© QTronic | Daimler

Simulation of Networked ECUs for Drivability Calibration

Drivability calibration of a vehicle's engine and transmission controller largely defines the unique character of the vehicle. Today, calibration is mostly performed on the road and on test rigs. Recent advances regarding virtualisation of ECUs made it possible to establish Software-in-the-Loop simulation as a development tool as well. To this end, Daimler AG and QTronic GmbH jointly conducted a project to push the limits of system simulation. The responsible engineers explain, why this requires the virtualisation of the entire engine controller, and closed-loop simulation of virtual ECUs and plant models. They also report how the ECUs have been virtualised and integrated with existing plant models.

AUTHORS

Dr. René Linssen
is Engineer at the Daimler AG
in Stuttgart (Germany).

Frank Uphaus
is Teamleader Drivability
Calibration at the Daimler AG
in Stuttgart (Germany).

Dr. Jakob Mauss
is Director of the QTronic GmbH
in Berlin (Germany).

CHALLENGE

Drivability calibration is system development. Drivability calibration depends like no other calibration discipline on the interactions of networked powertrain controllers (xCUs) and on the physical behaviour of the powertrain components. The diversity of subsystems that have to be modelled has posed a significant challenge to effective virtualisation of drivability calibration in the past. The bottleneck was the virtualisation of xCUs, not modelling the physical behaviour of the powertrain hardware. In particular the most complex subsystem, the engine controller (ECU) which holds software developed by the OEM and by the ECU-supplier has been widely recognised as a significant challenge for virtualisation.

However, full virtualisation of the powertrain as a system of powertrain hardware and xCUs is – besides innovative powertrain test rigs [1] – one of the few tools that have the potential to master the challenges created by ever growing complexity, limited resources and shrinking development times.

For this reason, Daimler AG and QTronic GmbH jointly conducted a project to push the limits of software in the loop (SiL) technology in order to establish system simulation as a tool for powertrain calibration.

This article reports project results on the simulation of networked xCUs for drivability calibration of Mercedes-Benz passenger cars. We explain why the virtualisation of the entire engine controller is required here, how this has been achieved, how the resulting virtual ECU has been integrated with plant models and conclude with an outlook on future work.

WHY SYSTEM SIMULATION?

The essential difference between system simulation as presented here and simulation for design and dimensioning of components and aggregates is the higher number of modules to be considered. A short example from drivability calibration shows how many modules are required to address a simple task in a virtual environment.

FIGURE 1 shows the initial phase of a full load acceleration of a vehicle with manual gearbox and diesel engine. The maneuver starts after the vehicle has been brought to a halt after phase 1. The transition between phase 1 (gray) and 2 (white) is determined by the change in accelerator pedal value as can be seen by the green line jumping from 0 % to 100 %. Caused by the increase in injected fuel and resulting effective torque, the engine revs up. The speed increase is limited by the amount of fuel injected due to smoke limitation constraints. This is a first example of coupling plant model and control software. This closed loop requires a model of the air path and the information of the pressure and temperature condition at the inlet valve for the function calculating the maximum allowed amount of fuel.

During the third phase (gray), the pedal value remains at 100 %, the

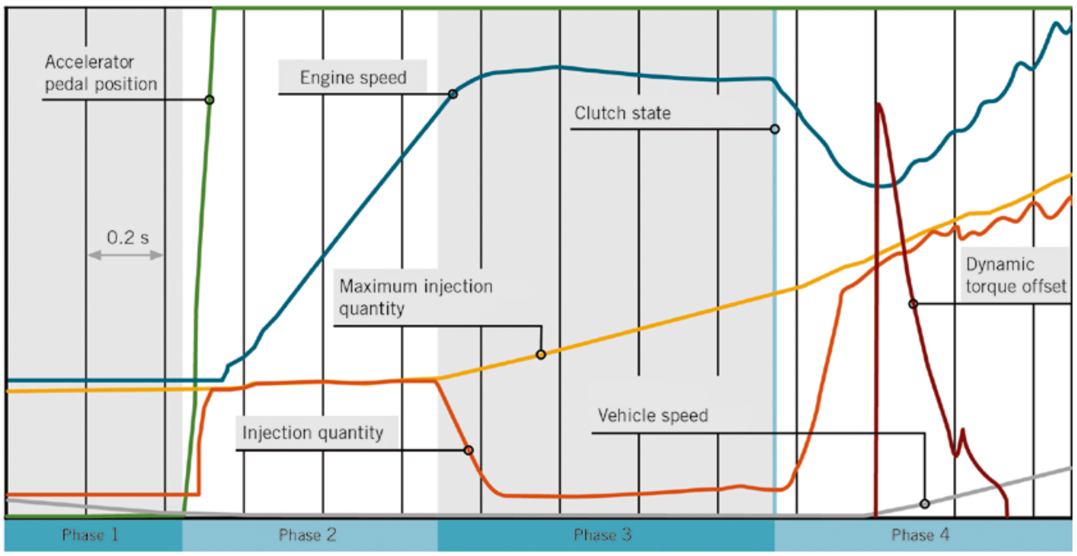


FIGURE 1 Initial phase full load acceleration (© Daimler)

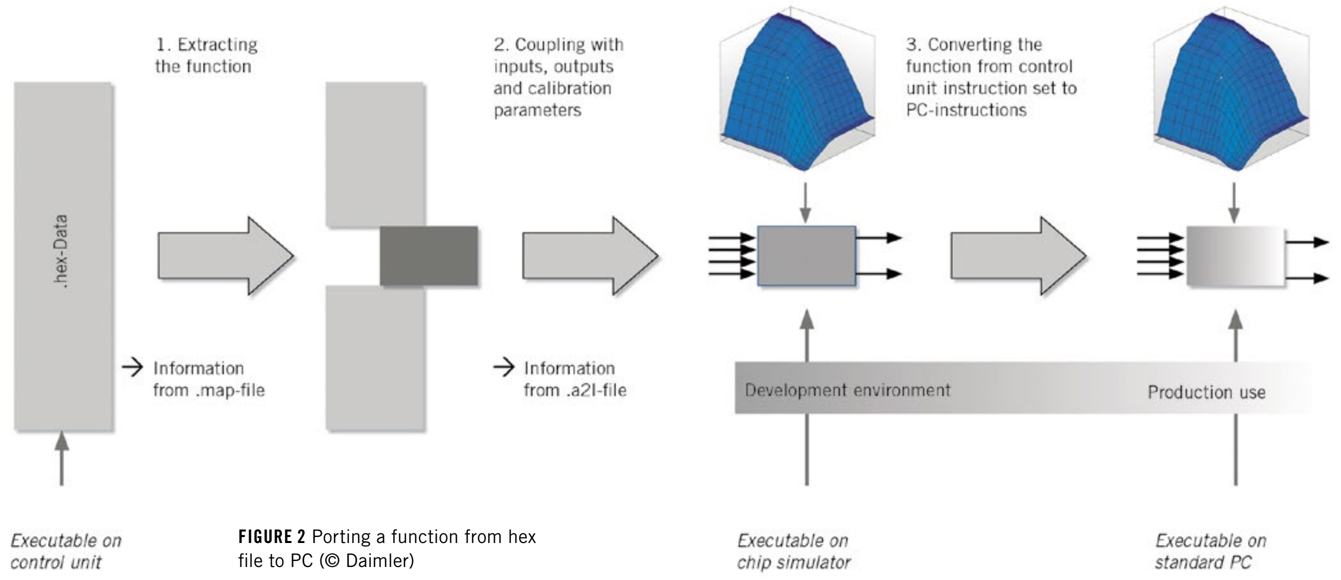

1. Extracting the function
2. Coupling with inputs, outputs and calibration parameters
3. Converting the function from control unit instruction set to PC-instructions
.hex-Data
Information from .map-file
Information from .a2l-file
Development environment
Production use
Executable on control unit
Executable on chip simulator
Executable on standard PC

FIGURE 2 Porting a function from hex file to PC (© Daimler)

amount of fuel being injected is reduced however. This is another closed loop between plant model and control software. In case of the clutch being open (clutch state at 100 %, gearbox not in neutral) the engine speed is limited to a lower value than its usual maximum allowed rotational speed. In the end, this enforces a constraint on the kinetic energy of the engine in case of a sudden, probably unintended closing of the clutch. Detecting this state requires knowledge of the state of the pedals and manual gearbox as well as the engine speed.

In the fourth phase (white), the injected fuel quantity is being influenced by two additional coupled systems. As a powertrain has the characteristics of a spring-mass-damper-system, a sudden increase in torque at the engine side will result in powertrain oscillations [5]. These oscillations are in a frequency range that is perceptible by the vehicle occupant. Their amplitude can be reduced by a dynamic torque offset (see "dynamic torque offset", brown line in phase 4). By adjusting the injected fuel quantity ("actor") based on measured engine and wheel speeds ("sensor") the engine torque can be influenced dynamically using closed loop control. Simulating this behavior requires parts of the control software as well as an elastic powertrain model, ideally including backlash.

Another limitation of the fuel quantity can be active in phase 4. In this measurement, the injected quantity is close to, but not reaching, the maximum allowed quantity. Calculation of this limit again requires coupling of a plant model and control software using sensors (boost pressure and additional information) and actuators (primarily injected fuel quantity in addition to various actuator positions of the air path).

In the fourth phase closing the clutch causes acceleration of the vehicle. To gain a realistic trend of the vehicle's speed the parameters of the clutch and wheel slip as well as the vehicle model need to be determined with high accuracy. Closed-loop interaction of plant model and control software as used for electronic stability control or anti-slip control requires additional control units which will not be included in the work presented here. A short maneuver with a duration of less than 3 s has clearly shown the need for a detailed coupling of actuators to sensors (plant model) and coupling by control software from sensor to actuator (control software model). Phenomenological completeness can only be achieved using an integrated system simulation. The remainder of this article describes the actual components of the system simulation and an assessment whether phenomenological completeness can be achieved.

VIRTUALISED CONTROL UNITS

Two control units were required for the integrated system simulation: a superordinate powertrain control unit and the actual engine control unit. There are substantial differences concerning input/output characteristics, sources of the control unit's software and supported bus communication.

– Powertrain: Functions running on the powertrain control unit usually are independent of the actual engine (diesel, gasoline, electric motor). The powertrain control unit supports several CAN busses as well as a Flexray bus. There are several analog ports like the one for acquiring the accelerator and the gear stick position as well as digital ports like SENT actuators (Single Edge Nibble Transmission, [6]). The control unit's software is developed at Daimler and available in graphical representation as MATLAB/Simulink model and as C Code.
– Engine: The engine control unit can be seen as remote controlled by the powertrain control when it comes to the actual torque demand values. The engine control unit's responsibility is to convert torque demands into fuel quantities and ensure actual injection into the combustion chamber as well as to perform closed-loop control on gas properties on the engine inlet and outlet side to meet emission requirements and perform necessary component protection. Due to this control unit being specific for an engine type, the vast majority of sensor–actuator couplings can be found here. There are several CAN busses. Parts of the control software are developed at

Daimler, other parts supplied by the ECU manufacturer. By linking together both parts, one binary is being created and used subsequently.

The powertrain control unit has been virtualised based on source code, in this case C Code, of all application software tasks running on the RTOS (real-time operating system) of the control unit. Basic software components, such as drivers for receiving and sending CAN-messages or for communication with sensors as well as the RTOS itself, are provided by the simulation environment (QTronic Silver, [2]). This way, the compiled C Code can be used on a PC without modifying the code base. The emulator code required is partly generated automatically from available files (address to label file "a2l" or can bus files "dbc") – this works for CAN, sensors and actuators – and partly configured manually. Names and execution times of the tasks that need to be run by the emulated RTOS need to be configured manually as an example. The build process in principle is a script that compiles existing C code for execution on a Windows PC instead of compiling C code for target processor of the powertrain control unit.

Virtualisation of the engine control unit is substantially different, as the software in part is provided by the supplier of the control. In addition, the close interaction between in-house and supplier software makes it very hard to separate both parts. To still be able to provide the integrated system simulation with a virtual engine control unit, a chip simulator (c.f. [3]) was used, even for those parts that stem from in-house source code. This requires information from three sources: The binary file of the control unit software (hex file), start addresses of the tasks (map file) as being generated in conjunction with the binary by the linker and the address-2-label file (a2l file) as known from engine calibration work.

The real control unit can thus be turned into a virtual control unit the same way as described for the powertrain control unit: The simulator provides again drivers for CAN, sensors and actuators as well as an RTOS for scheduling the tasks initially, time based or crank-angle based. In contrast to the previous implementation, the tasks are now being executed by interpreting the program code from the binary (hex-file) using a chip simulator. Thus, the speed of execution of the tasks drops by a factor of 5 to 10 compared to compiled C Code. The virtual engine control unit runs with half real time on a typical PC. **FIGURE 2** shows the process for generating a model of a single task ("function"). To virtualise the entire ECU, this process is repeated for each task of the ECU.

The initial effort for engine control unit virtualisation was much larger than the one for the powertrain control device as there was no proven and mature process model for control unit virtualisation. Instead a process model for the virtualisation of control units was developed simultaneously. The effort for upgrading to new software depends on the amount of changes and is a fraction of the initial effort.

FIGURE 3 shows the structure of the software of the engine control unit and separates the in-house from the supplier software. Coupling elements and functions performing hardware access are marked in addition. Functions performing hardware access cannot be run on the chip simulator, as they access on-chip peripherals such as controllers for CAN, FlexRay and analog-digital conversion. Such on-chip peripherals are not supported by the chip simulator. The number of hardware-access functions however is much smaller than the number of regular ECU functions. In addition, changes to functions performing hardware access are seldom. This explains the large difference between initial effort of virtualisation when all the functions were identified and solutions for these special functions were developed and the subsequent upgrades to new software versions using the process model developed in conjunction.

The user of the integrated system simulation has access to all calibration parameters of the virtualised control units. Common calibration tools (such as INCA or CANape) are supported.

PLANT MODELS

For the sake of simplicity the powertrain chosen for demonstrating the integrated system simulation was a manual gearbox. The plant models required were an engine model, a gearbox model including differential for a front wheel drive vehicle, side shaft models, wheel models (driven axle and non-driven axle) and a vehicle model.

Out of the possibilities of engine models at Daimler, an already existing and proven diesel engine model based on the dSpace ASM (Automotive Simulation Models) model library [4] was chosen. This model had been in use at a Hardware-in-the-Loop test bench (HiL). The HiL-model had the advantage of complying with the signal interface of the real

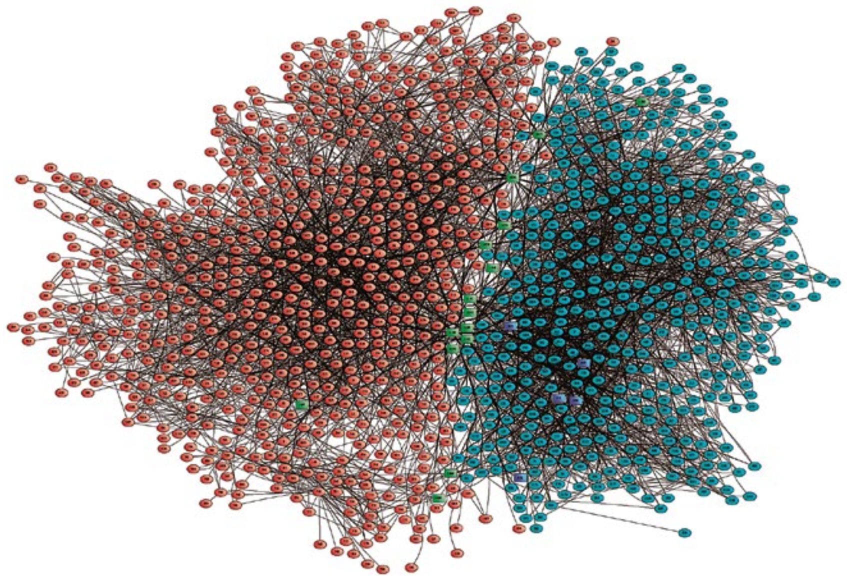


FIGURE 3 Structure of the engine control software with in-house software (right), supplier software (left), and software for coupling and hardware access (green and blue rectangles) (© Daimler)

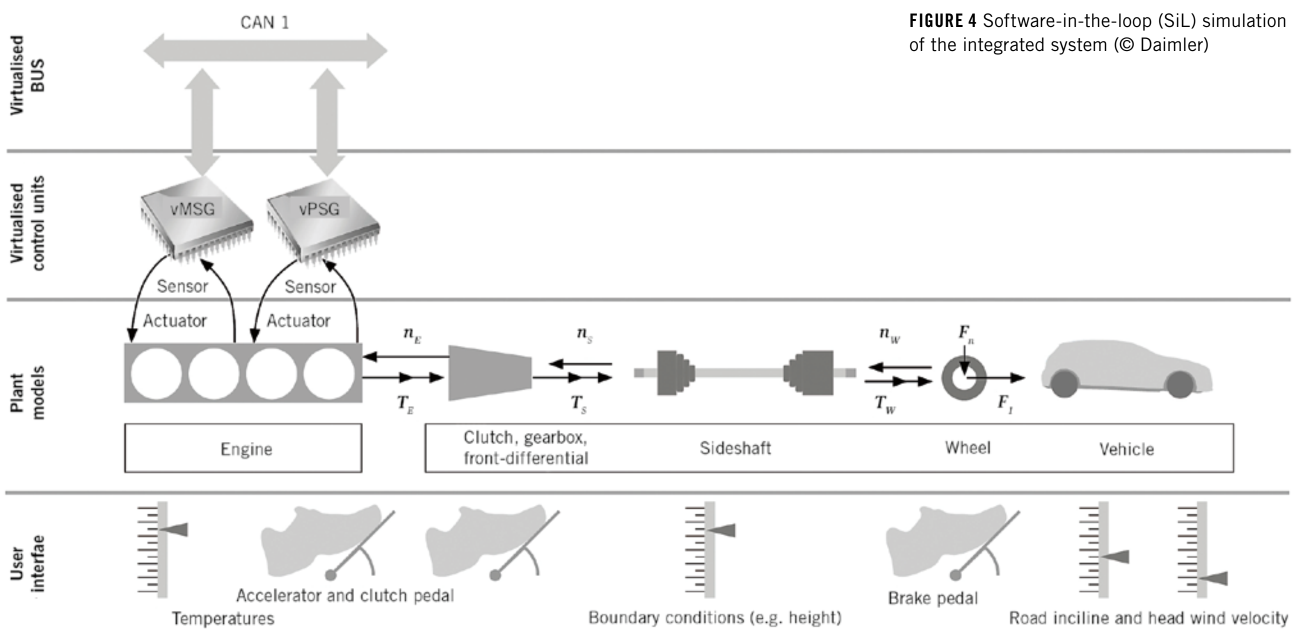


FIGURE 4 Software-in-the-loop (SiL) simulation of the integrated system (© Daimler)

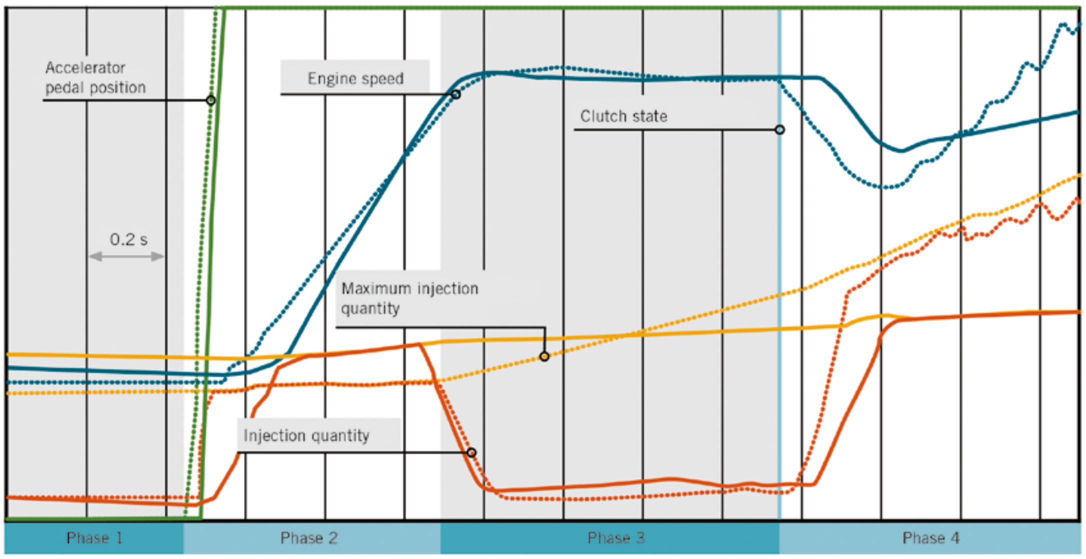
and thus also with the virtualised control unit. In most cases, removal of the HiL-hardware-specific input and output blocks of the engine model and direct use of the open signals was sufficient to make it usable in the integrated system simulation. The engine model closes the loop from sensors to actuators. Ultimately, approximately 40 signal connections between engine model and virtualised engine control unit have to be considered. The powertrain was modelled in Matlab/Simulink using SimDriveline by means of a physics based approach instead of signal based approach. Inputs of the powertrain are the effective torque of the engine and the states of the three remaining powertrain actuators (clutch, brake pedal and gear chosen) as well as two boundary conditions (road incline and head wind velocity). Outputs are various rotational speeds, in particular engine speed and wheel speed, which provide important feedback to the control units.

INTEGRATION

FIGURE 4 shows the structure of the system model with the two virtualised control units and the two plant models. Subdividing the powertrain model into components is possible (due to physical modelling) but provides no added value here as the focus of the simulation is not on the plant side.

Validation is being performed using characteristic maneuvers for drivability calibration. The maneuver used here is the same as was used for deriving the necessity of an integrated system simulation. The maneuver is conducted in the SiL- environment by driving the simulation with measured data ("replay"). This kind of validation is essential for proving

FIGURE 5 Validation of the maneuver (dashed lines are measurements) (© Daimler)



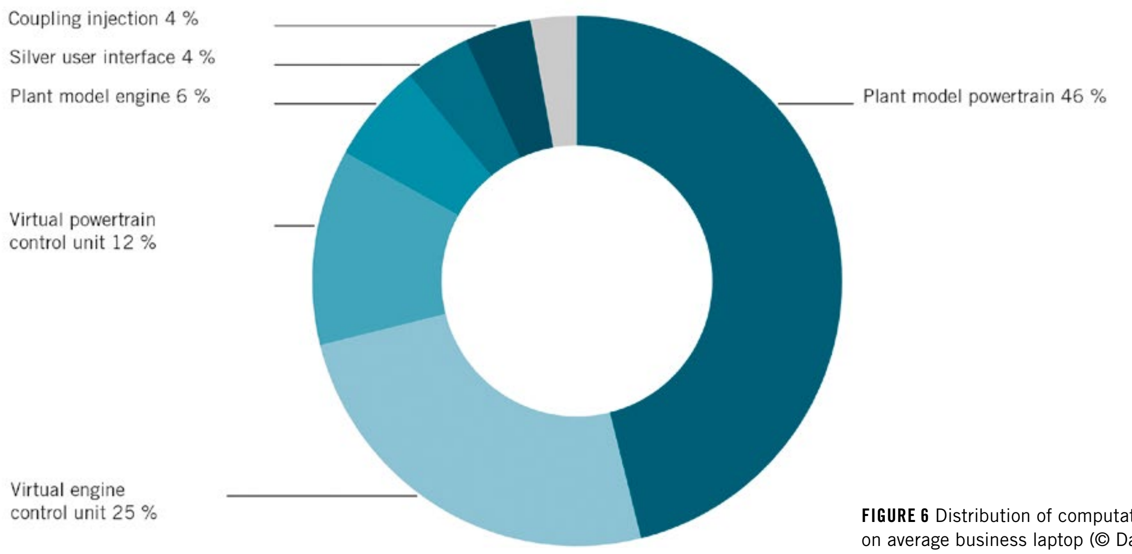


FIGURE 6 Distribution of computation time on average business laptop (© Daimler)

the usefulness of the SiL-simulation for use cases from drivability calibration. The first step in the development of the SiL-simulation was ensuring phenomenological completeness.

FIGURE 5 shows both measured and predicted signals for the maneuver using the measured values of gear, accelerator pedal position and clutch. There is good agreement between measurement and simulation. The speed gradient in phase 2 (white) is slightly larger compared to the reference measurement. This is a result of a higher maximum injection quantity which in turn is a result from differences in starting conditions between simulation (engine was just started before the maneuver) and measurement (just had completed the previous maneuver).

In phase 3 (gray) and 4 (white) the maximum injection quantity rises much faster in reality than the simulated quantity. Investigations showed that a sluggish increase in boost pressure, caused by a turbocharger inertia chosen much larger than actually present caused this behavior. We are currently updating the respective model parameters to better fit measured behavior.

SIMULATION PERFORMANCE

Measurement with Silver, **FIGURE 6** shows, that the plant model of the powertrain has highest computation time requirements. This is a result of interaction between two models: The powertrain plant model provides engine speed and position values requiring torque values whereas the engine plant model provides torque information requiring engine speed and position. This currently forces the powertrain plant model to be executed more often than actually required by the dynamics of the powertrain plant. By continuous optimisation of specific SiL properties, maturity and performance are constantly being improved. Notice the surprisingly low computational costs of the engine model that was previously used at a HiL test bench.

SUMMARY AND OUTLOOK

Powertrain simulation with multiple xCUs has reached a new level at Daimler. This has been enabled by the ability to virtualise entire control units in a way that includes both, supplied control software and OEM-specific functions. Virtualisation of the communication (CAN, Flexray) using real network topology was required as well. Integrating the resulting xCUs with appropriate plant models created a development platform usable not only for function- and software development but also for calibration. Phenomenological completeness of the platform has been demonstrated. Use-cases for the new development platform develop rapidly.

In the future, the set of powertrain variants covered by the platform will grow continuously. Quality and computation speed of the required plant models will be continuously improved. To cope with the growing number of users and development departments involved, the exchange of Silver modules (plant models, virtual xCUs) will be simplified using a rights management. Chip Simulation will be further developed together with QTronic, especially with a focus on implementing Autosar interfaces in Silver. This approach has shown its potential with the implementation of the virtual CAN and LIN bus. Further elements of the base software will follow. As a result, chip simulation will depend less on specific engineering and will become a standard tool for powertrain development. Full ECU simulation is a key tool to master future challenges.

REFERENCES

[1] Uphaus, F.; Gebhardt, A. ; Kirschbaum, F.; Pillas, J.; C. A. Malonga Makosi, R. Binz: Road to ... where? Methods and tools in calibrating drivability, 10th Symposium on Automotive Powertrain Control Systems, 2014, Berlin

[2] Mauss J.: Virtuelle Steuergeräte für die Antriebsentwicklung. 17. MTZ-Fachtagung VPC - Simulation und Test 2015, Hanau near Frankfurt am Main, 30.09 - 01.10.2015

[3] Mauss, J.; Simons, M.: Chip simulation of automotive ECUs. 9. Symposium Steuerungssysteme für automobile Antriebe, 20. -21.09.2012, Berlin. In: Nietschke und Predelli (Hrsg.): Steuerungssysteme für Automobile Antriebe, expert Verlag, 2012

[4] Schütte, H.; Plöger, M.: Hardware-in-the-Loop Testing of Engine Control Units – A Technical Survey, in SAE Technical Paper Series, No. 2007-01-0500, 2007

[5] Pillas, J.; Krischbaum, F.: Model based calibration of a load change reaction minimizing control function using hybrid state space models. Speech Daimler, 12. Internationales Stuttgarter Symposium 2012 Automobil- und Motorentechnik

[6] Single Edge Nibble Transmission, SAE J2716 SENT