The PowerTrain Library: New Concepts and New Fields of Application

Christian Schweiger[1] Mike Dempsey[2] Martin Otter[1]

[1]German Aerospace Center (DLR), Oberpfaffenhofen, 82234 Weßling, Germany
[2]Claytex Services Limited, Warwickshire, CV35 8XB, United Kingdom

Abstract

Version 2.0 of the PowerTrain library will be released in March 2005. This article presents the new release, which is enriched by optional consideration of 3D effects, a simpler signal bus concept, new components and example models for flexible drivelines, 4wd drivelines and hybrid vehicles. In addition, various new driver models have been added.

1 Introduction

The PowerTrain library [6] is a licensed Modelica package providing components for modelling vehicle powertrains. It is also used for the modelling of gearboxes with speed and torque dependent losses. The available components range from simple, easy to use parts to very sophisticated components. All the components are open and can be extended and modified by the user.
In December 2002, version 1.0 was finished and since then several new developments have been incorporated into the library. In addition, new concepts in the Modelica language have been applied leading to improvements aimed at delivering better interoperability between the different automotive model libraries available. These are described in Section 3. In Section 4, new fields of application are described and new driver models are presented in Section 5.

2 Previous library status

In the past, the library provided standard and planetary gearboxes with speed and torque dependent losses, table-based engine and simple driver models, and components required to model the longitudinal dynamics of vehicles, as well as a range of detailed examples. Version 1.0 of the library contained 45 reusable components and 10 examples, cf. Figure 1. A number of components originally developed for the PowerTrain library have been incorporated into the Modelica standard library (version 1.5) since they are of general interest.

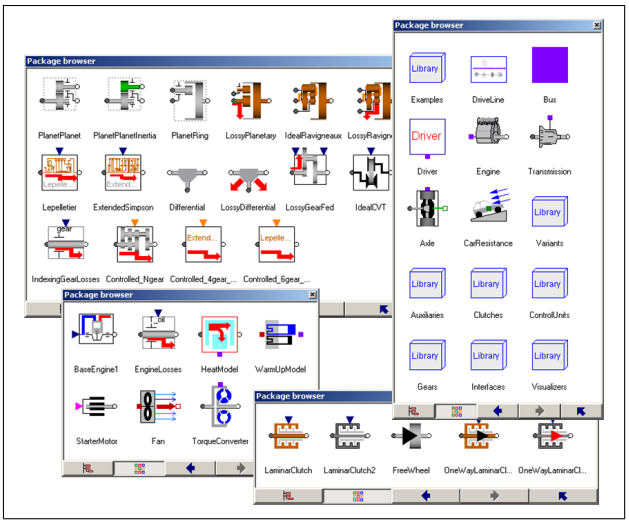
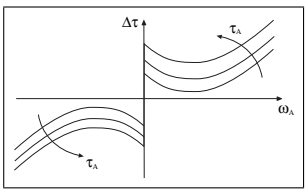
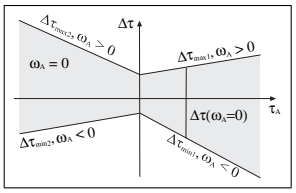


Figure 1: Components of the PowerTrain library and some sublibraries

A very important property of the PowerTrain library is the robust and efficient handling of *speed and torque dependent friction* [8] as illustrated in Figure 2, which occurs when considering gear mesh efficiency (due to gear teeth friction) and bearing friction. This novel type of friction handling was used in many components in version 1.0, especially for planetary gears, Ravigneaux gears, Lepelletier gears, extended Simpson gears and differential gears.
Large system models often become difficult to understand as there can be a large number of signals that need to be passed between the model's top-level components. To overcome this problem, the PowerTrain library used a signal bus as shown in Figure 3. The idea was that all the signals that have to be exchanged by the components are included on the bus. The com-



(a) Dependency on speed ω_A



(b) Dependency on torque τ_A

Figure 2: Speed and torque dependent friction $\Delta\tau$

ponents are then simply connected to the bus and do not have any signal connections to other components.
Another common modelling problem is that models with varying levels of detail are required for different tasks. To reduce the number of different models that need to be saved a model architecture was implemented that would allow components contained in the top-level objects to be easily swapped for other compatible models with different levels of detail. The ability to swap the components was realized by making use of the replaceable model features of Modelica. As a consequence, only one model, cf. Figure 3, is necessary to model many different driveline configurations.

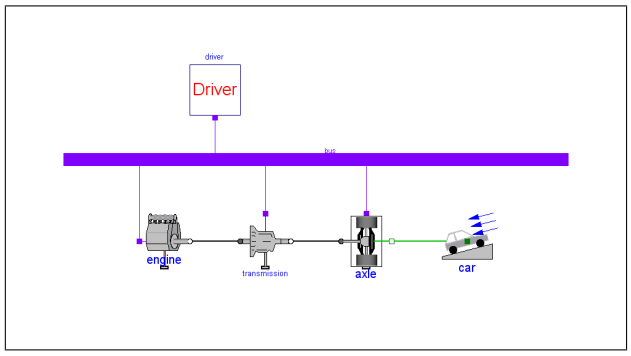


Figure 3: Model Driveline consists of a typical driveline from which all the main variants provided in the library can be selected, e.g., three different types of detailed automatic gearbox models but also user-defined gearboxes. The components need only a single connection to the *signal bus* in order to exchange signals among them.

The gearbox and shaft components can be animated, see Figure 4 for an example, which is useful for plausibility checking and demonstration purposes. Animation can be switched off by a parameter. In this case, the complete animation code is removed from a model in order to get efficient simulation code, e.g., for hardware-in-the-loop simulations.
A number of control systems as shown in Figure 5 were included in the library. These are used to control the engine and transmission models. The control

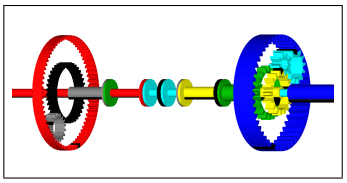


Figure 4: Animation of 6-speed automatic gearbox of Lepelletier type

systems for automatic transmissions are implemented in such a way that they support any number of gears. The driver interacts with these controllers by setting the gearbox mode to be used (P, R, N, D, 1, 2, 3, . . .).

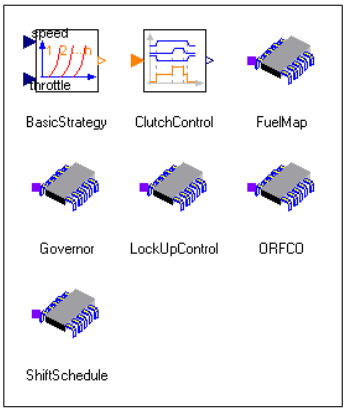


Figure 5: Control units included in the library

Several sophisticated example models are provided which serve as a starting point in developing user-specific models. Especially, examples are provided for power consumption calculation and analysis of shift strategies based on detailed models of 4- and 6-speed automatic gearboxes

3 New concepts and components

Several new concepts and components have been incorporated into version 2.0 of the PowerTrain library. They are described in the following sections.

3.1 Incorporation of 3D effects

In [9], a concept for reproducing the three-dimensional (3D) mechanical effects of one-dimensionally (1D) modelled powertrains has been presented. The idea is to model transmission elements with their mostly 1D rotating behaviour in a convenient way with 1D model components. Due to the simplicity of the 1D equations, this results in very efficient simulation code. When these 1D components are mounted

on systems moving in 3D space using the Modelica.Mechanics.MultiBody [7] library a number of important effects, such as support torques and gyroscopic torques, are missing. By including adaptor models and a 3D inertia component it is possible to incorporate these missing effects.
These 3D effects are incorporated in version 2.0 of the PowerTrain library. By default, 3D effects are turned off to get fast simulations, which is especially important for real-time purposes [3, 10]. The 3D effects can be turned on through the use of a parameter. This has been implemented using the new Modelica feature *conditional declarations* that have been introduced in version 2.2 of the Modelica language specification. The idea is illustrated using the example in Listing 1 and the object diagram in Figure 6.

Listing 1: Example demonstrating conditional declarations

```
model DampedInertia
  import Modelica.Mechanics.Rotational;

  extends Rotational.Interfaces.TwoFlanges;

  parameter Boolean damping=true;

  Rotational.Inertia inertia;
  Rotational.Damper damper(d=10) if damping;
  Rotational.Fixed fixed if damping;
equation
  connect(inertia.flange_a, flange_a);
  connect(inertia.flange_b, flange_b);
  connect(damper.flange_a, fixed.flange_b);
  connect(damper.flange_b, flange_b);
end DampedInertia;
```

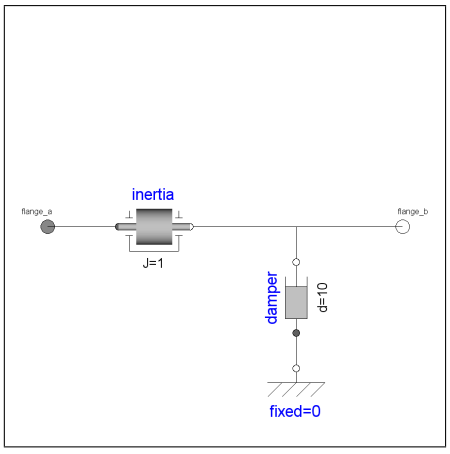


Figure 6: Object diagram of model *DampedInertia*

The declaration of the components *damper* and *fixed* is dependent on the Boolean parameter *damping*. These components are instantiated only if *damping* has the value true. Otherwise these components are not instantiated and the connect statements referring to them are ignored. The advantage in comparison to simply setting the damping coefficient *d* of *damper* to zero is that the equations of the disabled components are removed from the model and from the generated code, leading to more efficient simulations.
This feature is now used to incorporate 3D effects into the PowerTrain library. To include 3D effects into the components, a MultiBody connector is required but for the simple 1D case, it is desirable to remove these connectors. Therefore, the base class shown in Listing 2 was implemented which is inherited by the affected components of the PowerTrain library.

Listing 2: Base class for components with optional 3D effects

```
partial model ThreeD
  import Modelica.Mechanics.MultiBody;
  parameter Boolean enable3D=true;
  MultiBody.Interfaces.Frame_a
                    frame_a if effectiveEnable3D;
protected
  outer MultiBody.World world;
  parameter Boolean effectiveEnable3D=
                     world.enable3D and enable3D;
end ThreeD;
```

This base class allows the 3D effects to be switched on or off in two ways:

- The base class provides a Boolean parameter *enable3D*, which can be used to disable the 3D effects for a particular component.

- It is also possible to enable the 3D effects for all the components within a model by use of a single global setting. This is implemented using the inner-outer concept of Modelica, which is already used in the MultiBody library for providing global settings (e.g. for default animation and gravity field). These definitions are set in the inner component *world* which must be added to the top level of the model. All the components within a model access the global settings through a respective outer component *world*. The component MultiBody.World has been extended by adding a Boolean parameter *enable3D*.

Both parameters have to be true for the 3D behaviour to be modelled. If either of them is false then the 3D MultiBody *frame_a* as well as the additional equations for modelling the 3D effects are removed and only 1D behaviour is modelled. The described base class is

used throughout the PowerTrain library when 3D effects might be included. By default, only 1D behaviour is modelled and all the code for the 3D effects is removed during the code generation phase. By inheriting from this class, it is easy to enable 3D effects and then place the complete powertrain models onto 3D moving parts without neglecting any 3D effects.
Figure 7 shows a model which combines components of the PowerTrain and the VehicleDynamics library [2]. The 3D effects are modelled in the powertrain. First investigations, see Figure 8, of such a model have been performed in [5].

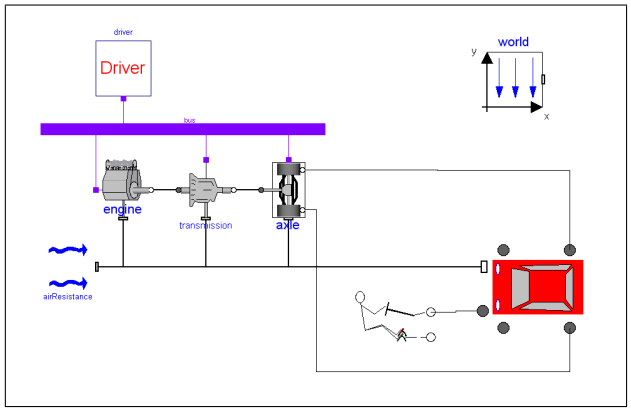


Figure 7: Powertrain model with 3D effects combined with vehicle dynamics model

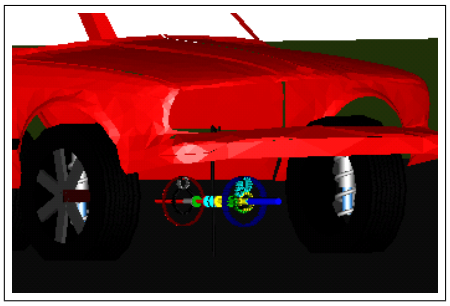


Figure 8: Animation of a joint powertrain and vehicle dynamics model

3.2 Expandable connectors

The signal bus concept introduced in version 1.0 of the PowerTrain library was not easy to extend to user-specific needs. The recommended method for changing the signal bus was to extend from the library and simultaneously replace the bus with a user-defined bus, ending up in a complicated structure.
For the *Vehicle Model Architecture* described in Section 3.4 an improved bus concept — *expandable connectors* — has been developed and included in version 2.2 of the Modelica language specification. In the simplest case, an expandable connector is merely an empty connector, see Listing 3. This connector class can be instantiated in different components, e.g, Source and Integrator in Listing 3, and it is possible to connect to components in the expandable connector, even though they are not defined in the class definition of the bus. The various connect statements are evaluated at compile time and the union of all referenced variables is used to build the actual bus connector. During translation a check is made to ensure that every signal read from the bus is defined exactly once.

Listing 3: Example package demonstrating bus realisation using an *expandable connector*

```
package BusTest
  import Modelica.Blocks;

  expandable connector Bus
  end Bus;

  model Source
    Bus bus;
    Blocks.Sources.Sine sine;
  equation
    connect(sine.y, bus.dq);
  end Source;

  model Integrator
    Bus bus;
    Blocks.Continuous.Integrator integrator;
  equation
    connect(integrator.u, bus.dq);
    connect(integrator.y, bus.q);
  end Integrator;

  model Example
    Bus bus;
    Source source;
    Integrator integrator;
  equation
    connect(bus, source.bus);
    connect(bus, integrator.bus);
  end Example;
end BusTest;
```

A simulation result of model *Example* in Listing 3 is shown in Figure 9. Although the bus connector was defined as empty, the two contained variables can be plotted.

The concept of expandable connectors leads to an enormous gain in flexibility. The bus definition in the PowerTrain library has been changed to that shown in Listing 3. This allows a user to extend the bus in a very convenient way: Just a connection must be drawn between a signal port and the bus, see Figure 10. In addition a variable name on the bus must be provided.

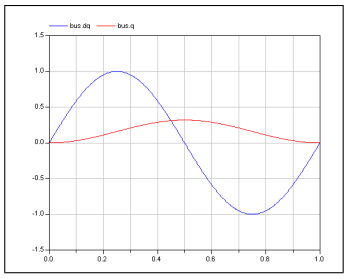


Figure 9: Simulation result of model *BusTest.Example*

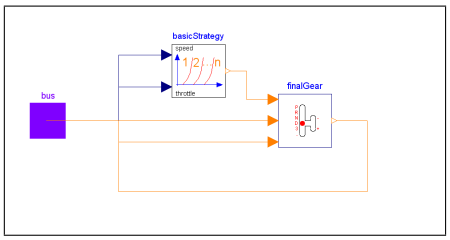


Figure 10: Example model *PowerTrain.ControlUnits.-ShiftSchedule*

3.3 New gears with losses

In the previous version of the PowerTrain library, the two components *Gears.PlanetPlanet* and *Gears.PlanetRing* were provided so that any type of planetary gearbox could be constructed. These elements have been improved so that speed and torque dependent losses are now taken into account. An example is shown in Figure 11 where a planetary gear of the Wolfrom type, with losses, is constructed using the PlanetPlanet and PlanetRing components.

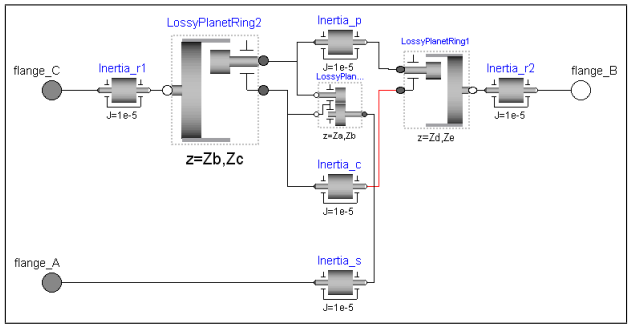


Figure 11: Object diagram of Wolfrom type planetary gearbox with losses implemented using the improved PlanetPlanet and PlanetRing components

The overall gear ratio and efficiency of a planetary gearbox constructed using these basic elements can be calculated using the model *PowerTrain.Examples.WolfromEfficiency* shown in Figure 12, provided that the number of teeth on each of the gearwheels and the efficiencies of each mesh (gear teeth contact) are known.

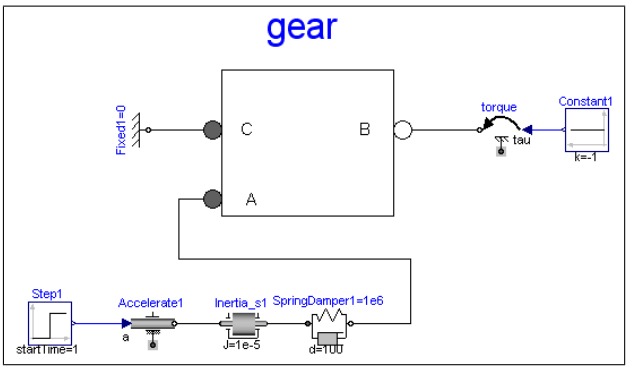


Figure 12: Object diagram of test model to determine gear ratio and gear efficiency between axis A and B of the Wolfrom planetary gearbox

The gear ratio and the gear efficiency between axis A and B are computed as follows:

- At the output (axis B) a unit torque is applied as a load.
- At the input (axis A) a unit acceleration is applied for 1 s. This means that the speed of axis A starts at zero, and rises linearly to 1 rad/s during the first 1 s of the simulation and then remains constant at 1 rad/s. Since the speed is constant, the inertias inside the gear do not have an effect for the power distribution.
- To avoid possible problems with the non-uniqueness of solutions of friction elements when forcing the wheel to rotate according to the desired acceleration, the forced movement of the flange is not directly required. Instead, the acceleration component drives a very stiff spring which in turn drives the gear flange.
- The gear ratio is the ratio of the angular velocities of flange_A and flange_B at the end of the simulation (say at 2 s).
- The gear efficiency is the ratio of the cut-torques of flange_A and flange_B divided by the gear ratio.
- The above two numbers can most easily be determined from the simulation, in Dymola, by clicking in the plot window on *Advanced* and then setting $t = 2$ in the input field *Time*. This displays the values of all variables in the variable browser at $t = 2$.

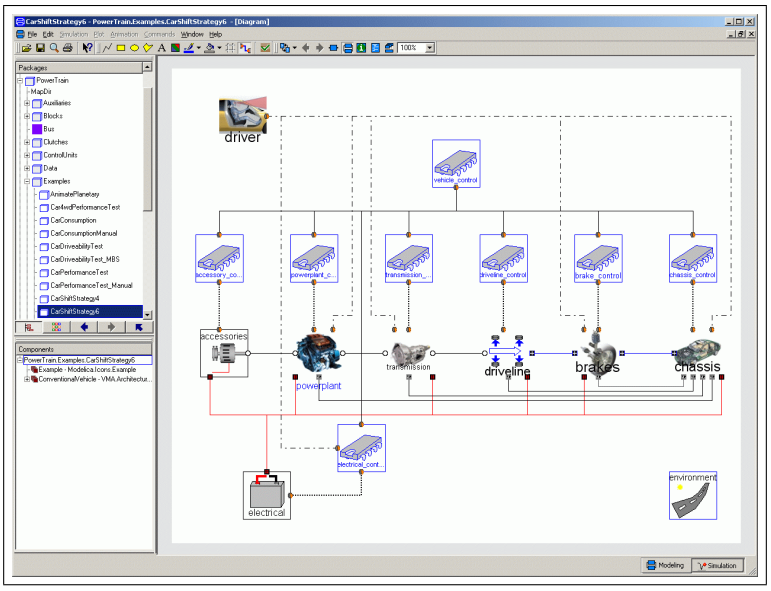


Figure 13: Screenshot of Dymola showing example model on Vehicle Model Architecture basis

It would not be possible to determine these values using a static model where the gear shafts are not rotating. This is because the friction between the teeth would be in the stuck mode and the friction torques are then **computed** implicitly from the requirement that the shaft accelerations are zero. This is correctly described by the Modelica model and therefore does not allow the efficiency to be calculated using a static model.

3.4 Vehicle Model Architecture

Within Ford Motor Company's Powertrain Research Department an architecture for modelling of vehicles has been developed and reimplemented in Modelica. The resulting Modelica package was presented in [11] and is freely redistributable in source code form. In order to promote interoperability with other libraries in the automotive area, this architecture will be used in the PowerTrain library. An example model following this architecture is shown in Figure 13.

4 New fields of application

Version 2.0 of the PowerTrain library has been extended by including a wider range of driver models and new components in a number of new application areas. Example models demonstrating the usage of the new components have also been included in the library and these are described below.

4.1 Flexible driveline models

An area of increasing interest is the modelling of vibrations and oscillatory responses within the whole powertrain. These effects are required when attempting to simulate driveability, shift quality or other similar effects that are likely to introduce oscillatory torques into the powertrain system. The study of these effects has required the development of additional driveline component models that include additional effects such as stiffness, damping and backlash. The first key component required was the flexible shaft, which introduces the ability to model the twisting of a shaft, such as the propshaft or driveshafts. In it's simplest form the flexible shaft consists of two rotational inertias connected by a linear spring-damper. In this form the shaft can be used to model low frequency effects such as shuffle, which occurs in the 2..10 Hz range.

The flexible shaft can easily be adjusted to model higher frequency effects as it can contain a variable number of spring-dampers and inertia components. This is possible through the use of a parameter n to specify how many spring-damper blocks the flexible shaft model should contain. The effective stiffness and damping of each spring-damper block is adjusted based on the parameter n. The flexible shaft contains $n+1$ inertias and the total inertia of the shaft is evenly distributed across these. The implementation of the flexible shaft is shown in Listing 4.

When developing a model to simulate driveability or shift quality it is important to include the reaction of the powertrain within the vehicle. As the entire pow-

Listing 4: Flexible shaft implementation

```
model FlexibleShaft
  extends Modelica.Mechanics.Rotational.Interfaces.TwoFlanges;

  parameter PowerTrain.Types.TorsionalStiffness c=1 "Stiffness";
  parameter PowerTrain.Types.TorsionalDamping d=0 "Damping";
  parameter Modelica.SIunits.Inertia J=1 "Inertia";
  parameter Integer n(min=1) = 1 "Number of spring-dampers";

  Modelica.Mechanics.Rotational.Inertia inertia[n + 1](each J=J/(n + 1));
  Modelica.Mechanics.Rotational.SpringDamper springDamper[n](each c=c*n, each d=d);
equation
  connect(flange_a, inertia[1].flange_a);
  for i in 1:n loop
    connect(springDamper[i].flange_a, inertia[i].flange_b);
    connect(springDamper[i].flange_b, inertia[i + 1].flange_a);
  end for;
  connect(inertia[n + 1].flange_b, flange_b);
end FlexibleShaft;
```

ertrain is suspended within the vehicle by a number of mounts it can move within the vehicle and have a significant impact on the overall vehicle response. Previously the PowerTrain library did not consider these effects and a number of components have been developed specifically for this task.

Within the PowerTrain library an example has been included showing how to model the reaction of a differential on its mounts for a rear-wheel drive vehicle. In this example the driveline is modelled using the 1D flexible shaft for the propshaft and driveshafts, the differential is modelled using MultiBody components and is connected to the appropriate shafts via the Shaft1D_MBS, see Figure 14.

The Shaft1D_MBS model is used to couple 1D rotating components directly to components in the MultiBody library. This component relates the rotational speed and torque in the 1D connector to the speed and torque on the specified axis in the MultiBody connector.

The differential component is modelled using the MultiBody library and reacts the torques in the driveline onto the differential mount points. The actual differential mounts form part of the chassis subsystem and are discussed below. The differential includes the rotating inertias of the various internal components, backlash referred to the diff input and the mass, inertia and geometry of the complete differential assembly.

The type of mounts typically used to suspend the powertrain within the vehicle are designed to react forces in the x, y and z directions and they leave the powertrain free to rotate. These have been modelled using a series of three ActuatedPrismatic joints that are used to react the forces applied to the mount in three directions. A spherical joint is used at the side of the mount

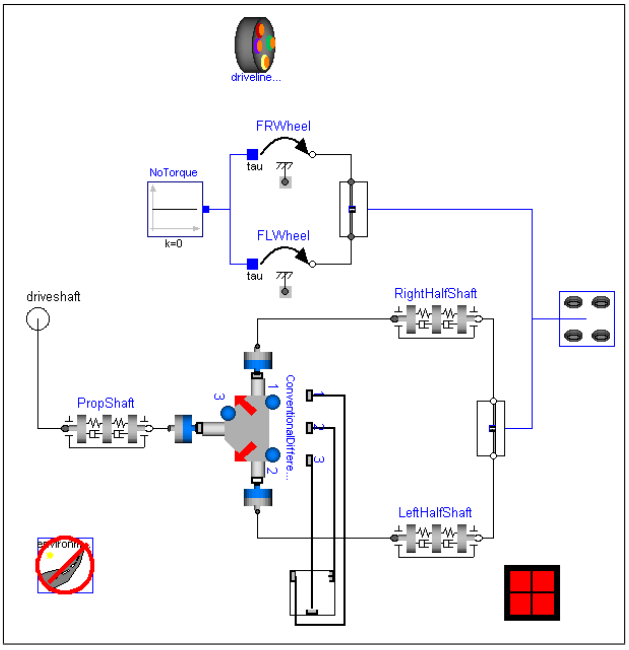


Figure 14: Driveline model that includes the movement of the differential on its mounts

that should be connected to the driveline component being suspended.

A number of tyre slip models have been included in the new version of the PowerTrain library. The available slip models include a simple linear slip model, a Pacejka slip model and the Rill slip model. These have been implemented for longitudinal slip only and consider the vertical load acting on the tyre.

4.2 4wd drivelines

A growing number of vehicles are being developed with all-wheel drive, e.g. sports utility vehicles (SUV) and commercial vehicles. There are a wide-range of

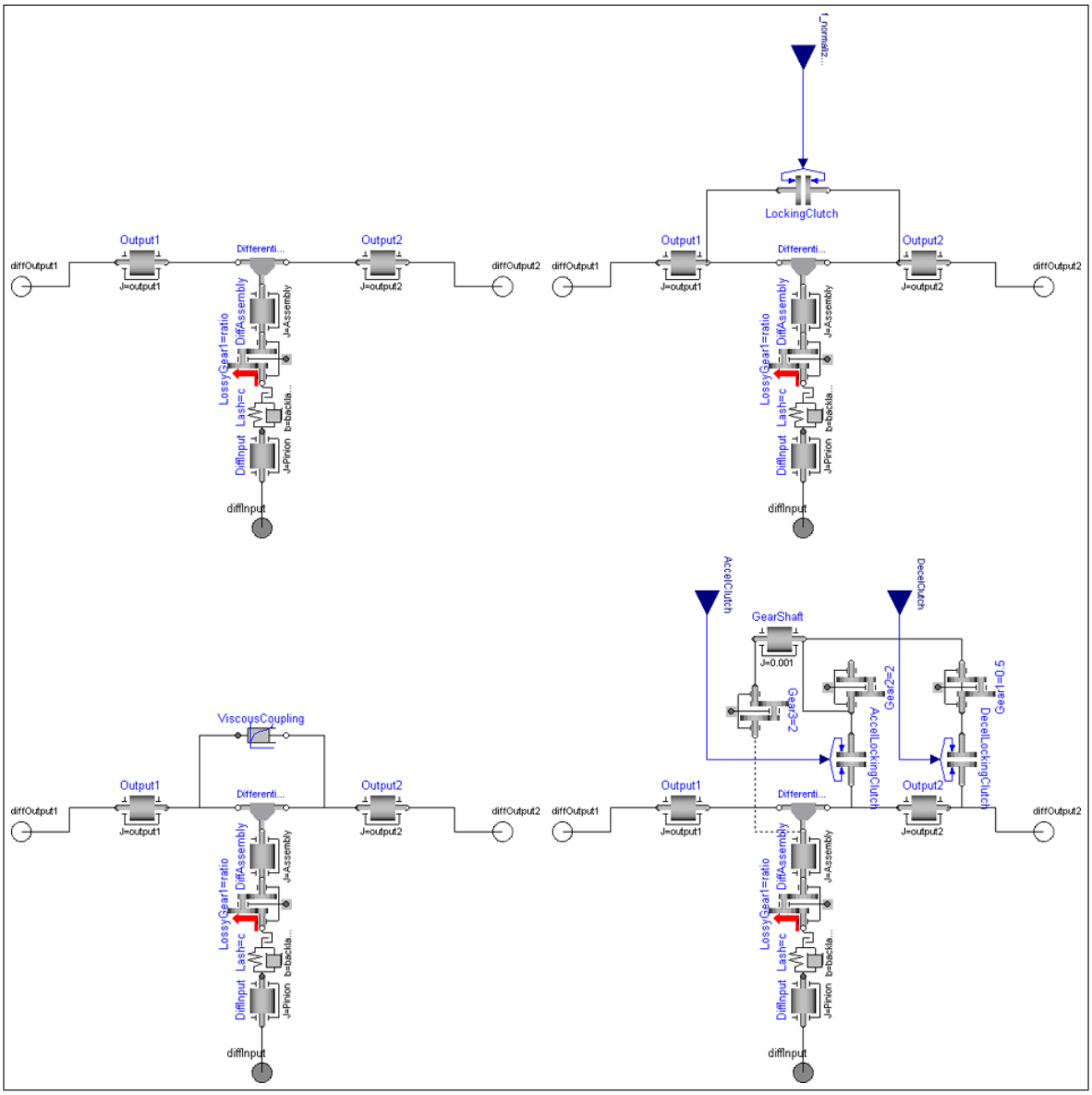


Figure 15: Some of the differential models available in the PowerTrain library. Clockwise from the top-left, conventional differential, simple active differential, torque vectoring differential, viscous differential

possible ways to deliver an all-wheel drive vehicle and components have been included to enable the modelling of the most common types and some of the most advanced. Available models include simple open differentials, viscous differentials, and two types of active differential. The various differential models have all been implemented as 1D rotational systems with only the conventional differential described previously using a MultiBody approach.

All the differential models provided are based around the use of an epicyclic differential unit. The different configurations of active and passive locking mechanisms are then placed around this core epicyclic unit and work in different ways to control the behaviour of the differential. Figure 15 shows four of the differential models available.

For the simple active differential and the torque vectoring differential control systems have also been provided. An example of a four-wheel drive vehicle, that uses three of these simple active differentials, has been added to the library. In this case, the control system has been designed to control each differential separately with the sole objective being to maximise traction. Each differential controller looks at the output shaft speeds from its differential and acts to reduce the difference in speed. It should be possible for some slip to occur between the shafts to allow for cornering and this can be defined through the controller parameters.

In addition to the range of differential models, a power take-off (PTO) style transfer box has been provided. In

some four wheel drive applications this type of transfer box is used instead of a centre differential. The key difference between using a differential and a PTO style transfer box is that the ratio between the input and each output shaft is fixed in the PTO style box whereas this ratio can vary when a differential is used.

4.3 Hybrid vehicles

With several automotive manufacturers and suppliers working on hybrid vehicles, there was a necessity to provide corresponding models in order to support concept studies in this area. Models have been included for batteries, motors and the associated controllers to meet this need.
The objective is to deliver models suitable for concept study work so that minimal data is required to develop a working model of a hybrid concept and to test out the functionality. Two hybrid vehicle examples have been included, one based on a vehicle using an Integrated Starter-Generator (ISG) and another based on a series-parallel hybrid style vehicle similar to the *Toyota Prius*. Both examples have been configured to run drive cycle simulations.
The battery model included in the PowerTrain library is based on the Saft capacitance model, which was originally developed in P-Spice [4] and has also been used in the Advisor [1] simulation tool. Figure 16 shows the circuit diagram used for the battery model. Capacitor *Cb* is very large and represents the ability of the battery to store charge chemically, the capacitor *Cc* is small and represents the surface effects of a spiral-wound cell. The three resistances represent the terminal resistance (*Rt*), end resistance (*Re*) and capacitor resistance (*Rc*).

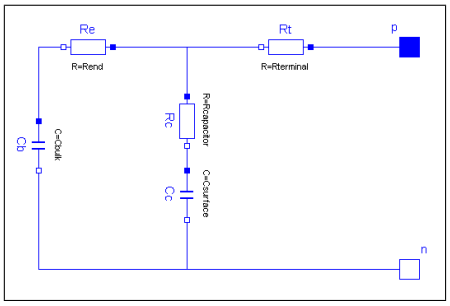


Figure 16: Circuit diagram of the battery model

The power electronics required to transfer energy from the battery to the electric motor have been simplified so that the simulation performance is maintained. An accurate model of the power electronics would require a large number of high frequency effects to be modelled and this would limit the suitability of the library for concept studies.

5 Other enhancements

This section describes the driver models, which have been added to the library, and changes concerning tables.

5.1 New driver models

The range of driver models provided with the PowerTrain library has been expanded to cover a wider range of tests. In addition to the existing cycle driver there are now driver models designed to carry out performance tests and driveability tests. There are also variants for use with both manual and automatic gearboxes.
The cycle driver models are based around the use of a PI controller that actuates either the brake or accelerator pedal to control the vehicle speed so that it follows a defined speed-time profile. A number of drive cycles are included by default such as the NEDC, EPA City and Highway cycles. It is possible to define your own additional drive cycles for use with the driver model. By varying the PI gains, the behaviour of the driver can be altered allowing the driver model to be tuned to match a range of different driving styles. The version of the cycle driver used with manual gearboxes also controls the clutch pedal and gear lever. The shift points are usually defined in the drive cycle to occur at particular points in time and driver starts to change gear at these points.
The driveability driver models are used to perform tip-in and tip-out tests in fixed gears, or fixed gearbox mode in the case of automatic transmissions. The tests start with the driver controlling the vehicle speed to an initial value and then accelerating and decelerating the vehicle between defined speeds using only the throttle. The brakes will not be used to decelerate the vehicle. For manual gearbox vehicles it is normal to define the tip-in and tip-out speeds as engine speeds. Due to the effect of the torque converter, it is more usual to define the tip-in and tip-out speeds using vehicle speed for automatic gearbox equipped vehicles.
The performance driver is used to perform standing start acceleration tests. The version used with automatic gearboxes can perform both an idle start or stall start acceleration test. In both versions the accelerator pedal position for the acceleration test can be defined so it is possible to assess the part-throttle accel-

eration performance as well as the wide open throttle (WOT) performance. The version used with manual gearboxes will change gear when a defined engine speed is reached. If in-gear acceleration times are required the driveability driver model should be used and the tip-in and tip-out speeds set to be the minimum and maximum speeds for the given gear.

5.2 Replaceable tables

All tables in the library have now been declared as replaceable. This change was made as many customers do not often have data in a form that is compatible with the tables in the Modelica standard library. Instead, they are forced to use their own, proprietary data format and their own table implementations. It was difficult for them to use these in combination with the PowerTrain library in the past.

6 Conclusions and Outlook

Version 2.0 of the PowerTrain library offers several new features, which open many new applications. New Modelica language elements allow a clean implementation of the new features and make it easier for users to adapt the library to their own specific needs.

Acknowledgements

For fruitful discussions, the authors would like to thank Jochen Köhler and his colleagues from ZF Friedrichshafen AG.
In addition, thanks to Michael Tiller from Ford Motor Company for pushing forward the concept of expandable connectors.
This work was in parts supported by *Bayerisches Staatsministerium für Wirtschaft, Infrastruktur, Verkehr und Technologie* under contract AZ300-3245.2-3/01 for the project *Test und Optimierung elektronischer Fahrzeug-Steuergeräte mit Hardware-in-the-Loop-Simulation* in the years 2001–2003.

References

[1] *ADVISOR*. URI `http://www.ctts.nrel.gov/analysis/advisor.html`.

[2] J. ANDREASSON, *VehicleDynamics library*, in Proceedings of the 3rd International Modelica Conference, Linköping, Sweden, November 2003, Modelica Association and Linköping University, pp. 11–18.

[3] H. ELMQVIST, S. E. MATTSSON, H. OLSSON, J. ANDREASSON, M. OTTER, C. SCHWEIGER, AND D. BRÜCK, *Realtime Simulation of Detailed Vehicle and Powertrain Dynamics*, in Electronics Simulation and Optimization (SAE 2004 World Congress), Detroit, USA, March 8–11, 2004, SAE International. Document Number: 2004-01-0768.

[4] V. H. JOHNSON, A. A. PESARAN, AND T. SACK, *Temperature-Dependent Battery Models for High-Power Lithium-Ion Batteries*, in 17th Electric Vehicle Symposium, Montreal, Canada, October 16–18, 2000.

[5] D. MAUERMANN, *Echtzeitsimulation detaillierter Fahr- und Antriebsstrangdynamik*, diploma thesis, Hochschule für Technik, Wirtschaft und Kultur Leipzig (FH), Fachbereich Elektrotechnik und Informationstechnik, July 2004. Compiled at Deutsches Zentrum für Luft- und Raumfahrt e. V.

[6] M. OTTER, M. DEMPSEY, AND C. SCHLEGEL, *Package PowerTrain. A Modelica Library for Modeling and Simulation of Vehicle Power Trains*, in Proceedings of the 1st Modelica Workshop, Lund, Sweden, October 2000, Modelica Association, pp. 23–32.

[7] M. OTTER, H. ELMQVIST, AND S. E. MATTSSON, *The New Modelica MultiBody Library*, in Proceedings of the 3rd International Modelica Conference, Linköping, Sweden, November 2003, Modelica Association and Linköping University, pp. 311–330.

[8] C. PELCHEN, C. SCHWEIGER, AND M. OTTER, *Modeling and Simulating the Efficiency of Gearboxes and of Planetary Gearboxes*, in Proceedings of the 2nd International Modelica Conference, Oberpfaffenhofen, Germany, March 2002, Modelica Association and Institute of Robotics and Mechatronics, Deutsches Zentrum für Luft- und Raumfahrt e. V., pp. 257–266.

[9] C. SCHWEIGER AND M. OTTER, *Modelling 3D Mechanical Effects of 1D Powertrains*, in Proceedings of the 3rd International Modelica Conference, Linköping, Sweden, November 2003, Modelica Association and Linköping University, pp. 149–158.

[10] C. SCHWEIGER, M. OTTER, AND G. CIMANDER, *Objektorientierte Modellierung mit Modelica zur Echtzeitsimulation und Optimierung von Antriebssträngen*, in Steuerung und Regelung von Fahrzeugen und Motoren – AUTOREG 2004, VDI/VDE-GMA, ed., no. 1828 in VDI-Berichte, Düsseldorf, Germany, März 2004, VDI-Verlag, pp. 639–650.

[11] M. TILLER, P. BOWLES, AND M. DEMPSEY, *Development of a Vehicle Model Architecture in Modelica*, in Proceedings of the 3rd International Modelica Conference, Linköping, Sweden, November 2003, Modelica Association and Linköping University, pp. 75–86.