# Physical Modeling with Modelica and Dymola and Real-Time Simulation with Simulink and Real Time Workshop

**H. Elmqvist[1], M. Otter[2], and C. Schlegel[2]**

[1]Dynasim AB, Research Park Ideon, S–223 70 Lund, Sweden, email: Elmqvist@Dynasim.se

[2]DLR, P.O.B. 1116, D-82230 Wessling, Germany, email: Martin.Otter@DLR.de/Clemens.Schlegel@DLR.de

## 1 Introduction

A new language for physical modeling called Modelica has been developed in an international effort. An overview of the Modelica language will be given in the context of modeling and simulating a complex industrial system: hardware-in-the-loop simulation of an automatic gearbox. The electronic control unit hardware has been tested against a simulated model of the gearbox.

Several tools have been used to solve the problem. The first issue is the modeling. Writing the equations from scratch would be very tedious. A structuring of the model according to the physical components and being able to reuse available submodels is essential. The new Modelica modeling language and a model library for drive trains have been used.

The graphical environment of Dymola has been used for composing the model. The natural graphical representations are shown below. Efficient code is essential for real-time simulation. The Modelica translator of Dymola has been used to symbolically translate the differential-algebraic equations (DAE) to explicit state space form and to generate C code in S function MEX file format.

The S-function has been used for offline simulation in Simulink which was also used for configuring the interface to the control hardware. Real Time Workshop has been used for generation of the code for dSPACE hardware. The time for the evaluation of one Euler step is 0.65 ms on a 300 MHz DEC alpha processor.

## 2 Modelica - A Unified Object-Oriented Language for Physical Systems Modeling

A new language for physical modeling called Modelica has been developed in an international effort. The main objective is to make it easy to exchange models and model libraries. The design approach builds on non-casual modeling with true ordinary differential and algebraic equations and the use of object-oriented constructs to facilitate reuse of modeling knowledge. There are already several modeling language based on these ideas available from universities and small companies. There is also significant experience of using them in various applications. The aim of the Modelica effort is to unify the concepts and design a new uniform language for model representation.

The work started in the continuous time domain since there is a common mathematical framework in the form of differential-algebraic equation (DAE) systems. The short range goal was to design a modeling language based on DAE systems with some discrete event features to handle discontinuities and sampled systems. The design should allow an evolution to a multi-formalism, multi-domain, general-purpose modeling language.

There are presently about 15 members from 13 universities and industries in six countries in the Modelica design group. H. Elmqvist is the chairman. Information on the Modelica effort, including modeling requirements, rationale and definition of the Modelica language, is available on WWW at http://www.Dynasim.se/Modelica/.

The activity started in September 1996 as an effort within the ESPRIT project "Simulation in Europe Basic Research Working Group (SiE-WG)". In February 1997 the Modelica design effort became a Technical Committee within the Federation of European Simulation Societies, EUROSIM. The first version of the Modelica language definition was finished in September 1997. The work is now focused on the design of standard function and model libraries.

An overview of the Modelica language will be given in the context of modelling an automatic gearbox.

# 3 Hardware-in-the-loop Simulation of Gearboxes

Comfort standards regarding gearshift of automatic gearboxes of vehicles increase permanently. Gearshift comfort is mainly influenced by control of the switching elements of a gearbox: Clutches and freewheels. Control is performed mostly by an electronic control unit (ECU). Fine tuning of the ECU and the switching elements is essential to optimize gearshift comfort.

Hardware-in-the-loop (HIL) simulation is used more and more in order to speed up the development cycle. For the problem treated, such a setup consists of the ECU-hardware and a realtime simulation of all other components interacting dynamically with it: The whole drive train of engine, hydrodynamic torque converter, gearbox, differential gearbox and longitudinal dynamics of the vehicle.

Modeling such a system is difficult since the structure of the system varies during each gearshift: Depending on the actual control, different wheels and freewheels are engaged or disengaged. The integrator has to be stopped and the new structure has to be determined in accordance to the actual forces imposed on the gearwheels by the switching elements. In the example treated, the automatic gearbox has 6 switching elements leading to $2^6 = 64$ possible configurations of the system.

It is shown how such a variable structure system can be modeled with Modelica. The Modelica-translator of the simulation environment Dymola, together with Dymola's symbolic engine, is used to generate efficient code suited for realtime simulation which e.g. requires a state space form of the model. A new technique, which meets the realtime requirements like fixed sampling time, is used to iteratively find a consistent configuration after a gearshift. Therefore a hardware-in-the-loop (HIL) simulation of automatic gearboxes becomes possible. The efficiency of the generated code and the sampling frequency necessary to reproduce offline simulation results enables the usage of digital signal processors.

# 4 Model of Automatic Gearbox

Gearshift dynamics can only be simulated if the input- and output-torques of the gearbox represent a real-life vehicle maneuver. Therefore, at least the engine and the longitudinal dynamics of the vehicle has to be modeled. A Modelica model of such a system is given in figure 1 as a composition or object diagram using Dymolas object diagram editor. Every icon in the composition diagram is an instance of a Modelica class and represents a component of the drive train. Mechanical flanges of the components are modelled by Modelica connector classes and are characterized by small square boxes in the icons. A line between two connectors defines a rigid, mechanical coupling between the respective components. Dashed lines respresent directed signal flow connections.

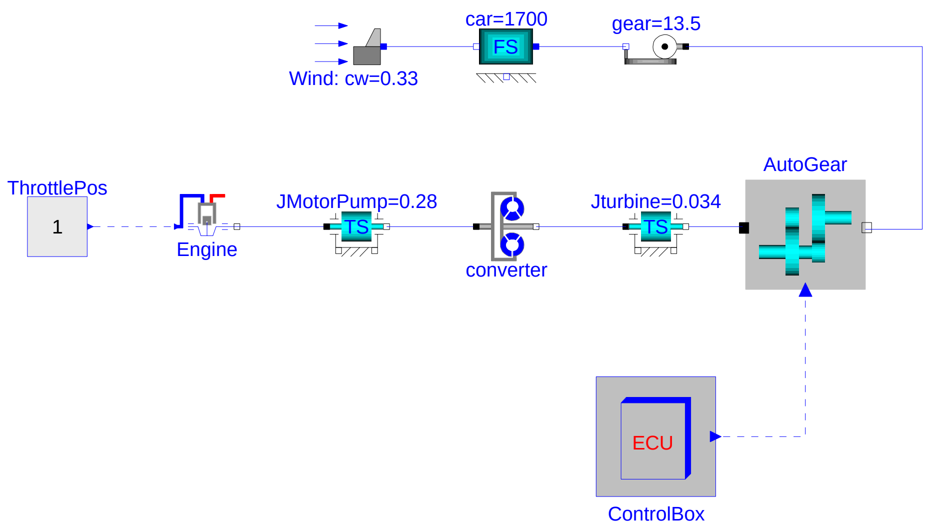



Figure 1: Composition diagram of vehicle drive train.

The driving part of the system in figure 1 consists of the engine, the torque converter and appropriate inertias. Engine and converter are modeled by tables. The driven part incorporates the automatic gearbox, the mass of the car, and aerodynamic forces. Component “ControlBox” represents the ECU which generates the gearshift signals. For offline simulations simple ramp functions are used. During HIL simulations these signals are generated by the (hardware) ECU and fed to the simulation processor via appropriate interfaces like digital IO and CAN bus.

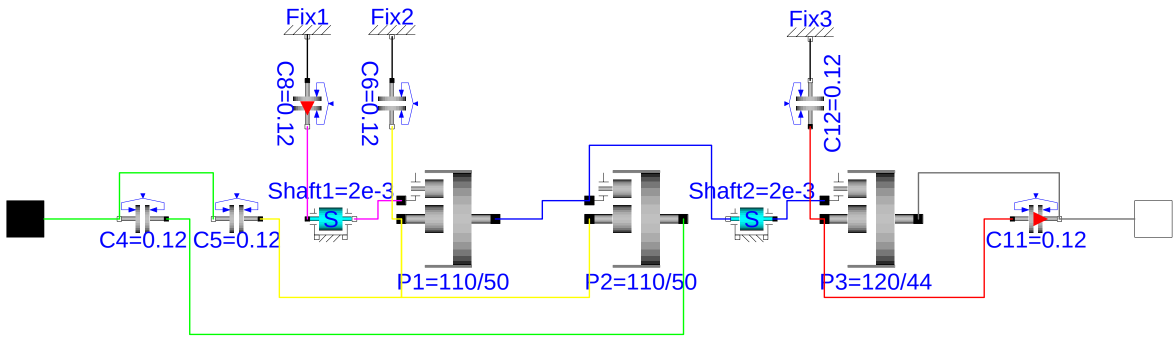



Figure 2: Composition diagram of automatic gear box “AutoGear”.

In figure 2 the Modelica model of the automatic gearbox (component “AutoGear”) is shown. It is set up by three standard planetary wheelsets, by shafts to model wheel inertias, by clutches and by combined clutch/freewheel elements (= one-way clutches).

## 5 Modelica Models of Basic Drive Train Library Components

In the previous section the vehicle drive train model was built up by the connection of basic components, like shafts and clutches, using composition diagrams. The most important components from the available drive train library are now discussed in more detail. All drive train components have mechanical flanges which are used to connect components rigidly together and which are defined by the following Modelica connector class (a connector class is a class usable in connections):

```
connector DriveCut
    AngularVelocity     : w "angular velocity of cut";
    AngularAcceleration: a "angular acceleration of cut";
    flow Torque         : t "cut-torque";
end DriveCut;
```

The keyword **flow** indicates that the Torque variables should be summed to zero at a connection point. In other words, *Torque*s are through variables. Connector class DriveCut defines the three variables “w,a,t” using basic type classes defined in the Modelica base library (type classes are classes which do not have equations):

```
type AngularVelocity     = Real(quantity ="AngularVelocity", unit ="1/s");
type AngularAcceleration = Real(quantity ="AngularAcceleration", unit ="1/s2");
type Torque              = Real(quantity ="Torque", unit ="N.m");
```

All the three variable types are real variables with a defined quantity and unit. Some typical components using this connector class are shown in table 1.

Model class *Shaft* describes a shaft with inertia and two mechanical flanges $p$ and $n$. The first two equations contain the relationship between the kinematic variables of the two flanges. The last equation is Newtons law. The third equation depends on a Boolean parameter and defines whether the angular velocity of the element is used as a state variable or not. This is e.g. needed if two shafts are connected rigidly together by a gearbox, since only one of the two shafts can have state variables in such a case in order to be able to transform the overall system to state space form. Model class *Gear* is an ideal gearbox without inertia and finally model class *DriveSpringS* is a rotational spring where the spring torque is used as state variable.

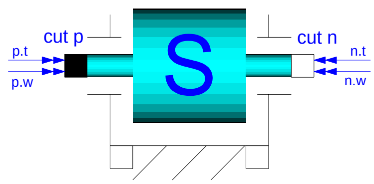
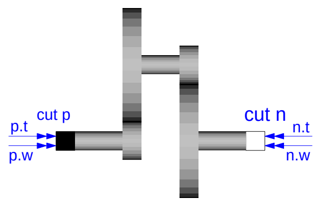
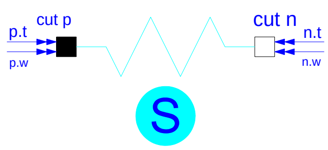
<table>
<tr><th>Graphical Representation</th><th>Modelica description</th></tr>
<tr><td>
</td><td>model Shaft<br>    parameter Inertia J “shaft inertia”;<br>    parameter Boolean state = true;<br>    DriveCut $p, n$;<br>equation<br>    $p.w + n.w = 0$;<br>    $p.a + n.a = 0$;<br>    if state then<br>        der$(p.w) = p.a$;<br>    end if<br>    $J * p.a = p.t - n.t$;<br>end Shaft;</td></tr>
<tr><td>
</td><td>model Gear<br>    parameter real ratio “gear ratio”;<br>    DriveCut $p, n$;<br>equation<br>    $p.w + ratio * n.w = 0$;<br>    $p.a + ratio * n.a = 0$;<br>    $ratio * p.t = n.t$;<br>end Gear</td></tr>
<tr><td>
</td><td>model DriveSpringS<br>    parameter Real c “spring constant”<br>    DriveCut $p, n$;<br>equation<br>    der$(p.t) = c * (p.w + n.w)$;<br>    $p.t = n.t$;<br>end DriveSpringS;</td></tr>
</table>

Table 1: Components of Modelica drive train library.

# 6 Realtime Simulation of Automatic Gearbox

The Modelica translator of the Dymola modelling and simulation environment takes the described Modelica model of the vehicle drive train and the automatic gearbox as an input, generates a differential-algebraic equation system (DAE) and transforms the DAE to state space form by symbolic manipulation and graph theoretical algorithms. The original sorted equations contain a linear system of 49 simultaneous equations, due to the shafts connected via rigid gearboxes and due to the clutch equations. Via tearing this system of equations is reduced to 10 linear equations which are solved by standard numerical procedures whenever the model is evaluated. The final equations are stored as C code which is compiled and linked as an S-function MEX file.

The simulation results from SIMULINK can be imported to the 3D animation module of Dymola. Dymola presents a 3D view with hidden surfaces removed and shading as shown in figure 3. Dymola uses the OpenGL standard, i.e. hardware accelerators on graphics boards are utilized. The visual 3D model is automatically generated from the composition diagrams of figure 1 because the components of the drive train library contain instances of built-in visual shapes like cylinders and gear wheels. The drive train components have some additional visual parameters like radius and position along the shaft.

For realtime simulation Matlab CMEX code can be generated and used in Mathworks Realtime Workshop. The time for the evaluation of one Euler step was measured as 4.7 ms on a 50 MHz Texas Instruments C40 DSP and 0.65 ms on a 300 MHz DEC alpha processor (both PC plugin cards by dSPACE Inc.). A typical gearbox ECU sampling time is 10 ms. This offers enough margin for model refinement and IO using todays typical HIL simulation processors.

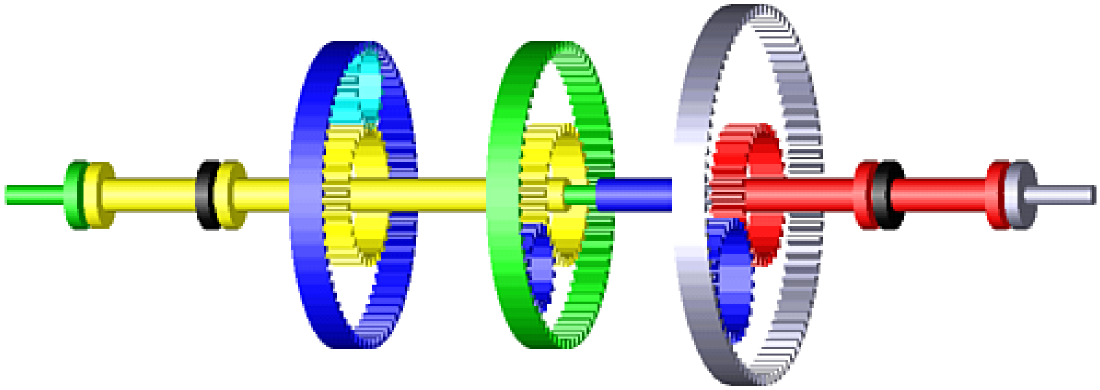


Figure 3: Automatically created animation of gear box components.

# References


[1] Elmqvist H., Brück D. and Otter M.: *Dymola – User's Manual*, Version 3.0, Dynasim AB, Lund, Sweden, 1996.

[2] Elmqvist H., Mattsson S.E.: *An introduction to the physical modeling language Modelica*. Proc. ESS '97 European Simulation Symposium, Passau, Germany, October 19-22, 1997.

[3] Modelica: *A Unified Object-Oriented Language for Physical Systems Modeling*. Modelica homepage: http://www.dynasim.se/Modelica/.

[4] Otter M.: *Object-Oriented Modeling of Drive Trains with Friction, Clutches and Brakes*. Proc. ESM'94 European Simulation Multiconference, Barcelona, Spain, June 1.-3., pp. 335-340, 1994.

[5] Otter M., Schlegel C. and Elmqvist H.: *Modeling and Realtime Simulation of an Automatic Gearbox using Modelica*. Proc. ESS '97 European Simulation Symposium, Passau, Germany, October 19-22, 1997.