-CODE>
                </MAXIMUM>
              </EXECUTION-TIME-CONSTRAINT>
              <EXECUTION-ORDER-CONSTRAINT>
                <SHORT-NAME>Execution Order</SHORT-NAME>
                <EXECUTION-ORDER-CONSTRAINT-TYPE>ORDINARY-EOC</EXECUTION-ORDER-CONSTRAINT-TYPE>
                <ORDERED-ELEMENTS>
                  <EOC-EXECUTABLE-ENTITY-REF>
                    <SHORT-NAME>EO_ABS_T</SHORT-NAME>
                    <DIRECT-SUCCESSOR-REFS>
                      <DIRECT-SUCCESSOR-REF DEST="EOC-EXECUTABLE-ENTITY-REF">/BBW_SWC_Description/Constraints/TimingConstraints/Execution Order/EO_BrakeActuatorLDM</DIRECT-SUCCESSOR-REF>
                    </DIRECT-SUCCESSOR-REFS>
                    <EXECUTABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/ABS_T_Information/ABS_T//ABS_T</EXECUTABLE-REF>
                  </EOC-EXECUTABLE-ENTITY-REF>
                  <EOC-EXECUTABLE-ENTITY-REF UUID="">
                    <SHORT-NAME>EO_BrakeActuatorLDM</SHORT-NAME>
                    <EXECUTABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/BrakeActuatorLDM_Information/BrakeActuatorLDM//BrakeActuatorLDM</EXECUTABLE-REF>
                  </EOC-EXECUTABLE-ENTITY-REF>
                  <EOC-EXECUTABLE-ENTITY-REF S="">
                    <SHORT-NAME>EO_BrakePedalLDM_T</SHORT-NAME>
                    <DIRECT-SUCCESSOR-REFS>
                      <DIRECT-SUCCESSOR-REF DEST="EOC-EXECUTABLE-ENTITY-REF">/BBW_SWC_Description/Constraints/TimingConstraints/Execution Order/EO_BrakeTorqMap</DIRECT-SUCCESSOR-REF>
                    </DIRECT-SUCCESSOR-REFS>
                    <EXECUTABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/BrakePedalLDM_T_Information/BrakePedalLDM_T//BrakePedalLDM_T</EXECUTABLE-REF>
                  </EOC-EXECUTABLE-ENTITY-REF>
                  <EOC-EXECUTABLE-ENTITY-REF>
                    <SHORT-NAME>EO_BrakeTorqMap</SHORT-NAME>
                    <DIRECT-SUCCESSOR-REFS>
                      <DIRECT-SUCCESSOR-REF DEST="EOC-EXECUTABLE-ENTITY-REF">/BBW_SWC_Description/Constraints/TimingConstraints/Execution Order/EO_GlobalBrakeController</DIRECT-SUCCESSOR-REF>
```

```xml
                  </DIRECT-SUCCESSOR-REFS>
                  <EXECUTABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/BrakeTorqMap_Information/BrakeTorqMap//BrakeTorqMap</EXECUTABLE-REF>
                </EOC-EXECUTABLE-ENTITY-REF>
                <EOC-EXECUTABLE-ENTITY-REF>
                  <SHORT-NAME>EO_GlobalBrakeController</SHORT-NAME>
                  <DIRECT-SUCCESSOR-REFS>
                    <DIRECT-SUCCESSOR-REF DEST="EOC-EXECUTABLE-ENTITY-REF">/BBW_SWC_Description/Constraints/TimingConstraints/Execution Order/EO_ABS_T</DIRECT-SUCCESSOR-REF>
                  </DIRECT-SUCCESSOR-REFS>
                  <EXECUTABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/GlobalBrakeController_Information/GlobalBrakeController//GlobalBrakeController</EXECUTABLE-REF>
                </EOC-EXECUTABLE-ENTITY-REF>
              </ORDERED-ELEMENTS>
            </EXECUTION-ORDER-CONSTRAINT>
            <LATENCY-TIMING-CONSTRAINT UUID="">
              <SHORT-NAME>LatencyConstraint_RR</SHORT-NAME>
              <LATENCY-CONSTRAINT-TYPE>REACTION</LATENCY-CONSTRAINT-TYPE>
              <SCOPE-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/BBW_SWC_Description/EventChains//EC_Sequence_RR</SCOPE-REF>
              <MAXIMUM>
                <CSE-CODE>20</CSE-CODE>
              </MAXIMUM>
            </LATENCY-TIMING-CONSTRAINT>
            <PERIODIC-EVENT-TRIGGERING UUID="">
              <SHORT-NAME>Periodic_Constraint_11</SHORT-NAME>
              <EVENT-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_ABS_RR_Pt</EVENT-REF>
              <PERIOD>
                <CSE-CODE>5</CSE-CODE>
              </PERIOD>
            </PERIODIC-EVENT-TRIGGERING>
          </TIMING-REQUIREMENTS>
        </SWC-TIMING>
      </ELEMENTS>
    </AR-PACKAGE>
    <AR-PACKAGE UUID="">
      <SHORT-NAME>EcuResource</SHORT-NAME>
      <AR-PACKAGES>
        <AR-PACKAGE UUID="">
          <SHORT-NAME>HwCategory</SHORT-NAME>
          <ELEMENTS>
            <HW-CATEGORY S="" UUID="">
              <SHORT-NAME>ProcessingUnit</SHORT-NAME>
              <CATEGORY />
              <HW-ATTRIBUTE-DEFS>
                <HW-ATTRIBUTE-DEF>
                  <SHORT-NAME>FrequencyHz</SHORT-NAME>
                  <CATEGORY />
                  <IS-REQUIRED>true</IS-REQUIRED>
                </HW-ATTRIBUTE-DEF>
              </HW-ATTRIBUTE-DEFS>
            </HW-CATEGORY>
          </ELEMENTS>
        </AR-PACKAGE>
        <AR-PACKAGE UUID="">
          <SHORT-NAME>HwElement</SHORT-NAME>
          <ELEMENTS>
            <HW-ELEMENT>
              <SHORT-NAME>CPU0</SHORT-NAME>
              <CATEGORY>CPU</CATEGORY>
              <HW-TYPE-REF DEST="HW-TYPE">/BBW_SWC_Description/EcuResource/HwType/QuartzCPU0</HW-TYPE-REF>
              <HW-CATEGORY-REFS>
                <HW-CATEGORY-REF DEST="HW-CATEGORY">/BBW_SWC_Description/EcuResource/HwCategory/ProcessingUnit</HW-CATEGORY-REF>
              </HW-CATEGORY-REFS>
```

```
                  </HW-ELEMENT>
                  <HW-ELEMENT>
                     <SHORT-NAME>CPU1</SHORT-NAME>
                     <CATEGORY>CPU</CATEGORY>
                     <HW-TYPE-REF DEST="HW-TYPE">/BBW_SWC_Description/EcuResource/HwType/QuartzCPU1</HW-TYPE-REF>
                     <HW-CATEGORY-REFS>
                        <HW-CATEGORY-REF DEST="HW-CATEGORY">/BBW_SWC_Description/EcuResource/HwCategory/ProcessingUnit</HW-CATEGORY-REF>
                     </HW-CATEGORY-REFS>
                  </HW-ELEMENT>
                  <HW-ELEMENT UUID="">
                     <SHORT-NAME>CPU2</SHORT-NAME>
                     <CATEGORY>CPU</CATEGORY>
                     <HW-TYPE-REF DEST="HW-TYPE">/BBW_SWC_Description/EcuResource/HwType/QuartzCPU2</HW-TYPE-REF>
                     <HW-CATEGORY-REFS>
                        <HW-CATEGORY-REF DEST="HW-CATEGORY">/BBW_SWC_Description/EcuResource/HwCategory/ProcessingUnit</HW-CATEGORY-REF>
                     </HW-CATEGORY-REFS>
                  </HW-ELEMENT>
               </ELEMENTS>
            </AR-PACKAGE>
            <AR-PACKAGE>
               <SHORT-NAME>HwType</SHORT-NAME>
               <ELEMENTS>
                  <HW-TYPE>
                     <SHORT-NAME>QuartzCPU0</SHORT-NAME>
                     <HW-ATTRIBUTE-VALUES>
                        <HW-ATTRIBUTE-VALUE>
                           <V BLUEPRINT-VALUE="">3.0E8</V>
                        </HW-ATTRIBUTE-VALUE>
                     </HW-ATTRIBUTE-VALUES>
                  </HW-TYPE>
                  <HW-TYPE>
                     <SHORT-NAME>QuartzCPU1</SHORT-NAME>
                     <HW-ATTRIBUTE-VALUES>
                        <HW-ATTRIBUTE-VALUE>
                           <V BLUEPRINT-VALUE="">3.0E8</V>
                        </HW-ATTRIBUTE-VALUE>
                     </HW-ATTRIBUTE-VALUES>
                  </HW-TYPE>
                  <HW-TYPE>
                     <SHORT-NAME>QuartzCPU2</SHORT-NAME>
                     <HW-ATTRIBUTE-VALUES>
                        <HW-ATTRIBUTE-VALUE>
                           <V BLUEPRINT-VALUE="">3.0E8</V>
                        </HW-ATTRIBUTE-VALUE>
                     </HW-ATTRIBUTE-VALUES>
                  </HW-TYPE>
               </ELEMENTS>
            </AR-PACKAGE>
         </AR-PACKAGES>
      </AR-PACKAGE>
      <AR-PACKAGE>
         <SHORT-NAME>EventChains</SHORT-NAME>
         <ELEMENTS>
            <SWC-TIMING>
               <TIMING-DESCRIPTIONS>
                  <TIMING-DESCRIPTION-EVENT-CHAIN UUID="">
                     <SHORT-NAME>EC_1_FL</SHORT-NAME>
                     <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_BrakePedalLDM</STIMULUS-REF>
                     <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/EventBrakeTorqueMap</RESPONSE-REF>
                  </TIMING-DESCRIPTION-EVENT-CHAIN>
                  <TIMING-DESCRIPTION-EVENT-CHAIN UUID="">
                     <SHORT-NAME>EC_1_FR</SHORT-NAME>
                     <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_BrakePedalLDM</STIMULUS-REF>
            <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/EventBrakeTorqueMap</RESPONSE-REF>
          </TIMING-DESCRIPTION-EVENT-CHAIN>
          <TIMING-DESCRIPTION-EVENT-CHAIN>
            <SHORT-NAME>EC_1_RL</SHORT-NAME>
            <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_BrakePedalLDM</STIMULUS-REF>
            <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/EventBrakeTorqueMap</RESPONSE-REF>
          </TIMING-DESCRIPTION-EVENT-CHAIN>
          <TIMING-DESCRIPTION-EVENT-CHAIN>
            <SHORT-NAME>EC_1_RR</SHORT-NAME>
            <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_BrakePedalLDM</STIMULUS-REF>
            <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/EventBrakeTorqueMap</RESPONSE-REF>
          </TIMING-DESCRIPTION-EVENT-CHAIN>
          <TIMING-DESCRIPTION-EVENT-CHAIN>
            <SHORT-NAME>EC_2_FL</SHORT-NAME>
            <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/EventBrakeTorqueMap</STIMULUS-REF>
            <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_GlobalBrakeController</RESPONSE-REF>
          </TIMING-DESCRIPTION-EVENT-CHAIN>
          <TIMING-DESCRIPTION-EVENT-CHAIN UUID="">
            <SHORT-NAME>EC_2_FR</SHORT-NAME>
            <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/EventBrakeTorqueMap</STIMULUS-REF>
            <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_GlobalBrakeController</RESPONSE-REF>
          </TIMING-DESCRIPTION-EVENT-CHAIN>
          <TIMING-DESCRIPTION-EVENT-CHAIN S="">
            <SHORT-NAME>EC_2_RL</SHORT-NAME>
            <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/EventBrakeTorqueMap</STIMULUS-REF>
            <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_GlobalBrakeController</RESPONSE-REF>
          </TIMING-DESCRIPTION-EVENT-CHAIN>
          <TIMING-DESCRIPTION-EVENT-CHAIN>
            <SHORT-NAME>EC_2_RR</SHORT-NAME>
            <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/EventBrakeTorqueMap</STIMULUS-REF>
            <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_GlobalBrakeController</RESPONSE-REF>
          </TIMING-DESCRIPTION-EVENT-CHAIN>
          <TIMING-DESCRIPTION-EVENT-CHAIN>
            <SHORT-NAME>EC_3_FL</SHORT-NAME>
            <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_GlobalBrakeController</STIMULUS-REF>
            <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_ABS_FL_Pt</RESPONSE-REF>
          </TIMING-DESCRIPTION-EVENT-CHAIN>
          <TIMING-DESCRIPTION-EVENT-CHAIN S="">
            <SHORT-NAME>EC_3_FR</SHORT-NAME>
            <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_GlobalBrakeController</STIMULUS-REF>
            <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_ABS_FR_Pt</RESPONSE-REF>
          </TIMING-DESCRIPTION-EVENT-CHAIN>
          <TIMING-DESCRIPTION-EVENT-CHAIN>
            <SHORT-NAME>EC_3_RL</SHORT-NAME>
            <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/BBW_SWC_Description/EventChains/Events/Event_GlobalBrakeController</STIMULUS-REF>
```

```
                    <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_ABS_RL_Pt</RESPONSE-REF>
                </TIMING-DESCRIPTION-EVENT-CHAIN>
                <TIMING-DESCRIPTION-EVENT-CHAIN>
                    <SHORT-NAME>EC_3_RR</SHORT-NAME>
                    <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_GlobalBrakeController</
STIMULUS-REF>
                    <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_ABS_RR_Pt</RESPONSE-REF>
                </TIMING-DESCRIPTION-EVENT-CHAIN>
                <TIMING-DESCRIPTION-EVENT-CHAIN>
                    <SHORT-NAME>EC_4_FL</SHORT-NAME>
                    <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_ABS_FL_Pt</STIMULUS-REF>
                    <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_pLDM_Brake_FL</RESPONSE-REF>
                </TIMING-DESCRIPTION-EVENT-CHAIN>
                <TIMING-DESCRIPTION-EVENT-CHAIN UUID="">
                    <SHORT-NAME>EC_4_FR</SHORT-NAME>
                    <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_ABS_FR_Pt</STIMULUS-REF>
                    <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_pLDM_Brake_FR</RESPONSE-REF>
                </TIMING-DESCRIPTION-EVENT-CHAIN>
                <TIMING-DESCRIPTION-EVENT-CHAIN>
                    <SHORT-NAME>EC_4_RL</SHORT-NAME>
                    <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_ABS_RL_Pt</STIMULUS-REF>
                    <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_pLDM_Brake_RL</RESPONSE-REF>
                </TIMING-DESCRIPTION-EVENT-CHAIN>
                <TIMING-DESCRIPTION-EVENT-CHAIN>
                    <SHORT-NAME>EC_4_RR</SHORT-NAME>
                    <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_ABS_RR_Pt</STIMULUS-REF>
                    <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_pLDM_Brake_RR</RESPONSE-REF>
                </TIMING-DESCRIPTION-EVENT-CHAIN>
                <TIMING-DESCRIPTION-EVENT-CHAIN>
                    <SHORT-NAME>EC_Sequence_FL</SHORT-NAME>
                    <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_BrakePedalLDM</STIMULUS-REF>
                    <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_pLDM_Brake_FL</RESPONSE-REF>
                    <SEGMENT-REFS>
                        <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_1_FL</SEGMENT-REF>
                        <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_2_FL</SEGMENT-REF>
                        <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_3_FL</SEGMENT-REF>
                        <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_4_FL</SEGMENT-REF>
                    </SEGMENT-REFS>
                </TIMING-DESCRIPTION-EVENT-CHAIN>
                <TIMING-DESCRIPTION-EVENT-CHAIN>
                    <SHORT-NAME>EC_Sequence_FR</SHORT-NAME>
                    <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_BrakePedalLDM</STIMULUS-REF>
                    <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_pLDM_Brake_FR</RESPONSE-REF>
                    <SEGMENT-REFS>
                        <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_1_FR</SEGMENT-REF>
                        <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_2_FR</SEGMENT-REF>
                        <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_3_FR</SEGMENT-REF>
                        <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
```

```
BBW_SWC_Description/EventChains//EC_4_FR</SEGMENT-REF>
              </SEGMENT-REFS>
            </TIMING-DESCRIPTION-EVENT-CHAIN>
            <TIMING-DESCRIPTION-EVENT-CHAIN>
              <SHORT-NAME>EC_Sequence_RL</SHORT-NAME>
              <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_BrakePedalLDM</STIMULUS-REF>
              <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_ABS_RL_Pt</RESPONSE-REF>
              <SEGMENT-REFS>
                <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_1_RL</SEGMENT-REF>
                <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_2_RL</SEGMENT-REF>
                <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_3_RL</SEGMENT-REF>
                <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_4_RL</SEGMENT-REF>
              </SEGMENT-REFS>
            </TIMING-DESCRIPTION-EVENT-CHAIN>
            <TIMING-DESCRIPTION-EVENT-CHAIN>
              <SHORT-NAME>EC_Sequence_RR</SHORT-NAME>
              <STIMULUS-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_BrakePedalLDM</STIMULUS-REF>
              <RESPONSE-REF DEST="TD-EVENT-SWC-INTERNAL-BEHAVIOR">/
BBW_SWC_Description/EventChains/Events/Event_pLDM_Brake_RR</RESPONSE-REF>
              <SEGMENT-REFS>
                <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_1_RR</SEGMENT-REF>
                <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_2_RR</SEGMENT-REF>
                <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_3_RR</SEGMENT-REF>
                <SEGMENT-REF DEST="TIMING-DESCRIPTION-EVENT-CHAIN">/
BBW_SWC_Description/EventChains//EC_4_RR</SEGMENT-REF>
              </SEGMENT-REFS>
            </TIMING-DESCRIPTION-EVENT-CHAIN>
          </TIMING-DESCRIPTIONS>
        </SWC-TIMING>
        <SWC-TIMING>
          <SHORT-NAME>Events</SHORT-NAME>
          <TIMING-DESCRIPTIONS>
            <TD-EVENT-SWC-INTERNAL-BEHAVIOR UUID="">
              <SHORT-NAME>EventBrakeTorqueMap</SHORT-NAME>
              <COMPONENT-IREF>
                <TARGET-COMPONENT-REF DEST="SW-COMPONENT-PROTOTYPE">/
BBW_SWC_Description/SwCompossitions/RootSwComposition/pBrakeTorqueMap</
TARGET-COMPONENT-REF>
              </COMPONENT-IREF>
              <RUNNABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/
SwComponents/BrakeTorqMap_Information/BrakeTorqMap//BrakeTorqMap</RUNNABLE-REF>
              <TD-EVENT-SWC-INTERNAL-BEHAVIOR-TYPE>RUNNABLE-ENTITY-ACTIVATED</
TD-EVENT-SWC-INTERNAL-BEHAVIOR-TYPE>
            </TD-EVENT-SWC-INTERNAL-BEHAVIOR>
            <TD-EVENT-SWC-INTERNAL-BEHAVIOR UUID="">
              <SHORT-NAME>Event_ABS_FL_Pt</SHORT-NAME>
              <COMPONENT-IREF>
                <TARGET-COMPONENT-REF DEST="SW-COMPONENT-PROTOTYPE">/
BBW_SWC_Description/SwCompossitions/RootSwComposition/ABS_FL_Pt</
TARGET-COMPONENT-REF>
              </COMPONENT-IREF>
              <RUNNABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/
SwComponents/ABS_T_Information/ABS_T//ABS_T</RUNNABLE-REF>
              <TD-EVENT-SWC-INTERNAL-BEHAVIOR-TYPE>RUNNABLE-ENTITY-ACTIVATED</
TD-EVENT-SWC-INTERNAL-BEHAVIOR-TYPE>
            </TD-EVENT-SWC-INTERNAL-BEHAVIOR>
            <TD-EVENT-SWC-INTERNAL-BEHAVIOR UUID="">
              <SHORT-NAME>Event_ABS_FR_Pt</SHORT-NAME>
              <COMPONENT-IREF>
                <TARGET-COMPONENT-REF DEST="SW-COMPONENT-PROTOTYPE">/
```

```
BBW_SWC_Description/SwCompossitions/RootSwComposition/ABS_FR_Pt</
TARGET—COMPONENT—REF>
                  </COMPONENT—IREF>
                  <RUNNABLE—REF DEST="RUNNABLE—ENTITY">/BBW_SWC_Description/
SwComponents/ABS_T_Information/ABS_T//ABS_T</RUNNABLE—REF>
                  <TD—EVENT—SWC—INTERNAL—BEHAVIOR—TYPE>RUNNABLE—ENTITY—ACTIVATED</
TD—EVENT—SWC—INTERNAL—BEHAVIOR—TYPE>
                </TD—EVENT—SWC—INTERNAL—BEHAVIOR>
                <TD—EVENT—SWC—INTERNAL—BEHAVIOR UUID="">
                  <SHORT—NAME>Event_ABS_RL_Pt</SHORT—NAME>
                  <COMPONENT—IREF>
                    <TARGET—COMPONENT—REF DEST="SW—COMPONENT—PROTOTYPE">/
BBW_SWC_Description/SwCompossitions/RootSwComposition/ABS_RL_Pt</
TARGET—COMPONENT—REF>
                  </COMPONENT—IREF>
                  <RUNNABLE—REF DEST="RUNNABLE—ENTITY">/BBW_SWC_Description/
SwComponents/ABS_T_Information/ABS_T//ABS_T</RUNNABLE—REF>
                  <TD—EVENT—SWC—INTERNAL—BEHAVIOR—TYPE>RUNNABLE—ENTITY—ACTIVATED</
TD—EVENT—SWC—INTERNAL—BEHAVIOR—TYPE>
                </TD—EVENT—SWC—INTERNAL—BEHAVIOR>
                <TD—EVENT—SWC—INTERNAL—BEHAVIOR UUID="">
                  <SHORT—NAME>Event_ABS_RR_Pt</SHORT—NAME>
                  <COMPONENT—IREF>
                    <TARGET—COMPONENT—REF DEST="SW—COMPONENT—PROTOTYPE">/
BBW_SWC_Description/SwCompossitions/RootSwComposition/ABS_RR_Pt</
TARGET—COMPONENT—REF>
                  </COMPONENT—IREF>
                  <RUNNABLE—REF DEST="RUNNABLE—ENTITY">/BBW_SWC_Description/
SwComponents/ABS_T_Information/ABS_T//ABS_T</RUNNABLE—REF>
                  <TD—EVENT—SWC—INTERNAL—BEHAVIOR—TYPE>RUNNABLE—ENTITY—ACTIVATED</
TD—EVENT—SWC—INTERNAL—BEHAVIOR—TYPE>
                </TD—EVENT—SWC—INTERNAL—BEHAVIOR>
                <TD—EVENT—SWC—INTERNAL—BEHAVIOR UUID="">
                  <SHORT—NAME>Event_BrakePedalLDM</SHORT—NAME>
                  <COMPONENT—IREF>
                    <TARGET—COMPONENT—REF DEST="SW—COMPONENT—PROTOTYPE">/
BBW_SWC_Description/SwCompossitions/RootSwComposition/pBrakePedalLDM</
TARGET—COMPONENT—REF>
                  </COMPONENT—IREF>
                  <RUNNABLE—REF DEST="RUNNABLE—ENTITY">/BBW_SWC_Description/
SwComponents/BrakePedalLDM_T_Information/BrakePedalLDM_T//BrakePedalLDM_T</
RUNNABLE—REF>
                  <TD—EVENT—SWC—INTERNAL—BEHAVIOR—TYPE>RUNNABLE—ENTITY—ACTIVATED</
TD—EVENT—SWC—INTERNAL—BEHAVIOR—TYPE>
                </TD—EVENT—SWC—INTERNAL—BEHAVIOR>
                <TD—EVENT—SWC—INTERNAL—BEHAVIOR UUID="">
                  <SHORT—NAME>Event_GlobalBrakeController</SHORT—NAME>
                  <COMPONENT—IREF>
                    <TARGET—COMPONENT—REF DEST="SW—COMPONENT—PROTOTYPE">/
BBW_SWC_Description/SwCompossitions/RootSwComposition/pGlobalBrakeController</
TARGET—COMPONENT—REF>
                  </COMPONENT—IREF>
                  <RUNNABLE—REF DEST="RUNNABLE—ENTITY">/BBW_SWC_Description/
SwComponents/GlobalBrakeController_Information/GlobalBrakeController//
GlobalBrakeController</RUNNABLE—REF>
                  <TD—EVENT—SWC—INTERNAL—BEHAVIOR—TYPE>RUNNABLE—ENTITY—ACTIVATED</
TD—EVENT—SWC—INTERNAL—BEHAVIOR—TYPE>
                </TD—EVENT—SWC—INTERNAL—BEHAVIOR>
                <TD—EVENT—SWC—INTERNAL—BEHAVIOR UUID="">
                  <SHORT—NAME>Event_pLDM_Brake_FL</SHORT—NAME>
                  <COMPONENT—IREF>
                    <TARGET—COMPONENT—REF DEST="SW—COMPONENT—PROTOTYPE">/
BBW_SWC_Description/SwCompossitions/RootSwComposition/pLDM_Brake_FL</
TARGET—COMPONENT—REF>
                  </COMPONENT—IREF>
                  <RUNNABLE—REF DEST="RUNNABLE—ENTITY">/BBW_SWC_Description/
SwComponents/BrakeActuatorLDM_Information/BrakeActuatorLDM//BrakeActuatorLDM</
RUNNABLE—REF>
                  <TD—EVENT—SWC—INTERNAL—BEHAVIOR—TYPE>RUNNABLE—ENTITY—TERMINATED</
TD—EVENT—SWC—INTERNAL—BEHAVIOR—TYPE>
```

```xml
                </TD-EVENT-SWC-INTERNAL-BEHAVIOR>
                <TD-EVENT-SWC-INTERNAL-BEHAVIOR UUID="">
                  <SHORT-NAME>Event_pLDM_Brake_FR</SHORT-NAME>
                  <COMPONENT-IREF>
                    <TARGET-COMPONENT-REF DEST="SW-COMPONENT-PROTOTYPE">/BBW_SWC_Description/SwCompossitions/RootSwComposition/pLDM_Brake_FR</TARGET-COMPONENT-REF>
                  </COMPONENT-IREF>
                  <RUNNABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/BrakeActuatorLDM_Information/BrakeActuatorLDM//BrakeActuatorLDM</RUNNABLE-REF>
                  <TD-EVENT-SWC-INTERNAL-BEHAVIOR-TYPE>RUNNABLE-ENTITY-TERMINATED</TD-EVENT-SWC-INTERNAL-BEHAVIOR-TYPE>
                </TD-EVENT-SWC-INTERNAL-BEHAVIOR>
                <TD-EVENT-SWC-INTERNAL-BEHAVIOR UUID="">
                  <SHORT-NAME>Event_pLDM_Brake_RL</SHORT-NAME>
                  <COMPONENT-IREF>
                    <TARGET-COMPONENT-REF DEST="SW-COMPONENT-PROTOTYPE">/BBW_SWC_Description/SwCompossitions/RootSwComposition/pLDM_Brake_RL</TARGET-COMPONENT-REF>
                  </COMPONENT-IREF>
                  <RUNNABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/BrakeActuatorLDM_Information/BrakeActuatorLDM//BrakeActuatorLDM</RUNNABLE-REF>
                  <TD-EVENT-SWC-INTERNAL-BEHAVIOR-TYPE>RUNNABLE-ENTITY-TERMINATED</TD-EVENT-SWC-INTERNAL-BEHAVIOR-TYPE>
                </TD-EVENT-SWC-INTERNAL-BEHAVIOR>
                <TD-EVENT-SWC-INTERNAL-BEHAVIOR UUID="">
                  <SHORT-NAME>Event_pLDM_Brake_RR</SHORT-NAME>
                  <COMPONENT-IREF>
                    <TARGET-COMPONENT-REF DEST="SW-COMPONENT-PROTOTYPE">/BBW_SWC_Description/SwCompossitions/RootSwComposition/pLDM_Brake_RR</TARGET-COMPONENT-REF>
                  </COMPONENT-IREF>
                  <RUNNABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/BrakeActuatorLDM_Information/BrakeActuatorLDM//BrakeActuatorLDM</RUNNABLE-REF>
                  <TD-EVENT-SWC-INTERNAL-BEHAVIOR-TYPE>RUNNABLE-ENTITY-TERMINATED</TD-EVENT-SWC-INTERNAL-BEHAVIOR-TYPE>
                </TD-EVENT-SWC-INTERNAL-BEHAVIOR>
              </TIMING-DESCRIPTIONS>
            </SWC-TIMING>
          </ELEMENTS>
        </AR-PACKAGE>
        <AR-PACKAGE>
          <SHORT-NAME>Mapping</SHORT-NAME>
          <ADMIN-DATA>
            <SDGS>
              <SDG GID="_conversion">
                <SDG GID="PROCESS-TO-MACHINE-MAPPING-SET">
                  <SDG GID="PROCESS-TO-MACHINE-MAPPINGS">
                    <SDG GID="PROCESS-TO-MACHINE-MAPPING" />
                    <SD GID="_mixed" />
                  </SDG>
                  <SD GID="_mixed" />
                </SDG>
                <SD GID="_path">PROCESS-TO-MACHINE-MAPPING-SET@0/ELEMENTS@2</SD>
                <SD GID="_target">org.artop.aal.autosar4430</SD>
              </SDG>
            </SDGS>
          </ADMIN-DATA>
          <ELEMENTS>
            <SWC-BSW-MAPPING>
              <RUNNABLE-MAPPINGS>
                <SWC-BSW-RUNNABLE-MAPPING>
                  <SWC-RUNNABLE-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/ABS_T_Information/ABS_T//ABS_T</SWC-RUNNABLE-REF>
                </SWC-BSW-RUNNABLE-MAPPING>
              </RUNNABLE-MAPPINGS>
            </SWC-BSW-MAPPING>
```

```
      </ELEMENTS>
    </AR-PACKAGE>
    <AR-PACKAGE>
      <SHORT-NAME>OS</SHORT-NAME>
      <ELEMENTS>
        <SYSTEM>
          <SHORT-NAME>MICROSAR</SHORT-NAME>
          <CATEGORY>Operating System</CATEGORY>
        </SYSTEM>
      </ELEMENTS>
    </AR-PACKAGE>
    <AR-PACKAGE UUID="">
      <SHORT-NAME>SwComponents</SHORT-NAME>
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>ABS_T_Information</SHORT-NAME>
          <ELEMENTS>
            <APPLICATION-SW-COMPONENT-TYPE>
              <SHORT-NAME>ABS_T</SHORT-NAME>
              <INTERNAL-BEHAVIORS>
                <SWC-INTERNAL-BEHAVIOR UUID="">
                  <SHORT-NAME />
                  <EVENTS>
                    <TIMING-EVENT S="" UUID="">
                      <SHORT-NAME>stimulus_5ms</SHORT-NAME>
                      <ADMIN-DATA>
                        <SDGS>
                          <SDG GID="_conversion">
                            <SDG GID="OFFSET">
                              <SD GID="_mixed">0.0</SD>
                            </SDG>
                            <SD GID="_path">OFFSET@3/</SD>
                            <SD GID="_target">org.artop.aal.autosar4460</SD>
                          </SDG>
                        </SDGS>
                      </ADMIN-DATA>
                      <START-ON-EVENT-REF DEST="RUNNABLE-ENTITY">/
 BBW_SWC_Description/SwComponents/ABS_T_Information/ABS_T//ABS_T</
START-ON-EVENT-REF>
                      <PERIOD>0.005</PERIOD>
                    </TIMING-EVENT>
                  </EVENTS>
                  <RUNNABLES>
                    <RUNNABLE-ENTITY>
                      <SHORT-NAME>ABS_T</SHORT-NAME>
                      <MINIMUM-START-INTERVAL>0.0</MINIMUM-START-INTERVAL>
                      <SYMBOL />
                    </RUNNABLE-ENTITY>
                  </RUNNABLES>
                </SWC-INTERNAL-BEHAVIOR>
              </INTERNAL-BEHAVIORS>
            </APPLICATION-SW-COMPONENT-TYPE>
          </ELEMENTS>
        </AR-PACKAGE>
        <AR-PACKAGE>
          <SHORT-NAME>BrakeActuatorLDM_Information</SHORT-NAME>
          <ELEMENTS>
            <APPLICATION-SW-COMPONENT-TYPE S="" UUID="">
              <SHORT-NAME>BrakeActuatorLDM</SHORT-NAME>
              <INTERNAL-BEHAVIORS>
                <SWC-INTERNAL-BEHAVIOR>
                  <EVENTS>
                    <TIMING-EVENT>
                      <SHORT-NAME>stimulus_6ms</SHORT-NAME>
                      <ADMIN-DATA>
                        <SDGS>
                          <SDG GID="_conversion">
                            <SDG GID="OFFSET">
                              <SD GID="_mixed">0.0</SD>
                            </SDG>
```

```xml
                      <SD GID="_path">OFFSET@3/</SD>
                      <SD GID="_target">org.artop.aal.autosar4460</SD>
                    </SDG>
                  </SDGS>
                </ADMIN-DATA>
                <START-ON-EVENT-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/BrakeActuatorLDM_Information/BrakeActuatorLDM//BrakeActuatorLDM</START-ON-EVENT-REF>
                <PERIOD>0.006</PERIOD>
              </TIMING-EVENT>
            </EVENTS>
            <RUNNABLES>
              <RUNNABLE-ENTITY S="">
                <SHORT-NAME>BrakeActuatorLDM</SHORT-NAME>
                <MINIMUM-START-INTERVAL>0.0</MINIMUM-START-INTERVAL>
              </RUNNABLE-ENTITY>
            </RUNNABLES>
          </SWC-INTERNAL-BEHAVIOR>
        </INTERNAL-BEHAVIORS>
      </APPLICATION-SW-COMPONENT-TYPE>
    </ELEMENTS>
  </AR-PACKAGE>
  <AR-PACKAGE>
    <SHORT-NAME>BrakePedalLDM_T_Information</SHORT-NAME>
    <ELEMENTS>
      <APPLICATION-SW-COMPONENT-TYPE S="" UUID="">
        <SHORT-NAME>BrakePedalLDM_T</SHORT-NAME>
        <INTERNAL-BEHAVIORS>
          <SWC-INTERNAL-BEHAVIOR>
            <EVENTS>
              <TIMING-EVENT>
                <SHORT-NAME>stimulus_2ms</SHORT-NAME>
                <ADMIN-DATA>
                  <SDGS>
                    <SDG GID="_conversion">
                      <SDG GID="OFFSET">
                        <SD GID="_mixed">0.0</SD>
                      </SDG>
                      <SD GID="_path">OFFSET@3/</SD>
                      <SD GID="_target">org.artop.aal.autosar4460</SD>
                    </SDG>
                  </SDGS>
                </ADMIN-DATA>
                <START-ON-EVENT-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/BrakePedalLDM_T_Information/BrakePedalLDM_T//BrakePedalLDM_T</START-ON-EVENT-REF>
                <PERIOD>0.002</PERIOD>
              </TIMING-EVENT>
            </EVENTS>
            <RUNNABLES>
              <RUNNABLE-ENTITY>
                <SHORT-NAME>BrakePedalLDM_T</SHORT-NAME>
                <MINIMUM-START-INTERVAL>0.0</MINIMUM-START-INTERVAL>
                <REENTRANCY-LEVEL>MULTICORE-REENTRANT</REENTRANCY-LEVEL>
              </RUNNABLE-ENTITY>
            </RUNNABLES>
          </SWC-INTERNAL-BEHAVIOR>
        </INTERNAL-BEHAVIORS>
      </APPLICATION-SW-COMPONENT-TYPE>
    </ELEMENTS>
  </AR-PACKAGE>
  <AR-PACKAGE>
    <SHORT-NAME>BrakeTorqMap_Information</SHORT-NAME>
    <ELEMENTS>
      <APPLICATION-SW-COMPONENT-TYPE UUID="">
        <SHORT-NAME>BrakeTorqMap</SHORT-NAME>
        <SHORT-NAME-PATTERN />
        <INTERNAL-BEHAVIORS>
          <SWC-INTERNAL-BEHAVIOR>
            <EVENTS>
```

```
                    <TIMING-EVENT UUID="">
                      <SHORT-NAME>stimulus_3ms</SHORT-NAME>
                      <ADMIN-DATA>
                        <SDGS>
                          <SDG GID="_conversion">
                            <SDG GID="OFFSET">
                              <SD GID="_mixed">0.0</SD>
                            </SDG>
                            <SD GID="_path">OFFSET@3/</SD>
                            <SD GID="_target">org.artop.aal.autosar4460</SD>
                          </SDG>
                        </SDGS>
                      </ADMIN-DATA>
                      <START-ON-EVENT-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/BrakeTorqMap_Information/BrakeTorqMap//BrakeTorqMap</START-ON-EVENT-REF>
                      <PERIOD>0.003</PERIOD>
                    </TIMING-EVENT>
                  </EVENTS>
                  <RUNNABLES>
                    <RUNNABLE-ENTITY S="">
                      <SHORT-NAME>BrakeTorqMap</SHORT-NAME>
                      <MINIMUM-START-INTERVAL>0.0</MINIMUM-START-INTERVAL>
                    </RUNNABLE-ENTITY>
                  </RUNNABLES>
                </SWC-INTERNAL-BEHAVIOR>
              </INTERNAL-BEHAVIORS>
            </APPLICATION-SW-COMPONENT-TYPE>
          </ELEMENTS>
        </AR-PACKAGE>
        <AR-PACKAGE UUID="">
          <SHORT-NAME>GlobalBrakeController_Information</SHORT-NAME>
          <ELEMENTS>
            <APPLICATION-SW-COMPONENT-TYPE S="">
              <SHORT-NAME>GlobalBrakeController</SHORT-NAME>
              <CATEGORY />
              <INTERNAL-BEHAVIORS>
                <SWC-INTERNAL-BEHAVIOR>
                  <EVENTS>
                    <TIMING-EVENT S="" UUID="">
                      <SHORT-NAME>stimulus_4ms</SHORT-NAME>
                      <ADMIN-DATA>
                        <SDGS>
                          <SDG GID="_conversion">
                            <SDG GID="OFFSET">
                              <SD GID="_mixed">0.0</SD>
                            </SDG>
                            <SD GID="_path">OFFSET@3/</SD>
                            <SD GID="_target">org.artop.aal.autosar4460</SD>
                          </SDG>
                        </SDGS>
                      </ADMIN-DATA>
                      <START-ON-EVENT-REF DEST="RUNNABLE-ENTITY">/BBW_SWC_Description/SwComponents/GlobalBrakeController_Information/GlobalBrakeController//GlobalBrakeController</START-ON-EVENT-REF>
                      <PERIOD>0.004</PERIOD>
                    </TIMING-EVENT>
                  </EVENTS>
                  <RUNNABLES>
                    <RUNNABLE-ENTITY>
                      <SHORT-NAME>GlobalBrakeController</SHORT-NAME>
                      <MINIMUM-START-INTERVAL>0.0</MINIMUM-START-INTERVAL>
                    </RUNNABLE-ENTITY>
                  </RUNNABLES>
                </SWC-INTERNAL-BEHAVIOR>
              </INTERNAL-BEHAVIORS>
            </APPLICATION-SW-COMPONENT-TYPE>
          </ELEMENTS>
        </AR-PACKAGE>
      </AR-PACKAGES>
    </AR-PACKAGE>
    <AR-PACKAGE UUID="">
      <SHORT-NAME>SwCompossitions</SHORT-NAME>
      <ELEMENTS>
        <COMPOSITION-SW-COMPONENT-TYPE>
          <SHORT-NAME>RootSwComposition</SHORT-NAME>
          <COMPONENTS>
            <SW-COMPONENT-PROTOTYPE>
              <SHORT-NAME>ABS_FL_Pt</SHORT-NAME>
              <TYPE-TREF DEST="APPLICATION-SW-COMPONENT-TYPE">/BBW_SWC_Description/SwComponents/ABS_T_Information/ABS_T</TYPE-TREF>
            </SW-COMPONENT-PROTOTYPE>
            <SW-COMPONENT-PROTOTYPE>
              <SHORT-NAME>ABS_FR_Pt</SHORT-NAME>
              <TYPE-TREF DEST="APPLICATION-SW-COMPONENT-TYPE">/BBW_SWC_Description/SwComponents/ABS_T_Information/ABS_T</TYPE-TREF>
            </SW-COMPONENT-PROTOTYPE>
            <SW-COMPONENT-PROTOTYPE>
              <SHORT-NAME>ABS_RL_Pt</SHORT-NAME>
              <TYPE-TREF DEST="APPLICATION-SW-COMPONENT-TYPE">/BBW_SWC_Description/SwComponents/ABS_T_Information/ABS_T</TYPE-TREF>
            </SW-COMPONENT-PROTOTYPE>
            <SW-COMPONENT-PROTOTYPE>
              <SHORT-NAME>ABS_RR_Pt</SHORT-NAME>
              <TYPE-TREF DEST="APPLICATION-SW-COMPONENT-TYPE">/BBW_SWC_Description/SwComponents/ABS_T_Information/ABS_T</TYPE-TREF>
            </SW-COMPONENT-PROTOTYPE>
            <SW-COMPONENT-PROTOTYPE UUID="">
              <SHORT-NAME>pBrakePedalLDM</SHORT-NAME>
              <TYPE-TREF DEST="APPLICATION-SW-COMPONENT-TYPE">/BBW_SWC_Description/SwComponents/BrakePedalLDM_T_Information/BrakePedalLDM_T</TYPE-TREF>
            </SW-COMPONENT-PROTOTYPE>
            <SW-COMPONENT-PROTOTYPE>
              <SHORT-NAME>pBrakeTorqueMap</SHORT-NAME>
              <TYPE-TREF DEST="APPLICATION-SW-COMPONENT-TYPE">/BBW_SWC_Description/SwComponents/BrakeTorqMap_Information/BrakeTorqMap</TYPE-TREF>
            </SW-COMPONENT-PROTOTYPE>
            <SW-COMPONENT-PROTOTYPE>
              <SHORT-NAME>pGlobalBrakeController</SHORT-NAME>
              <TYPE-TREF DEST="APPLICATION-SW-COMPONENT-TYPE">/BBW_SWC_Description/SwComponents/GlobalBrakeController_Information/GlobalBrakeController</TYPE-TREF>
            </SW-COMPONENT-PROTOTYPE>
            <SW-COMPONENT-PROTOTYPE>
              <SHORT-NAME>pLDM_Brake_FL</SHORT-NAME>
              <TYPE-TREF DEST="APPLICATION-SW-COMPONENT-TYPE">/BBW_SWC_Description/SwComponents/BrakeActuatorLDM_Information/BrakeActuatorLDM</TYPE-TREF>
            </SW-COMPONENT-PROTOTYPE>
            <SW-COMPONENT-PROTOTYPE>
              <SHORT-NAME>pLDM_Brake_FR</SHORT-NAME>
              <TYPE-TREF DEST="APPLICATION-SW-COMPONENT-TYPE">/BBW_SWC_Description/SwComponents/BrakeActuatorLDM_Information/BrakeActuatorLDM</TYPE-TREF>
            </SW-COMPONENT-PROTOTYPE>
            <SW-COMPONENT-PROTOTYPE>
              <SHORT-NAME>pLDM_Brake_RL</SHORT-NAME>
              <TYPE-TREF DEST="APPLICATION-SW-COMPONENT-TYPE">/BBW_SWC_Description/SwComponents/BrakeActuatorLDM_Information/BrakeActuatorLDM</TYPE-TREF>
            </SW-COMPONENT-PROTOTYPE>
            <SW-COMPONENT-PROTOTYPE>
              <SHORT-NAME>pLDM_Brake_RR</SHORT-NAME>
              <TYPE-TREF DEST="APPLICATION-SW-COMPONENT-TYPE">/BBW_SWC_Description/SwComponents/BrakeActuatorLDM_Information/BrakeActuatorLDM</TYPE-TREF>
            </SW-COMPONENT-PROTOTYPE>
          </COMPONENTS>
          </COMPOSITION-SW-COMPONENT-TYPE>
        </ELEMENTS>
      </AR-PACKAGE>
    </AR-PACKAGES>
  </AR-PACKAGE>
</AR-PACKAGES>
<AR-PACKAGE UUID="1730698a-4ed2-4ab4-bb7d-dfdafdbd72ad">
  <SHORT-NAME>AUTOSAR_Project</SHORT-NAME>
  <AR-PACKAGES>
    <AR-PACKAGE UUID="43d54b22-1b52-49a6-b3a1-dd9c75d52f7f">
      <SHORT-NAME>BaseTypes</SHORT-NAME>
      <CATEGORY>STANDARD</CATEGORY>
      <ELEMENTS>
        <SW-BASE-TYPE UUID="c6a5ae11-6097-4e64-8ced-71efc1e94d2d">
          <SHORT-NAME>Base_16.0bit</SHORT-NAME>
          <CATEGORY>FIXED_LENGTH</CATEGORY>
          <BASE-TYPE-SIZE>16</BASE-TYPE-SIZE>
          <BASE-TYPE-ENCODING xsi:nil="true" />
          <MEM-ALIGNMENT>0</MEM-ALIGNMENT>
          <BYTE-ORDER>MOST-SIGNIFICANT-BYTE-LAST</BYTE-ORDER>
          <NATIVE-DECLARATION xsi:nil="true" />
        </SW-BASE-TYPE>
        <SW-BASE-TYPE UUID="9aa2944d-cac5-4a7f-8809-6584000192a8">
          <SHORT-NAME>Base_8.0bit</SHORT-NAME>
          <CATEGORY>FIXED_LENGTH</CATEGORY>
          <BASE-TYPE-SIZE>8</BASE-TYPE-SIZE>
          <BASE-TYPE-ENCODING xsi:nil="true" />
          <MEM-ALIGNMENT>0</MEM-ALIGNMENT>
          <BYTE-ORDER>MOST-SIGNIFICANT-BYTE-LAST</BYTE-ORDER>
          <NATIVE-DECLARATION xsi:nil="true" />
        </SW-BASE-TYPE>
      </ELEMENTS>
    </AR-PACKAGE>
    <AR-PACKAGE UUID="95f01a97-98dc-4628-a135-88b77aa79072">
      <SHORT-NAME>CompuMethods</SHORT-NAME>
      <CATEGORY>STANDARD</CATEGORY>
    </AR-PACKAGE>
    <AR-PACKAGE UUID="4470200c-00c0-42b2-8065-7453a59ae345">
      <SHORT-NAME>Constraints</SHORT-NAME>
      <ELEMENTS>
        <SWC-TIMING UUID="44de6e2c-1f99-4464-a9df-976a4253c137">
          <SHORT-NAME>TimingConstraints</SHORT-NAME>
          <TIMING-REQUIREMENTS>
            <EXECUTION-TIME-CONSTRAINT UUID="69dc11c3-f8eb-4396-a3b5-6af3678bb201
  ">
              <SHORT-NAME>GrossUpper</SHORT-NAME>
            </EXECUTION-TIME-CONSTRAINT>
          </TIMING-REQUIREMENTS>
        </SWC-TIMING>
      </ELEMENTS>
    </AR-PACKAGE>
  </AR-PACKAGES>
</AR-PACKAGE>
</AUTOSAR>
```