Rail Vehicle Control System Integration Testing Using Digital Hardware-in-the-loop Simulation

Peter Terwiesch* Thomas Keller Erich Scheiben

ABB Corporate Research
Information Technology Dept.
Speyerer Str. 4, D-69115 Heidelberg
Fax: +49-6221-59-6253

ABB Daimler-Benz Transportation
Propulsion Systems
Dept. BAA
CH-8050 Zürich, Switzerland

* Author to whom correspondence should be addressed.

Abstract: Control systems for converter-controlled rail vehicles are orders of magnitude more complex than controllers for previous generations of vehicles. While the dynamic behaviour of previous generations of vehicles was to a large extent determined by its power components alone, an important part of the dynamics of modern vehicles is shaped by real-time software, distributed computing and inter-controller communication. To ensure proper operation of the vehicle on track, an integration test of the vehicle control system is performed before initial roll-out. In order to achieve a maximum test depth and to minimize risk and cost, this test is achieved by connecting the original vehicle control system to a real-time dynamic vehicle simulator in closed loop operation. The present paper describes concept, evaluation, and operation of a digital hardware-in-the-loop simulator for testing control-relevant parts of the vehicle. Particular emphasis is put on the hybrid nature of the underlying simulation problem and its inherent causality variations due to the combination of discrete switching effects, e.g., in diodes and controlled converters, with continuous system parts, e.g., differential equations for currents or mechanical system parts.

Keywords: simulation, real time systems, digital control, hybrid system, variable causality, hardware in the loop.

1 Introduction

1.1 Integration Tests Using Closed-loop Real-time Simulation

Traditionally, many technical systems are tested by putting them into their designated working environment and seeing whether they perform according to expectation. Given the importance and complexity of modern digital control systems, including advanced control strategies implemented in far more than 100'000 lines of code per project and distributed over several controller boards with multiple processors each and real-time bus communication between them, the traditional *ad hoc* approach is no longer adequate, and concern must be given to system integration, testing, and verification.

This demand is driven by the factors:

• *risk* (loss of human life or capital)

• *cost* (test in target system can be prohibitively expensive)
• *availability* (target system or designated working environment are not available)
• *coverage* (not all test states can be reached during regular operation)

One of the most powerful, but also most demanding, tests for dedicated real-time control systems (frequently also referred to as *embedded systems*) is to connect the inputs and outputs of the control system under test to a real-time simulation of the target process. As this implies that all control loops are closed via the simulator, this method is often called *hardware-in-the-loop* simulation.

A particular merit of this approach is that it even permits a gradual change-over from simulation to actual application, as it allows to start from a pure simulation and to gradually integrate real electrical and mechanical subsystems into the loop as they become available.

Depending on requirements and process dynamics, three conceptual for control system testing exist:
• self-test/self-simulation mode designed as part of embedded system under test
• use of another control system as a simulator
• dedicated real-time simulation system

The advantage of the first approach is that it requires little or no additional hardware and that it can be activated during maintenance stops without further external tools. However, a built-in self-test alone is hardly sufficient for finding communication bottle-necks or even design flaws, and its fault coverage may often be insufficient. The second approach is relatively popular in an industrial development environment, as it delivers the advantages of full-scale hardware-in-the-loop simulation at comparatively low cost. It saves on wiring, interfacing, and conversion, as controller and simulator are made of identical hardware and are thus easily connected. This aspect is particularly important when several thousand signals need to be exchanged between control system and simulator, as is the case in power plant or chemical process control. One limitation of this approach is that it typically fails to work for extremely fast systems, such as systems with non-negligible electrical dynamics, where the computing power sufficient for running a controller may be an order of magnitude too slow for executing a detailed process simulation. Another limitation lies in the programming of this simulation, since elementary control function blocks are rather ill-suited for the implementation of complex simulation systems.

For very demanding requirements only a dedicated real-time simulation system can offer the required computing power. For a long time, this has been the domain of analog and hybrid simulators, e.g., [2]. The third approach is more costly than the other two in terms of initial investment, as it requires a special hardware interface to the control system.

The present paper describes such a dedicated real-time simulation system and its use for the integration testing of the line and the propulsion converter control system hardware and software of

an electrical locomotive. The inputs from the control system under test to the simulator are discrete firing pulses from Adtranz´ proprietary control system to the power converters, and its outputs include measured currents, voltages, and drive speeds from the simulated vehicle to the control system. The discrete and continuous dynamics of the main electric circuits and main mechanical modes of a locomotive are simulated in a real-time closed loop with a hybrid control system, resulting in a challenging hybrid control and simulation problem.

Figure 1 shows the generic structure of such a system, including the possibility to connect pieces of actual process hardware to the control system and the simulated process for those parts of the system where no proper models are available.

1. 2 Controlled System: Electrical Locomotive

Today's rail propulsion systems deliver a functionality in terms of energy efficiency and power factor that was unheard of before the days of computer-controlled power converters (section 2.1). This revolutionary increase in functionality is accompanied by an ever growing control system complexity (section 2.2).

1.2.1 Loco 2000

Despite differences in functionality between different classes of rail vehicles, we concentrate our process description of electrical rail vehicles on locomotives, as locomotive projects are one important class of applications for the simulator described here and as locomotives typically have often more demanding control requirements and functionality then, e.g., trams or metros.

Within the class of electrical locomotives, we choose as an example "Loco 2000", the pride of Swiss Federal Railways. This class of vehicles employs 3-phase propulsion and GTO (=gate turn-off thyristor) converters to reach maximum power of 6100 kW and top speed of 230 km/h. Weighing in at 81 tons, it is the first member of a family of locomotives on which several further European and overseas railway projects have been based [8]. Together with the power transformer, the converters and their controllers are really the "gut" of the locomotive. They have the task of converting the 16 2/3 Hz single-phase AC supply into a three-phase AC source of variable amplitude and frequency suitable for supplying the induction motor drives. The flow of energy conducted by the line converter does not equal that conducted by the propulsion converter at every instant. The difference is either stored or supplied by the DC link circuit, in banks of metallised paper capacitors.

Figure 2 gives a schematic overview of the Loco 2000's electrical system. Electrical energy is fed in from the catenary through the pantograph, and is then transformed and rectified to feed an intermediate-link capacitor. In Loco 2000 Re460, this capacitor is split in two symmetric halves, providing the voltages +0.5 V_d, 0, -0.5 V_d, where V_d is the DC link voltage. A computer-controlled GTO inverter bridge (propulsion converter) converts this DC energy to a three-phase AC system that feeds the driving motor. Note that this flow of energy can also be reversed for regenerative braking, which converts energy of motion back to electrical energy that is fed back into the line.

1.2.2 Control Structure

Figure 3 summarizes the most important control functions of an electrical locomotive as relevant to this paper. The main control signals are the GTO firing pulses, which determine whether the GTO thyristors are conducting or not. These signals are generated by the line converter controller and the motor inverter controller. For the line side, the control signals are generated by a pulse width modulator. By varying the widths of the different voltage impulses, the amplitude and phase-angle of the 4-quadrant controller voltage can be adjusted to fit the needs of stability and energy. The motor controller produces GTO firing pulses to provide the motors with a variable-frequency three-phase AC supply. To generate the corresponding pattern of control impulses and for driving the motor under optimal conditions at all times the motor frequency range is divided into 3 control ranges: indirect self control, direct self control and full impulse method [18, 19]. Each control range has its own requirements with regards to hardware-in-the-loop simulation and has therefore been examined individually.

3 Hybrid Simulation Problem

3.1 Simulator Requirements

The goal of the proposed real-time simulator is to ensure proper operation of the control system on the actual vehicle. Tests are performed on the closed-loop system obtained by connecting the actual locomotive control system to the simulator. The basic structure of the hardware-in-the-loop simulation is shown in Fig. 4. GTO firing pulses from the control system are the most important inputs to the simulator, while electrical quantities (voltage, current) and mechanical quantities (torque, speed) are the most important outputs. Within the simulator, mathematical models of converters, filters, drives, and mechanics are used to represent the electro-mechanical behavior of the vehicle. Obviously not only the dynamics of the vehicle itself, but also interaction with its environment (wheel-rail contact and electrical network) need to be simulated. An example of a typical feature tested by the simulator is the amplitude of the torque ripple produced by the induction motors at different operating points. Another typical measurement is the transient shape of motor and line currents when switching from acceleration to braking.

The setup described in Fig. 4 imposes stringent real-time requirements on the simulator, as the control system is running with a cycle time of 40-60µs and as computation of currents from firing pulses is extremely sensitive (1µs jitter in the processing may noticeably affect the outcome of the simulation). This need for accurate processing of switched currents and the fast control system sampling leads to a required simulation *frame time* of about 30µs, which is sufficient for the evaluation of the electro-mechanical dynamics provided that switching events are receiving a faster special treatment. Here, frame time is the calculation time needed for one simulation step with the entire model. At the end of each frame time the simulator communicates with the connected control system, meaning that the frame time is the granularity of real-time synchronization between simulator and connected control system. Note that a conceptual alternative to the short frame times needed here can be to synchronize control system and simulator, using identical sampling instants. While this somewhat relaxes the computation speed requirements, this approach appears viable only when a control system has been explicitly designed for such testing.

A further important requirement is that the programming of the simulation system should be user-friendly and configurable for the end users, who are typically vehicle experts, but without in-depth real-time simulation knowledge. This need translates into the requirement of using a graphical, block-oriented, simulation language to configure and parameterize the simulation model.

3.2 Simulator Platform

Several commercial vendors are addressing the real-time simulation market with their hardware and software platform solutions. One of them is dSPACE, a German company that offers a fast, modular real-time hardware, together with a software interface to Matlab/Simulink [17]. dSPACE systems are widely used for rapid prototyping and hardware-in-the-loop simulation in many application areas, especially in the automotive industry. Based on DEC Alpha processors and TMS320C40 signal processors they offer scaleable computing power. The processors are connected by fast point-to-point connections, avoiding the bottleneck of a common bus. Every C40 processor has its own peripheral bus for connecting I/O boards, allowing parallel I/O on different processors. A number of very powerful I/O boards is available.

dSPACE offers a user-friendly development environment based on Simulink as a graphical modelling environment [15]. Simulink is widely used for off-line simulations, resulting in the existence of adequate company-internal model libraries and commercial and university support toolboxes for our purposes. This is one of the reasons for using Simulink as a front-end, despite the drawbacks of Simulink and similar languages to imply fixed causality in its block diagrams and to properly support event handling.

Since causality variations and discrete events (e.g., firing pulses) in an otherwise continuous model (e.g., first-order ODEs describing a motor) contribute an important part to our modeling problem and since we could not find suitable off-the-shelf solutions for our problem, we had to give this aspect further consideration and solve it for ourselves. The following sections describe our hybrid simulation problem in more detail, introduce a pragmatic solution to circumvent the above mentioned deficiencies of fixed-causality simulation languages, and show results obtained with the fully operating simulator.

3.3 Hybrid Simulation Problem: Discrete and Continuous Subsystem

The vehicle simulation model, as many others, consist of an *analog block* with a number of first-order ordinary differential equations (ODEs) for the continuous-valued system states x and a *binary block* with combinatorial and sequential equations for the discrete-valued system states X. The two blocks are coupled through switches (change of continuous input variables or equation structure by digital signal) and comparators (binary result from comparison of continuous variable with threshold) as shown in Fig. 5. While there is a wealth of methods available for simulation of either block [3,4,5], a great challenge both in non-real-time and real-time simulation is to properly account for the links between continuous and discrete simulation blocks.

This coupling between discrete and continuous subsystems is one of the main reasons why *digital* real-time simulation is taking so long to replace *hybrid* simulation computers: hybrid computers contain comparators (analog to discrete) and switches (discrete to analog) that implement this link very naturally [1,2]. In digital simulation, on the other hand, three types of problems resulting from the link of discrete to continuous blocks can be distinguished (with examples from our system):

- external events (Fig. 6), e.g., firing pulses from the control system to the converters
- state (or internal) events (Fig. 7), e.g., idealized diodes that conduct or block depending on voltage and current conditions (blocks as long as neither voltage nor current are positive)
- time events, e.g., waiting time conditions

The latter two have in common that a continuous variable passes a threshold, so that a comparator generates an event. This event will typically operate on a switch, so that it may significantly change the continuous system in its structure, causality, or parameters. In an electric circuit, containing elements such as capacitors or inductances, the event structurally changes the state equations, i.e., the differential equations governing the dynamics of the system.

In real-time simulation, handling state and external events are the major problems, since the limited frame time, corresponding to a fixed external step size, normally does not permit adaptive step-

sizing or iterations around the event. These two types of events thus call for a specific treatment depending on the exact requirements. Time events, on the other hand, are usually known in advance and can be incorporated by using adequate synchronization instants, so that they are normally not the main problem of real-time simulation.

There are different ways to approach the handling of state and external events in a digital hardware-in-the-loop simulation. External event problems can often be solved by one of the following approaches:

- small frame time
- interrupt-driven integration

and state event problems by

- small frame times
- event time registration and correction

The best approach to event problems is in all cases small frame times. It is therefore always advisable to examine the exact requirements of the system before implementing different approaches, as they may lead to much bigger frame times due to computational effort spent on iterations and corrections. However, when the dynamics of the system are very fast in comparison to the computational qualities, one of the other approaches must be used, see section 3.3.

3.4 Integration of Continuous Part

The choice of the right numerical integration method for the continuous part may greatly contribute to the power of a digital real-time simulator. This section summarizes some considerations for selecting appropriate algorithms, compare [9,11].

Mathematically, the continuous part of the system that needs to be simulated in real-time is usually described by a set of first-order ordinary differential equations

$$\frac{dx}{dt} = f(x,u,t)$$

Here x is the system state vector (containing elements such as position, speed, voltage of a capacitor, current through an inductance etc.), u represents external inputs (such as manual switches, or signals coming from the control hardware), and t stands for time.

In general, this type of system can be integrated using several approaches, which are distinguished by the following properties [9]:

- *explicit / implicit methods:* Explicit methods only use past values of x (up to the present time t_k), whereas implicit methods also use the new value $x(t_{k+1})$, which they find by iteration. Implicit methods are particularly suitable for stiff systems, i.e., for systems in which very slow and very fast dynamics need to be considered simultaneously.

• *single-step / multi-step methods:* Single-step methods only use one past value, $x(t_n)$, for computing the new value, $x(t_{n+1})$. Multi-step methods use more than one past value, e.g., $x(t_n)$ and $x(t_{n-1})$.. As single-step methods need smaller frame times or more evaluations of the model in the interval [t_n , t_{n+1}] to achieve the same accuracy as multi-step methods (as long as the state trajectories can be approximated by polynomials), they are also computationally more expensive.

Note that real-time simulation does not permit to use predictor/corrector methods beyond the current frame, since no knowledge of u_{k+1} is available. As long as there are no abrupt changes in the state trajectories, multi-step methods, such as the Adams-Bashforth algorithm, offer better performance and/or accuracy than single-step methods. However, as soon as switching needs to be accounted for, single-step methods such as the explicit Euler algorithm must be used, since an extrapolation of the past is no longer desirable and would lead to inaccurate results. Note however that these algorithms require small step-sizes and have comparatively poor stability properties [9]. Note that modern algorithms capable of event detection [23] are not suited for real-time simulation without modifications (such as bounds on the number of iteration, which in turn would effect their accuracy) due to the need to synchronize with an external cycle time of 30µs.

3.5 Switch Simulation Problem

As described above, events, also termed discrete mode transitions, due to idealised models of switches introduce two problems [12,13]:

- The causality of the continuous model part will change for every mode-transition, e.g., if voltage was the input and current was the output of a model before the event, voltage would be the output and current the input after it.
- State events have to be detected and continuous mode-equations re-arranged and re-initialised for every mode-transition.

The problem of variable causality is illustrated by modelling a switch located between the transformer and the rectifier of a locomotive. A switch model between the two blocks has inputs U_d and I_q and outputs U_{st} and I_d (Fig. 8).

Now consider the change of causality depending on the state of the switch. When the switch is *closed*, we have $U_{st} = U_d$ and $I_d = I_q$, as shown in Fig. 8 (upper). Model causality is different when the switch is *opened*, since then $I_d = 0$ and $I_q = 0$. Instead of U_{st}, I_q is given and the voltage U_{st} is undefined (Fig. 8, lower). However, I_q is also an output of the transformer. In order to resolve this conflict properly, the transformer equations have to be solved for U_{st} instead of I_q.

Implicit simulation languages, such as ACSL [7] are able to describe events, but are not directly suited for real-time simulation. Explicit simulators such as Simulink do not support such equations. One can obviously program all cases manually, but a system with n switches would then have 2^n different representations, depending on the state of the switches [14,6,10].

Alternatively, we propose a pragmatic method to estimate the undefined open switch voltage U_{st}.

4 Switch Approximation for Fixed-Causality Simulation

Section 4 discusses a pragmatic solution to the problem of digital real-time simulation of switched systems as described in section 3.5. Switching occurs both on the line side and the motor converter side of the vehicle and two issues, namely variable causality and accurate event processing need to be dealt with. Without a solution to the causality problem, the decomposition of the system model to subsystem models would not be possible, resulting in a monolithic model that is hard to engineer and difficult to distribute over multiple processors. The solution to the second problem, accurate processing of switching events, replaces the coarse intregration of firing pulses with a time resolution limited to the sampling time by a more accurate mechanism.

4.1 Open Switch Voltage Estimation

An open switch has to set the voltage U_{st} in such a way that the current I_q becomes zero. Our pragmatic approach is to use an estimator with an approximate model for determining U_{st} (Fig. 9). The estimator is based on a first-order approximate model of the open circuit, e.g. an R-L-link for a transformer (Fig. 10). U_q, R, and L represent the model of the transformer. Assuming that R and L are approximately known, the unknown voltage U_q is estimated by applying a voltage U_{st} and evaluating the resulting change in the current I_q. Note that an R-L-link is a reasonable choice in our case, as both types of converter loads (transformer and induction motor) are inductive. For the values of R and L we choose the resistance and leakage inductance of the corresponding coil, neglecting the coupling of other coils. As shown later, modelling errors result in a leakage current.

The R-L-link in Fig. 10 is described by the following discrete equation:

$$L\,\frac{I_q(k) - I_q(k-1)}{\Delta t} = -R\,I_q(k-1) + U_q(k-1) - U_{st}(k-1) \qquad (1)$$

At time k we know $I_q(k)$ and $I_q(k\text{-}1)$, the new and the previous current from the transformer, and $U_{st}(k\text{-}1)$, the voltage over the switch from the previous time step. With these values the unknown source voltage U_q during the last time step can be estimated in the following way:

$$\hat{U}_q(k-1) = L\frac{I_q(k) - I_q(k-1)}{\Delta t} + RI_q(k-1) + U_{st}(k-1) \qquad (2)$$

At zero current this is the desired open switch voltage. But if the switch is opened with non-zero current (which is physically impossible but often happens in a simulation due to sampling effects), a non-zero leakage current remains as a simulation error. This simulation error can be forced to converge to zero by adding a correcting term to the estimated source voltage:

$$U_{st}(k) = \hat{U}_q(k-1) + k_{pso} I_q(k) \qquad (3)$$

The leakage current, and thus the simulation error, will then decrease with the time constant

$$\tau = \frac{L}{R + k_{pso}}. \qquad (4)$$

We normalize the estimator gain k_{pso} with

$$k_{pso0} = \frac{L}{\Delta t} - R, \qquad (5)$$

the gain for $\tau = \Delta t$. The estimator gain is now described through the relative factor k_{corr}:

$$k_{corr} = \frac{k_{pso}}{k_{pso0}} \qquad (6)$$

The following factors contribute to the leakage current:

- *Time delay:* The voltage estimation is delayed by one time step.
- *Switching off a non-zero current:* Opening the switch with a non-zero current flowing results in voltage peaks, since the current through the inductivity L cannot change discontinuously. The current then fades away with a time constant given by the estimator parameters, as shown above.
- *Error in L:* A mismatch between the inductivity L in the estimator and the true inductivity in the circuit leads to oscillations or even instability. The effect can be reduced by allowing a larger time constant in the estimator (section 4.2).

Figure 9 illustrates that even for a non-zero leakage current I_q the current I_d on the other side of the open switch can be set to zero. In order to properly parameterize this estimator, two kinds of parameters must be chosen: internal model parameters and estimator gain.

The following discussion assumes an R-L-link as an internal model of the estimator, as described above. In our example, the choice of R is not critical. It will be added to k_{pso} and can normally be neglected, namely if $R\ \Delta t / L \ll 1$. On the other hand, the choice of L is important. Assuming that the real circuit is an R-L circuit according to Fig. 10 with L_{eff} instead of L, it can be shown that the estimator is only stable, if L_{eff} is in the range

$$\frac{k_{corr} + 2}{4} L < L_{eff} < 2L \qquad (7)$$

When the inductivity of the open circuit is known exactly, the estimator can be tuned to make the leakage current disappear after one time step (L = L_{eff}, k_{corr} = 1). However, in many cases L_{eff} is not known a priori, since it depends on the state of other switches. Then for simplicity an ad hoc decision is taken for choosing L. Often the inductance of the branch which is directly connected to the switch is used, e.g. the leakage inductance of a transformer coil. The stability range (7) can be extended at the lower end by choosing a small value for k_{corr}, This makes the system more robust against changes in L_{eff} at the cost of a slower decreasing leakage current. As we see, the choice of k_{corr} is a trade-off between stability and performance and must be examined experimentally. Reasonable values of k_{corr} lie in the range 0..1.

Typical values in our transformer-rectifier example (Fig. 10) are R = 0.02Ω, L = 0.005H, Δt = 25e-6s. With $R\ \Delta t\ /\ L$ = 1e-4 << 1 we can neglect R in equations (5) and (6). For different choices of k_{corr} we have the following results (see also section 5.2):

k_{corr} = 1: fast decreasing leakage current ($\tau = \Delta t$), but not robust (oscillations when error in L).

k_{corr} = 0.25: still fast ($\tau = 4\Delta t$), no oscillations, robust until $L_{eff} \approx L/2$.

k_{corr} = 0: very slow (τ = L/R = 0.25s), no oscillations, robust until $L_{eff} = L/2$

This shows that selecting k_{corr} is not a critical design decision.

By modelling a diode as an ideal switch, the same method can also be applied to diodes. We have modelled the diodes of the line converter in this way. With the proposed method the behavior of the locomotive is simulated qualitatively correct even with only the diodes working (switching pulses off). There is a small leakage current $I_q \neq 0$ in the transformer which normally can be neglected, especially when the diodes are used only for charging the DC link capacitor at startup.

An alternative approach would be to zero the current in other parts of the model (e.g. transformer block) when the switch is open. This decreases the leakage current by at least an order of magnitude, but depends on changes in the model blocks (e.g. transformer) and additional, artificial, connections between them. We have decided against this approach, since it would come at the cost of giving up the current modularity, since artificial control signals would be needed to transmit the state of the switches to the blocks. Instead of setting the current to zero it is also possible to design an internal event correction method which calculates the exact time of the event and adds a correcting term in the next simulation step. This method has the same drawbacks concerning modularity as setting the current to zero.

4.2 Time Stamping

While the frame time needed to process the electrical model equations is in the order of 30µs, switching delays and switching jitter must be handled with a resolution of 1µs or faster in order not

to loose simulation fidelity. Since 1µs frame time would be out of question for the given models with today's real-time simulators, an alternative method for handling the influence of switching events in the model is needed for the particular application under investigation.

Our solution exploits that all converter firing pulses in the vehicle model occur at the input of voltage integrators, where they switch between zero and a slow-changing voltage value. Consequently, this integration of firing pulses can be approximated with high fidelity by approximating the ideal continuous integrator with an accurate time-voltage product instead of the usual coarse discrete-time integration. Since the voltage is changing only slowly and can thus be considered constant over one frame time, it is sufficient to accurately register the switching times (with 25ns resolution in our application) in order to produce a high-quality time-voltage product at the output of the integrator. This approximation is first-order accurate and neglects the cross-couplings between different integrators of the model. Simulation studies have shown this time-stamping method to be far more precise than sampling of the switch state and every frame time, and the remaining real-time integration error compared to an accurate off-line solution is definitely negligible in view of any model quality that can be achieved for the type of system under investigation.

5 Results

5.1 Implementation

The real-time simulation is implemented according to the modularization shown in Fig. 4, including a model of the controller used in off-line simulations and sensitivity studies. In on-line simulations the controller model is exchanged against the actual Adtranz controller hardware and software, and a real-time hardware-in-the-loop simulation is performed. Taking special care of switching events as described in section 4 requires some of the lower-level parts of the model to be coded in C rather than Simulink. However, this part remains hidden from the simulation end-user, who only gets to see the vehicle topology and parameters as a Simulink models.

As the locomotive consists of two almost symmetrical parts, only half of the locomotive (half the line converters and inverters, one intermediate circuit and one bogie) need to be considered and the coupling of the two bogies through rail and carbody is neglected in the current version of the simulator. The model is entirely based on ordinary differential equations as found in the literature, e.g., [21] for the electrical and [22] for the mechanical part of the vehicle. Practical tests have shown that the approximation of the half-loco is sufficient in almost all cases, with few exceptions

such as the fine tuning of adhesion controllers, where the leading axles are conditioning the rail for the following ones.

Before actually building up the complete system, benchmark tests have been conducted together with dSPACE. The model was carefully partitioned over several processors to improve the frame time, and the optimal configuration was determined. Such an optimum generally exists, since there is a natural bound on the number of processors due to communication needs, which at some point increase the frame and reaction time more than a further processor can reduce the computation time. Not all parts of the model actually require a high sampling rate. The mechanical subsystem, for instance, has very slow dynamics in comparison to the rest of the model and can therefore be sampled slower and connected by a zero-order hold, thereby decreasing the computational load.

5.2 Hardware

Benchmarking different configurations, the final hardware architecture has been chosen. Figure 11 shows how the model is partitioned in the final configuration. With two 300Mhz, 64bit, DEC alpha processors and six C40 digital signal processors a frame time of less than 30 μs has been achieved.

The converter switching pulses are read in with a time stamping hardware with a resolution of 25ns. The time stamp information is used in the simulation in order to increase the accuracy and more than one pulse per integration step can be handled. This prepares the simulator also for future applications with higher switching frequencies, such as those possible with high-power IGBT (insulated gate bipolar transistor) converters.

A more detailed description of the underlying dSpace platform system and of their application to the problem described here from a platform point of view can be found in [20]. All processors are mounted on PC/AT-compatible boards in two different boxes. Each box is connected via ISA bus to the host PC, a PentiumPro 200 with Windows NT, which is used as an engineering workstation and operator interface. The different utilized dSpace component types are:

- Alpha Processor Boards as designated working horses, evaluating the entire model. Clock frequency of 300MHz, 64 bit resolution. Communication through dual port memory to a DS1003 board, practically no computational overhead.
- DS1003 board based on the Texas Instruments processor TMS320C40 with 50MHz clock frequency and 32 bit resolution, mainly used for communication and not for the evaluation of model equations. Also used to read the registered trigger pulses from digital waveform capture board (DS5001). Three types of communication channels:
    - *Parallel 8-bit links:* 6 8-bit parallel links per board for inter-processor communication with 20 MB/s transmission rate. All DS1003 boards are interconnected by these links.

- *32-bit PHS-Bus:* peripheral high-speed (PHS) bus connects peripheral boards to DS1003 boards with 20 MB/s. Several peripheral boards can be connected to the same bus.
- *Dual Port Memory:* The dual port memory (DPM) is the third possibility to connect a board to the DS1003 board. It is used for the alpha processor board.

- Peripheral Boards: the simulator comprises four different types of peripheral for inputs and outputs.
  - *Time-stamp Board.* The DS5001 board is used to register trigger pulses coming from the control electronics. It consists of 16 channels, each storing in 1 bit the state of the signal (high or low) and in 31 bits the time of the transition with a resolution of 25ns.
  - *Binary I/O.* The DS4001 board consists of 32 binary I/O configurable in groups of 8 bits as inputs or outputs. We use this board for slow binary signals such as control signals for circuit breakers and their feedback signals.
  - *D/A-Converter Board.* The DS2102 is a high resolution high speed digital to analog converter board with 16 bits resolution and a settling time of 2.5µs for 6 parallel channels.
  - *Waveform generator board.* The DS2301 consists of 6 parallel channels, each comprising a TMS320C31 processor and a D/A-converter. Each processor is separately programmable in C, which we use for the generation of incremental speed sensor signals. This calculation is performed asynchronously to the rest of the simulation with a resolution of 6µs.

The described digital real-time simulator replaces a previous hybrid simulator [2] of significantly higher cost, with more frequent maintenance requirements, which filled a large air-conditioned computer room. The new simulator is considerably less costly, requires less maintenance and requires only the space of two desktop PCs. Comparing the fidelity of both simulators against actual process measurements showed a very high level of fidelity for both, with no clear accuracy advantage for either of them.

5.2 Numerical Results for Switches

The simulation in Fig. 12 shows the development of the voltage U_{st} and the current I_q during switching in an R-L-link with U_d = 0. With the chosen value k_{corr} = 0.25 the leakage current decreases with a time constant $\tau = 4\Delta t$. Notice the voltage peaks at switching off due to forcing the current in the inductance to zero. The detail plot in Fig. 13 shows that the estimation scheme is robust against a leakage inductance mismatch (L = 0.003 instead of 0.005). It can be seen that the scheme produces perfect stationary accuracy and that the transient estimation error converges back to zero with a time constant of about 0.3ms, which is definitely acceptable for our purpose.

5.3 Numerical Results for Vehicle Simulation

Closed-loop simulation of the entire vehicle is demonstrated on the example of a comparison between field-weakening direct self control, which is used in the upper speed range, against indirect self control, which is used in the lower speed range of the vehicle [18, 19].

For the line side the top row of Fig. 14 shows the primary-side current of the vehicle during a transition between driving and regenerative braking. In order to make the time scales equal, the later braking part of the indirect self control is omitted here. Especially for the field-weakening direct self control plot one can clearly see the 180° phase change in the current, which is needed in order to feed back electrical energy into the traction mains. The second row of Fig. 14 shows how the intermediate link voltage is affected by the transition. Note, however, that a comparison of the control schemes based on these voltages is not straightforward due to the different motor speeds in both cases.

Regarding the motor side, Fig. 15 shows the spatial trajectory of the stator flux for one of the traction motors. The hexagonal flux figure of direct self control is easily distinguished from the circular indirect self control flux figure. It can also be seen that 1.5 seconds after the braking command a stationary trajectory has already been reached in the left case, whereas the flux amplitude is still growing in the right case. Figure 16 shows a detail from this transition phase, namely the electrical motor torque and a related stator phase current. Due to the higher motor speed the sinus approximation of the phase current in direct self control with field weakening needs to be made with fewer switching actions (hence the hexagonal flux figure) compared to the lower stator frequency, where a much better sine wave approximation is reached (hence the almost circular flux figure).

6 Conclusions and Perspective

We have presented selected results from a digital real-time simulation of traction vehicles. Particular emphasis was placed on the simulation of converter switching phenomena in the otherwise continuous system. Since this hybrid nature of the problem does not permit an exact solution in real time, we have presented an approximation method based on an estimator, which has satisfied our requirements. Distributing the locomotive model over several processors yields a sampling time of 30 μs for the continuous part, while discrete switching is resolved even more accurately.

Overall, closed-loop real-time simulation of the vehicle using the actual control system considerably simplifies following vehicle tests on track and can also be used to reconstruct and further investigate possible problems encountered during vehicle operation. The digital simulator described here has

entered a domain that had been reserved to analog or hybrid simulators in the past due to the extremely fast electrical process dynamics and the complexity of handling discrete causality changes in otherwise continuous systems in real time. The graphical programming of the entirely digital simulator greatly simplifies the control system integration test procedure.

Although a difficult task, we hope that simulation platform suppliers will make hybrid real-time simulation even more user-friendly in future versions. One promising approach can be to do the modeling on a higher level, without initially fixing the causalities, as is supported by object-oriented modeling tools such as Dymola [16]. An intelligent translator can then resolve the causalities analytically for a small number of discrete combinations, or, knowing the network and possible switch combinations, resolve them approximately by finding an appropriate estimator as demonstrated in our example.

We would like to take the opportunity to thank a number of people who have supported us throughout this work: R. Graf, P. Wenk, P. Kulli, B. Wixinger, and P. Häse from ABB Daimler-Benz Transportation (Switzerland) Ltd; A. Petersen and S. Menth from ABB Corp. Research Ltd, and U. Kiffmeier from dSPACE (Germany).

References

[1] H. Bühler: A driving simulator for investigating the static and dynamic characteristics of speed regulating systems for traction vehicles, Bulletin Oerlikon No.364/365, pp15-22, 1965.

[2] H.P. Wenk: SIMSTAR-Ein modernes Simulationswerkzeug. ABB Verkehrssysteme AG, *Roll-on,* no.3, 1992.

[3] G.A. Korn, J.V. Wait: Digital continuous-system simulation, Prentice-Hall, 1978.

[4] F.E. Cellier: Continuous System Modeling. Springer-Verlag, 1991.

[5] A. Naim: Systems modeling and computer simulation. M. Dekker, 1988.

[6] O. Ruhle: Echtzeitsimulation schneller transienter Vorgänge mit Hilfe von Parallelrechnern, VDI-Fortschrittsber., Reihe 20, Nr. 127, 1994.

[7] Technical Committee on Continuous Simulation Languages, Simulation Councils, Inc. (SCi): The SCi Continuous System Simulation Language", *Simulation*, no.9, pp.281-303, 1967.

[8] Locomotive 2000 product description, ABB Transportation Systems Ltd, Schweizer Eisenbahn-Revue 10/1991.

[9] G.H. Golub, J.M. Ortega: Scientific computing and differential equations: an introduction to numerical methods. Academic Press, 1992.

[10] O. Rathjen: Digitale Echtzeit-simulation: Simulation einer Hoch-spannungs-Gleichstrom-Übertragung. Vieweg, 1993.

[11] T. Hopkins, C. Phillips: Numerical methods in practice: using the NAG Library. Addison-Wesley, 1988.

[12] G.M. Asher, V. Eslamdoost: A novel causality changing method for the bond graph modelling of variable topology switching circuits. In Proc. IMACS Symposium, Lille 1991, pp371-376.

[13] J.E. Strömberg, J. Top, U. Södermann: Variable causality in bond graphs caused by discrete effects. In Proc. of the First Int. Conf. on Bond Graph Modeling (ICBGM ‘93), SCS, San Diego, 1993.

[14] W. Borutzky, J.F. Broenink, K.C. Wijbrans: Graphical description of physical system models containing discontinuities. In Proc. ESM‘93, Lyon, 1993, pp203-207.

[15] H. Hanselmann: DSP in Control: The Total Development Environment. Int. Conf. on Signal Processing Applications and Technology, Boston, MA, 1995.

[16] H. Elmqvist, F.E. Cellier, M. Otter: Object-oriented modeling of hybrid systems. ESS’93, European Simulation Symposium, Delft, NL, Oct 25-28, 1993.

[17] Matlab and Simulink are trademarks of TheMathWorks, Inc., Natick, MA.

[18] M. Depenbrock: Direct self-control of inverter-fed induction machines. IEEE PESC, Blacksburg/USA, pp. 632-641, 1987.

[19] J. Steinke: Pulsbreitenmodulation eines Dreipunktwechselrichters für Traktionsantriebe im Bereich niedriger Drehzahlen. etz Archiv, vol. 11, no. 1, 1989.

[20] R. Otterbach: Geschwindigkeitsrausch trotz simulierter Fahrt. Elektronik, no. 12, pp. 126-133, 1997.

[21] M. P. Kazmierkowski, H. Tunia: Automatic control of converter-fed drives. Elsevier, 1994.

[22] P. Terwiesch, S. Menth, S. Schmidt: Relation between mechanical and electrical oscillations in rail vehicles. 7th Intl Conf Harmonics in Power Systems, Las Vegas NV, Oct 16-18 , 1997.

[23] J. Stoer and R. Bulirsch: Introduction to numerical analysis. New York: Springer-Verlag, 1980.

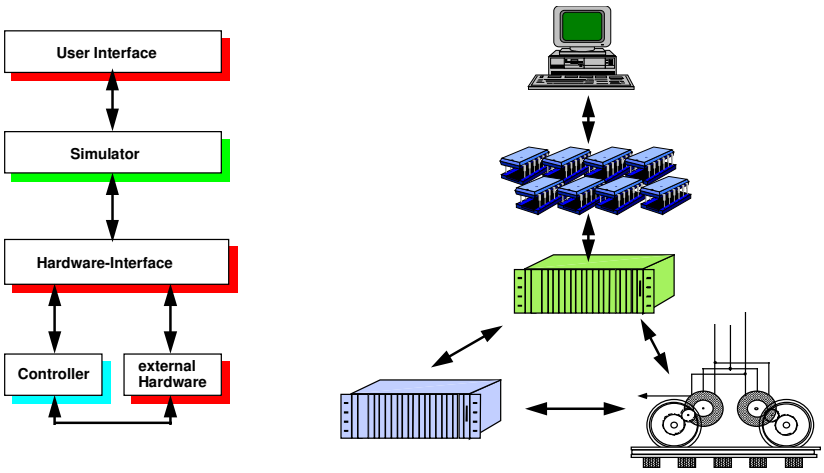

Fig. 1: Generic hardware-in-the-loop setup.

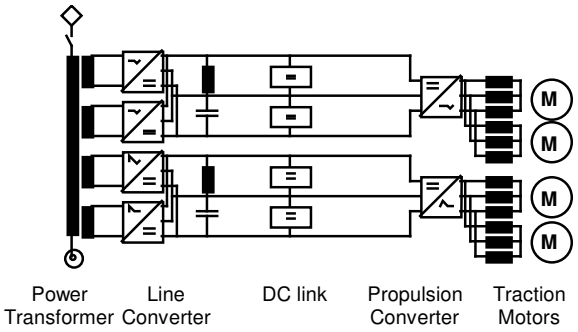

Fig. 2. Loc 2000 electrical system

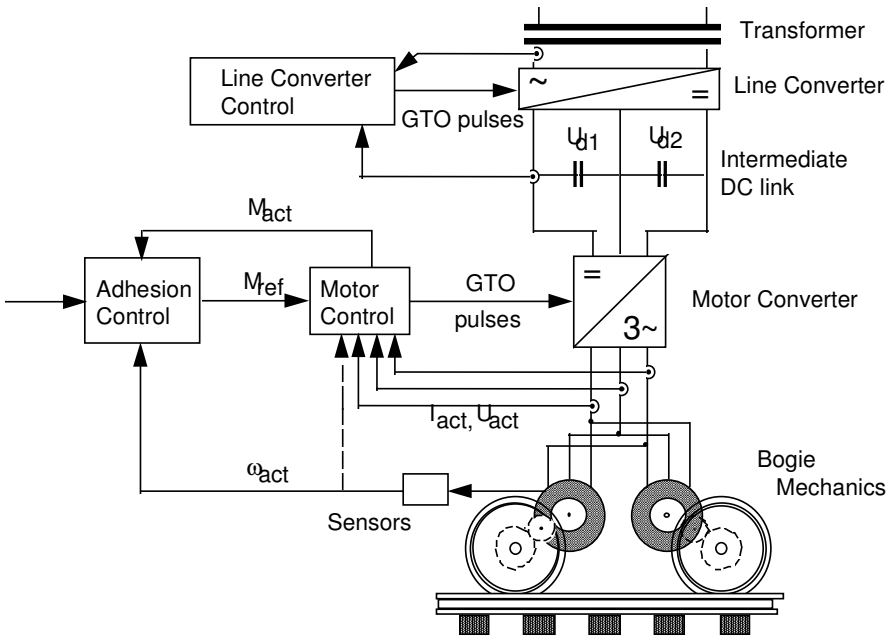

Fig. 3: Major control functions in an electrical locomotive.

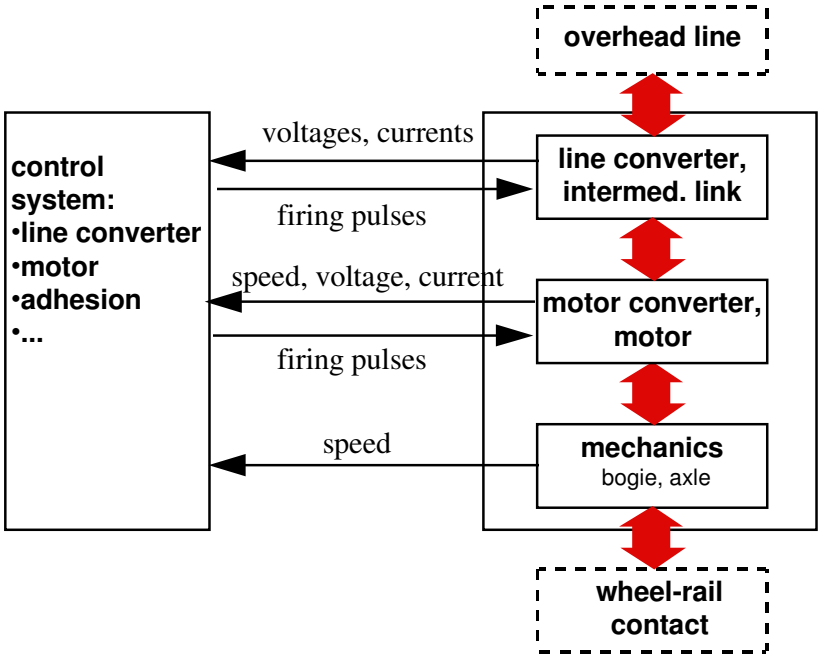

Fig. 4. Hardware-in-the-loop system structure

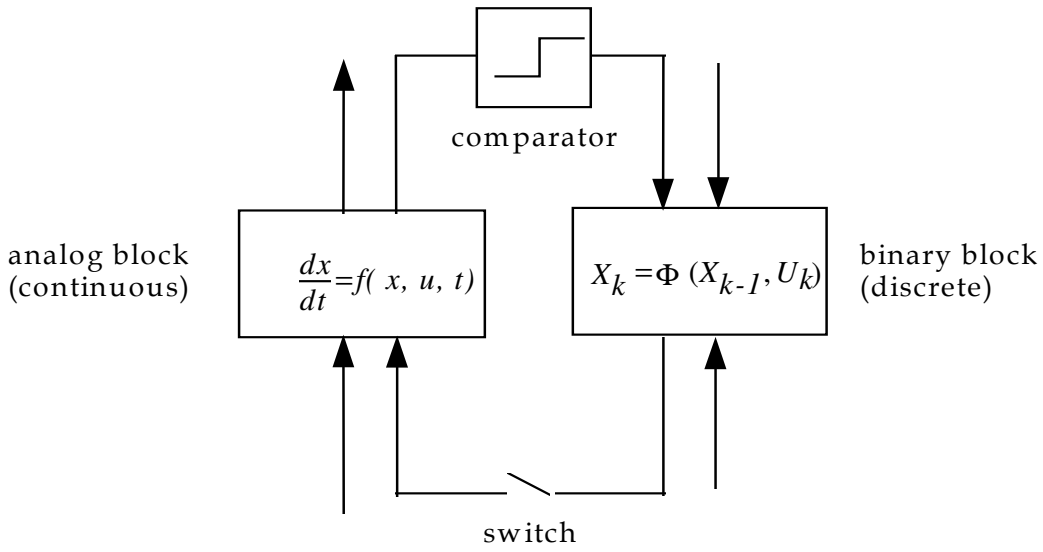

Fig. 5. Hybrid model structure

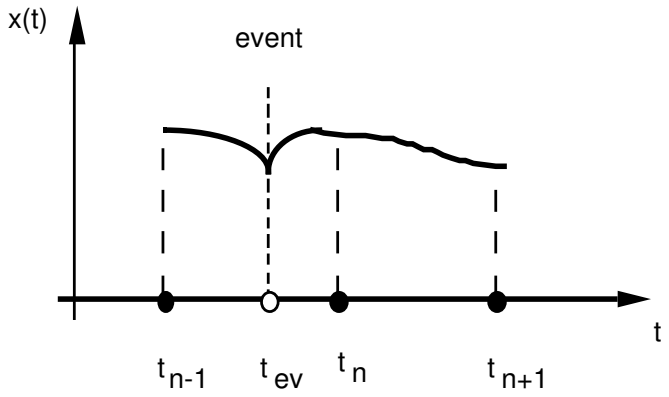

Fig. 6: External Event

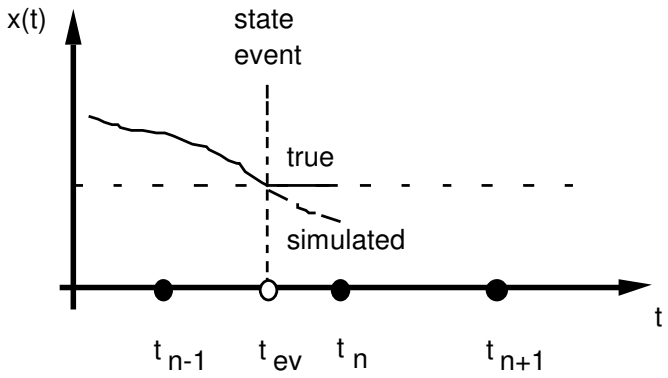

Fig. 7: Internal Event

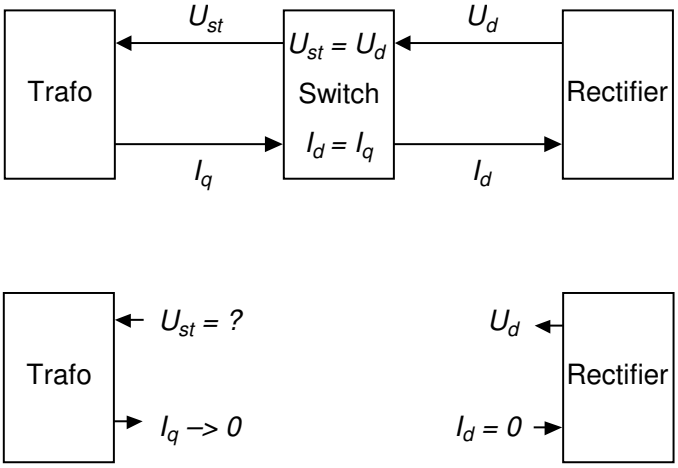

Fig. 8. Causality change in switch model: switch closed (upper), switch open (lower).

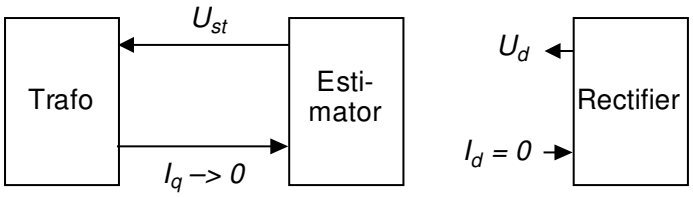

Fig. 9. Open Switch with Estimator

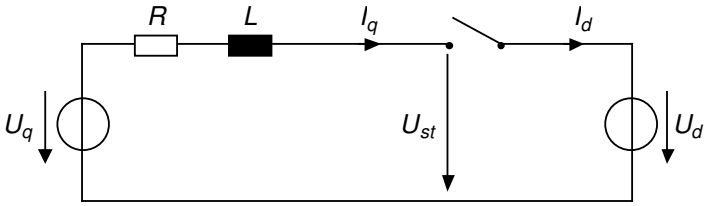

Fig. 10. Open Switch with R-L-Link

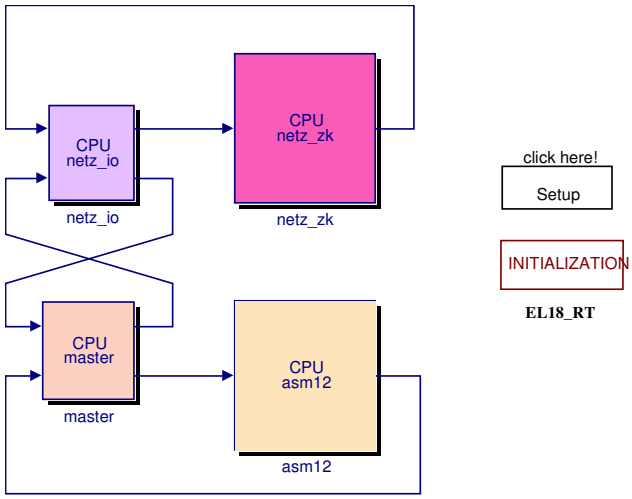

Fig. 11. Partitioning the Model.

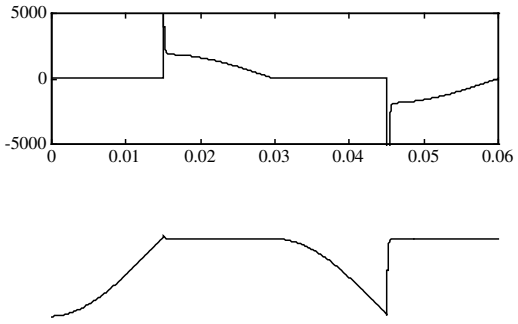

Fig. 12. Switch in R-L-Link. Switching times: 0.015 off, 0.03 on, 0.045 off. $k_{corr} = 0.25$.

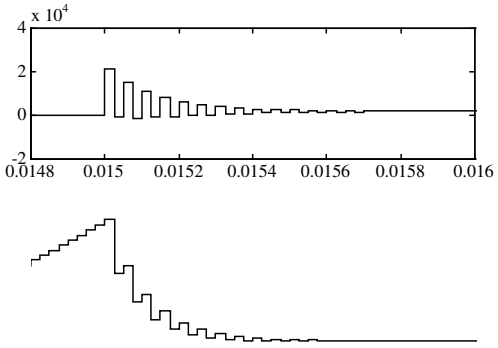

Fig. 13. Effect of modeling error in L with $k_{corr} = 0.25$

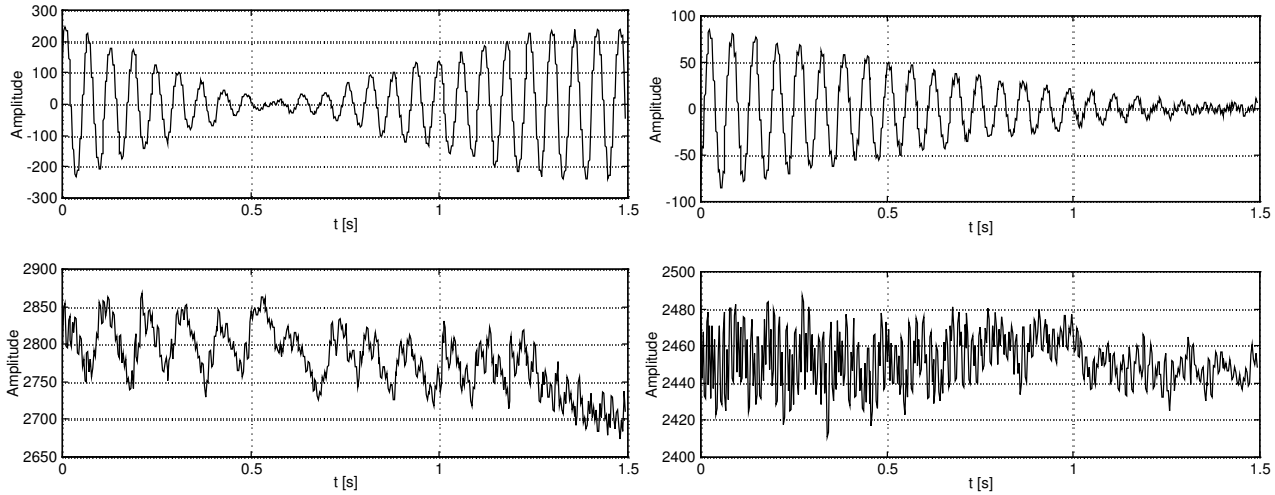

Fig. 14. Primary current (top row) and intermediate link voltage (bottom row) during change from driving to braking operation with field-weakening direct self control (left column) and indirect self control (right column).

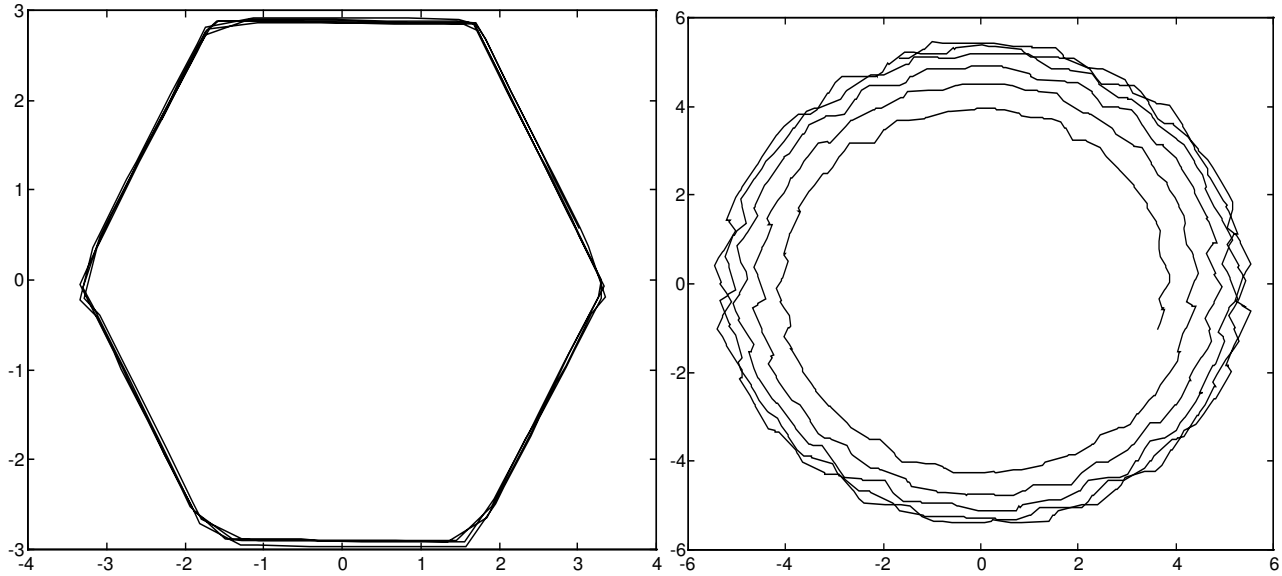

Fig. 15. Spatial stator flux trajectories during change from driving to braking operation with field-weakening direct self control (left) and indirect self control (right).

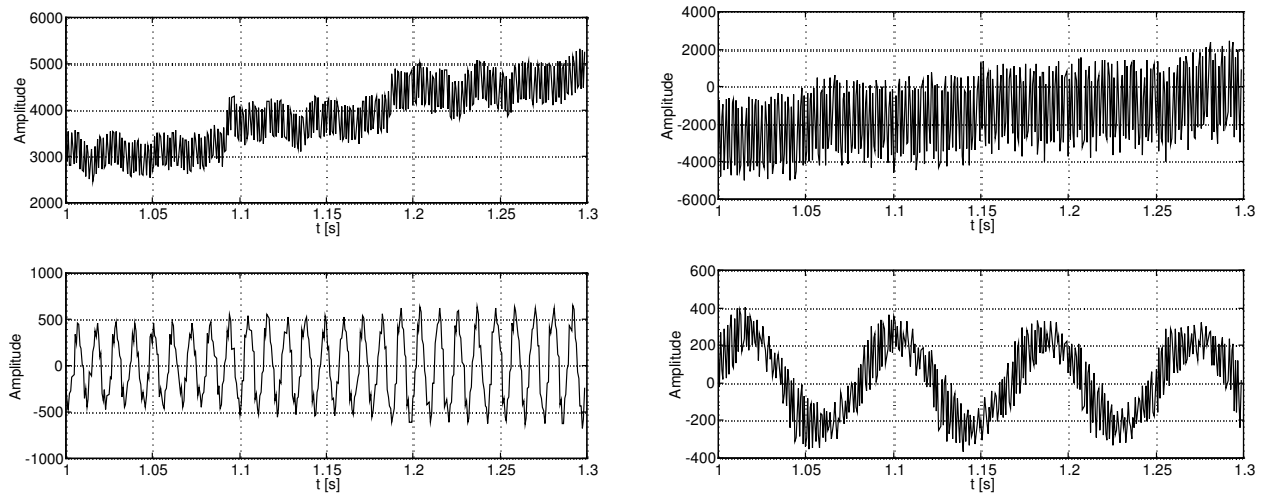

Fig. 16. Electrical motor torque (top row) and phase R motor current (bottom row) during change from driving to braking operation with field-weakening direct self control (left column) and indirect self control (right column).

About the Authors:

Peter Terwiesch received his M.Sc. degree in electrical engineering/automatic control from Karlsruhe Technical University, Germany, in 1991 and his Ph. D. degree in the same area from ETH Zurich, Switzerland, in 1994. He is a member of IEEE and its Control System Society and currently manager of the information technology and automation software department at ABB Corporate Research in Germany. The present paper is a result from his previous position at ABB Corporate Research in Switzerland, where he was managing the automatic control activities.

Thomas Keller holds an M.Sc. in electrical engineering from ETH Zurich, which he received in 1992. He is a group manager for propulsion system simulation and calculation within ABB Daimler-Benz Transportation (Switzerland).

Erich Scheiben received his M.Sc. in electrical engineering/automatic control from ETH Zurich in 1990. He is a member of IEEE and worked on the real-time simulation project in ABB Corporate Research (Switzerland) until he moved to ABB Daimler-Benz Transportation (Switzerland) together with the project in January 1997.