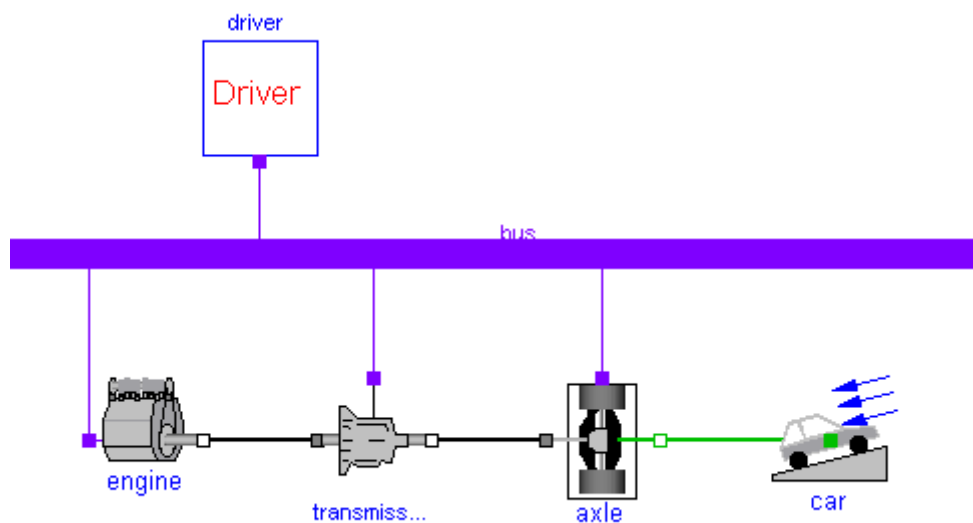




PowerTrain Library

Version 1.0

Tutorial



December 2002
German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Oberpfaffenhofen, Germany

Information: Dynasim AB
Research Park Ideon
SE-223 70 Lund
Sweden
Phone: +46 46 2862500
Fax: +46 46 2862501
E-mail: info@Dynasim.se
URL: <http://www.Dynasim.se>

Support: E-mail: support@dynasim.se

Development: German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Control Design Engineering
Postfach 11 16
82230 Wessling
Germany
E-mail: Christian.Schweiger@dlr.de
URL: <http://www.robotic.dlr.de/control>

Contributors: Martin Otter and Christian Schweiger, DLR, Germany
Ingrid Bausch-Gall, Bausch-Gall GmbH, Munich, Germany (<http://www.bausch-gall.de/>)
Mike Dempsey, Claytex Services, U.K. (<http://www.claytex.co.uk/>)
Clemens Schlegel, Schlegel Simulation GmbH, Germany (cs@schlegel-simulation.de)

Copyright ©1999-2002 by DLR. All Rights Reserved.

Contents

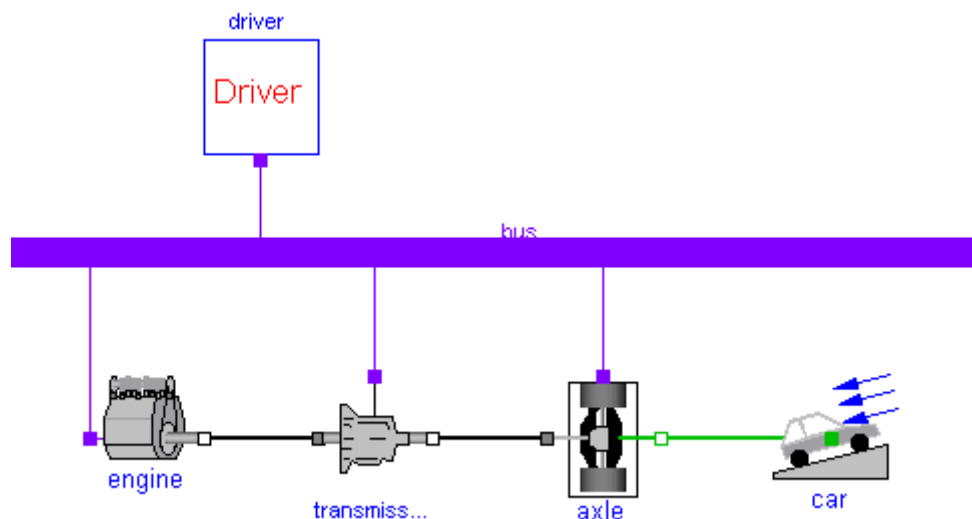
| | |
|---|-----------|
| Contents | 3 |
| 1 Introduction | 1 |
| 1.1 Main features of package PowerTrain, version 1.0 | 2 |
| 1.2 How to use this Library | 3 |
| 1.3 Acknowledgement | 3 |
| 2 Running a Demo Model | 4 |
| 3 A First Example | 8 |
| 3.1 Description of the first example | 8 |
| 3.2 Setting up the first example | 9 |
| 3.3 Simulation of the model | 12 |
| 4 DriveLine and Main Components | 15 |
| 4.1 Choice of drivers (package Variants.Drivers) | 16 |
| 4.2 Choice of engines (package Variants.Engines) | 16 |
| 4.3 Choice of transmissions (package Variants.Transmissions) | 17 |
| 4.4 Choice of axles (package Variants.Axles) | 18 |
| 4.5 Choice of car resistances (package Variants.CarResistances) | 18 |
| 5 Basic Components | 19 |
| 5.1 Package Clutches | 19 |
| 5.2 Package Gears | 22 |
| 5.3 Package Auxiliaries | 24 |
| 5.4 Package ControlUnits | 25 |
| 6 Concept of the Signal Bus | 27 |
| 6.1 First bus example | 27 |
| 6.2 Description of the bus | 27 |
| 6.3 First steps to connect a signal to the bus | 29 |
| 6.4 Model SimpleBusUsage | 31 |

| | | |
|-----------|--|-----------|
| 7 | Animation | 34 |
| 7.1 | Run a first example with animation | 34 |
| 7.2 | Animation parameters | 35 |
| 7.3 | Comments on the choice of colors | 39 |
| 8 | Demo Examples | 40 |
| 8.1 | List of all examples | 40 |
| 9 | Upgrade from Former PowerTrain Versions | 45 |
| 10 | Literature | 49 |
| 10.1 | Books | 49 |
| 10.2 | Articles | 49 |

Chapter 1

Introduction

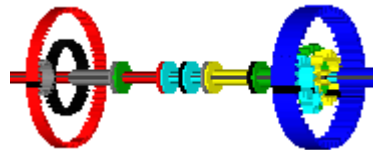
Library **PowerTrain** is a **licensed** Modelica package providing 1-dimensional rotational mechanical components for **vehicle power trains**. It is also useful in general for modeling of gear boxes. The library includes from easy to use to very sophisticated components to model power trains. It does not supply components for all specific needs. But, all components are open and can be modified or extended by the user. A typical screenshot is shown in the next figure



where the most important components are used to build-up a drive line. Note, that for all these components, different variants can be selected, e.g., a detailed 4-gear and a detailed 6-gear automatic transmission model.

1.1 Main features of package PowerTrain, version 1.0

- **45 user-callable components** (not counting interface objects), **10 example models**, 45 new components for the Modelica standard library (version 1.5) since of general interest and needed from the PowerTrain library.
- **Speed and torque dependent friction** can be handled in a robust and efficient way based on new research results. Therefore, gear efficiency is now easy to simulate. Many elements in the library, especially most clutches, gears, planetary gears, automatic gearboxes, engines, have now optional speed and torque dependent losses.
- **Bus concept** to get rid of the complex connection structures of signal connections and support the user to easily extend the bus with user-defined signals.
- **Variant selection** using the replaceable feature of Modelica. Basically, there is now just one model – PowerTrain.DriveLine – consisting of a typical drive line from which all the main variants provided in the library can be selected, e.g., 3 different types of detailed automatic gearbox models but also user-defined gearboxes.
- **Animation** of gears and shafts of a drive line as built-in feature of many components, see, e.g., the animation of a 6-speed automatic gearbox of Lepelletier type in the figure below. Animation can be switched off with a parameter to remove the animation code completely from a model, e.g., for hardware-in-the-loop simulation.



- **Control units** to control the various parts of a drive line, such as . All of them are parameterized with the control lever position of the driver (P,R,N,D,1,2,3,...) and do not depend on a specific number of gears.
- **Sophisticated examples** which may be used as a starting point for user drive line models. Especially, examples are provided for power consumption calculation and analysis of shift strategies based on detailed models of 4 and 6 speed automatic gear boxes.
- **Improved documentation** with this 54 pages tutorial and a 127 pages reference guide (both in pdf-format and in html-format for online-help).

1.2 How to use this Library

You should be familiar with Dymola and Modelica in order to use this package. PowerTrain Library version 1.0 needs Dymola version **5.0a** or later. Note, that the terms "package" and "library" are used interchangeably in this tutorial.

- **Running a Demo Model (cf. Section 2)** helps you with a quick start and gives a useful introduction to understand the concept of the library.
- **Build a first Model** gives a short step by step introduction for building a first model.
- **Model DriveLine (cf. Section 4)** combines the main components (engine, transmission, axle, resistance, driver) of a power train. For most of these components different variants can be used including your own versions.
- **Basic Components (cf. Section 5)** to build the main components (also useful for users building higher level components).
- **Concept of the Signal Bus (cf. Section 6)** explains the concept of the signal bus that is used to transport data between all components.
- **Concept of the Animation (cf. Section 7)** describes, how animation is set up and used.
- **Description of all Demo Examples (cf. Section 8)** describes the available demo models.
- **Upgrade (cf. Section 9)** models that have been developed with a former version of the PowerTrain library, e.g., with version 0.98.
- **Literature (cf. Section 10)** contains a list of useful books and literature.

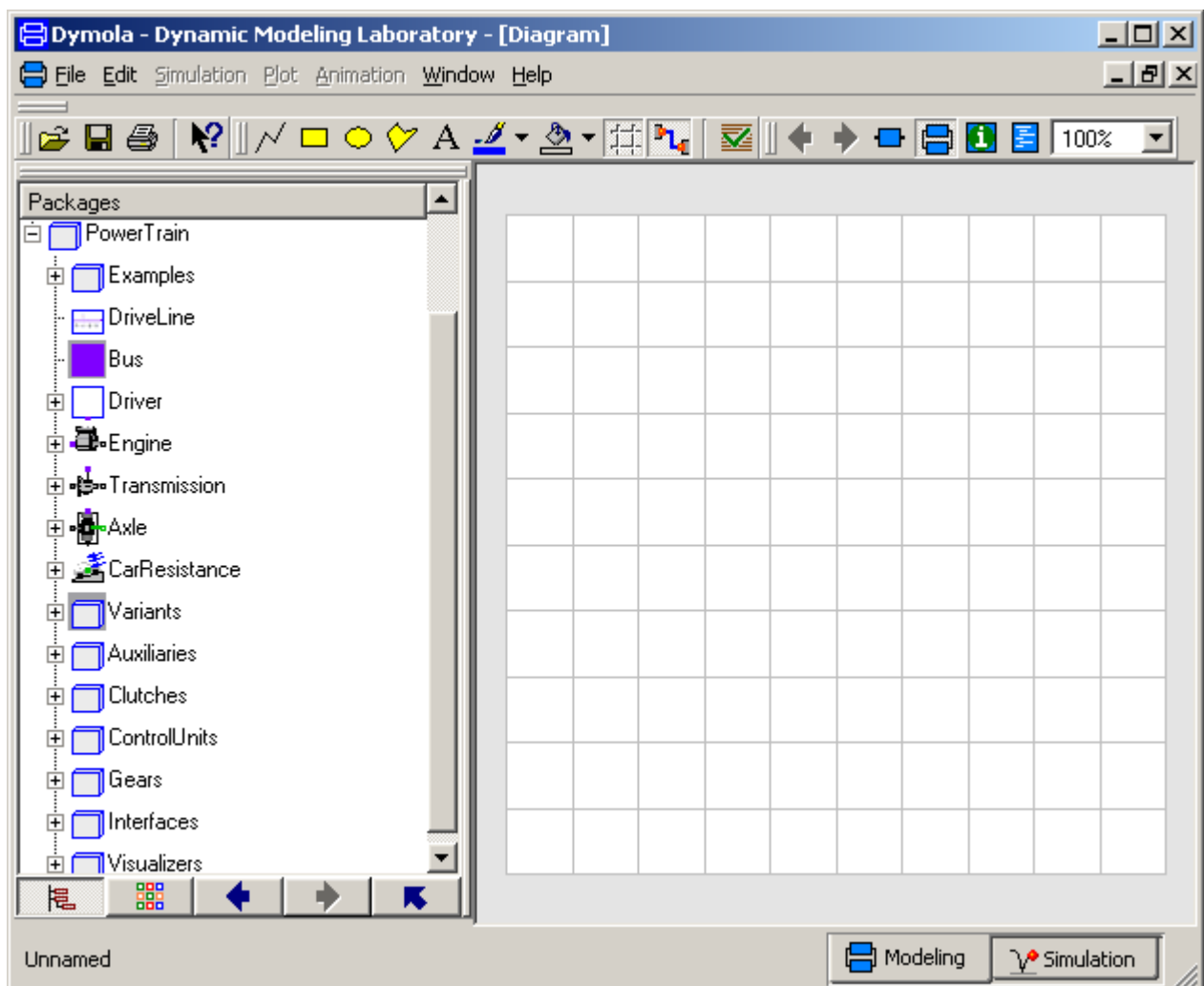
1.3 Acknowledgement

The development of this library was in parts supported by "Bayerisches Staatsministerium für Wirtschaft, Verkehr und Technologie" under contract 300-3245.2-3/01 for the project "Test und Optimierung elektronischer Fahrzeug-Steuergeräte mit Hardware-in-the-Loop-Simulation".

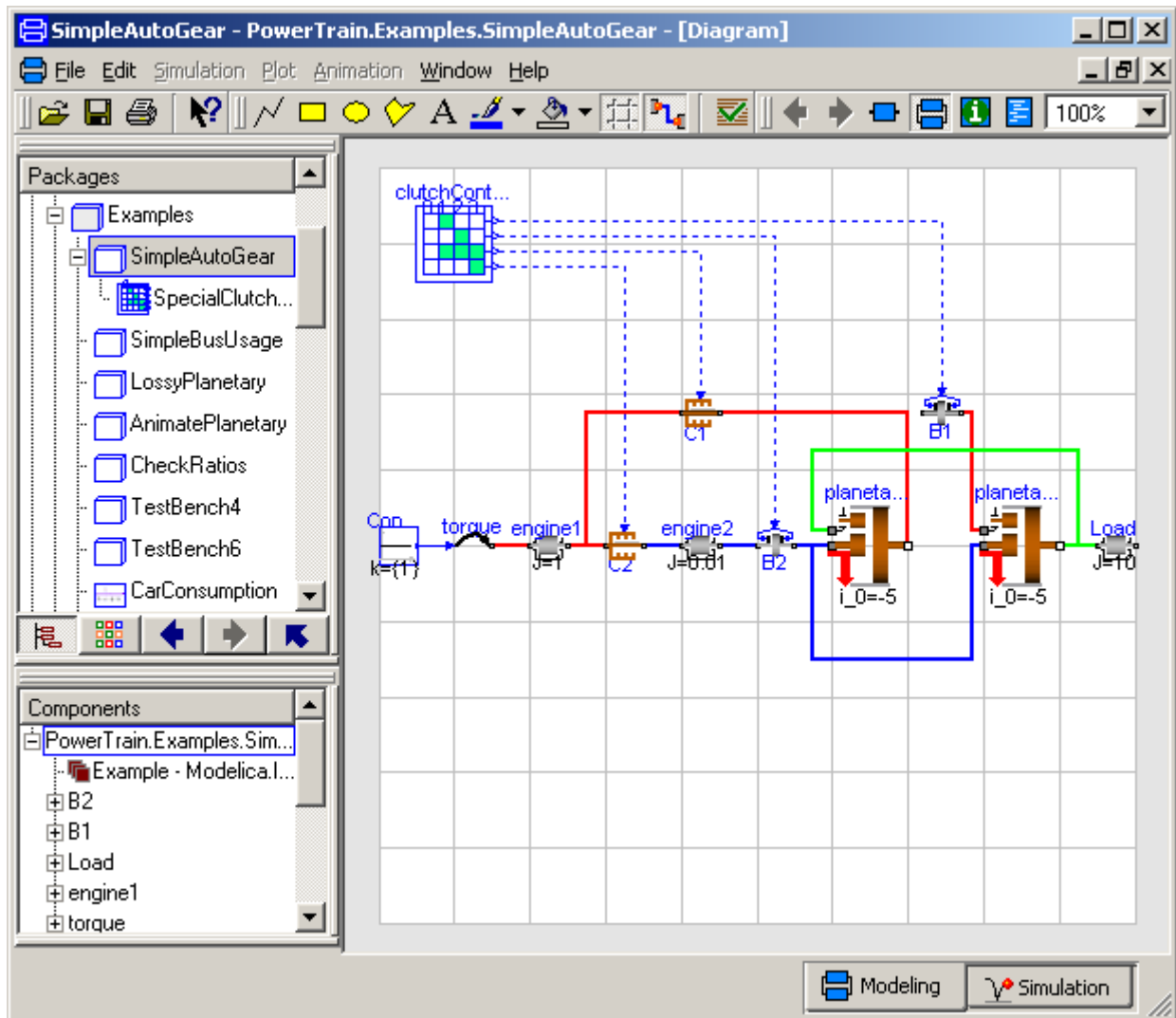
Chapter 2

Running a Demo Model

Open the PowerTrain Library from menus *File, Libraries, Power trains* and you see:



Double click on **Examples** and then on **SimpleAutoGear** and you get:



This example shows a very simple, basic structure of an automatic gearbox. The components of this model are from the following packages:

From package **PowerTrain**:

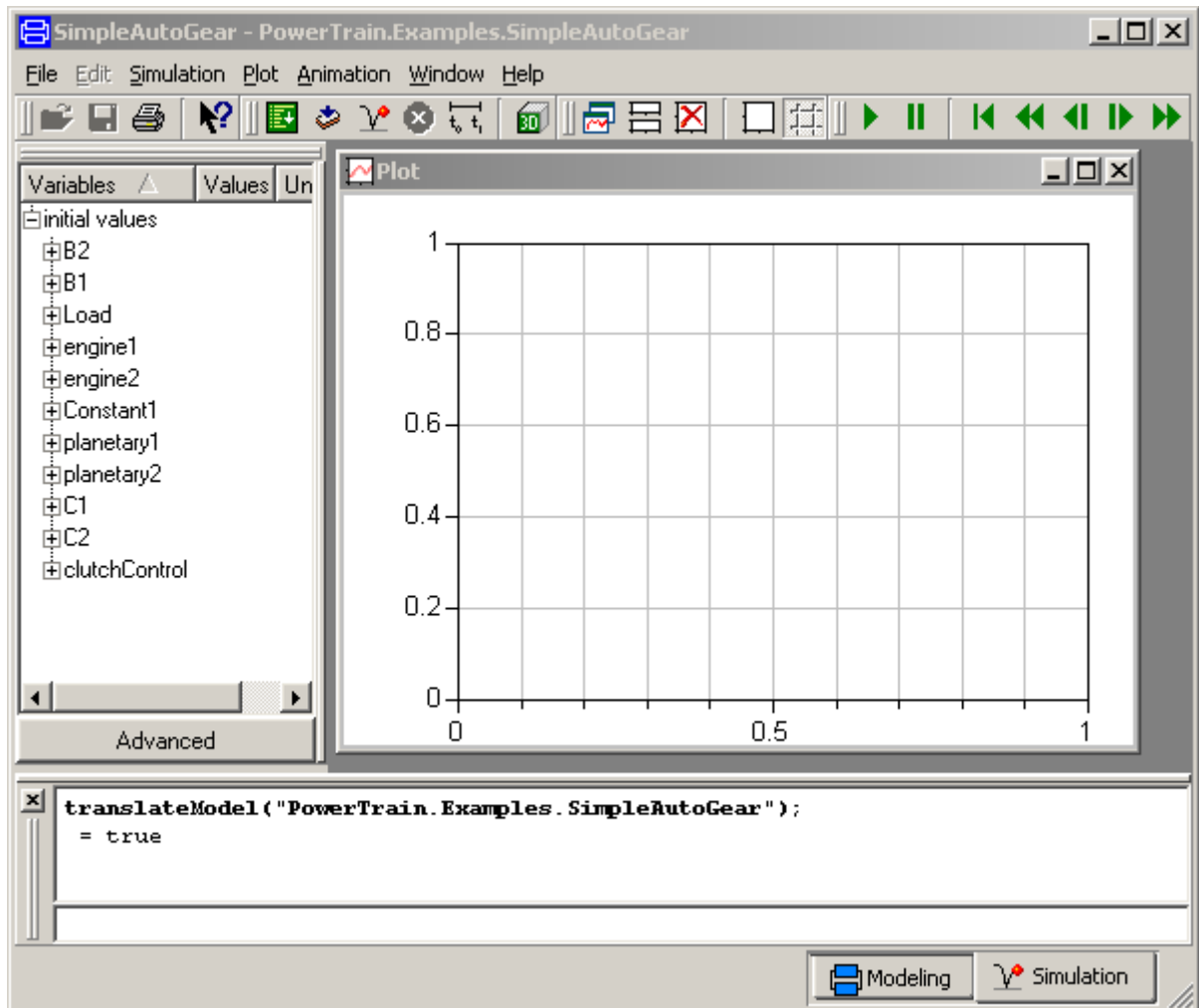
- Gears “planetary1” and “planetary2” are instances of Gears.LossyPlanetary and model planetary gears with mesh and bearing friction.
- The laminar clutches “C1” and “C2” are instances of Clutches.LaminarClutch.
- The control unit “clutchControl” is a very simple model to control the clutches and brakes of this automatic gearbox. It is an instance of Examples.SimpleAutoGear.SpecialClutchControl.

From package **Modelica**:

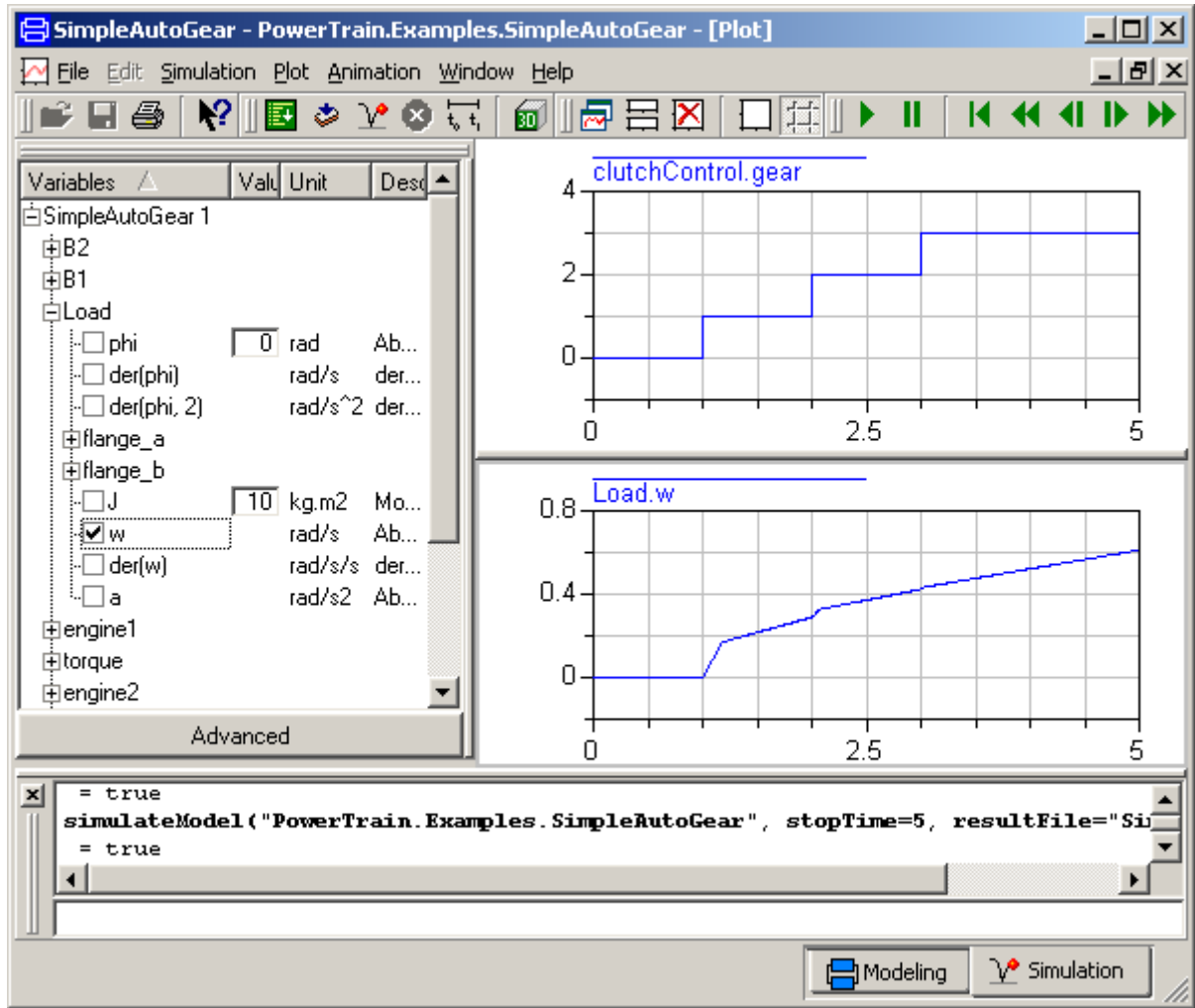
- The rotational inertias “engine1”, “engine2” and “load” are instances of Mechanics.Rotational.Inertia.
- The frictional brakes “B1” and “B2” are instances of Mechanics.Rotational.Brake
- The external torque “torque” transforms a signal to a driving torque. It is an instance from Mechanics.Rotational.Torque

- “Constant1” from Blocks.Sources models the engine in an extremely simplified way by providing a constant torque signal..

The system switches after every second to the next gear. Change now to the *Simulation* menu, set *Stop time* in *Setup ...* to 5 s and simulate the system. The model will be translated automatically, the simulation run will be performed and you get the window setup for generating plots:



Select now results you want to plot from the left part of the window. In the example, we choose the selected *gear* from component *clutchControl* and the speed *w* of the *Load*. The result is the following plot:



Chapter 3

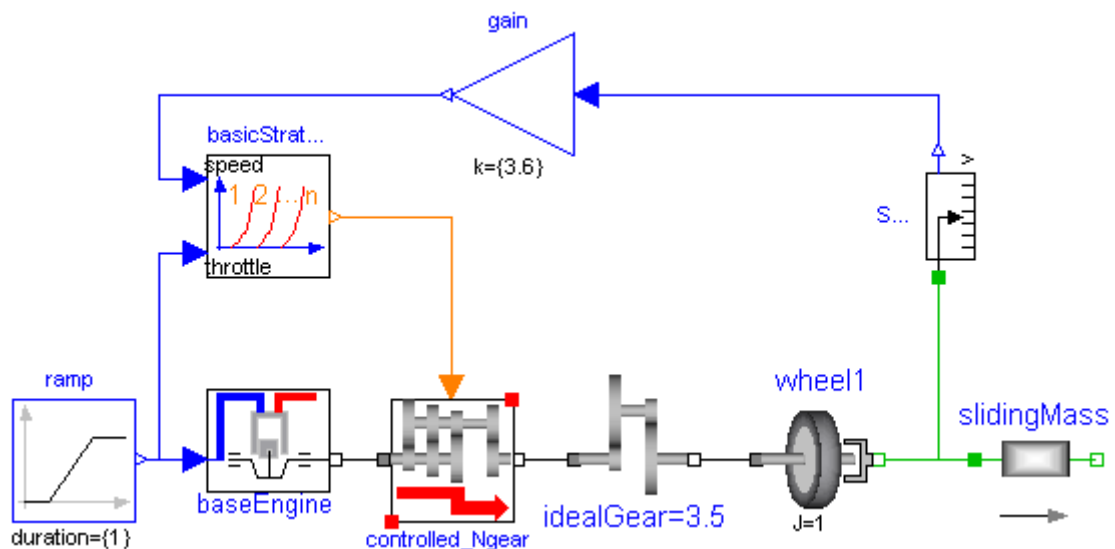
A First Example

You can either go through this example from top to bottom or choose from:

- explain the first example (cf. Section 3.1)
- setting up the example (cf. Section 3.2)
- simulate and see the results of the example (cf. Section 3.3)

3.1 Description of the first example

With the **first example** some basic elements are explained and it will be shown how these can be combined with elements from other packages.



- The diagram shows an engine with a gear moving a sliding mass, which could be a very simple model of a car.
- The simple “driver” is a ramp which applies the gas (throttle position) to the engine.
- The gearbox (controlled_Ngear) defines the gear ratio, depending on the selected gear.

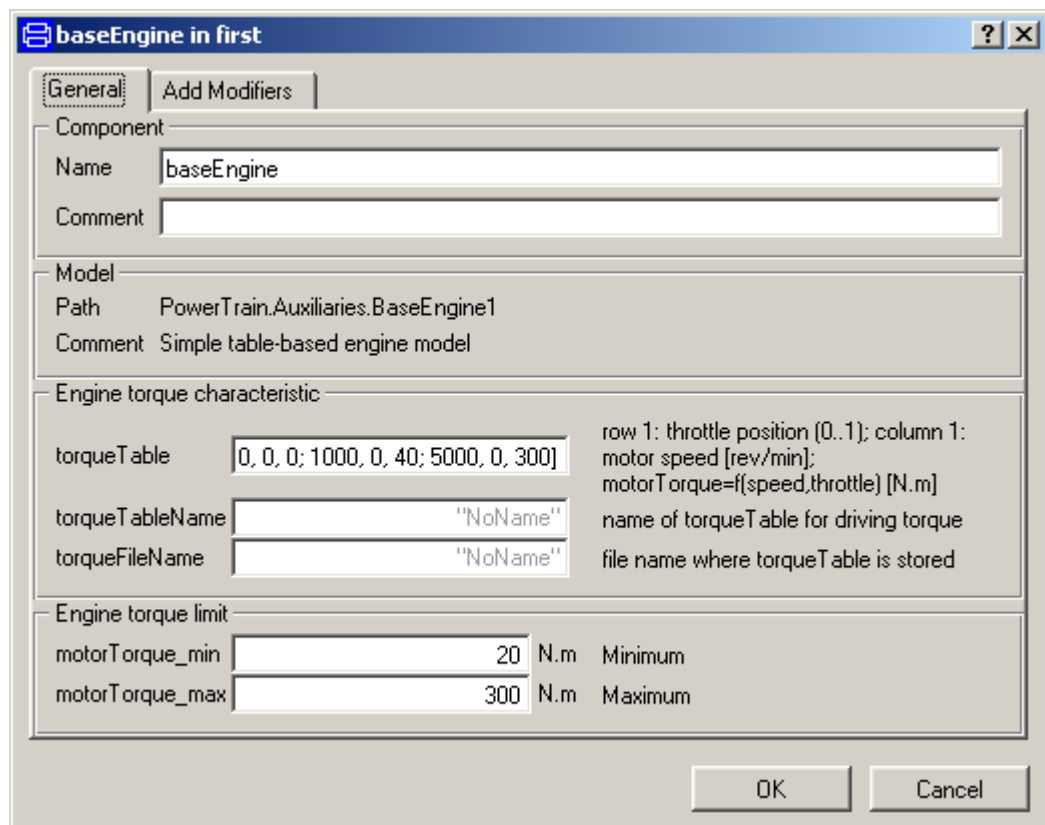
- The gear is chosen by the component “basicStrategy” based on car speed (in km/h, see Gain) and actual throttle position.
- The “idealGear” models the differential gear.
- The “wheel1” changes the rotational movement (output from the differential gear) to the translational movement of the moving mass (car).
- The speed sensor measures the speed (in m/s).

3.2 Setting up the first example

Generate the first model, described above, by combining the following elements and use the parameters as described below.

PowerTrain Package:

BaseEngine (sublibrary Auxiliaries) models a simple engine with respect to torque generation and fuel consumption based on tables. Parameter setting in model first.mo is:

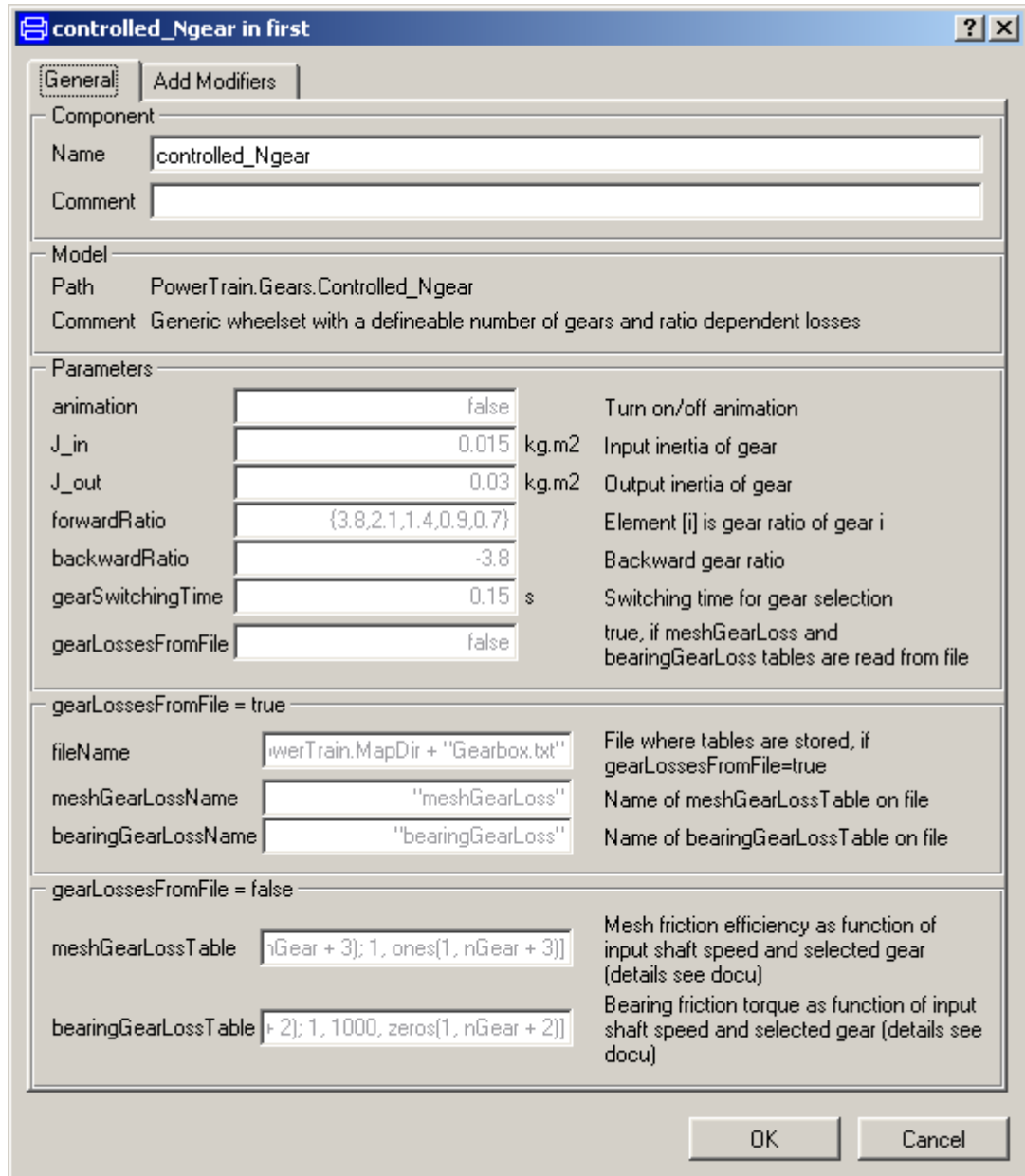


The parameters for motorTorque_min and motorTorque_max are changed. The torqueTable is

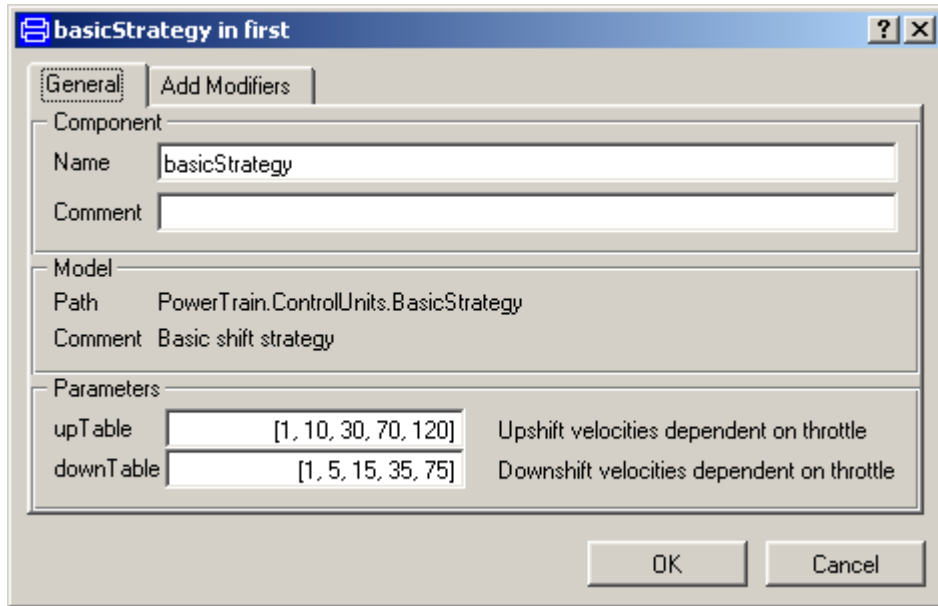
| | | |
|------|---|-----|
| 0 | 0 | 1 |
| 0 | 0 | 0 |
| 1000 | 0 | 40 |
| 5000 | 0 | 300 |

In Modelica syntax this matrix is defined as “[0,0,1; 0,0,0; 1000,0,40; 5000,0,300]”. This means, if throttle is 1 (full gas), the output torque of the engine is 40Nm at 1000 rev/min, 300 Nm at 5000 rev/min and interpolated linearly between these values.

Controlled_Ngear (sublibrary Gears) models a generic wheelset with a defineable number of gears and ratio dependent losses. All parameters are set to their defaults.

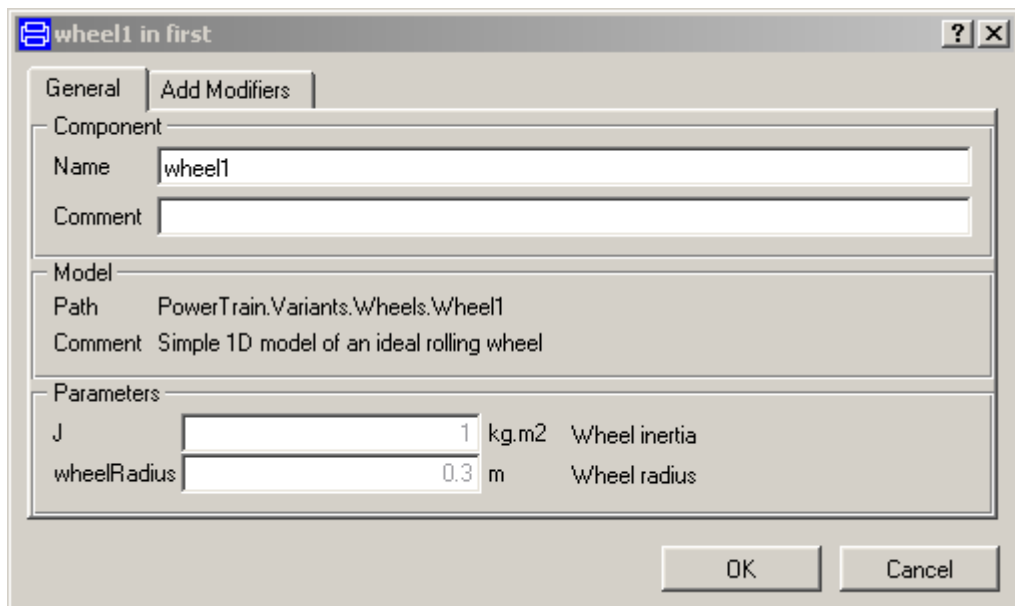


BasicStrategy (sublibrary ControlUnits) models a basic shift strategy. Input are speed of the vehicle and throttle position. Output is the selected gear as Integer value. Two tables define the upshift and downshift velocities, dependent on the throttle position. Throttle position is between 0 (no gas) and 1 (full gas). The values in the tables shall be given in km/h. Therefore the input speed signal has to be also in km/h which requires to multiply the speed of the speed sensor (which is given in m/s) with a factor of 3.6 using the Gain block (from the Modelica.Blocks Library). The values for the upTable are [1,10,30,70,120] km/h. The values for the downTable are [1,5,15,35,75] km/h. As we choose 5 values (switching velocities in km/h) for each of the tables, we defined a 5-gear strategy.



This model defines a gearbox with five gears. The gear ratios are defined in model “controlled_Ngear” by using the default values.

Wheel1 (sublibrary Variants.Wheels) is a simple 1D model of an ideal rolling wheel. It transfers rotational speed into translational speed for the mass (car). Use the default parameters:

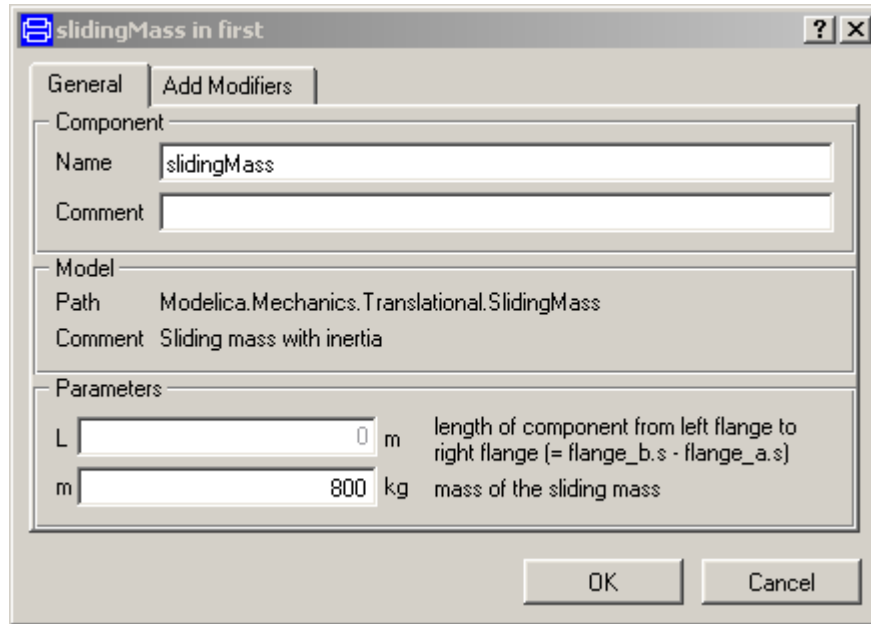


Modelica.Mechanics.Rotational Package:

IdealGear models a gearbox with a fixed gear ratio (without losses and elasticity). This model is used to define the ratio of a differential gear (ratio = 3.5).

Modelica.Mechanics.Translational Package:

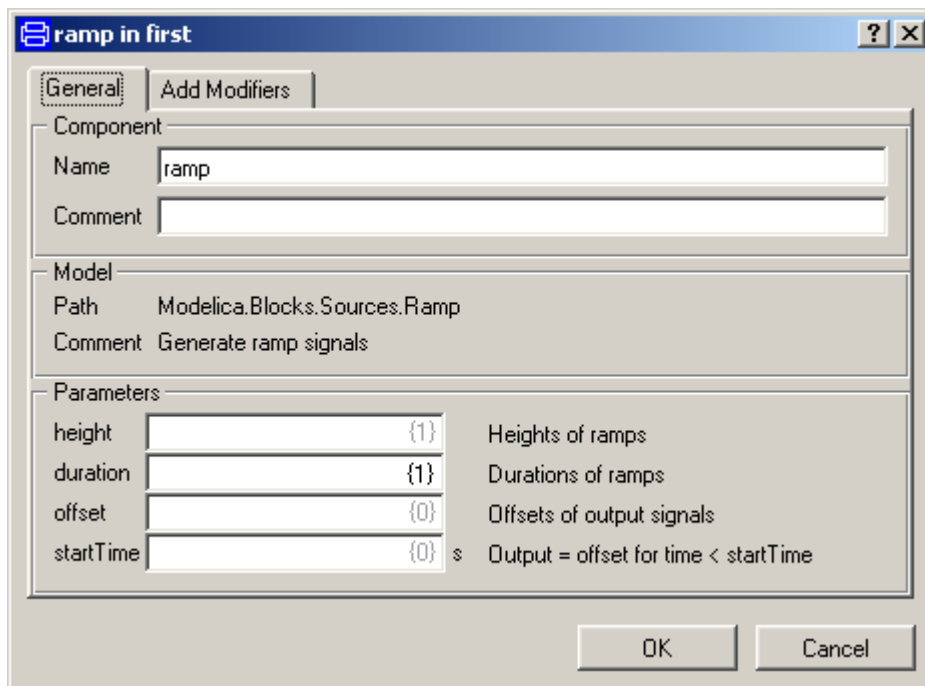
SlidingMass models a translational moving mass without resistances (mass = 800 kg).



Sensors.SpeedSensor measures the absolute velocity. This block has no parameters.

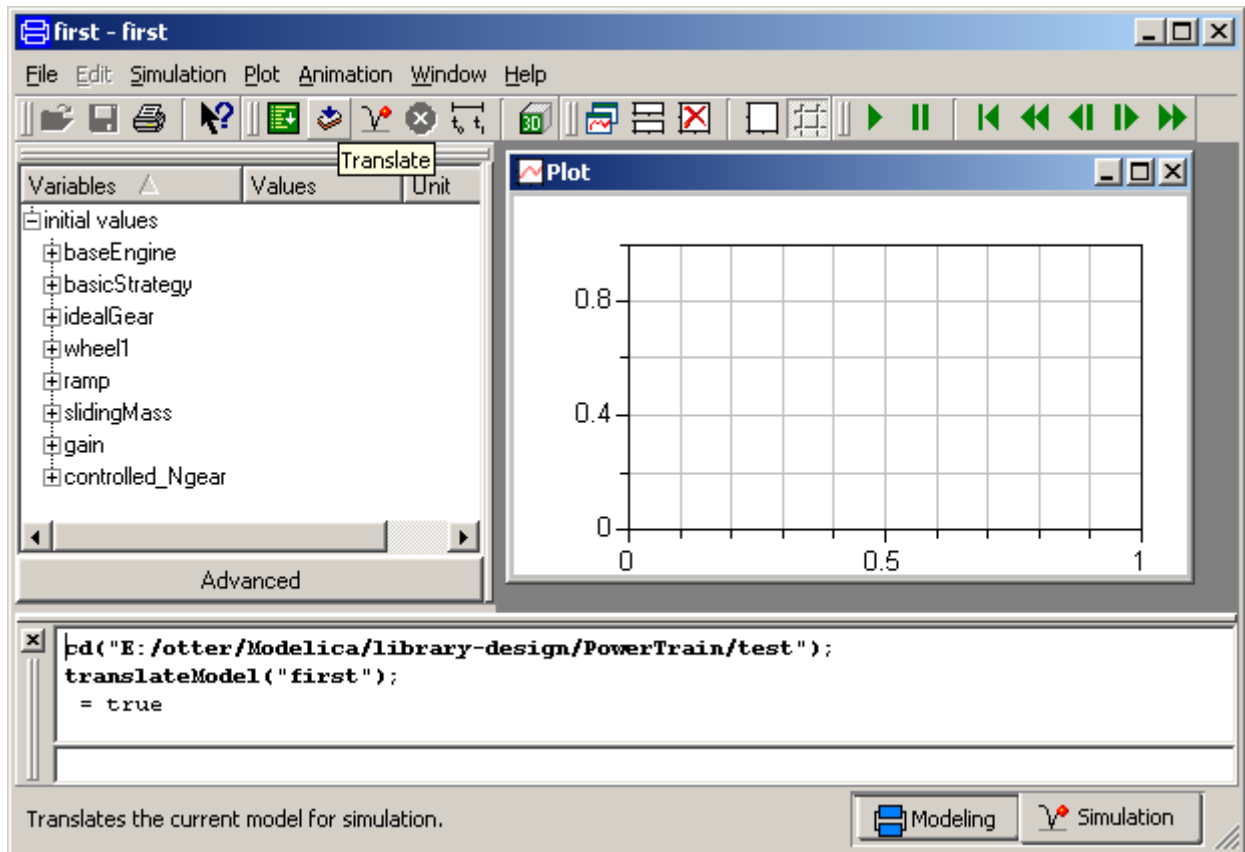
Modelica.Blocks.Sources Package:

Ramp is used to model a simple “driver”. It defines the throttle position between 0 (no gas) and 1 (full gas). It takes 1 s to open the throttle completely.

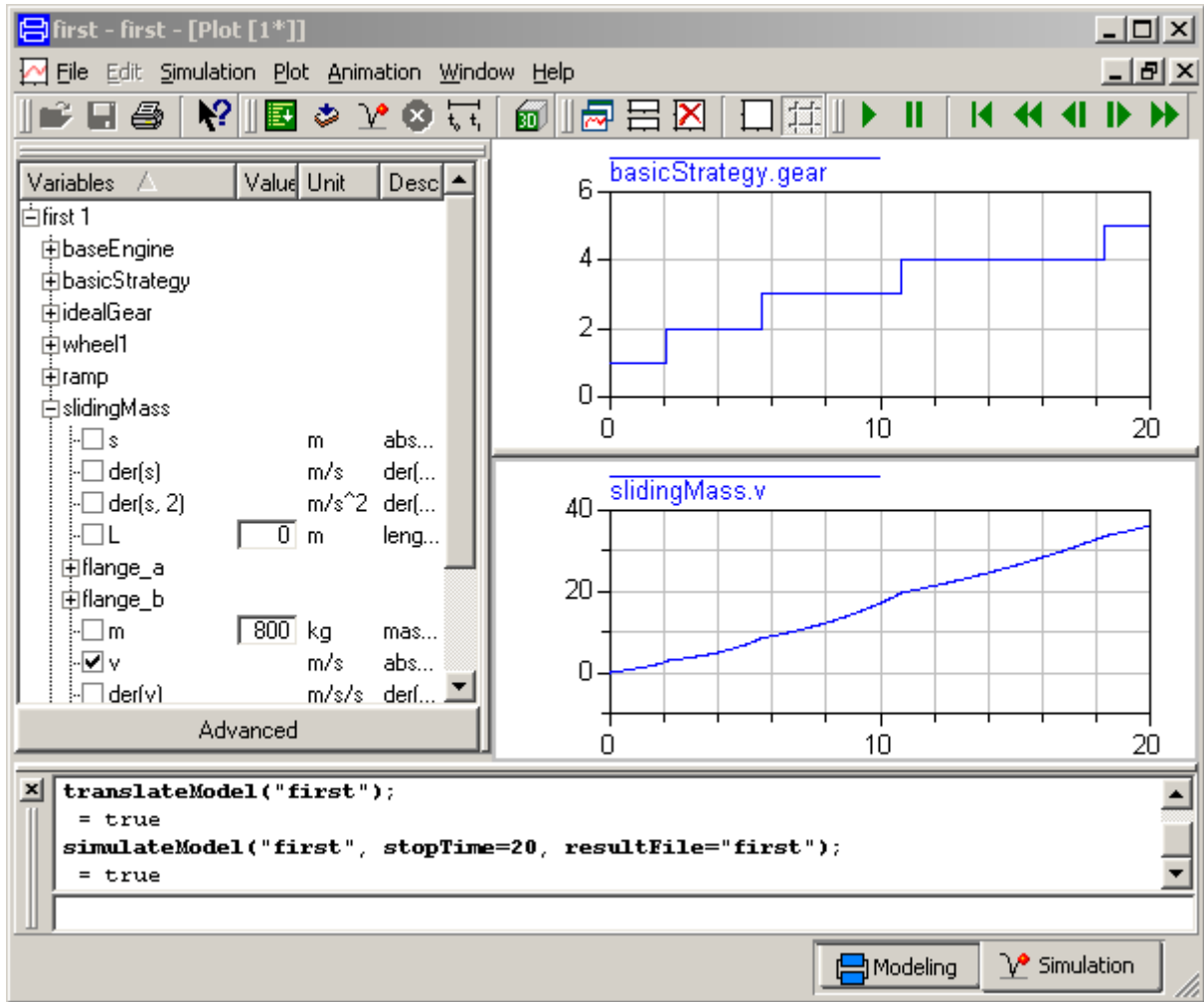


3.3 Simulation of the model

After you combined all components and selected the suggested parameters, translate your model.



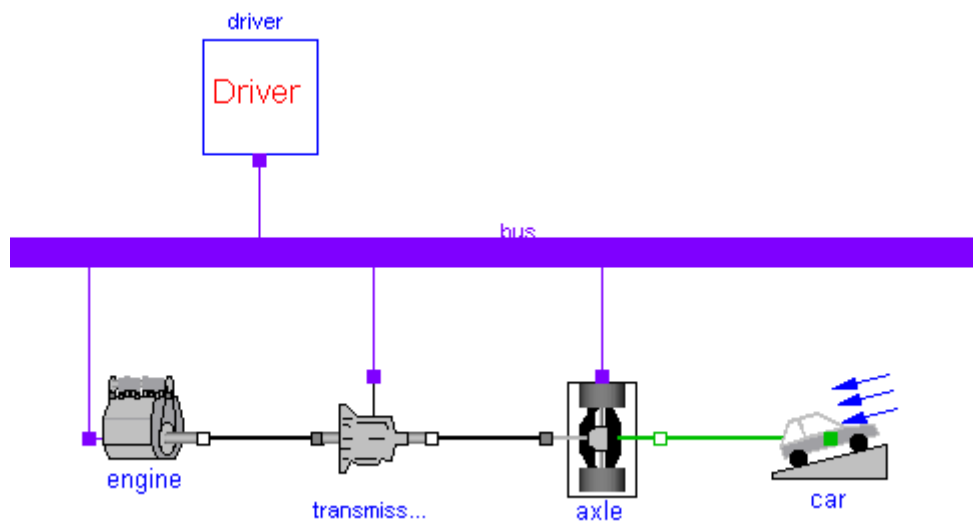
If the translation was successful, choose a simulation time of 20s. The following plot shows the gear shifting from 1 to 5 and the corresponding speed of the mass.



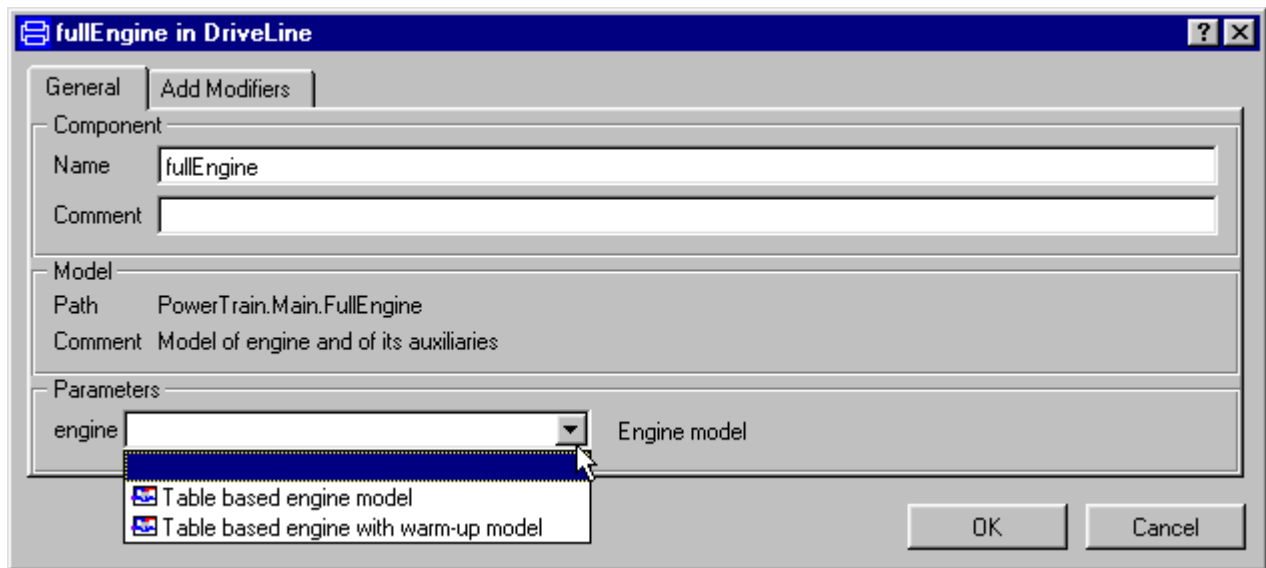
Chapter 4

DriveLine and Main Components

On the first hierarchical level of the PowerTrain library the main sublibraries are present, as well as the **main components** for the first setup of a general powertrain: Bus, Driver, Engine, Transmission, Axle and CarResistance. For all these components different variants exist (e.g., a specific 4- and 6-gear automatic transmission). Additionally, model **DriveLine** is a complete drive line with these components. All available variants, including user supplied variants, can be selected in the DriveLine model. To understand the concept, open model “DriveLine” from the top level of the PowerTrain Library.



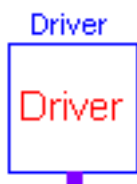
If you open, e.g., component Engine with a double click, you see the menu



and you can choose one of the suggested engine models. The components gearbox, axle, car and driver behave similarly. Below, the currently provided variants are shortly described.

4.1 Choice of drivers (package Variants.Drivers)

Currently, there is only one choice of a prepared driver model



PowerTrain.Driver

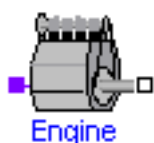
Models of drivers and drive cycles from PowerTrain.Variants.Drivers and from user definitions.



Variants.Drivers.Driver1

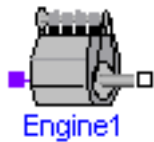
A basic driver model for driving cycle simulation with an automatic transmission. The model is based around a PI controller that is used to determine the desired throttle and brake pedal positions. The driver will only control one pedal at once and the pedal being controlled is decided by a logic system based on the desired speed and acceleration. The desired vehicle speeds are input by a CombiTable and are usually defined on a file (see file PowerTrain/maps/DrivingCycles.txt). The driving cycle that the driver follows is provided as a speed-time profile including control lever position

4.2 Choice of engines (package Variants.Engines)



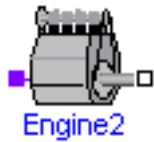
PowerTrain.Engine

Models of engines with controllers and auxiliaries from PowerTrain.Variants.Engines and from user definitions.



Variants.Engines.Engine1

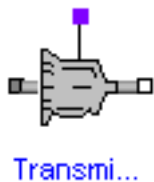
Engine based on a table with idle and speed limit controller. Input to the engine model is the desired throttle position. Additionally, a starter motor is present to start the engine.



Variants.Engines.Engine2

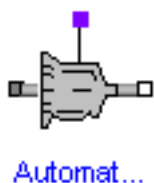
Simple engine model based on tables which includes an engine warm-up model. Inputs to the engine model are desired throttle position and fuel flow. The model determines the torque produced by the engine from a table of throttle position and engine speed (both signals from connector bus) taking into account frictional losses which depend on oil temperature. The warm-up model determines the heat rejected into the engine based on the fuel energy input and useful work output. Additionally, a starter motor is present to start the engine.

4.3 Choice of transmissions (package Variants.Transmissions)



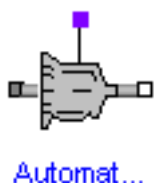
PowerTrain.Transmission

Transmission models from PowerTrain.Variants.Transmissions and from user definitions.



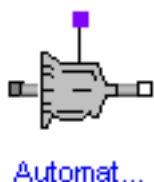
Variants.Transmissions.AutomaticNgear

Generic automatic wheelset with a defineable number of gears, ratio dependent losses and simple model of switching behaviour based on a first order filter.



Variants.Transmissions.Automatic4Gear

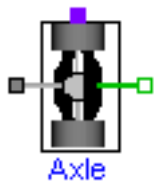
Detailed model of a controlled automatic wheelset with 4 forward gears of extended Simpson type. The automatic gear is modelled with laminar clutches, brakes and free-wheels.



Variants.Transmissions.Automatic6Gear

Detailed model of a controlled automatic wheelset with 6 forward gears of Lepelletier type. The automatic gear is modelled with laminar clutches, brakes and freewheels.

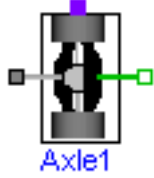
4.4 Choice of axles (package Variants.Axles)



Axle

PowerTrain.Axle

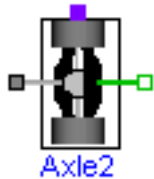
Models of front or rear axles with differential gear, wheels and brakes from PowerTrain.Variants.Axles and from user definitions.



Axle1

Variants.Axles.Axle1

Axle model including differential gear losses and a simple 1D model of an ideal rolling wheel.



Axle2

Variants.Axles.Axle2

Axle model including differential gear losses and a simple 1D model of a wheel with speed dependent wheel radius.

4.5 Choice of car resistances (package Variants.CarResistances)



CarResi...

PowerTrain.CarResistance

Models of car resistances from PowerTrain.Variants.CarResistances and from user definitions.



CarResi...

Variants.CarResistances.CarResistance1

Model of car resistances due to acceleration, slope of the road, wind and rolling resistance. Some of the data are given via combi-tables, i.e., the data may be given directly or read from file.



CarResi...

Variants.CarResistances.CarResistance2

Model of car resistances defined as a 2nd order polynomial of the car speed.

Chapter 5

Basic Components

Below the basic components provided in the PowerTrain library are shortly described. They are used to construct the main components discussed in “DriveLine and Main Components”. They are also useful for a user when constructing higher level components.

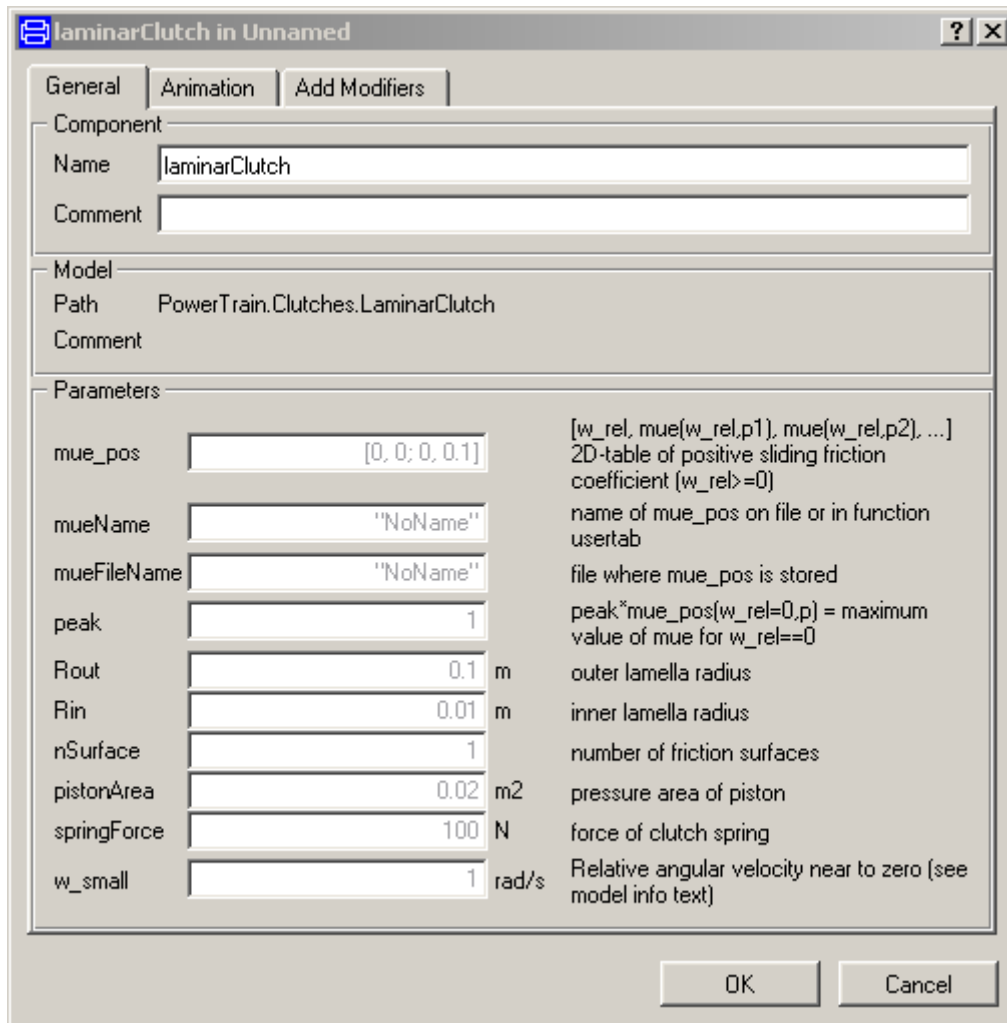
5.1 Package Clutches

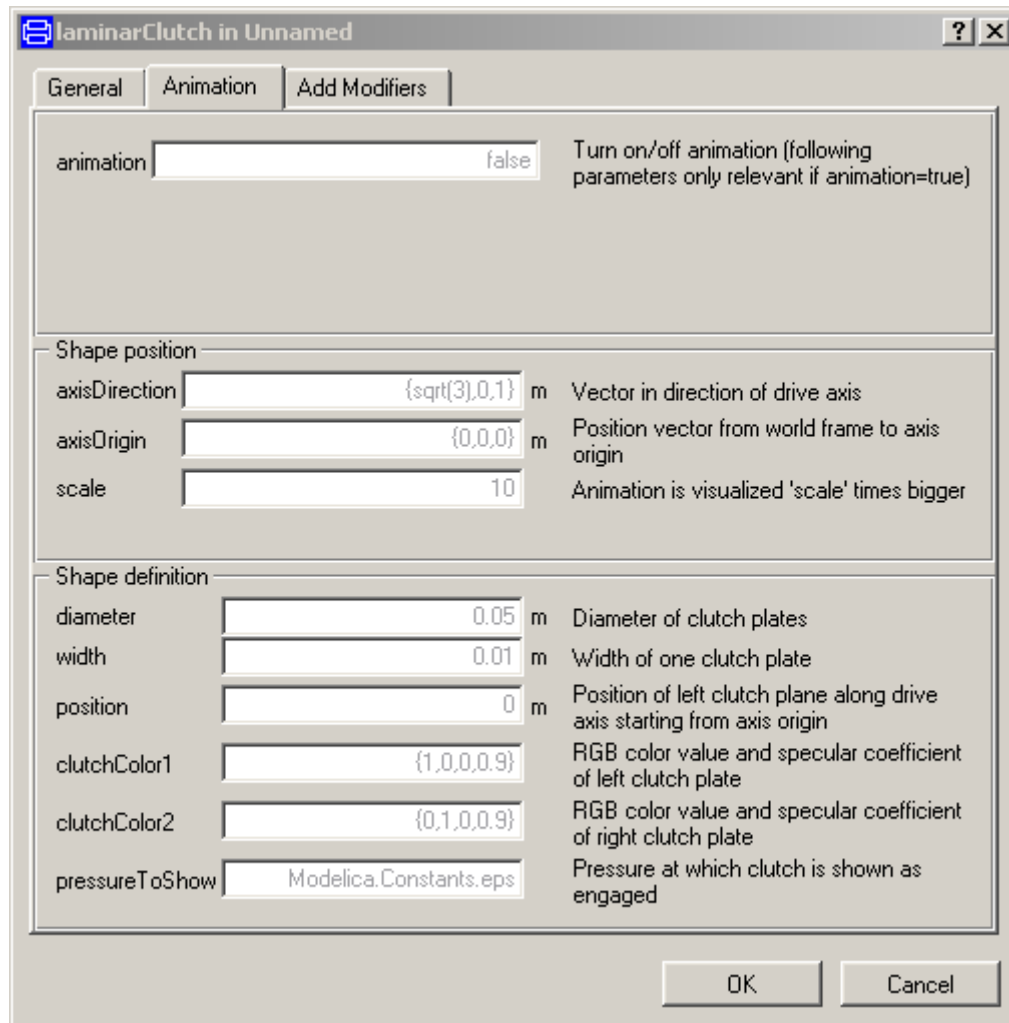
This package contains laminar clutches, free wheels and one-way-laminar clutches (= parallel connection of laminar clutch and free wheel). These components are based on the Coulomb friction model and have stuck and sliding phases. The implementation of the friction models are performed in a robust and numerically reliable way. Problems usually only occur, if two friction elements may lock (directly or indirectly) the same relative motion, since then no unique mathematical solution exists. Dymola is still able to handle this situation in most cases by continuing with a least squares solution (= pick one solution of the infinitely many that is as close as possible to the solution of the previous time instant). The friction phases in all these elements are characterized by the public Integer variable **mode** which can have the following values:

- mode = 2: clutch is **not active** (no frictional torques)
- mode = 1: clutch is **sliding in forward** direction
- mode = 0: clutch is **stuck** (no relative motion between clutch surfaces)
- mode = -1: clutch is **sliding in backward** direction

In package Modelica.Mechanics.Rotational also clutch and brake models are available. The essential difference to PowerTrain.Clutches is that the latter are parameterized by technological parameters directly available from a clutch manufacturer whereas the components in the Rotational library have a generic parameterization.

Components that have a dark orange color in the icon (LaminarClutch, OneWayLaminarClutch) can be optionally used in an animation to visualize when the component is active or not active. The parameter menus of model LaminarClutch are:





Clutches.LaminarClutch

Laminar clutch where the input signal is the pressure. Same as LaminarClutch with the only difference that the plates are defined via a middle radius instead of an inner and an outer radius.



Clutches.LaminarClutch2

Same as LaminarClutch with the only difference that the plates are defined via a middle radius instead of an inner and an outer radius.



Clutches.FreeWheel

Model of an ideal free wheel, i.e., the component does only allow a relative motion in one direction and locks the relative motion, if the sign of the speed would become negative.



Clutches.OneWayLaminarClutch

Same as LaminarClutch, however the speed cannot become negative (= parallel connection of LaminarClutch and FreeWheel. The ambiguity of the friction torques are resolved here in an appropriate way).

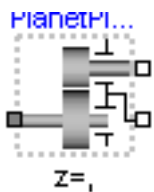


Clutches.OneWayLaminarClutch2

Same as OneWayLaminarClutch, with the only difference that the plates are defined via a middle radius instead of an inner and an outer radius.

5.2 Package Gears

This package contains different kinds of gears, such as planetary gears, Ravigneaux wheelsets, differential gears, CVT gear. Components that have a red arrow in their icon, such as LossyPlanetary, are modelled with mesh and bearing frictional losses (= torque and speed dependent friction with sliding and stuck model). The implementation of the friction models are performed according to a new theory in a robust and numerically reliable way. Similarly as for package Clutches above, the friction phases in all these elements are characterized by the public Integer variable **mode** (details see above). Components that have a dark orange color in the icon such as LossyPlanetary can be optionally used in a 3-dimensional animation to visualize the movement of the gear wheels.



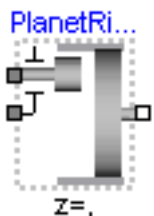
Gears.PlanetPlanet

is used to model together with “PlanetRing” any type of planetary gearbox. The usage can, e.g., be inspected in model “IdealRavigneaux”. Note, that no inertia is included in this model. By attaching inertias to the sun, carrier and ring flanges, all inertias of a planetary gear can be taken into account. Only the inertia of the planets with respect to their axes are always neglected. If this should be not the case, use component “PlanetPlanetInertia”. Also elasticity, damping or backlash are not included in this model.



Gears.PlanetPlanetInertia

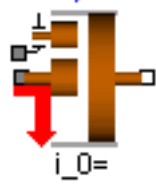
is identical to PlanetPlanet with the only exception that the inertia of the planets with respect to their axes are included in the model.



Gears.PlanetRing

is used to model together with “PlanetPlanet” any type of planetary gearbox (see, e.g., the implementation of IdealRavigneaux).

LossyPla...



Gears.LossyPlanetary

models any type of planetary gear that has three external flanges and is described by Willis' equation: $(w_A - w_B) = i_0 \cdot (w_C - w_B)$ where “ w_X ” is the speed of flange_X and “ i_0 ” is the so-called stationary gear ratio. For standard planetary gears “ $i_0 = z_r/z_s$ ”, where “ z_s ” is the number of teeth for the inner sun wheel and “ z_r ” is the number of teeth for the outer ring wheel (teeth numbers are taken negative for internal teeth). The model includes mesh and bearing friction effects and can be animated.

IdealRa...



Gears.IdealRavigneaux

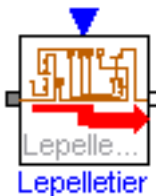
Ravigneaux wheel set without losses and with animation

LossyR...



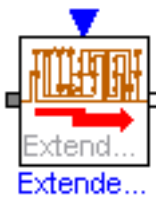
Gears.LossyRavigneaux

Ravigneaux wheel set with losses and with animation



Gears.Lepelletier

Detailed model of lepelletier wheelset for 6 speed automatic gearbox, modeled with inertias, clutches, brakes and free wheels. Component can be animated.



Gears.ExtendedSimpson

Detailed model of extended Simpson wheelset for 4 speed automatic gearbox, modeled with inertias, clutches, brakes and free wheels. Component can be animated.

Differen...



Gears.Differential

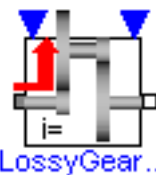
ideal differential gear without losses.

LossyDi...



Gears.LossyDifferential

differential gear with losses

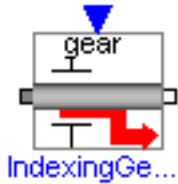


Gears.LossyGearFed

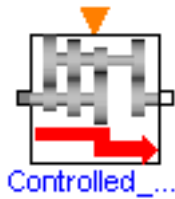
standard gear where the mesh and bearing friction losses can be defined by external signals. One has to be very careful when using this element, because “non-physical” signal connections can lead easily to nasty non-linear systems of equations. Be careful that the input signals do not depend non-linearly on torques of other components. As an example see the implementation of component “IndexingGearLosses” where this model is used in a suitable way.



Gears.IdealCVT
ideal CVT gear (= Continuous Varying Transmission) without losses



Gears.IndexingGearLosses
speed and driving torque dependent losses in an indexing gear as function of the selected gear



Gears.Controlled_Ngear
generic automatic wheelset with a defineable number of gears, ratio dependent losses and simple model of switching behaviour based on a first order filter. The input to this model is the selected gear.

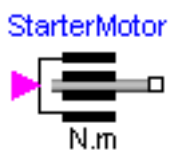


Gears.Controlled_4gear_ExtendedSimpson
detailed model of a controlled automatic wheelset with 4 forward gears of extended Simpson type. The automatic gear is modelled with laminar clutches, brakes and free-wheels and controlled by a simple clutch control system. The input to this model is the selected gear.

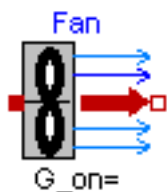


Gears.Controlled_6gear_Lepelletier
detailed model of a controlled automatic wheelset with 6 forward gears of Lepelletier type. The automatic gear is modelled with laminar clutches, brakes and freewheels and controlled by a simple clutch control system. The input to this model is the selected gear.

5.3 Package Auxiliaries



Auxiliaries.StarterMotor
is a simple model of a starter motor. When the boolean input signal ignition is true, the component outputs a fixed torque value. It has no inertia



Auxiliaries.Fan
is a simple fan model. Input is a threshold value for temperature. When the boolean input signal ignition is true, the component outputs a fixed torque value. It has no inertia

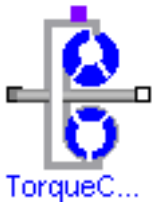


Auxiliaries.WarmUpModel
is a simple warm-up model that determines coolant and oil temperature based on a lumped mass system. An energy balance between 4 masses represents the engine block, coolant in the block, coolant in the radiator and oil system



Auxiliaries.EngineLosses

models frictional losses of engine and auxiliaries as a function of the engine speed and oil temperature



Auxiliaries.TorqueConverter

models a hydrodynamic torque converter consisting of pump, turbine and reactor, where the reactor is controlled by a clutch to allow the torque converter pump and turbine to be directly coupled together. The torque converter characteristic is primarily defined by a combi-table. The signal “lockUpClutchDemand” from the signal bus controls the lock-up clutch position. This element is usually used as coupler between the engine of a vehicle and an automatic gearbox.

5.4 Package ControlUnits



ControlUnits.BasicStrategy

basic shift strategy, depend on throttle position and speed



ControlUnits.ClutchControl

generates clutch pressure for selected gear



ControlUnits.ShiftSchedule

is a gear shift controller for use with an n-speed automatic gearbox



ControlUnits.LockUpControl

Torque Converter Lock-up clutch control for automatic transmissions



ControlUnits.FuelMap

Fuel flow controller with externally determined fuel cut-off. Fuel flow is the output from this model and is determined from tables of engine speed and throttle position



ControlUnits.ORFCO

Over Run Fuel Cut Off (ORFCO) Controller to shut-off the fuel flow into the engine when the engine speed is above a threshold value and the engine is decelerating

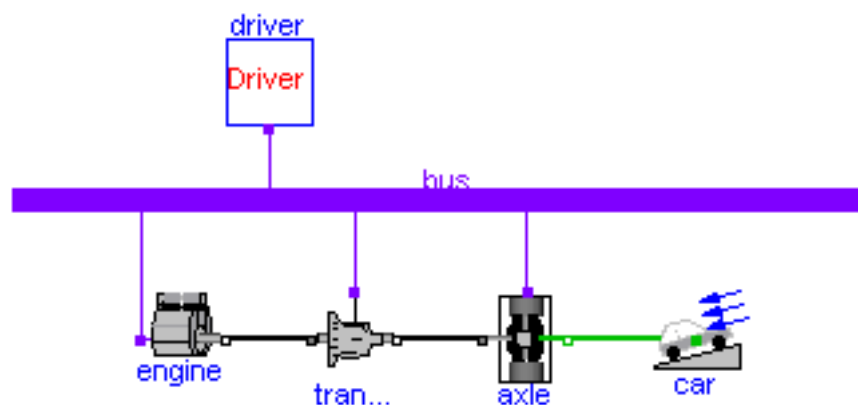
Chapter 6

Concept of the Signal Bus

The signal bus (connector PowerTrain.Bus) is used to exchange signals. It was designed to avoid many connections of input/output signals in the graphical diagram. The result are simulation models, which are easy to understand. Since this is of general interest for Modelica applications, the needed utility blocks have been included in the Modelica Standard Library (version 1.5). In the PowerTrain library the signal bus is defined as **replaceable** connector. This allows a user to exchange the bus definition in the PowerTrain library with his own definition of the bus (= all signals of the original bus + additional, user defined signals).

6.1 First bus example

A good **first example** is the model Driveline from the top level of the Powertrain Library.



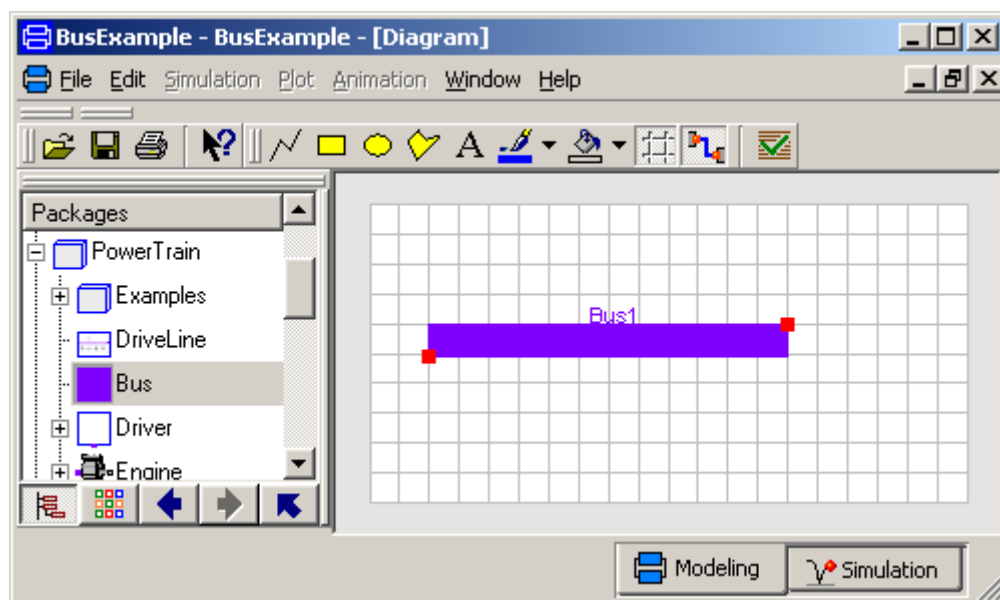
For a description of this model see the description of model DriveLine (cf. Section 4). The pink line in the middle with the description “bus” is the signal bus, which is an instance of connector PowerTrain.Bus with the instance name “bus”. The bus is usually drawn in such a way as shown in the above figure in order to provide the standard graphical representation of a bus.

6.2 Description of the bus

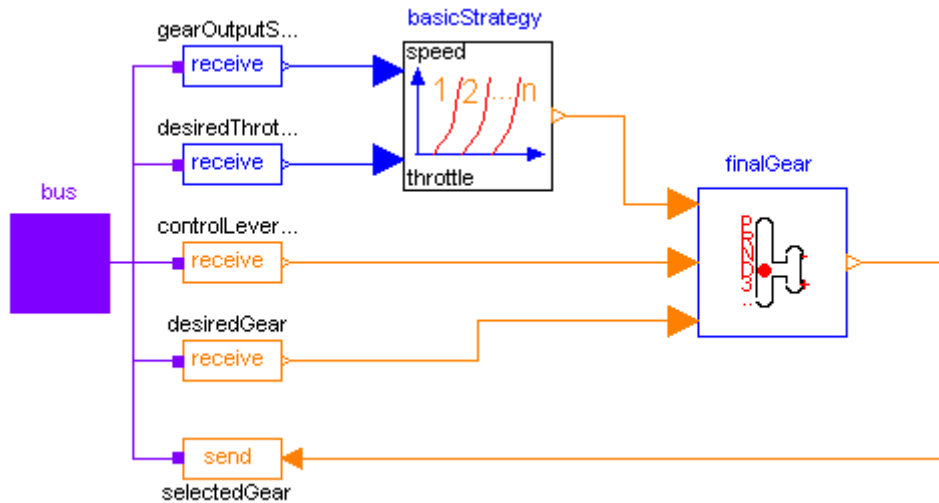
The component bus defines the connector for the signal bus. All signals, which are transferred by the bus to a component are defined in the bus. Currently, the bus contains the following signals:

| Name | Description | | | | | | | | | | | | | | |
|----------------------|--|---|---------|----|-----------------|----|------------------|---|-------------------------|---|-----------------|---|-------------------------------|---|------------------------|
| desiredThrottle | desired throttle position between 0 (closed) and 1 (fully open) | | | | | | | | | | | | | | |
| brake | brake position between 0 (no action) and 1 (maximum position) | | | | | | | | | | | | | | |
| controlLeverPosition | position of control lever <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>=</th> <th>meaning</th> </tr> </thead> <tbody> <tr> <td>-2</td> <td>P park position</td> </tr> <tr> <td>-1</td> <td>R drive backward</td> </tr> <tr> <td>0</td> <td>N neutral (no movement)</td> </tr> <tr> <td>1</td> <td>D drive forward</td> </tr> <tr> <td>2</td> <td>gear limited by "desiredGear"</td> </tr> <tr> <td>3</td> <td>use gear "desiredGear"</td> </tr> </tbody> </table> | = | meaning | -2 | P park position | -1 | R drive backward | 0 | N neutral (no movement) | 1 | D drive forward | 2 | gear limited by "desiredGear" | 3 | use gear "desiredGear" |
| = | meaning | | | | | | | | | | | | | | |
| -2 | P park position | | | | | | | | | | | | | | |
| -1 | R drive backward | | | | | | | | | | | | | | |
| 0 | N neutral (no movement) | | | | | | | | | | | | | | |
| 1 | D drive forward | | | | | | | | | | | | | | |
| 2 | gear limited by "desiredGear" | | | | | | | | | | | | | | |
| 3 | use gear "desiredGear" | | | | | | | | | | | | | | |
| desiredGear | maximum or desired gear if controlLeverPosition = 2 or 3 | | | | | | | | | | | | | | |
| ignition | true , if ignition is on, otherwise false | | | | | | | | | | | | | | |
| throttle | actual throttle position between 0 (closed) and 1 (fully open) | | | | | | | | | | | | | | |
| selectedGear | actual number of selected gear between -2 (park) and maximum gear (same meaning as controlLeverPosition) | | | | | | | | | | | | | | |
| gearOutputSpeed | angular velocity of transmission output shaft in [rad/s] | | | | | | | | | | | | | | |
| lockUpClutchDemand | force signal for lock-up clutch in torque converter between 0 (no action) and 1 (maximum pressure force) | | | | | | | | | | | | | | |
| fuelFlow | fuel flow in [g/s] | | | | | | | | | | | | | | |
| fuelOverRun | true to shut-off fuel flow into the engine; false if no action | | | | | | | | | | | | | | |
| vehicleSpeed | Speed of vehicle in [m/s] | | | | | | | | | | | | | | |
| engineSpeed | Angular velocity of engine crank shaft in [rad/s] | | | | | | | | | | | | | | |
| coolantTemperature | Temperature of cooling liquid in cooler in [K] | | | | | | | | | | | | | | |
| oilTemperature | Temperature of oil in oil pan in [K] | | | | | | | | | | | | | | |

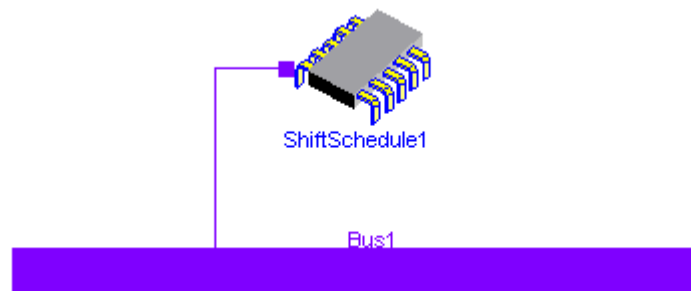
Components that have the pink bus connector can be connected to the signal bus. For usage of the bus, copy the bus in a new model and drag it to a long line.



Almost all control units use the signal bus to exchange signals. If you want to apply, e.g., the component ControlUnits.ShiftSchedule it needs four signals from the bus and returns two signals to the bus:



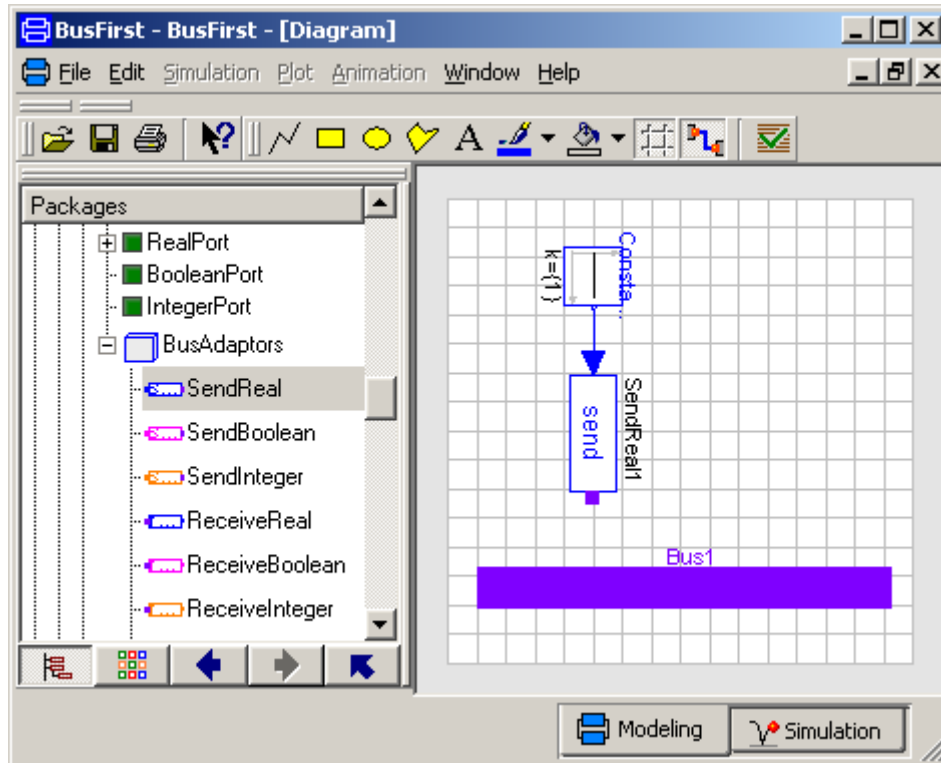
If you want to add this component to your model, you have to connect the pink bus connector to the bus.



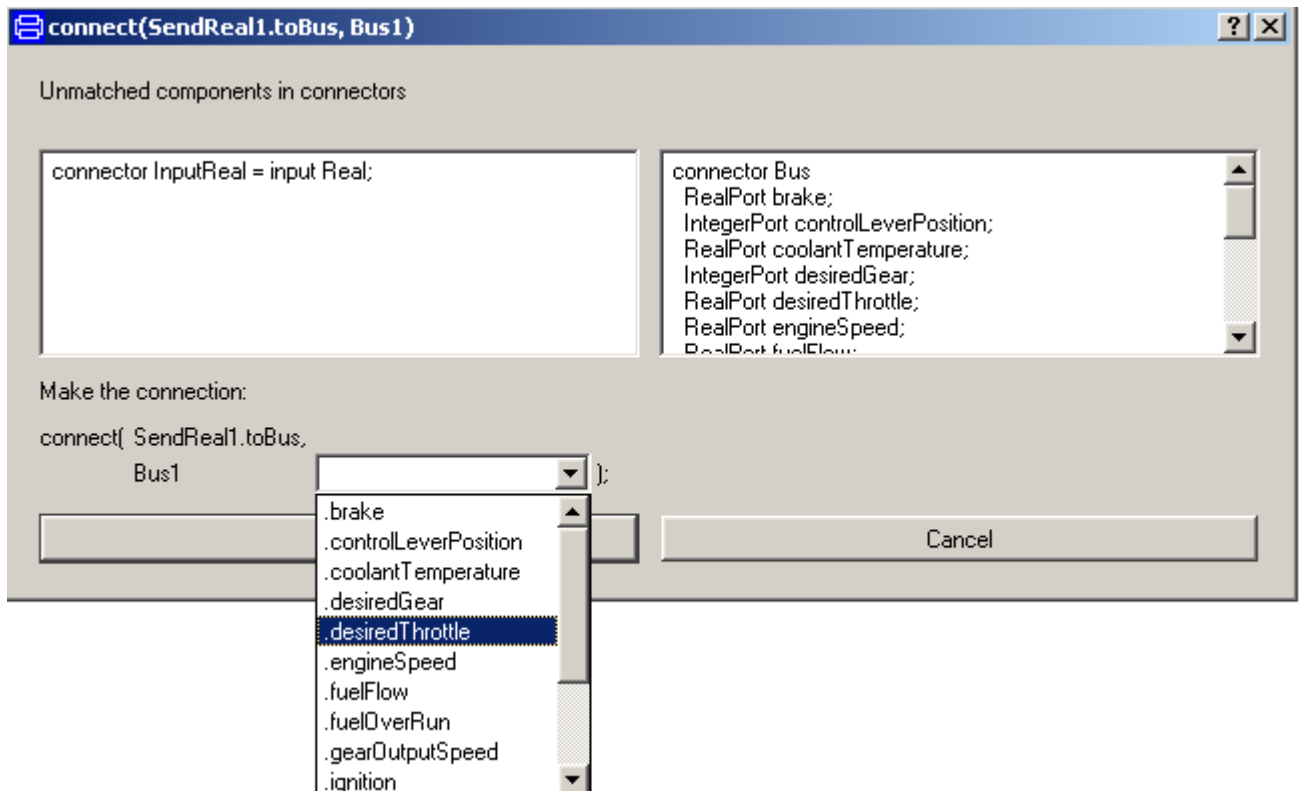
6.3 First steps to connect a signal to the bus

The following description shows, how you can send a real signal to the bus. In Modelica context this means that a signal on the bus is defined by an equation. To receive a signal is a similiar procedure.

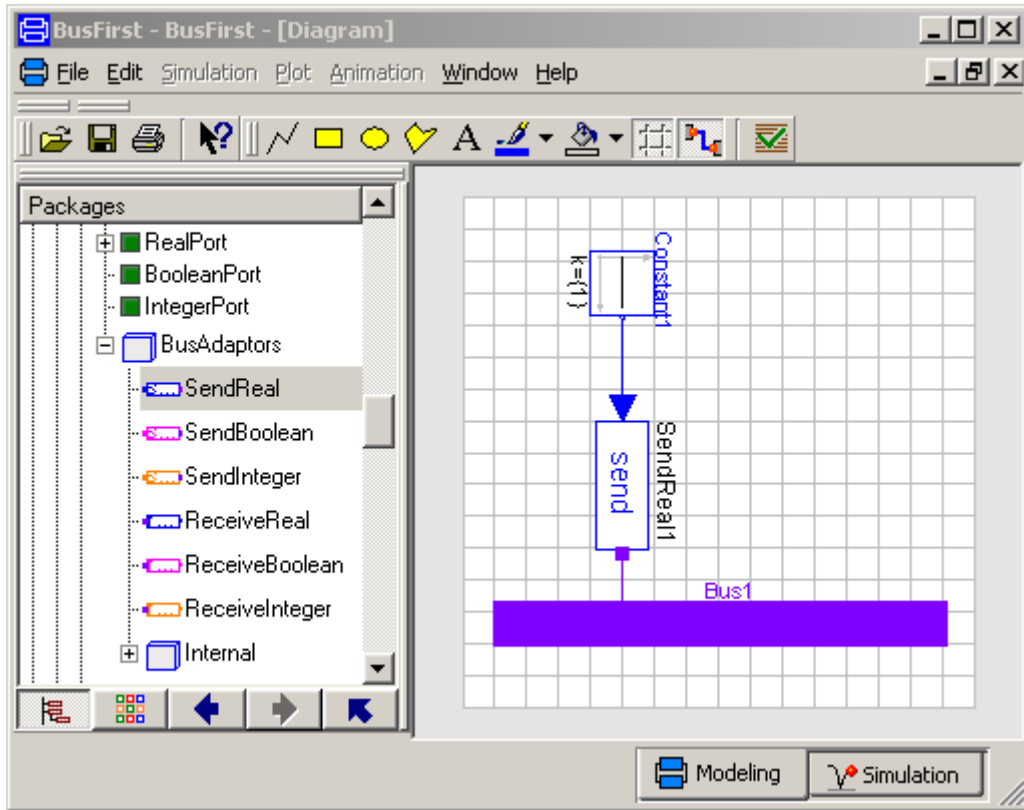
As example you may compute a desired throttle as a Real value. If you want to send this Real signal to the bus, you have to connect it first to the “SendReal” component from package Modelica.Blocks.Interfaces.BusAdaptors .



If you connect the bus connector of “SendReal” with the bus, the following menu appears on the screen:



Choose “desiredThrottle” from the menu and click the OK button and you get:

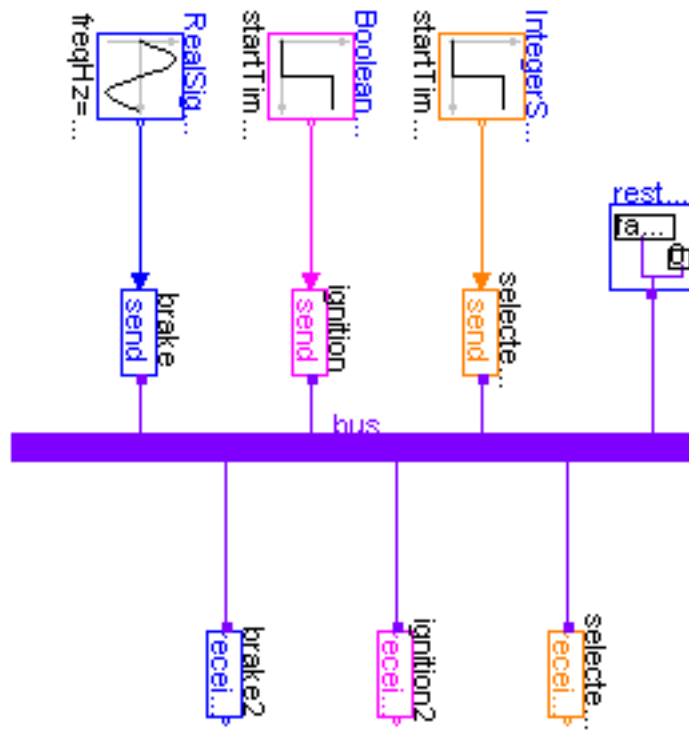


If you want to see, which signal was send to (or received from the bus) you double click on the line which connects the bus connector with the bus and you will see:

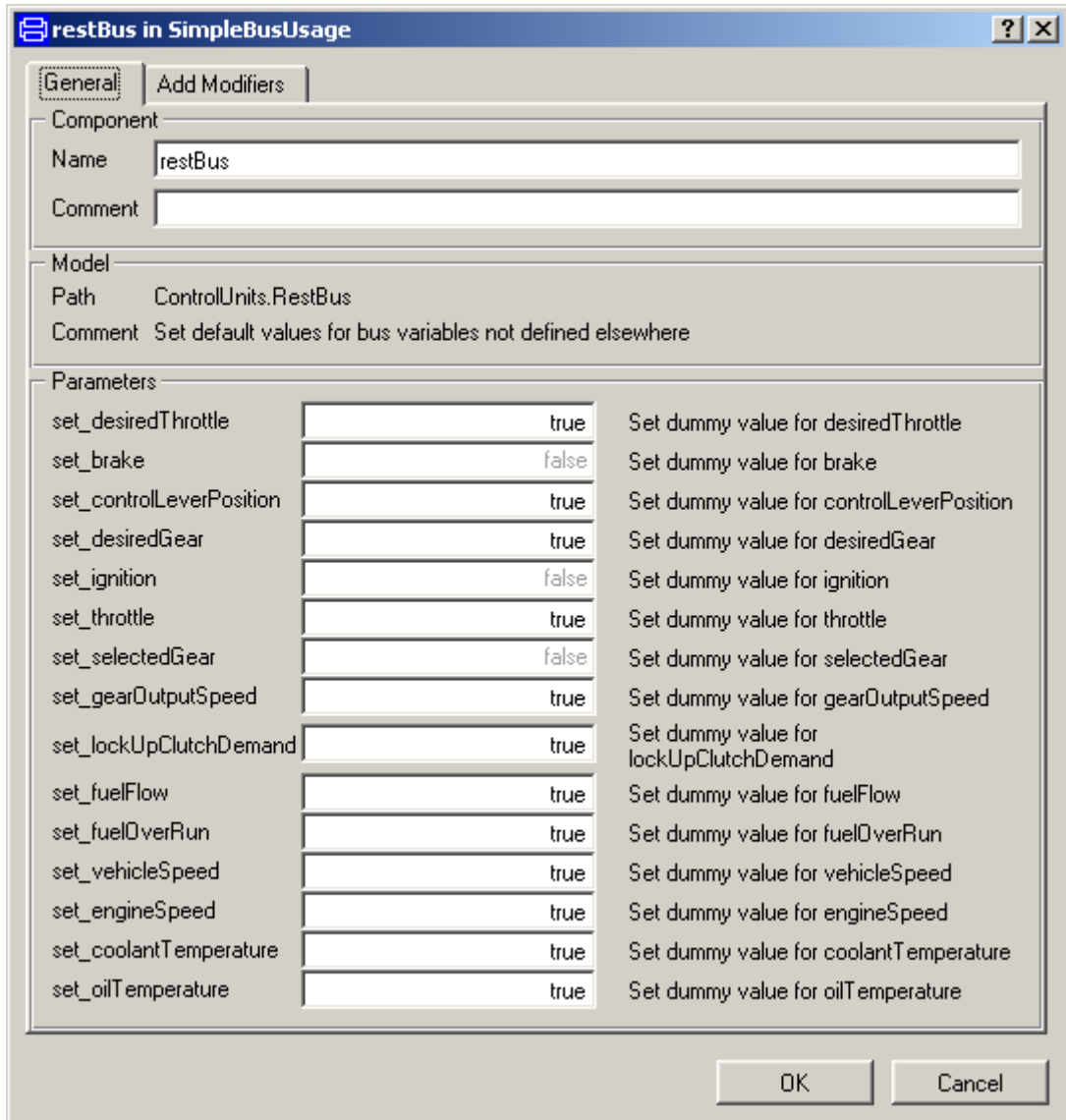


6.4 Model SimpleBusUsage

To learn more about the Bus, open the model “SimpleBusUsage” from package PowerTrain.Examples. This model shows, how different signals are sent to the bus and received from the bus.



The component “RestBus (in package PowerTrain.ControlUnits) allows to define default values of signals on the bus, if they are not defined elsewhere:



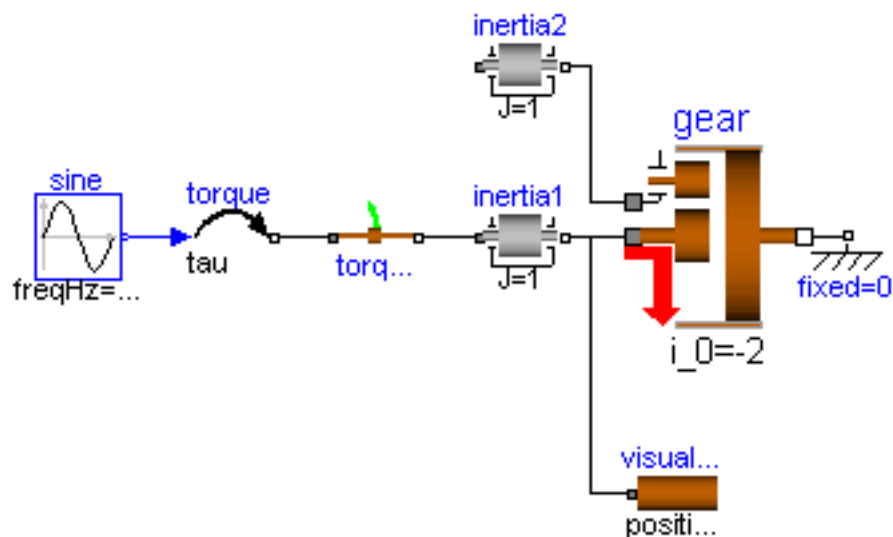
Chapter 7

Animation

Some components supply animation. The icons of such components have a dark orange color. You find such components in packages Clutches, Gears and Visualizers. Animation can be switched off for faster simulation. In this case, all equations needed for the animation are removed from the model before generating the code.

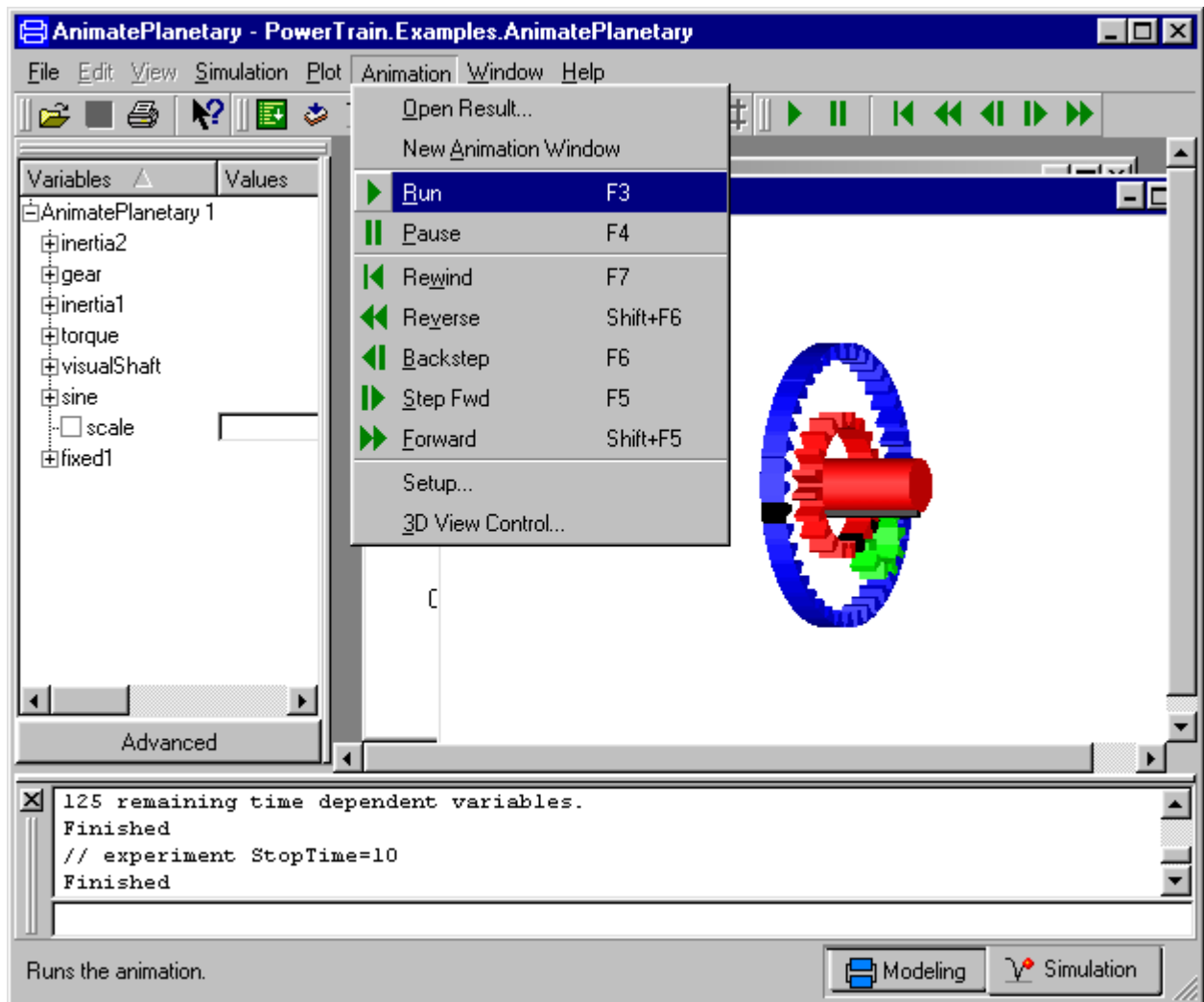
7.1 Run a first example with animation

An example with animation of a planetary gear is:



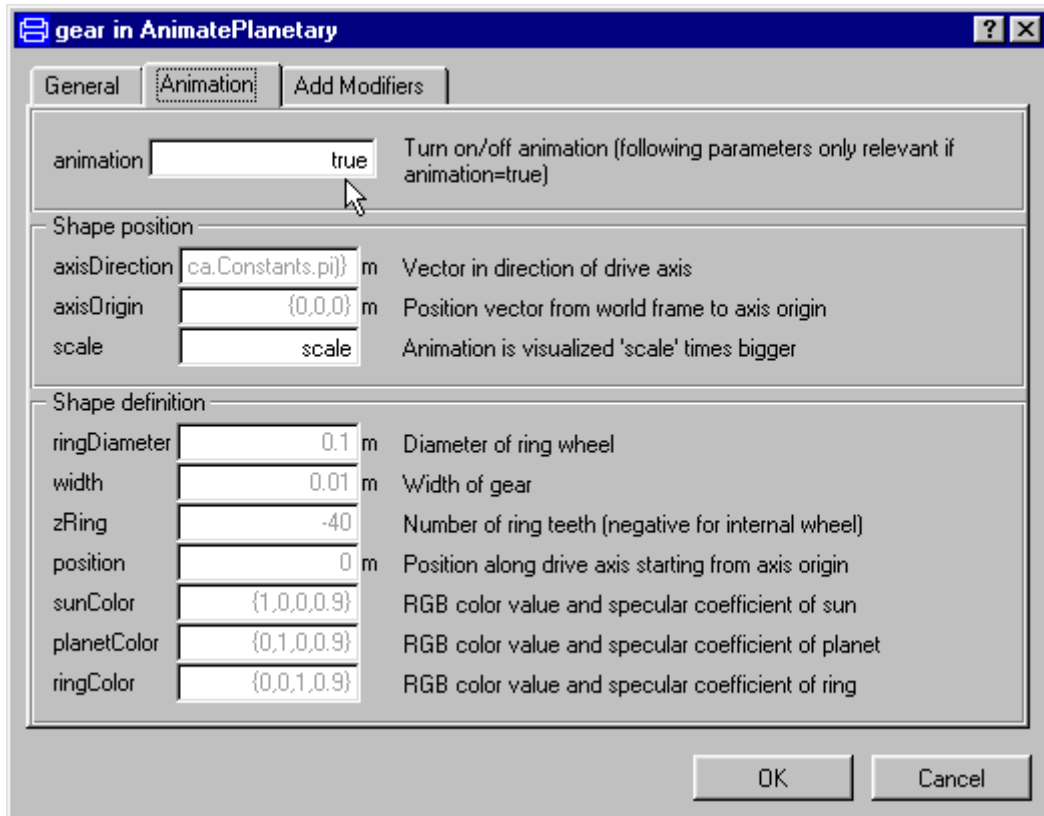
Set the stop time of this example to 100.

Attention: If the animation window does not open automatically (usually it is automatically opened when animation information is present), use the menu: Animation/New Animation. Now you can run the animation:



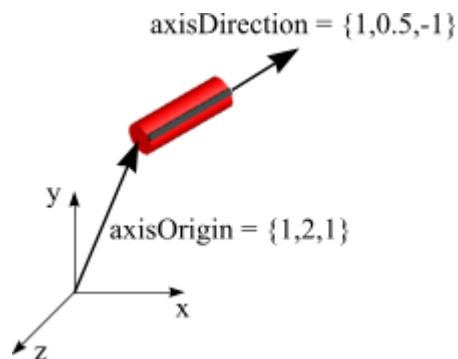
7.2 Animation parameters

Open example PowerTrain.Examples.AnimatePlanetary and double click on the component gear. You will get the following parameter window (choose subwindow animation):



If you choose the parameter **true** you have to fill the following parameters correctly. If you choose **false** the following parameters are without any meaning.

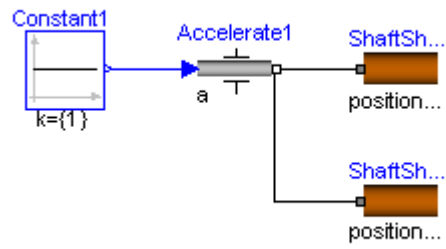
Parameters in shape position are:



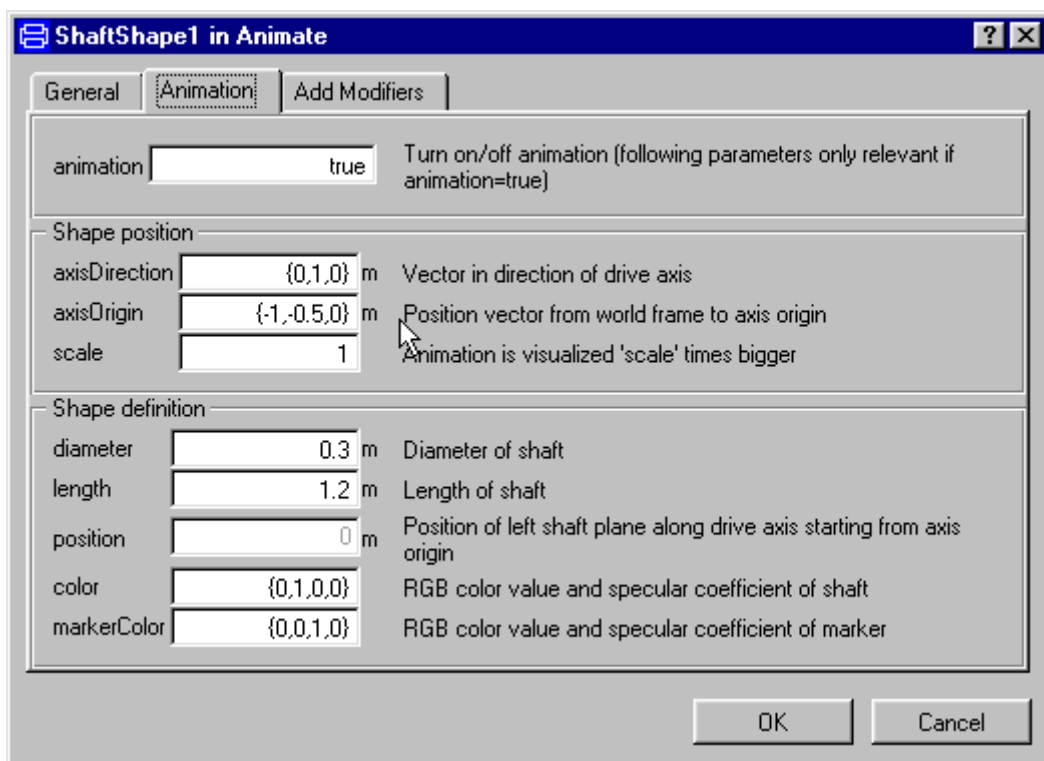
where

- **axisOrigin** is a vector from the basic (world) frame to the origin of the part
- **axisDirection** is a vector along the rotation axis of the part.

For a better understanding generate for example the following model (Component Accelerate is from package Modelica.Mechanics.Rotational, the Constant block is from the Modelica.Blocks.Sources, component Shaft is from package PowerTrain.Visualizers):

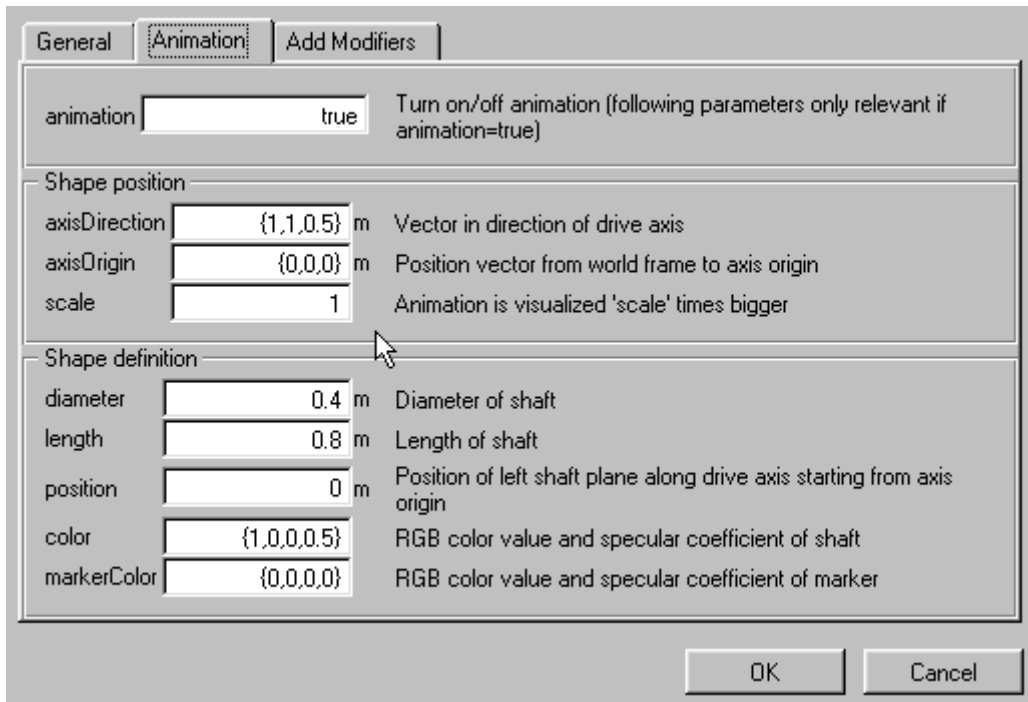


Choose the parameters in ShaftShape1:

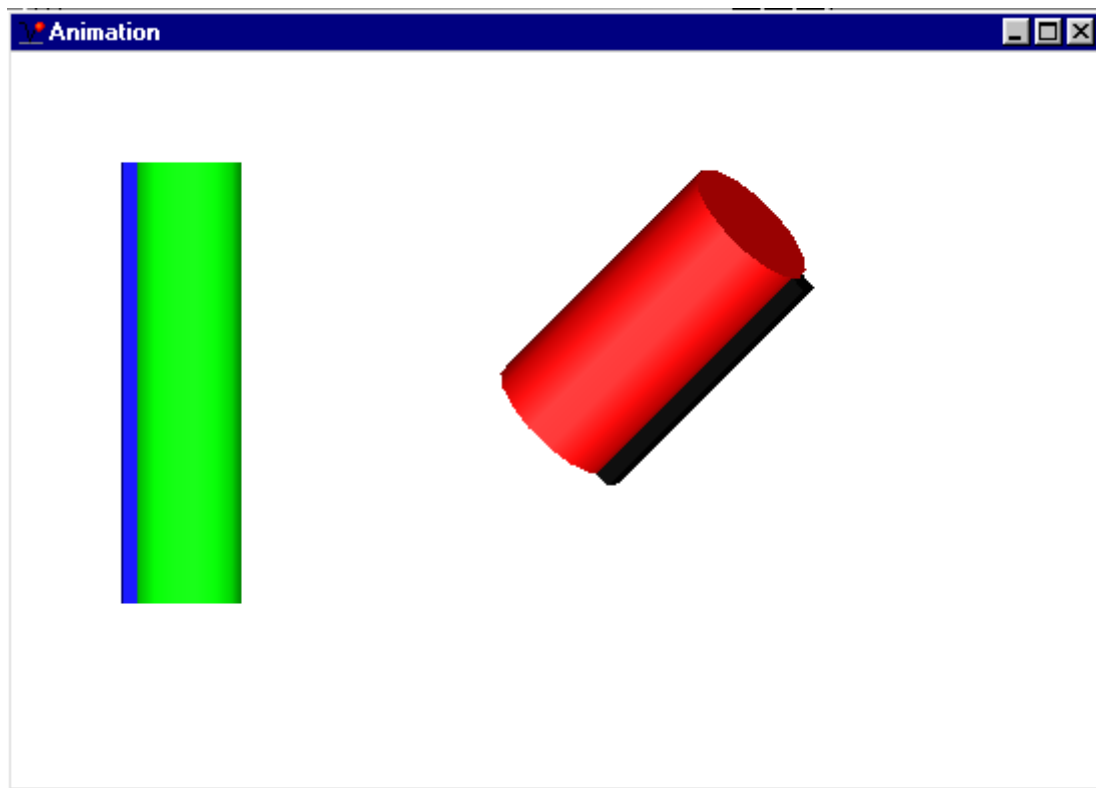


The position, rotation and size (diameter and length) of the shaft has been changed. The shaft shows directly into y-direction. The shaft is green (color = {0,1,0}) and the marker is blue (markerColor = {0,0,1}). The 4th value in the color parameters gives an impression of the illumination. 0 means show the real chosen value, 1 means illuminate the body.

Choose the parameters in ShaftShape2:



The position of the shaft has been changed, the shaft is red with a black marker. And you will get the following animation window:



Play with the parameters to understand their exact meaning.

7.3 Comments on the choice of colors

The color is given with RGB-(**R**ed-**G**reen-**B**lue) values. Best is you choose values between 0 and 1, which means how much of this basic color you take to mix the new color. The fourth value means the specular coefficient. The specular coefficient gives an impression of lighting on the body. Play with the example “Animate” to see the effects.

Examples:

- {0,0,0,0} is black,
- {1,0,0,0} is pure red,
- {0,1,0,0} is pure green,
- {0,0,1,0} is pure blue,
- {0,0,0,0} is black,
- {1,1,1,0} is white,
- {1,1,0,0} is yellow,
- {1,0,1,0} is magenta,
- {0,1,1,0} is turquoise,

All other choices are mixtures of these basic colors. The PowerTrain library uses a set of pre-defined constants in package Visualizers: Black, Red, Green, Blue, Yellow, Pink, Grey, Turquoise. You may use also use these constants in your models, e.g., “PowerTrain.Visualizers.Red”.

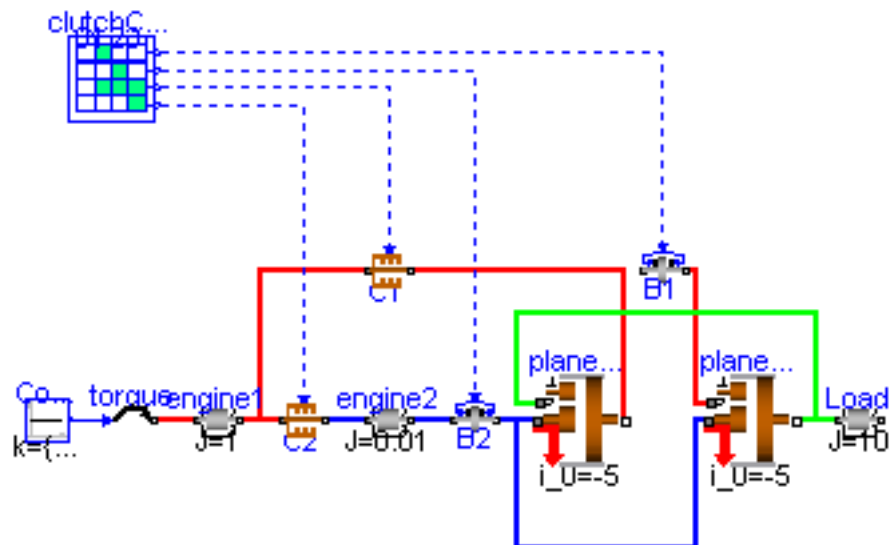
Chapter 8

Demo Examples

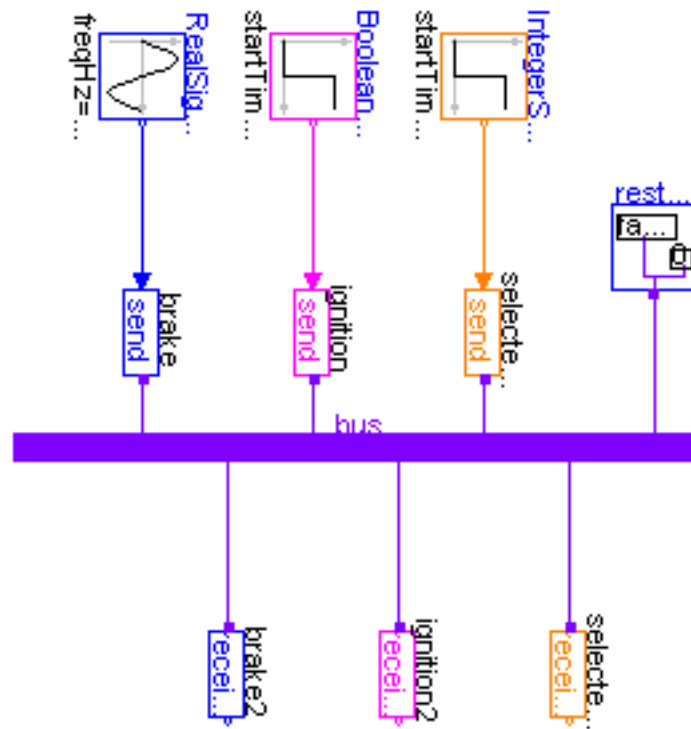
Several demo examples are delivered with the Powertrain Package. They help to understand the package and show examples of usage. All examples are described in the documentation of package PowerTrain.Examples.

8.1 List of all examples

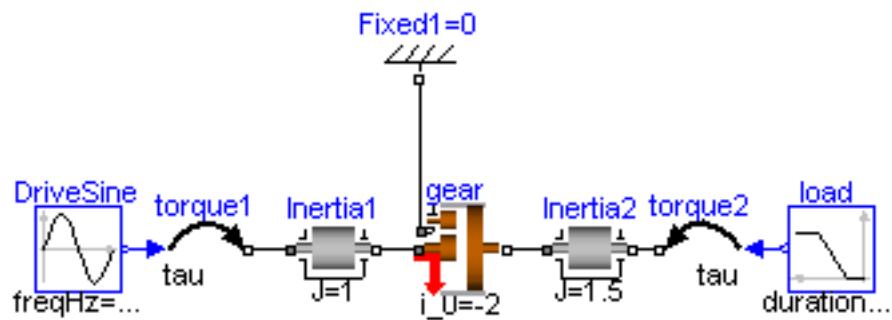
Examples.SimpleAutoGear: Simple automatic gearbox



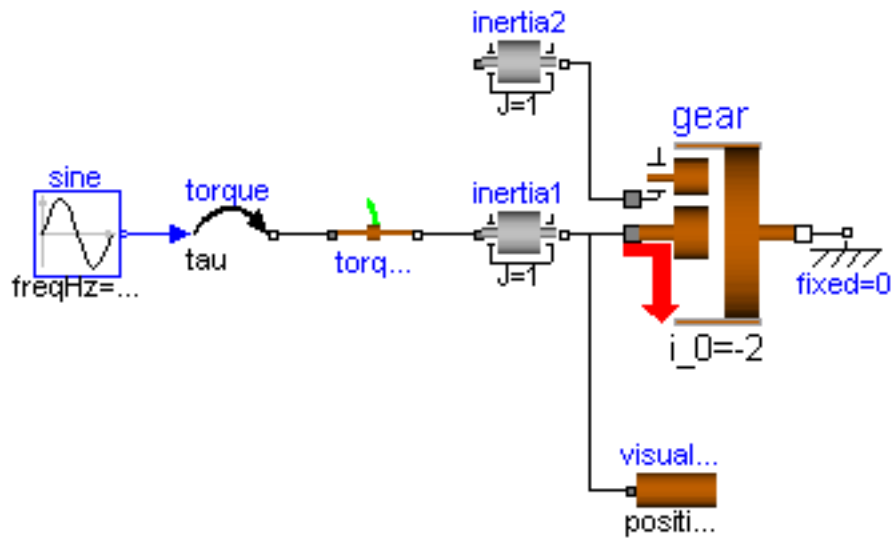
Examples.SimpleBusUsage: Demonstration of signal bus usage



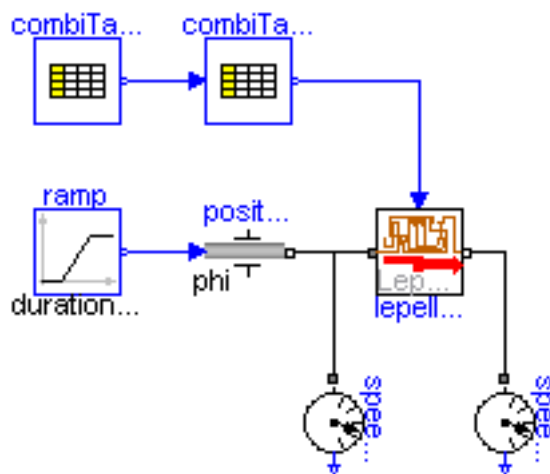
Examples.LossyPlanetary: Example to show that gear efficiency may lead to stuck motion



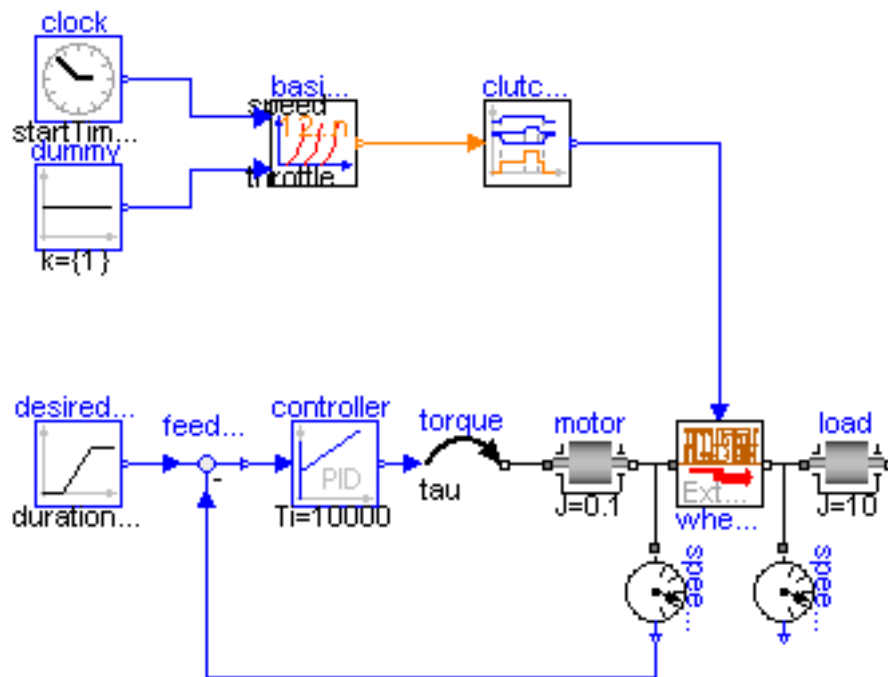
Examples.AnimatePlanetary: Animation of planetary gear



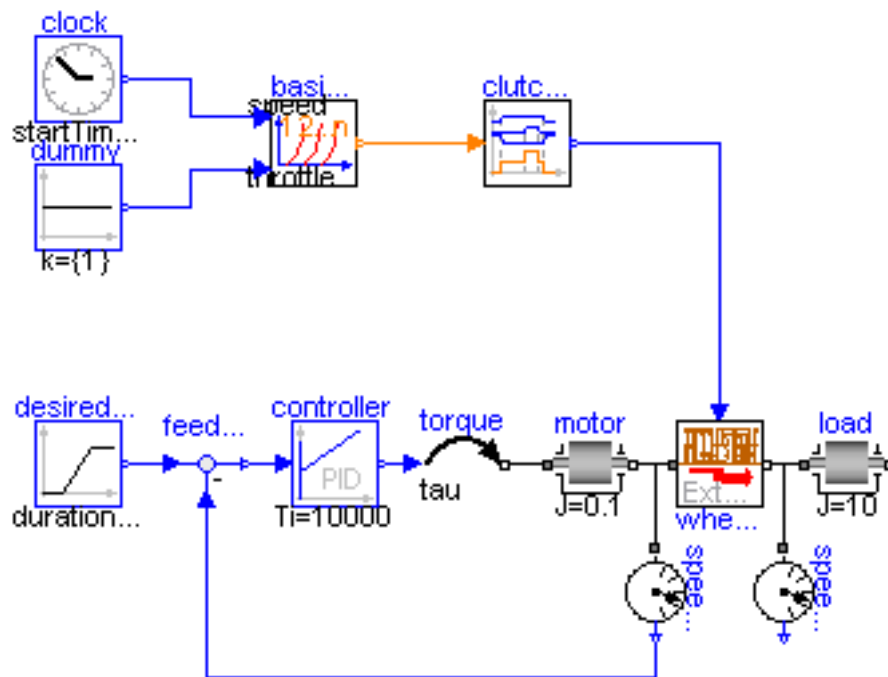
Examples.CheckRatios: Determine ratios of different gears



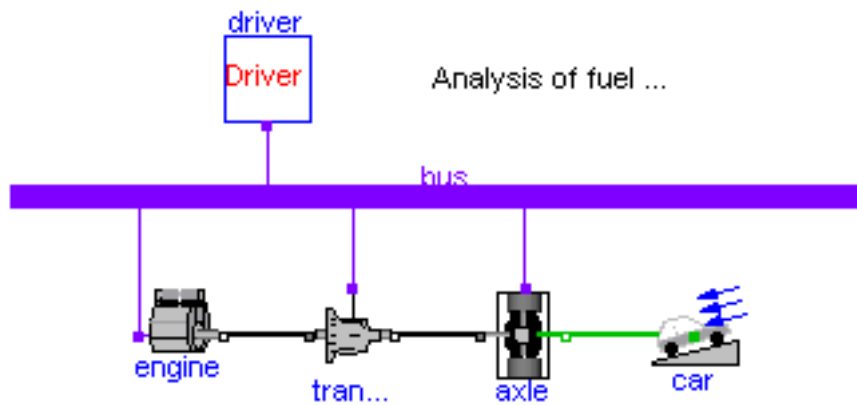
Examples.TestBench4s: Test 4-speed automatic gearbox



Examples.TestBench: Test 6-speed automatic gearbox

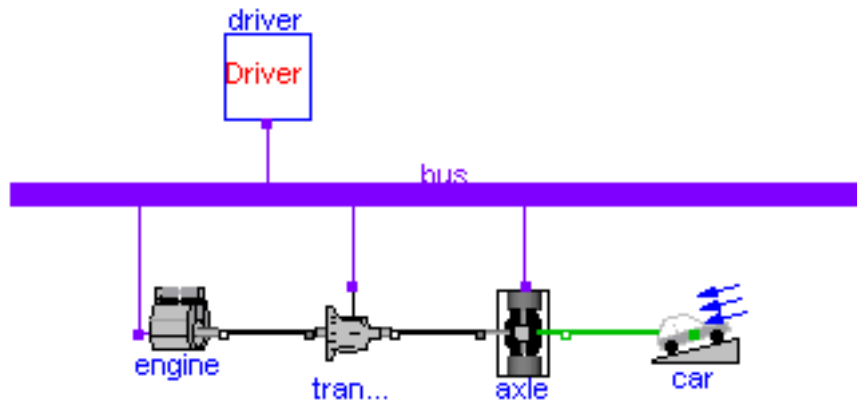


Examples.CarConsumption: Predict fuel consumption for prescribed driving cycle



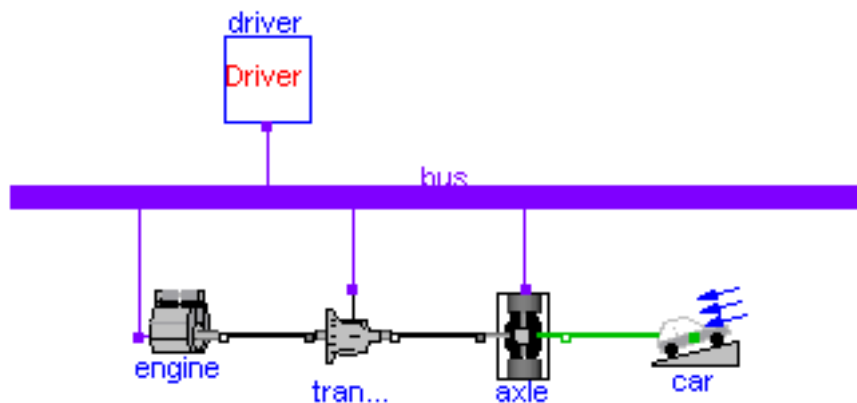
Examples.CarShiftStrategy4: Examine shift strategy of 4-speed automatic gearbox with detailed clutch models

analysis of shift strategy with ...



Examples.CarShiftStrategy6: Examine shift strategy of 6-speed automatic gearbox with detailed clutch models

analysis of shift strategy with ...

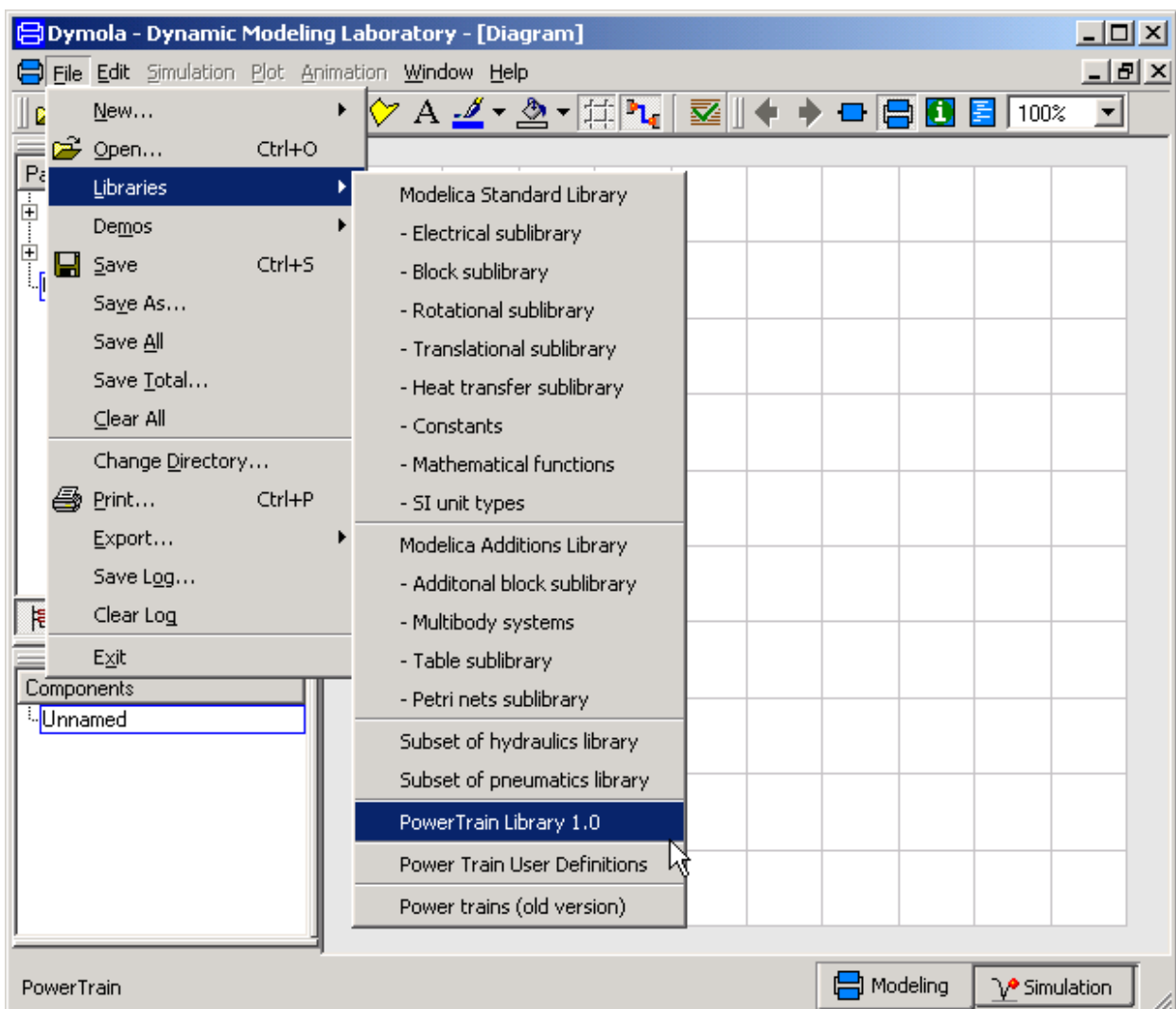


Chapter 9

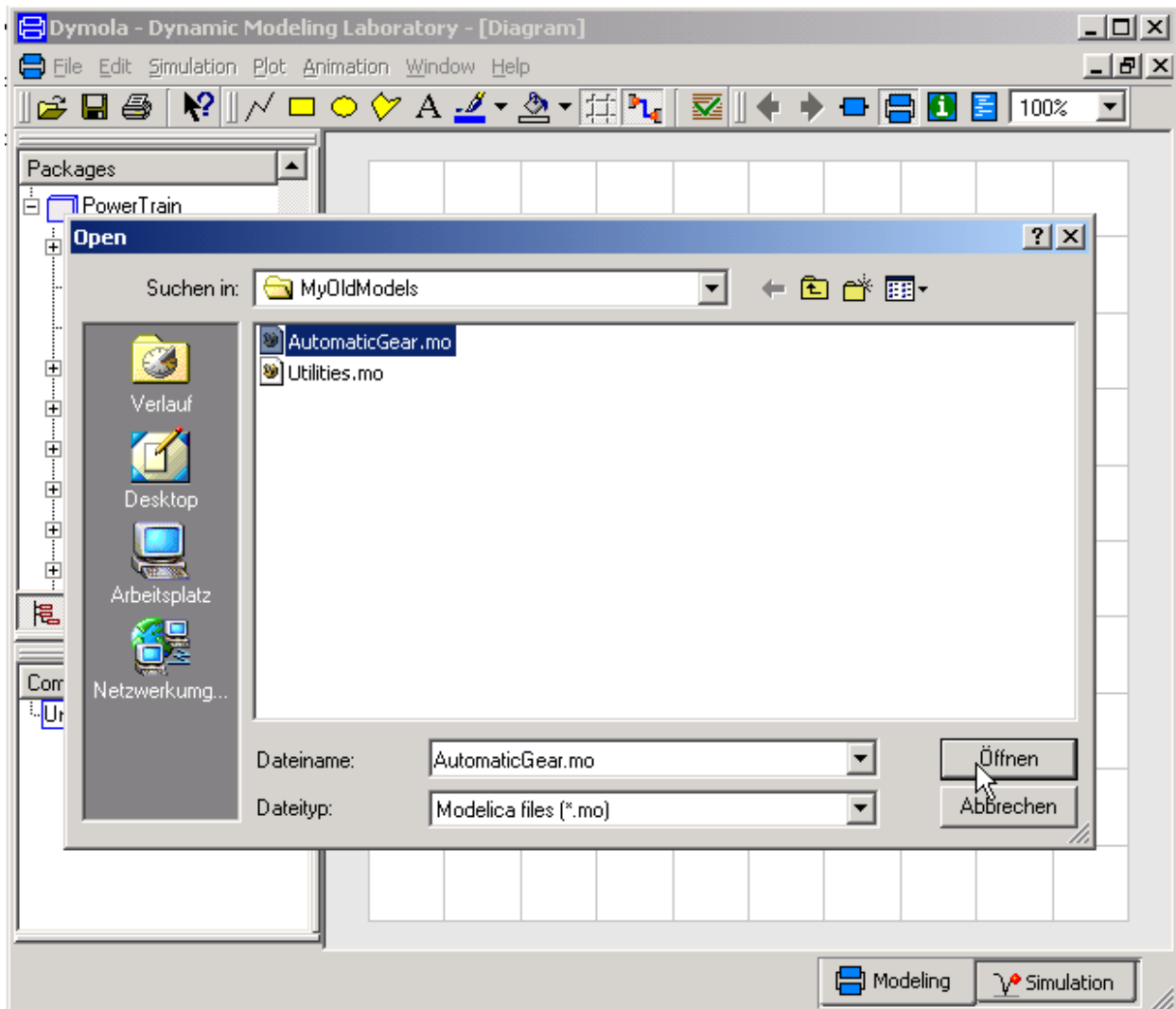
Upgrade from Former PowerTrain Versions

The PowerTrain library version 1.0 is not backward compatible. For users, who have developed models with a former version of the PowerTrain library and want to upgrade to the PowerTrain version 1.0, a conversion script is provided.

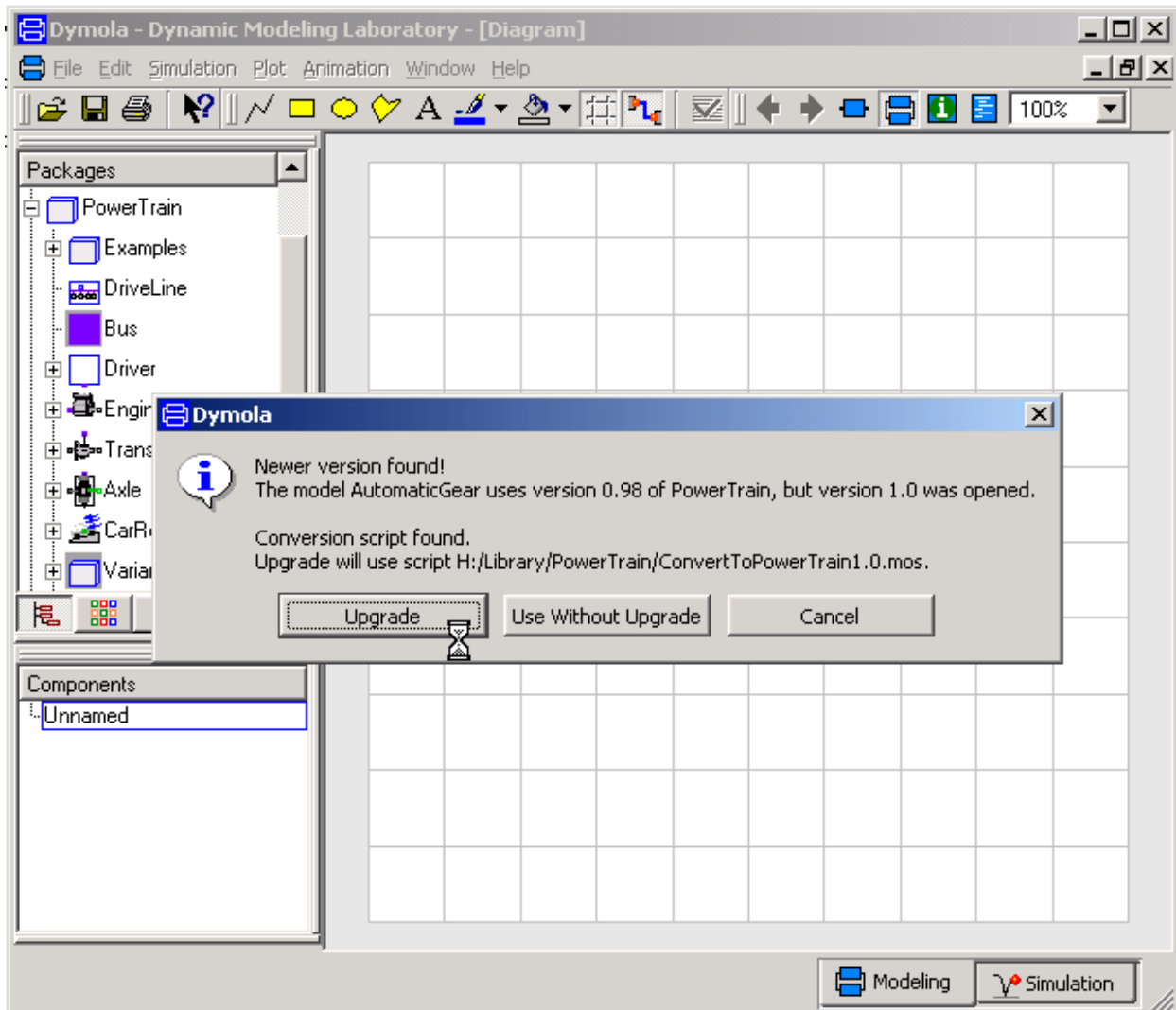
In order to use it, the PowerTrain library version 1.0 must be opened in Dymola via the File / Libraries menu (during installation of version 1.0, previous versions of the PowerTrain library are renamed. These versions are no longer needed).



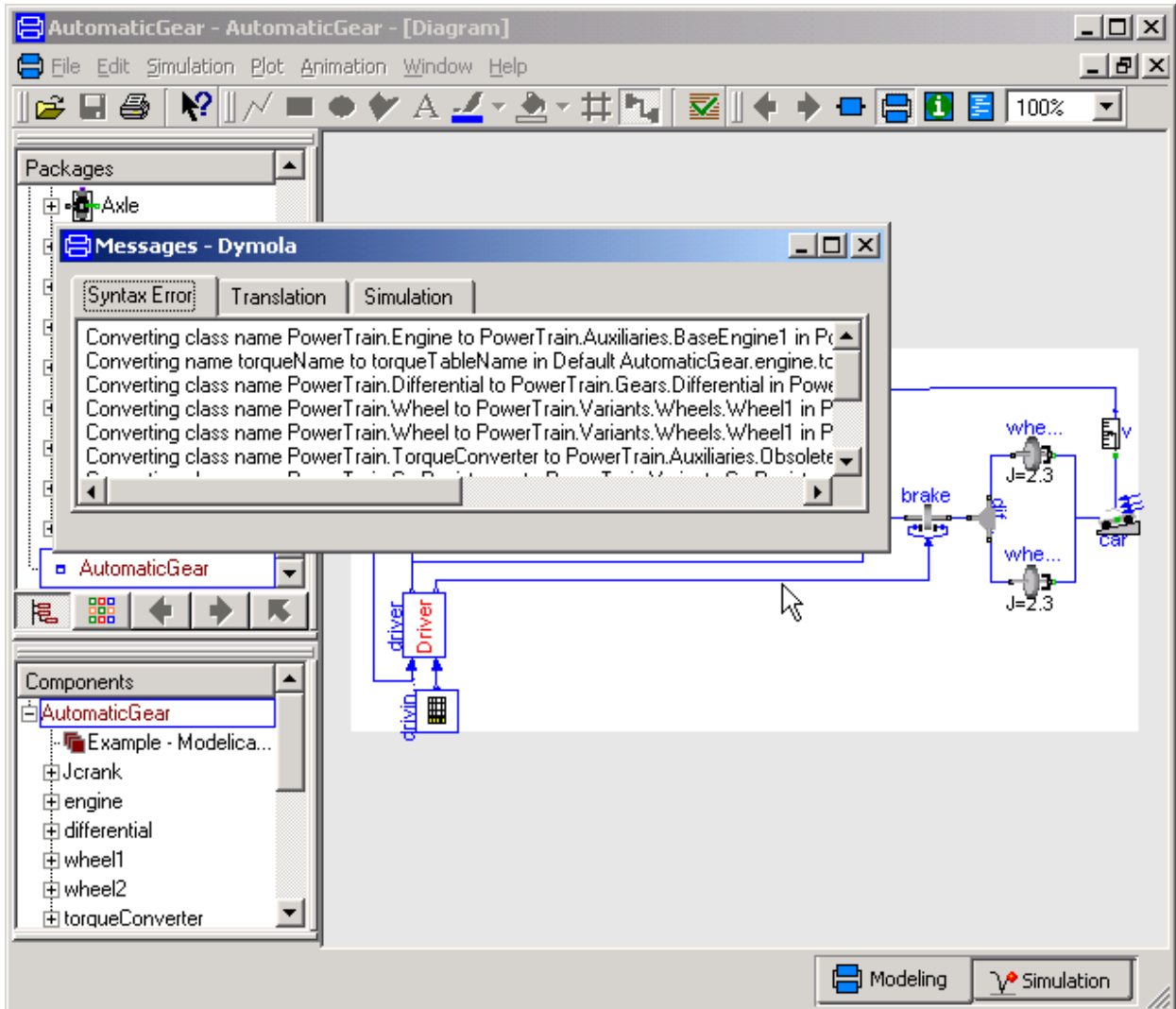
In the next step, a model which has been constructed with a former version of the PowerTrain library and which should be converted, must be opened in Dymola.



Dymola recognizes, that an old model is opened und will warn about this. The user is asked, if he wants to convert the model to be opened to a model using the PowerTrain library version 1.0. For converting, *Upgrade* must be clicked.



In the message window, Dymola reports which conversions are performed.



It is recommended to save the new model under a different name (*Save As ...*). Opening the next time, the new model uses the PowerTrain version 1.0 at once (the version numbers of the used Modelica libraries are stored in the models).

Chapter 10

Literature

10.1 Books

- Förster, H.J.: Automatische Fahrzeuggetriebe.
Springer-Verlag, Berlin Heidelberg New York, 1991
- Laschet, Andreas: Simulation von Antriebssystemen.
Springer-Verlag, Berlin Heidelberg New York, 1988, ISBN 3-540-19464-9
- Lechner G., Naunheimer H.: Fahrzeuggetriebe
Grundlagen, Auswahl, Auslegung und Konstruktion
Springer-Verlag, Berlin Heidelberg New York, 1994, ISBN 3-540-57423-9
- Looman, Johannes: Zahnradgetriebe
Grundlagen, Konstruktion, Anwendung in Fahrzeugen
Springer-Verlag, Berlin Heidelberg New York, 1996, ISBN 3-540-60336-0

10.2 Articles

- Schlegel C., Bross M., and Beater P.:
Schlegel Simulation GmbH, BMW München, and Universität-GH Paderborn
HIL-Simulation of the Hydraulics and Mechanics of an Automatic Gearbox
http://www.modelica.org/Conference2002/papers/p10_Schlegel.pdf
- Treffinger P. and Goedecke M.:
Development of Fuel Cell Powered Drive Trains With Modelica
DLR Stuttgart, Germany
http://www.modelica.org/Conference2002/papers/p16_Treffinger.pdf
- Pelchen C., Schweiger C., and Otter M.:
Modeling and Simulating the Efficiency of Gearboxes and of Planetary Gearboxes
ZF-Friedrichshafen, DLR Oberpfaffenhofen, Germany
http://www.modelica.org/Conference2002/papers/p33_Pelchen.pdf

-
- Hellgren J.:
Modelling of Hybrid Electric Vehicles in Modelica for Virtual Prototyping
Chalmers University of Technology, Göteborg, Sweden
http://www.modelica.org/Conference2002/papers/p32_Hellgren.pdf
 - Otter, M., Schlegel, C.:
Objektorientierte Modellierung Physikalischer Systeme, Teil 9: Modellierung von Antriebssträngen.
at Automatisierungstechnik, 47, 12, pp. A33-A36 (1999).
 - Otter, M., Elmqvist, H., Mattsson, S.E.:
Hybrid Modeling in Modelica based on the Synchronous Data Flow Principle.
1999 IEEE Symposium on Computer-Aided Control System Design, CACSD'99, Hawaii, pp.
151-157, August 22-27, 1999.
<http://www.modelica.org/papers/hybrid99.pdf>
 - Jacobson, Fredriksson, Hellgren, Karlsson, Scarpati, Templin, Vallejo:
Modelica Usage in Automotive Problems at Chalmers
Chalmers University, Sweden
<http://www.modelica.org/workshop2000/proceedings/Jacobson.pdf>
 - Tiller, Bowles et al.: Detailed Vehicle Powertrain Modeling in Modelica
Ford Motor Company, USA
<http://www.modelica.org/workshop2000/proceedings/Tiller.pdf>