Transmission Modeling in Modelica: A consistent approach for several software development platforms

Jochen Köhler, Michael Kübler, Julian King
ZF Friedrichshafen AG
Graf-von-Soden-Platz 1, D-88046 Friedrichshafen, Germany
{jochen.koehler, michael.kuebler, julian.king}@zf.com

Abstract

Simulation models play a fundamental role in the development of transmission control software. In the ideal case, the same model can be used throughout the whole development process from concept and design over implementation to system verification. The idea is to use one uniform model along this V-scheme. This leads to the requirement that simulations have to be able to run in real-time on hardware-in-the-loop platforms. On the other hand, very detailed models of some components might be needed during the early design phase.

Thus, a trade-off between modeling depth and computational performance has to be found. This may be achieved by selectively simplifying parts of the model that are prone to generating stiff sub-systems or a large number of state events.

Within this framework, the present paper introduces the Modelica simulation model of TraXon, the new modular transmission for heavy commercial vehicles by ZF. The model can be adapted to various needs by replacing components according to the required modeling depth and/or dynamical behavior.

After a brief overview of the ZF in-house Modelica libraries and the architecture of the TraXon model, some approaches and tools are described for evaluating and optimizing models with respect to real-time issues.

Keywords: ZF Modelica libraries; model simplification; performance analysis; hardware-in-the-loop;

1 Introduction

ZF is a global player in driveline and chassis technology. Being confronted with a huge variety of increasingly complex transmission concepts and continuously reduced development cycle times, simulation has traditionally been an essential part of transmission control software engineering at ZF. Detailed models reflecting the mechanical structure of the transmission as well as the physical behavior of its actuating components are central to this task.

This paper outlines the Modelica simulation model of the new ZF TraXon transmission. Here, the principles of object-orientated modeling inherent to Modelica greatly contribute towards balancing both modeling accuracy and real-time performance. Thus, one simulation model can be used throughout the whole development process.

2 Usage of Modelica in ZF

2.1 Motivation

Modelica was introduced at ZF over 10 years ago. The main intention was to standardize the modeling methodology for transmission systems and to share modeling know-how by creating component libraries accessible throughout the company [1].

Modelica nowadays represents the standard approach for modeling and simulating of a wide range of distinct transmission systems, particularly in the context of control algorithm development. Originally starting from one single ZF Modelica Library, more than ten different context-specific libraries have been developed so far, all based on the Modelica Standard Library [2].

2.2 ZF Modelica Libraries

While some of the above mentioned Modelica libraries are only employed for targeted investigations regarding, e.g., shifting comfort or combustion engine dynamics, others are regularly used in almost every transmission model formulation:

- ZFLib, encompassing central gear parts such as clutches, synchronizers, table-based engine formulations, controllers, chassis models, etc. It is based on the Modelica Standard Library and also con-

DOI
10.3384/ECP14096259

tains some extensions to the components therein.
- HybridLib, including extensions for hybrid powertrains (electrical machines, battery models and complete driving strategies for hybrid vehicles).
- FluidLib for pneumatic/hydraulic actuating cylinders and valves. This library was built upon Modelica.Media and Modelica.Fluid. The separation of fluid-mechanic components and the employed media is quite useful in order to design models that can be used for both hydraulics and pneumatics.

The motivation for developing in-house libraries rather than using commercial ones stems from the following requirements:

- Adaptability of components according to internal standards, e.g., with respect to data handling and parameterization [1].
- Preservation of ZF-specific modeling know-how inside components.
- Synergy effects when modeling closely related transmission concepts.
- Models perfectly tailored to the needs of ZF.

In particular, the central libraries are widely reused among distinct transmission modeling projects, thus leading to thoroughly tested components characterized by a high degree of reliability and robustness.

Dymola [3] represents the standard front-end tool. However, as Modelica is tool-independent current efforts also aim at evaluating the usability of the ZF Modelica libraries in SimulationX [4]. Then, users may choose the most suitable tool for their task.

2.3 Model Export

In addition to the unified modeling approach described above, also a standardized export process has been implemented. The possibility to make simulation models available on various software design and testing platforms is an essential requirement. One common software development and integration tool is Simulink [5] with its ability to export models to other platforms.

For this purpose, the Dymola-Simulink interface has been adapted in order to serve all simulation environments used at ZF, especially ZF-internal software-in-the-loop and hardware-in-the-loop platforms. One particular extension to the original interface is that different fixed-step solvers are attached to the exported models. These solvers can be selected just before running the simulation, so the user may decide whether he wants to use the Simulink solver or one of the built-in fixed-step integrators. Furthermore, the inner simulation time step can be defined independently from the Simulink solver step size. If the inner step size is smaller than the global step size a sample and hold mechanism is implemented for in- and outputs.

3 The TraXon simulation model

3.1 TraXon – the new modular transmission by ZF

TraXon (Figure 1) is an automatic commercial vehicle transmission platform with five modules for different driving applications, satisfying the most challenging requirements. It is prepared for intelligent networking with other vehicle systems, and sets new standards in the areas of efficiency, comfort, and application diversity [6].

Figure 1: TraXon transmission

3.2 TraXon simulation model: overview

The modular structure of the TraXon transmission is also reflected in the structure of the simulation model. Correspondingly, the TraXon gearbox model consists of six mechanical sub-systems in series, each one associated with its own (pneumatic) actuator concept (Figure 2). In particular, these sub-systems all have a uniform connector interface and can thus easily be rearranged to model a wide spectrum of distinct transmission topologies in a consistent manner.

Figure 3 exemplifies the general structure of such a sub-system (splitter). Note that there is a clear separation between the rotational part, mimicking the torque flow through a synchronizer depending on the position of the sleeve, and the actuator, which trans-

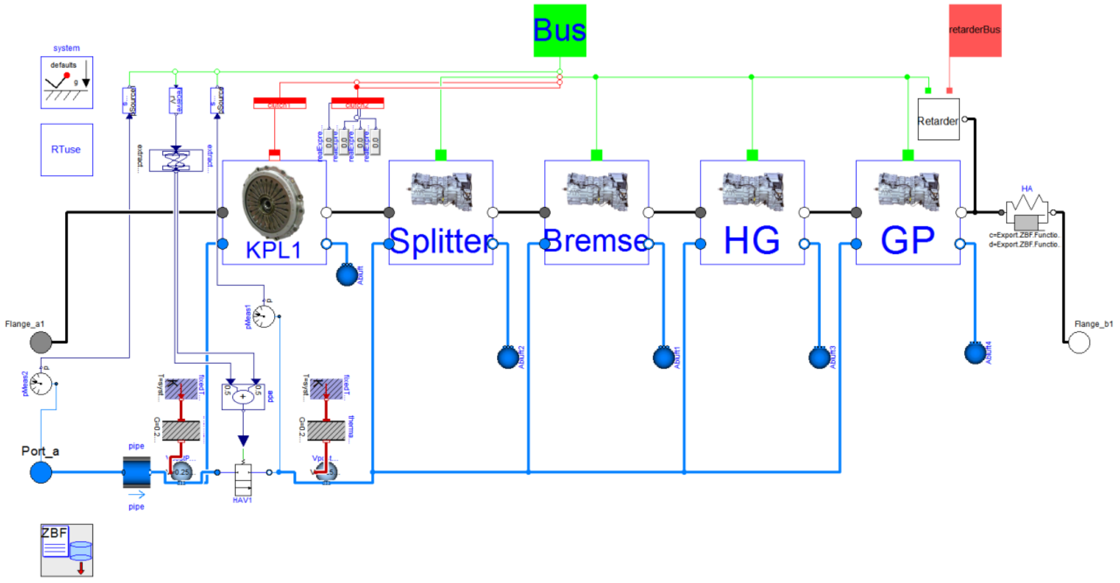


Figure 2: TraXon simulation model: engine clutch, splitter (synchronizer), brake, main gear (dog clutch transmission with specified shifting geometry), group transmission (synchronizer) and retarder

lationally moves the sleeve according to the pressure relationships inside a two-position pneumatic cylinder.

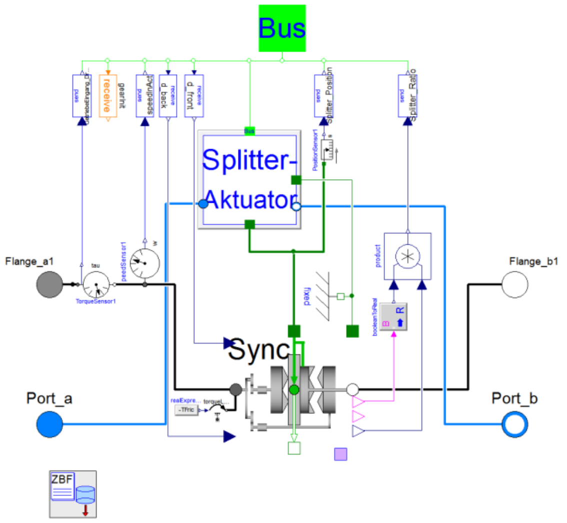


Figure 3: Splitter part of the TraXon transmission, modeling a synchronizer actuated by a pneumatic cylinder

By extending from a common interface, different actuator concepts (e.g., electro-magnetic or electro-mechanical) and actuator formulations with varying modeling depth can thus quickly be exchanged and simulated. This is especially important when the model needs to be simplified for real-time simulations.

3.3 Detailed modeling approach

The master model contains all relevant physical effects of the pneumatic actuators and can be employed for detailed MiL/SiL-investigations during function development.

3.3.1 Detailed actuator cylinder model

Almost all components of TraXon are actuated by pneumatic cylinders due to the fact that this kind of energy is already available on heavy trucks. The physical model of these actuator cylinders consists of replaceable variants of chamber and piston components. The most detailed piston model considers friction with stick-slip-effects. This results in high calculation effort, so the piston may also be replaced (using the “replaceable” mechanism of Modelica) by other variants building on continuous friction modeling approaches. End-stops are generally not taken into account; instead they are incorporated into the translational dynamics of actuated shifting device. This helps to reduce the number of equations without any effect on the overall behavior of the model. The detent of the piston is also modeled, assuring that the piston stays in a pre-defined position if not actuated.

Chambers are composed of a fluid volume and a translational flange, transmitting force according to the product of fluid pressure and plunger area. The latter may be constant or position-dependent as in the case of “three-position” cylinders, where the piston can be held at a center-position.

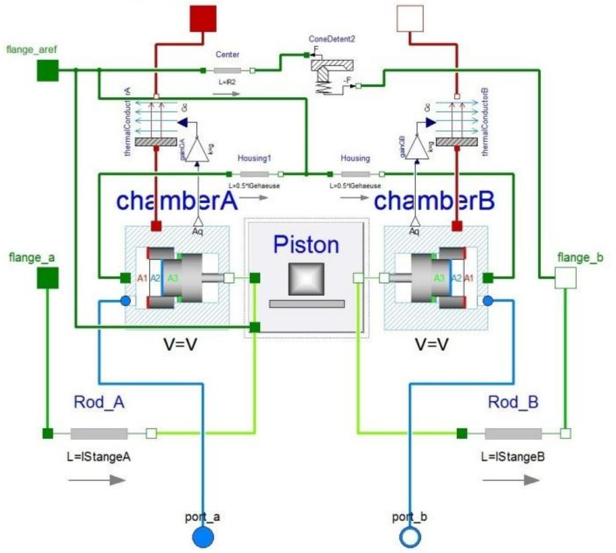


Figure 4: Detailed cylinder model using replaceable components

A major advantage here is that the modeling media are separated from the components so that the cylinder models can be used for pneumatic and also for hydraulic systems just by changing the media.

3.3.2 Detailed actuator valve model

The actuator valve models governing the venting behavior of the respective cylinders are divided into an electrical, mechanical and fluid-dynamic part.

The electrical part contains coil inductance and electro-magnetic force. Control signals are generated by Pulse-Width-Modulation. Simplifications are made in the magnetic part by neglecting hysteresis and using tables instead. The mechanical part consists of a translational spring-mass system for the valve tappet. End-stops are modelled and friction can be activated but is not used at the moment.

The fluid-dynamic part is divided into a geometrical and a fluid flow part, both of which are replaceable. The geometrical part yields the valve opening resulting from a specific position of the tappet. Different valve seating geometries can be realized, including flat seat and ball-drill/-cone seat. The fluid flow part calculates mass flow depending on the valve opening.

Here, changing from pneumatics to hydraulics is almost as easy as in the case of cylinders. The only component that has to be replaced is the inner valve containing the equations to calculate the pressure drop. This is due to the fact that a different behavior results dependent on the compressibility of the media (compressible (e.g., air) vs. incompressible (e.g., simple oil)).

3.4 Simplifications made for real-time computations

Since the TraXon simulation model aims at accompanying the entire software development process, there is a special constraint: real-time capability.

To meet this challenge, certain model simplifications have to be implemented in order to guarantee a fast and robust simulation on HiL-platforms, usually employing an Euler-forward algorithm with a fixed step size of 1 ms.

In this context, extensive use is made of Modelica's inherent feature to make certain model components easily exchangeable. In particular, by declaring the above-mentioned actuator parts as replaceable, cylinder and valve formulations with different levels of detail may directly be incorporated into the model without changing the surrounding structure. This feature is especially beneficial during model simplification steps as outlined below.

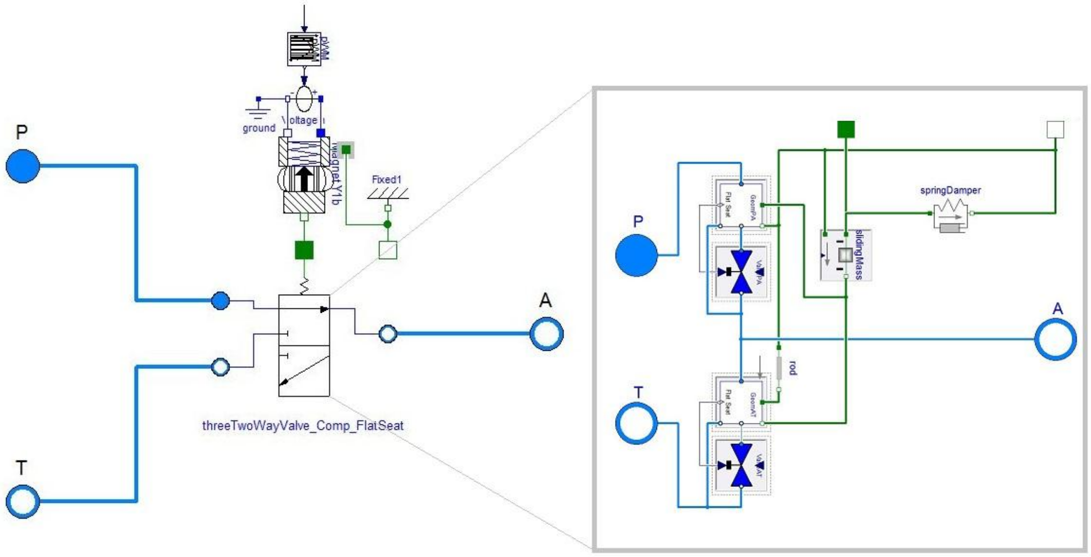


Figure 5: Detailed valve model composed of electrical, mechanical and fluid-dynamic part

3.4.1 Simplified actuator cylinder model

In the simplified cylinder model the plunger areas are parameterized such that the neutral position corresponds to a force equilibrium if both chambers have ambient pressure. Therefore, the detent force can usually be neglected. Friction is modeled in a velocity-dependent way in order to avoid stick-slip-events and thus the creation of many event iterations.

The volumes of the in- and outgoing pipes are lumped into the dead volume of the chambers. This avoids fast dynamics due to small fluid volumes and hence leads to better numerical stability.

3.4.2 Simplified actuator valve model

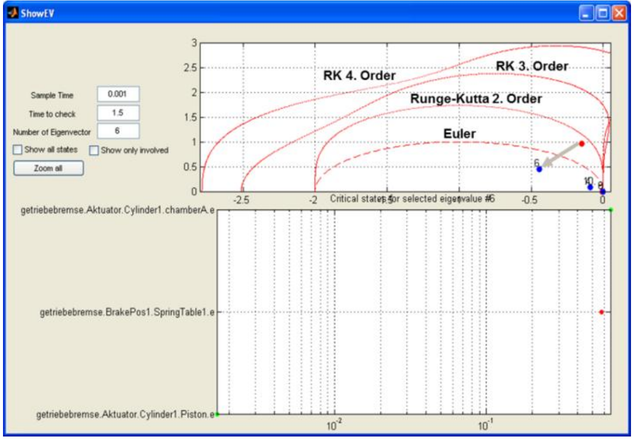
In contrast to the detailed valve model the electro-mechanical part is replaced by a second-order transfer function. Thus, higher frequencies of the valve piston movement can be avoided. Also the generation of events is reduced by neglecting end-stops of the piston.

3.5 Stability analysis

If simulation has to be performed with a fixed step solver, the stability region of the integrator has to be taken into account. Facing a complex model like the one introduced before, with multiple domains interacting with each other, it is difficult to identify the critical modes of the system, i.e., those modes that do not fulfill the stability requirements because their eigen-frequencies are too fast.

To overcome this problem a MATLAB [5] tool was implemented, which visualizes all eigenmodes and eigenvectors that can be computed by linearizing the nonlinear system at certain time points of interest.

The linearizing feature of Dymola is used for this purpose. The top panel of Figure 6 shows the stability regions for the implemented fixed step solvers.



Sample Time 0.001
Time to check 1.5
Number of Eigenvector 6
□ Show all states □ Show only involved

ShowEV
Zoom all
3
2.5
2
1.5
1
0.5
0
-2.5
-2
-1.5
-1
-0.5
0
RK 4. Order
RK 3. Order
Runge-Kutta 2. Order
Euler
6
10
8
Critical states for selected eigenvalue #6
getriebebremse.Aktuator.Cylinder1.chamberA.e
getriebebremse.BrakePos1.SpringTable1.e
getriebebremse.Aktuator.Cylinder1.Piston.e
10^{-2}
10^{-1}

Figure 6: Eigenmode analysis tool

The blue dots show all eigenmodes in the z-space. Each dot that does not fall within the stability region of the used solver has to be investigated. At the bottom of Figure 6, the eigenvector of one selected eigenmode is shown.

This links the eigenmode to the contributing state variables and thus allows for identifying the corresponding components in the model. To enhance this feature, special states are included in all components, representing the stored energy. The eigenvectors can be “filtered” according to these energy states, making it even easier to find the components of interest. With this information, the engineer may decide to change the model structurally or to adapt some physical parameters, in order to move the eigenmodes into the stability region without modifying the dynamical behavior too much.

3.6 Performance analysis

A model analysis after symbolic manipulation of the underlying equations shows the differences between the detailed and simplified modeling approach as discussed above, see Table 1. Note that the number of mixed real/discrete systems of equations is drastically reduced, hence less computational effort is expected during event iterations.

	Detailed model	Simplified model
Number of mixed real/discrete systems of equations	20	4
Number of states	180	124
Number of linear systems of equations	4	4
Max. size of linear systems of equations	8	8
Number of nonlin. systems of equations	12	7
Max. size of nonlin. systems of equations	17	17

Table 1: Model analysis before and after simplification

The performance increase by using the simplified modeling approach can be seen in Figure 7. This computational load can be made visible because the translated model is running in our own simulation framework by calling the “dsblock” C-function [7], and the time needed for its execution is measured at each time step. A very important effect that can be seen in this plot is the occurrence of events and the time needed to handle them. In general, finding consistent restarting conditions after an event is a serious problem for real-time simulations as it seems that at every event handling the calculation time required is at least two times higher than the average time without an event. Due to this effect the maximum al-

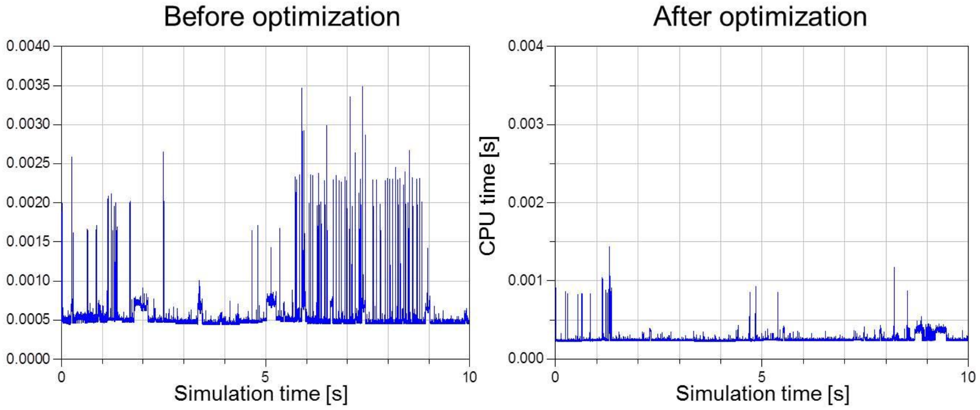


Figure 7: Computational performance before and after simplification

lowed calculation effort is half the integration step size – here 0.5 ms – to avoid time overruns.
Using this analysis tool, the average time interval needed for executing one solver step can be lowered by approximately 50%. Another important effect is the reduction of state events and the parallel decrease of iteration steps necessary for finding consistent restart conditions. When using the TraXon simulation model on hardware-in-the-loop platforms this is the key to avoid overruns.

4 Conclusions

The use of simulation models along the entire software development process is widely implemented at ZF. A standardized way to export models to different simulation platforms has been established and the potential of the Functional Mockup Interface in this regard is currently evaluated.
Modelers can benefit from the availability of in-house Modelica libraries and components that are extensively reused in many simulation models.
For real-time applications, object oriented modeling provides an elegant way to rapidly switch between different model formulations with a varying degree of detail. This is a key feature when the model has to be simplified in order to ensure a fast and robust execution. Nevertheless, the analysis methods inside Dymola with respect to this task may be improved. In particular, an automated way for ranking model variables according to their contribution to different timescales in the model would be highly desirable.
Furthermore, the generation of events and the subsequent iteration process represents a drawback of Modelica in the context of for real-time applications.

References

[1] Koehler J, Banerjee A. Usage of Modelica for transmission simulation in ZF. Proceedings of the 4th Modelica Conference 2005, Hamburg, Germany, Modelica Association, 7-8 March 2005

[2] Modelica Standard Library (Version 3.2), Modelica Association, 2010, https://www.modelica.org/libraries

[3] Dymola 2014, Dassault Systemes http://www.3ds.com/products-services/catia/capabilities/systems-engineering/modelica-systems-simulation/dymola

[4] SimulationX 3.6, ITI GmbH, http://www.itisim.com/simulationx/

[5] Matlab / Simulink, The Mathworks, http://mathworks.com

[6] ZF Friedrichshafen AG, www.zf.com/TraXon

[7] The DSblock model interface, https://www.modelica.org/documents/DSblock

DOI 10.3384/ECP14096259