Utilizing Object-Oriented Modeling Techniques for Composition of Operational Strategies for Electrified Vehicles

Sebastian Hämmerle
Vorarlberg Univ. of Appl. Sc.
Austria
Sebastian.Haemmerle@students.fhv.at

Marco Kessler
Modelon GmbH
Germany
Marco.Kessler@modelon.com

Markus Andres
Modelon GmbH
Germany
Markus.Andres@modelon.com

Abstract

The paper introduces a new concept of modeling the overall control unit of hybrid electric vehicles in *Modelica*. The work focuses on a structure which can simulate substantially different vehicle concepts without changing the structure of the control unit. Based on this universal implementation different scenarios can be simulated rapidly and consequently cheaply, including fundamentally different drive trains ranging from conventional to purely electrical including hybrid versions.

Keywords: object-oriented control unit modeling; control unit template modeling; object-oriented electrified vehicle modeling

1 Introduction

Ever increasing demands on fuel efficiency are requested from both politicians and customers. This should happen without increasing the costs and without compromising on performance of the vehicle [2]. Therefore, simulation in the concept stage of a vehicle development is getting more and more important, because with the aid of simulation design decisions can be supported in a more efficient overall process that is consequently resulting in cost savings [4]. This is especially true when designing complex systems which are state-of-the-art in the form of hybrid electric vehicles (HEV).

Nowadays, several different HEV topologies are state-of-the-art, for example series hybrid, parallel hybrid or power split hybrid. Furthermore, from every topology multiple modification levels are available. Until now, if a rough layout of a HEV has to be simulated, a very specific CU (control unit), specially adapted to each different vehicle topology has been modeled. This process is error-prone, time consuming and in turn associated with high costs. Due to these facts the manufacturers postulate the development of a universal CU for simulations.

Implementations of individual HEV topologies have been discussed many times in literature, but the implementation of a standardized template for a universal CU has not yet been provided.

Examples are given in [8] and in [9], where fuel consumption simulations - on basis of two commercial *Modelica* libraries: the *PowerTrain* library [5], [7] and the *SmartElectricDrives* library [1] - of different HEVs are discussed.

Another example of a library for simulation of HEVs is given in [3]. The used library is called *eVehicleLib* and can be used for fuel and energy consumption simulations as well as other purposes.

Both, the *PowerTrain*- and the *eVehicleLib* library contain models for process control, i.e. for the operating strategy but no template of a universal CU for different vehicle topologies. The concept presented in the paper fulfills this request of the manufacturers.

The paper focuses on HEV structures, what is reasonable as purely electric or conventional drive trains may be treated as a simplification of suitable hybrid versions. An example will be demonstrated in Section 5 where the hybrid car (BMW i8) is simulated as a conventional vehicle by just changing controller parameters.

2 Hybrid Electric Vehicles

Basically, HEVs can be classified on either the type of the power train or by their degree of hybridization. Speaking of type of power train, one can distinguish between series-, parallel- and power-split hybrids. The classification by degree of hybridization contains micro, mild and full hybrid.

Due to the fact that a HEV, in contrast with a conventional vehicle, has two energy converting systems -

DOI
10.3384/ECP1409679

usually a combustion engine (CE) and an electric machine (EM) - more degrees of freedom in generating traction energy are available. Which possibilities are available depends on the arrangement of the CE and the EM in the drive line, i.e. on the HEV topology [2].

3 Modeling Approach

To design a template of a universal CU, which can be used to simulate different kinds of HEVs, it must first be understood how the CU interacts with the remainder of the entire vehicle system. Therefore, a decomposition of an entire vehicle system into basic objects is necessary. The first division is done into four main parts: a drive environment, an environment, a vehicle itself and the control unit.

The drive environment emulates the human driver, i.e. it influences the vehicle in longitudinal direction with the accelerator and brake pedal (lateral direction is neglected). The schematic function of the drive environment is illustrated in Figure 1. Depending on a reference velocity (supplied by the driving cycle) and an actual velocity (supplied by the vehicle) the accelerator- and brake pedal position is computed by the driver. These signals are sent to the CU which computes signals for the vehicle. Therefore, a direct interaction between the CU and the drive environment is given.

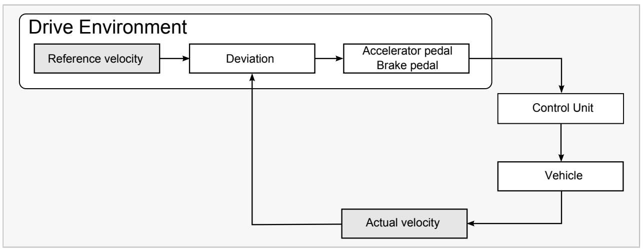

Figure 1: Schematic structure of the drive environment

In the vehicle it is distinguished between objects which are required for generating traction energy and the ones which are not, i.e. the vehicle is grouped into a drive line and auxiliary units, see in Figure 2. The CU provides on the one hand signals for the drive line, e.g. torque requirements for the CE and the EMs and on the other hand for the auxiliary units, e.g. the available power for the auxiliary units. Furthermore, sensor signals from the drive line are necessary in the CU, e.g. the actual vehicle speed. Therefore, the CU has a direct interaction between the objects from the vehicle.

The environment includes all effects from outside of the entire vehicle system and provides it to the vehicle, see Figure 3. The CU requires signals from the environment which as well influence the vehicle and therefore, an indirect interaction between the CU and the environment exists.

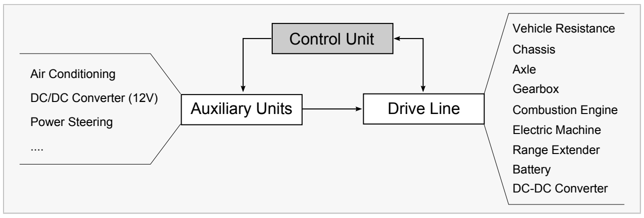

Figure 2: Schematic structure of the Vehicle

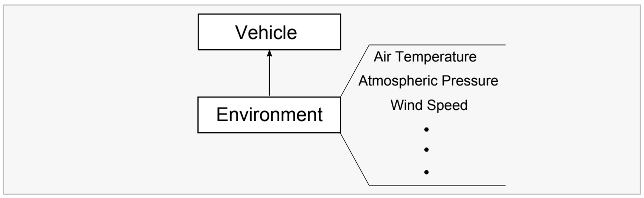

Figure 3: Schematic structure of the environment

Figure 4 shows the decomposition of the entire system in summarized form, whereas the interaction between the objects is indicated by arrows.

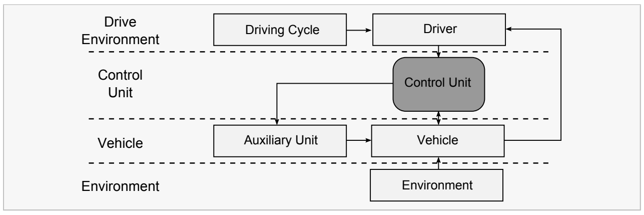

Figure 4: Schematic structure of an entire system

The interaction between the CU and the rest of the entire vehicle system is clearly defined. Now, the decomposition of the CU has to be carried out in a way that it can be reused for different HEV topologies. To make the structure flexible enough a controller with two levels is introduced (see Figure 5). In the primary layer the HCU (hybrid control unit) determines the current driving mode. Depending on the output of the primary layer the second level of hierarchy computes the desired signal for the corresponding drive line component (see Figure 5).

- BCU (brake control unit)
- ESCU (electric storage control unit)
- CCU (clutch control unit)
- GBCU (gearbox control unit)

- ECU (engine control unit)
- EDCU (electric drive control unit)

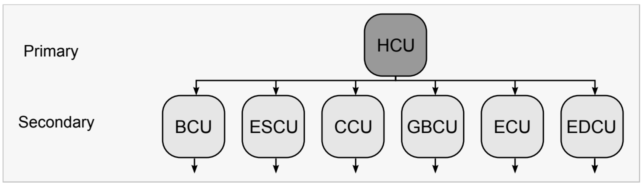

Figure 5: Schematic structure of the CU

To test the controller three parts are necessary, of which the first is to design a template for the overall system, which is shown in Figure 6. For this purpose a library with basic components was created to replace the corresponding parts of the template. As the focus of this paper is to present the structure of the controller the models for testing (*Drive Environment*, *Vehicle* and *Environment* in Figure 6) are not discussed here in more detail. Moreover a considerable amount of attention was paid to the fact that well defined interfaces of each model were used in the template and therefore every component model can be adapted or changed easily, thanks to Modelica's object-orientation. In the following section we will focus on the implementation of the universal control structure that is shown in the *Control Blocks* part of Figure 6.

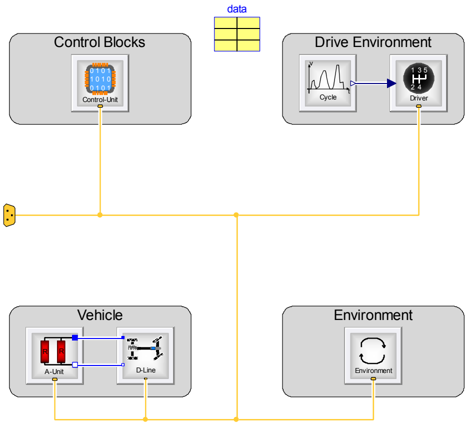

Figure 6: Template of the entire system

4 Universal Control Unit Modeling

Manufacturers tend to protect the control strategy as trade-secret because it represents one of their core knowledge. Therefore, the implementation of each part of the CU must be created in a way that it can easily be adapted and/or replaced. The structure of the implementation of the CU is based on Figure 5 and its implementation is shown in Figure 7.

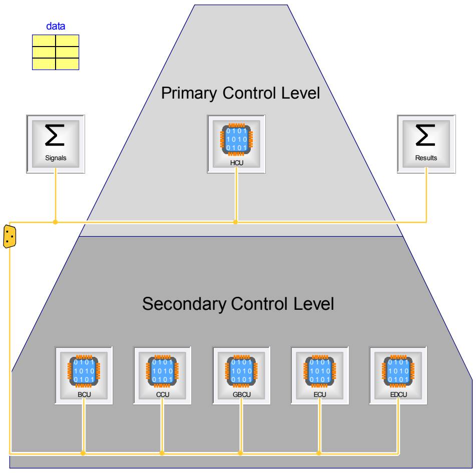

Figure 7: Template of the universal CU

4.1 Bus Structure

Looking at Figure 7 one can easily see, that the implementation is heavily relying on a bus structure. It was applied to the massive amount of control signals within the controller that have to be dealt with. Moreover it enhances the flexibility and maintainability of the overall controller. To reduce errors within connecting signals from the bus and as well enhancing maintainability connectors are introduced that only feed values from the bus to a signal connector of suitable type (Real, Integer, Boolean). Although those adapter models are very simple they are valuable in certain cases. When e.g. a variable on the bus is renamed only the connector has to be modified in contrast to every single connection relying on that variable.

In the current implementation the bus has a flat structure without any grouping of variables. After it turned out that the bus has close to 100 variables it would be well worth the effort to restructure the bus with a hierarchy related to e.g. Figure 4.

4.2 Primary Control Level

The primary control level solely consists of the HCU as shown in Figure 7. It is the centerpiece of the CU as it supervises all the other control units of the secondary control level. To implement a universal HCU it is necessary to consider all possible driving modes of all HEV topologies. These are

1. Standstill with CE off (`StandStill`)

2. Standstill with CE on (`StandStillEngine`)
3. Electric Driving (`ElectricDriving`)
4. Friction braking (`Braking`)
5. Regenerative braking (`RegenerativeBraking`)
6. Boosting (`Boosting`)
7. Shifting of the Operation Point (`Shifting-OperationPoint`)
8. Conventional Driving (`ConventionalDriving`)

whereas the names in the brackets denote the variable names used in the model itself.

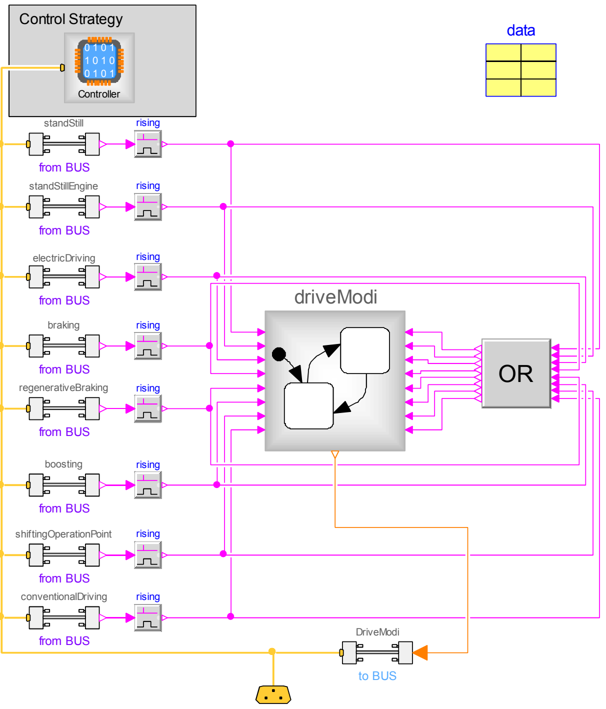

Figure 8: Implementation of the HCU

The HCU (Figure 8) has to decide which of those driving modes is active e.g. depending on driver inputs or internal states of the vehicle. The first prerequisite to be able to take that decision is to know which of those driving modes are available in the current vehicle configuration. Therefore parameters to activate driving modes independently of each other are provided in the `data` record shown in Figure 8.

In the current implementation independent models are used to decide which driving mode is active as shown in Figure 9. All of them have Boolean outputs each one representing a possible driving mode. This implementation guarantees maximum flexibility but is not the most straight-forward implementation.

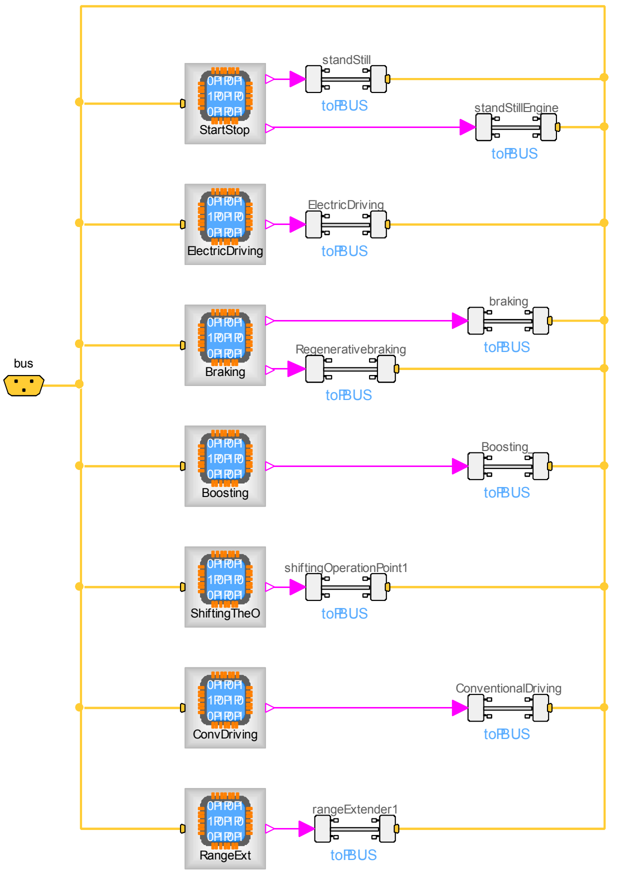

Figure 9: Implementation of the Control Strategy within the HCU

To have distinct behavior it has to be ensured that in each time step only one driving mode is active within the HCU. To ensure that, the modeling is done via the *state graph 2* library[1]. The implementation shown in Figure 10 is a combination of eight parallel loops, one for each driving mode. This model is used named *driveModi* in Figure 8. The loops consist of an activate and a deactivate transition and therefore of an on and an off step. The output of the HCU is an integer variable that is put to the bus indicating the current drive mode. This drive mode is then utilized by the secondary level controllers (Figure 7) to generate input signals for the drive train components. As a result of this structure it is not only possible to simulate HEVs of different structure, but also purely electric vehicles and conventional vehicles. Therefore, the modeler can simply choose a hybrid vehicle topology and based on that the possible and desired driving modes are enabled in the HCU.

The range extender operation is an exception to that rule. This mode can be interpreted using a source of power for the battery without any power directly

[1] For further information it is referred to [6]

transferred to the power train. Alternatively it could be seen as part of the auxiliary units as it does not generate traction force. It can therefore not be integrated in the integer variable that represents the driving mode, as some other driving mode has to be active to compute the control signals for the components of the drive train. Then the integer value for the driving mode would have to take two values. It is therefore represented by an additional Boolean variable on the bus.

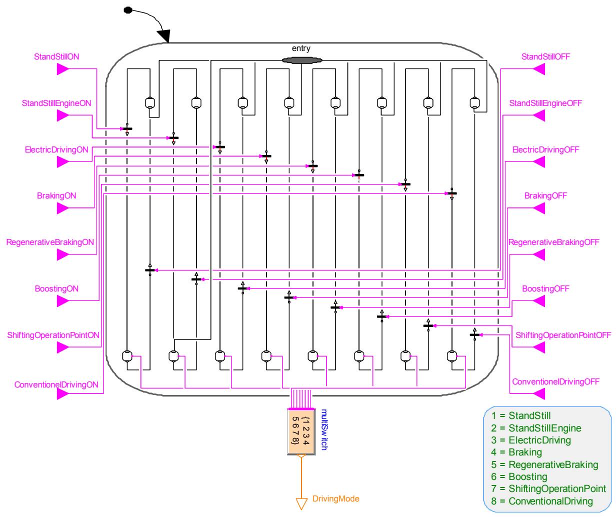

Figure 10: Driving mode selection via state graphs

Consequently the driving modes are represented by an integer variable (driving mode 1 - 8), which is computed via the state graphs and a Boolean variable (driving mode 9), which is computed in the Controller block.

Modifications

The driving modes are simple to adapt or to replace as only the Boolean output has to be fed with a meaningful value. If an additional driving mode should be implemented a new driving mode controller in the *Controller* block and an additional loop in the *DriveModes* block has to be implemented.

4.3 Secondary Control Level

In the following sections the second layer controllers are shown starting with the engine control unit that is discussed in a bit more detail whereas the others shall be described only briefly.

4.3.1 Engine Control Unit

The ECU is responsible to calculate the normalized torque request of the CE. Furthermore, the signals for the starter generator and the signals to bring the CE into idle mode or switch it off are computed in this controller. The main areas of functionality are shown on the left side of Figure 11. These are

- Conventional Driving
- Shifting the Operating Point and
- Boosting

represented by different paths through the model within the left three boxes in the model.

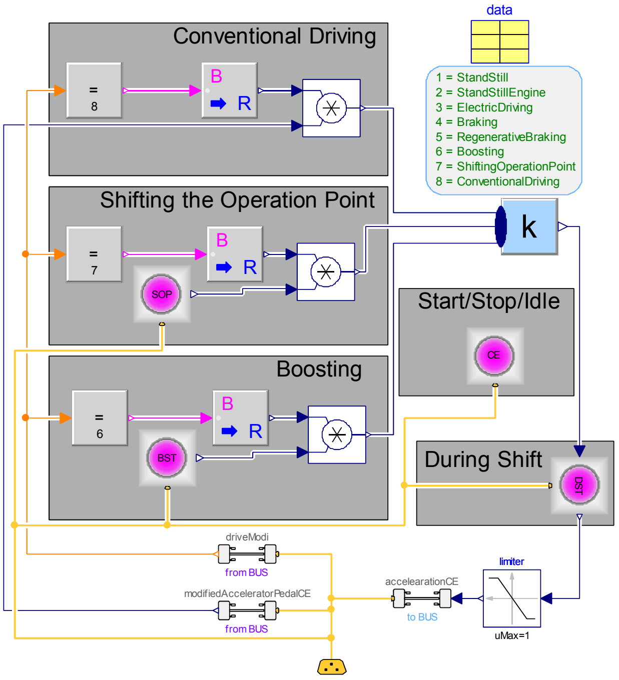

Figure 11: One possible controller of the combustion engine

They compute an output signal that represent the combustion engine's torque demand depending on the variable *driveModi* that is taken from the bus via the adapter shown in the lower part of Figure 11. The combination of compare block, Boolean to real converter and multiplication with the output basically act as an activation and deactivation of the single path. The *SOP* and *BST* blocks represent implementations of the functionality indicated by their name.

The additional blocks for *Start/Stop/Idle* and *During Shift* take responsibility that the CE acts like one would expect during the operation states indicated by their name.

Flexibility is granted by the possibility to replace the blocks for *SOP*, *BST* etc. or by replacing the overall *ECU* (Engine Control Unit) within the *CU*. Which possibility is more suitable has to be decided within the application that is targeted.

4.3.2 E-Drive Control Unit

The EDCU is responsible to calculate the normalized torque requests of the EMs. Furthermore, in this controller it is ensured that the computed signals are sent to the corresponding EM (in case more than one EM is available).

The implementation of the EDCU is split into different areas, i.e. one for each driving mode where an EM is involved. By this structure it is clearly defined how the normalized torque request signal for the EM is calculated in every driving mode. Moreover, by this implementation there is a clear separation enabling simple replacement of strategies depending on driving modes.

Due to the bigger amount of driving modes that the electric drives are included compared to the combustion engine the model gets more complex than the one in Figure 11 as one can see from Figure 12.

4.3.3 Gearbox Control Unit

The task of the GBCU is to evaluate the moment of shifting a gear up or down. In a real vehicle the shifting is dependent on different parameters, e.g. vehicle speed, slope of the road, actual gear level. To simplify the implementation it is assumed that the shifting is only dependent on the actual vehicle speed, i.e. the gear shift depends on predefined threshold values. The output is the actual gear level and is represented by an integer variable.

4.3.4 Clutch Control Unit

The CCU is responsible for opening and closing the clutches. The implementation is similar to the EDCU and ECU. That means that the CCU is split into different areas and, therefore, it is easy to adapt, expand or replace the controller.

4.3.5 Brake Control Unit

The BCU has the task to split the required brake torque into a recuperation brake and a friction brake amount. Two different regenerative braking strategies are implemented: the series-[2] and the parallel[3] recuperation strategy.

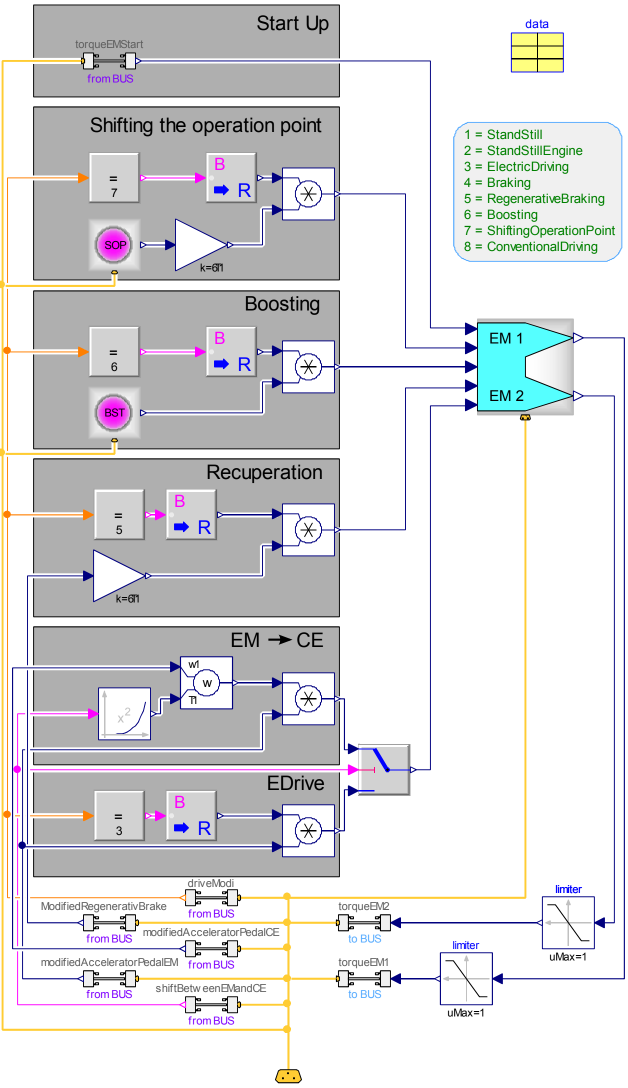

Figure 12: One possible controller of the electric machine

4.4 Other Models

In Figure 7 models besides the primary and secondary control level. These will be discussed in this section.

4.4.1 Signals

For simulation of a conventional vehicle the accelerator pedal position can be directly used to scale the

[2] As long as the required deceleration is not higher than the provided generator deceleration, the generator brake is solely used. When the required deceleration increases, the friction brake is added.

[3] The generator and the friction brake are acting together in a fixed ratio.

maximum torque of the CE. In comparison to that, HEVs have at least two energy converting systems. Therefore the accelerator pedal position must be used to scale the maximum available torque for forward motion, which in most cases results from a combination of the torques of the CE and the EM.

In this block the conversion of the accelerator- and brake pedal position into normalized input signals is implemented.

4.4.2 Results

The *Results* block is for the computation or display of values which are necessary for evaluating the simulation, for example the fuel consumption.

4.5 Auxiliary Units

The auxiliary units are all power units, which are not primarily needed for generating traction energy. Therefore they are not part of the HCU directly as it focuses on components for traction energy generation. Still in modern vehicles a multitude of auxiliary units can be found, e.g. an air conditioning system or a lube oil pump. The power controlling of the auxiliary units is realized via a priority list, i.e. the available electric power is shared depending on the driving mode.

Figure 13 illustrates the schematic function of the priority list. The required power from the drive line and from the auxiliary units is sent to the priority list block. This block splits the available power depending on the drive mode and sends the results to each block.

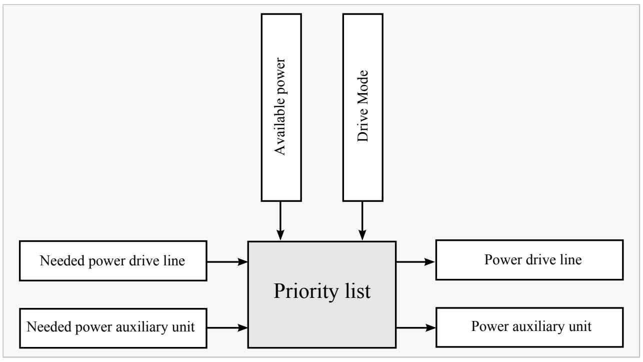

Figure 13: Schematic function of the priority list

5 Simulation

To demonstrate the universal applicability of the designed controller, two different vehicles are simulated. One target of simulations like these could be to judge their fuel consumption during common drive cycles. The intention was to pick one vehicle with very high complexity, which is why the BMW i8[4] was chosen. Additionally the BMW i8 was operated as it is intended to and as a conventional vehicle by deactivating the hybrid operation modes. The BMW i3[5] was picked due to its very different structure that has to be covered with the same controller.

The driving modes in shown the simulation results in Figures 15, 16 and 18 are referring to the ones listed in Section 4.2.

It is important, that not all parameters for the models were exactly known which make the absolute results differ from real world values. Still relative comparisons based on different operating strategies are feasible and demonstrate the flexibility of the designed controller.

5.1 BMW i8

It is exemplarily shown how a comparison between the fuel consumption of the BMW i8, used as a conventional vehicle, and the BMW i8, used as an HEV could be made (driving cycle: NEDC).

At first a fuel consumption simulation of the BMW i8 as a conventional vehicle is run. For that solely the conventional driving mode is enabled in the HCU, i.e. only stand still with engine on, friction braking, and conventional driving are possible.

The second step is to run the fuel consumption simulation of the BMW i8 as an HEV. Therefore, the suitable driving modes are enable in the HCU, i.e. start-stop automatic, regenerative braking, electric driving, boosting, shifting of the operation point and conventional driving.

The changes between the simulations come down to selecting the boolean parameters that enable or disable the driving modes mentioned above within the parameter window. No modifications of the single controllers in any level of hierarchy is necessary.

Result

The results of the fuel consumption simulations utilizing the NEDC are shown in Figure 14, whereby the dashed line represents the BMW i8 as a conventional vehicle and the constant line in the BMW i8 as a HEV. More important regarding this paper is how the controller behaves with respect to the driving modes that

[4] Is a special case of a parallel structure, called through the road hybrid or axle split hybrid.

[5] Is available as a purely electric vehicle or as series HEV

are activated. This is presented in Figure 15 and 16. In Figure 15 mainly the modes 1 (StandStill), 3 (ElectricDriving) and 5 (RegenerativeBraking) are active during the first 800 seconds of simulation. Afterwards the highway cycle starts, where 7 (ShiftingOperationPoint), and 8 (ConventionalDriving) are active.

If the BMW i8 is configured by a few clicks to be operated as a conventional vehicle only the modes 2, 4 and 8 are available resulting in Figure 16.

The fuel consumption of the BMW i8 as a conventional vehicle is 0.98 $\frac{1}{\text{NEDC}}$ (8.89 $\frac{1}{100\,\text{km}}$) and as an HEV 0.61 $\frac{1}{\text{NEDC}}$ (5.53 $\frac{1}{100\,\text{km}}$). In other words, the use of the BMW i8 as an HEV instead as a conventional vehicle effects a fuel saving of approximately 38 %.

BMW i3

This investigation focuses on the effect of a range extender (10 l tank volume) on the driving range of the BMW i3 (driving cycle: WLTP).

At first a driving range simulation with the BMW i3, used as a purely electric vehicle, is carried out. Therefore, the available driving modes are enabled in the HCU, i.e. start-stop automatic, electric driving and regenerative braking. Afterwards, the second drive range simulation is executed, with the difference that the range extender mode is additionally enabled in the HCU.

Result

The results of the driving range simulations are illustrated in Figure 17. The upper curve is driving range of the BMW i3 as a purely electric vehicle and the lower curve the driving range of the BMW i8 with an additional range extender.

The driving range of the purely electric vehicle is 112.4 km and with the additional range extender 287.7 km. This links to a surplus of driving range of approximately 175 km.

Figure 18 shows the driving modes that are active during the simulation of a WLTP cycle for the purely electrically driven BMW i3 without range extender. It shows that only the modes 1 (StandStill), 3 (ElectricDriving) and 5 (RegenerativeBreaking) are active in this case. The controller structure was the same as for the simulations carried out with the BMW i8, with necessary changes to the secondary level controllers due to other components used.

6 Conclusion

To summarize, the presented controller structure can be seen as a good starting basis for manufacturers for the simulation in the early design phase of a vehicle propulsion system. The presented implementation of a universal control unit provides a simple and time saving possibility to quickly simulate fundamental design changes in the system, which was the target for this development. Additionally the unification of the controller structure comes with other advantages like quicker orientation in non-familiar projects.

The presented implementation is in a prototype stadium and there are several parts that could be enhanced. These are e.g. the integration of the range extender mode that could be added to the auxiliary components instead of the driving modes. Another thing that should be reviewed thoroughly is the mixture of boolean and integer variables that are determine the driving mode. The current implementation goes for maximum flexibility but most likely more intuitive possibilities exist to solve this problem.

References

[1] J. V. Gragger, H. Giuliani, C. Kral, T. Bäuml, H. Kapeller, and F. Pirker. The SmartElectricDrives Library - Powerful Models for Fast Simulations of Electric Drives. *The Modelica Association*, 571-577, 2006.

[2] P. Hofmann. *Hybridfahrzeuge: Ein alternatives Antriebskonzept für die Zukunft*. Springer, 2010.

[3] S. Kutter and B. Bäker. eVehicleLib - Eine Modelica-Bibliothek zur Simulation von Fahrzeugen mit alternativen Antrieben. *Symposium Simulationstechnik - ASIM*, 2009.

[4] Q. Li. *Developing Modeling And Simulation Methodology For Virtual Prototype Power Supply System*. PhD thesis, Virginia Polytechnic Institute and State University, 1999.

[5] M. Otter, M. Dempsey, and C. Schlegel. Package PowerTrain. A Modelica library for modeling and simulation of vehicle power trains. *The Modelica Association*, 23-32, 2000.

[6] M. Otter and I. Dressler. Stategraph - A Modelica Library for Hierarchical State Machines. *The Modelica Association*, 569-578, 2005.

DOI
10.3384/ECP1409679

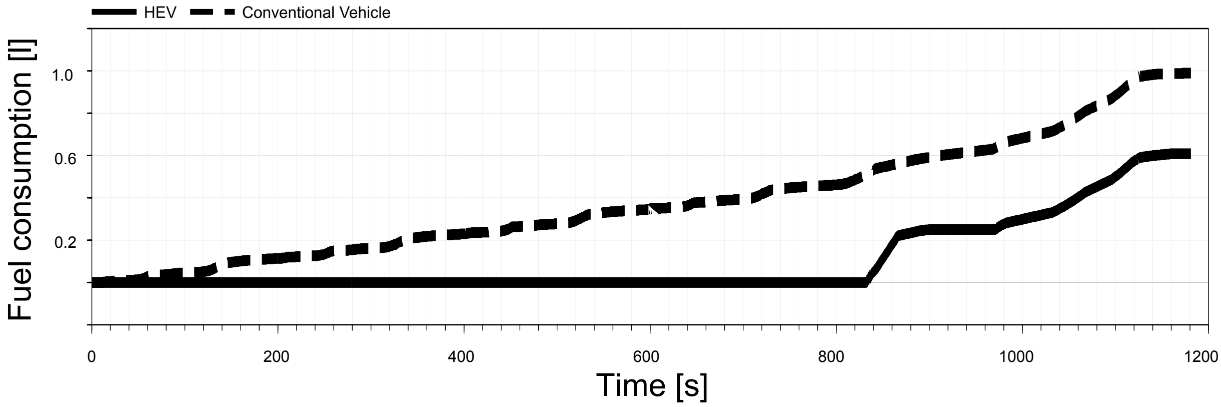

Figure 14: Fuel consumption

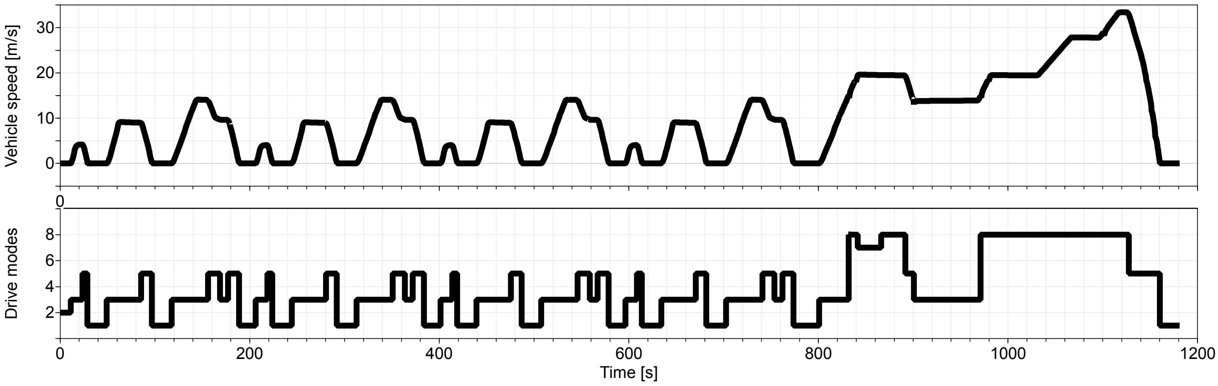

Figure 15: BMW i8 operated as a hybrid vehicle

[7] C. Schweiger, M. Dempsey, and M. Otter. The PowerTrain Library: New Concepts and New Fields of Application. *The Modelica Association*, pages 457–466, 2005.

[8] D. Simic and T. Bäuml. Implementation of Hybrid Electric Vehicles using the VehicleInterfaces and the SmartElectricDrives Libraries. *The Modelica Association*, pages 557–562, 2008.

[9] D. Simic, H. Giuliani, C. Kral, and J. V. Gragger. Simulation of Hybrid Electric Vehicles. *The Modelica Association*, pages 25–31, 2006.

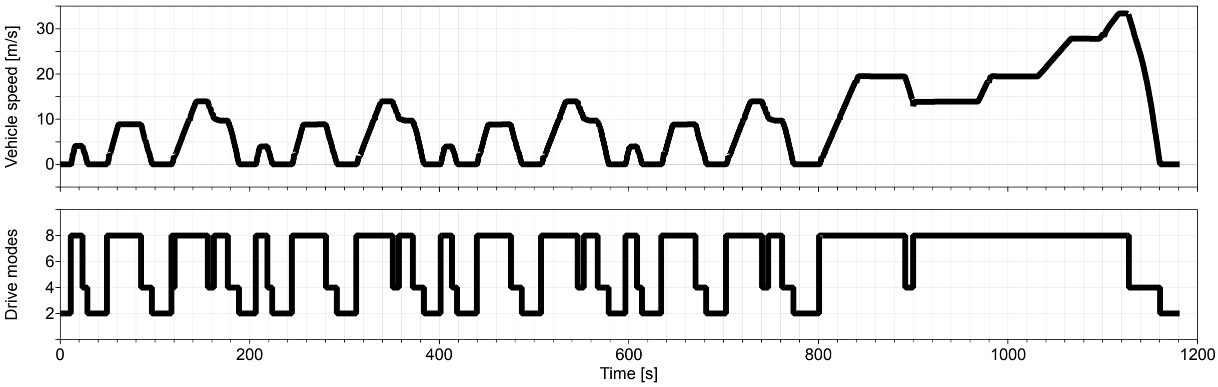

Figure 16: BMW i8 operated as a conventionally driven vehicle

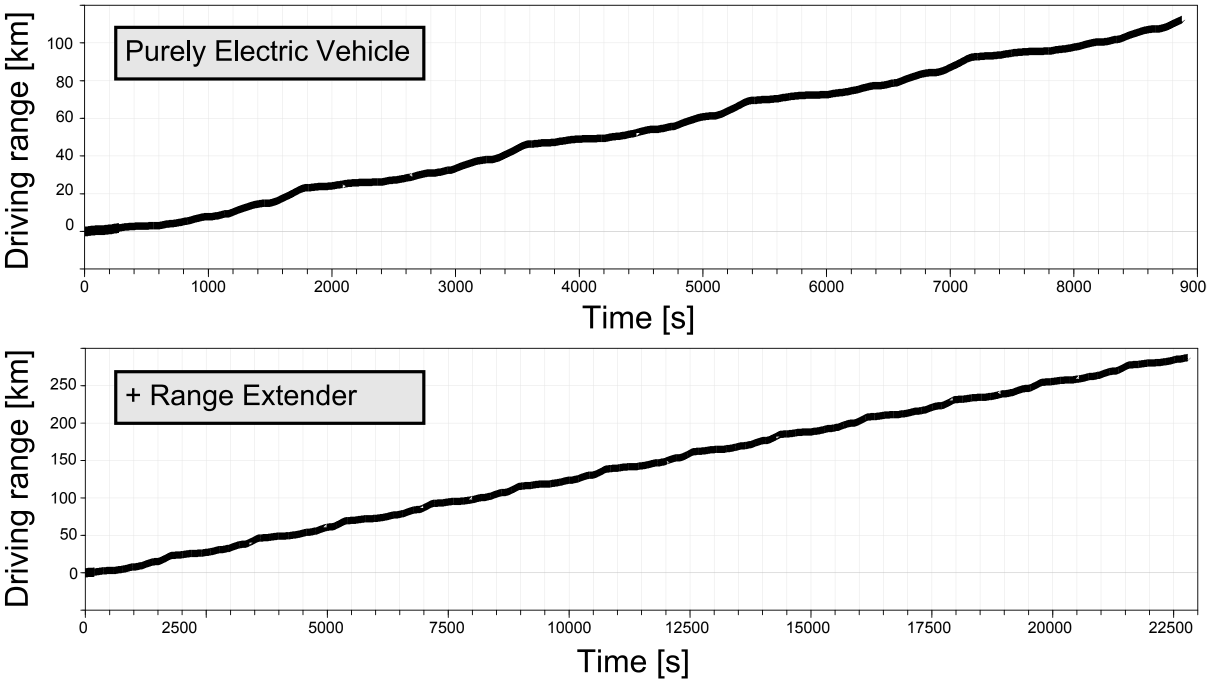

Figure 17: Driving range

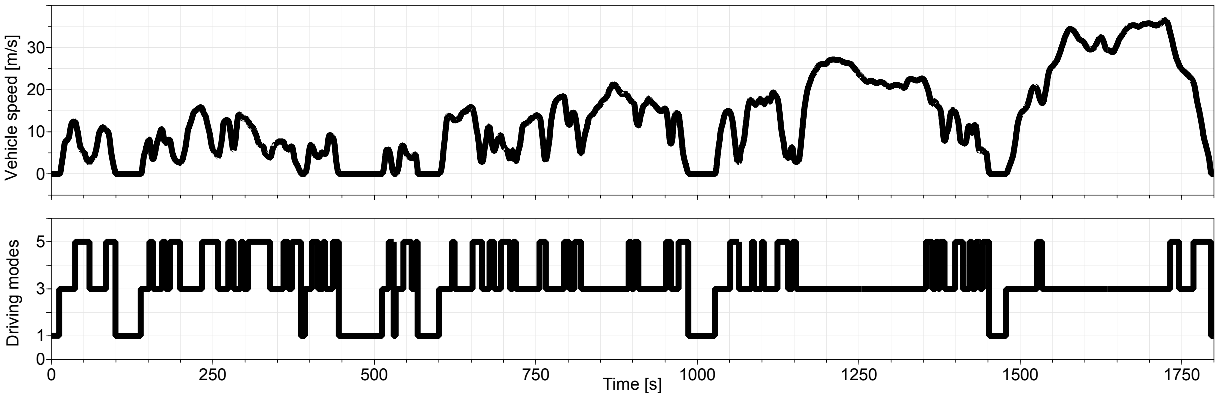

Figure 18: BMW i3 operated as a conventionally driven vehicle