Systematic Separation of Electrical Power Systems for Hardware-in-the-Loop Simulation

Axel Kiffe, Katrin Witting and Frank Puschmann
dSPACE GmbH
Rathenaustraße 26, 33102 Paderborn, Germany
Tel.: +49 / (0) – 5251 1638 – 0
Fax: +49 / (0) – 5251 161980
E-Mail: {akiffe, kwitting, fpuschmann}@dspace.de
URL: http://www.dspace.com

Keywords

Real time simulation, Circuits, Modelling, Field Programmable Gate Array (FPGA)

Abstract

The increasing complexity of power grids makes hardware-in-the-loop simulations of electrical power systems more and more interesting for the industry. The simulation models are typically large. They often have to be split and used on several hardware platforms to allow efficient computations. Finding an appropriate position for splitting the simulation model requires expert knowledge about the electrical power system itself and about the functionality and limitations of the simulation hardware. In this paper, we present criteria that have to be taken into account when splitting an electrical power system model into subsystems. These criteria can serve as a basis for a systematic model splitting algorithm, which requires no special expert knowledge.

1. Introduction

Environmentally friendly energy concepts involve regenerative energy as the main source of energy. Power is generated from infinite natural resources such as wind, water, and the sun with nearly no emissions. Smart grids and smart houses typically include various regenerative energy installations, such as photovoltaic and solar thermal devices as well as wind turbines. However, developing such power systems is challenging, because the energy output of regenerative energy systems fluctuates throughout the day. Depending on the weather conditions, there are times of increased power generation and times of low power generation. A rough estimate of the expected energy output can be taken from the weather forecasts – but this data is just as unreliable as the weather itself. This is why the systems have to be highly flexible. They need smart control units to cope with the variations in power generation and demand, as well as sensible energy management systems, which shift less time-critical actions to times of high energy output.

Due to the increasing complexity of power grids, the interest in hardware-in-the-loop (HIL) simulations for power electronics and power electrical circuits, as well as for smart grids, is also increasing. HIL simulation is a well-known approach for testing control units in the automotive industry: The real plant is replaced by a real-time system that captures the outputs of the control unit, computes the reaction of the plant and returns the control values to the control unit. This allows for testing in the laboratory under reproducible conditions and reduces the risk of accidents when the real plant is used.

Creating real-time-capable models of power grids is an ambitious task due to the high number of components and the fast switching of power electronics. In most applications, the model has to be split into submodels that can be simulated in parallel on multiple cores, processors or FPGAs for an efficient real-time simulation of power grids.

Different approaches for splitting large simulation models into submodels have been described in the relevant literature, e.g., [1], [6] and [11]. In general, there are several properties that contribute to successful model splitting. First of all, the coupling of the subsystems should simulate the real behavior and especially maintain stability. Moreover, the subsystems' computational complexity and memory consumption should be more or less evenly distributed.

This paper investigates the criteria that evaluate different positions of the separation interface and support the user in finding the suitable positions for an efficient model separation.

The outline of the paper is as follows: section 2 summarizes the relevant basics of circuit simulation and illustrates them with an example. Section 3 deals with the principal approach for splitting electrical circuits and outlines the important properties for a suitable partitioning into submodels. Section 4 illustrates the main idea of this paper. It presents a systematic model splitting process that enables the user to find adequate splitting positions. This section also describes the mathematical background for the main steps of the procedure. Afterwards, section 5 discusses splitting positions as per the examples. The paper closes with a conclusion in section 6.

2. Basics on the State-Space Approach for Circuit Simulation

Specialist literature provides various algorithms and modeling approaches for simulating electrical power systems, e.g., [4], [10]. However, a discrete state-space approach as described in [3] is used as the basis for the computations and evaluation in this paper. For this approach, the physical model of the circuit consists of a linear and a nonlinear part. While the linear part (variables with subscript *L*) describes all passive devices of the circuit by a state-space representation, the nonlinear part consists of switching conductances and logics (variables with subscript *NL*). Figure 1 provides an example for this approach.

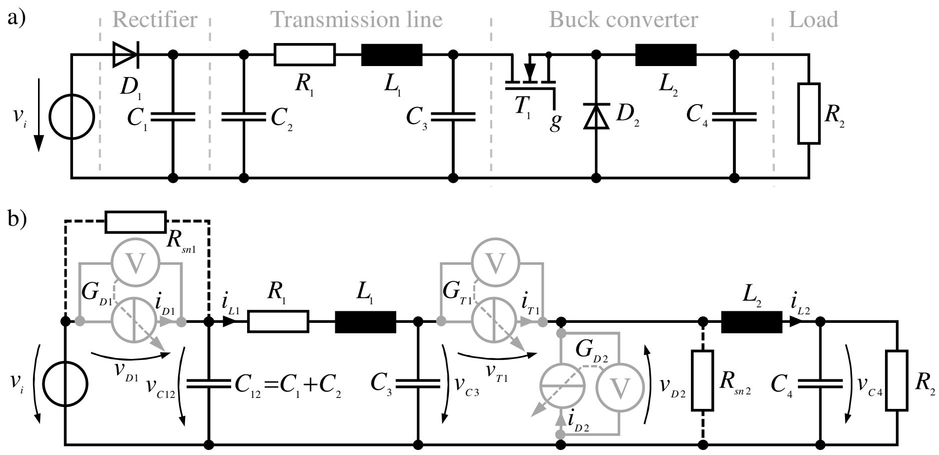


Fig. 1: Illustration of the state-space modeling approach.

The semiconductor switches in Fig. 1a are replaced by voltage-controlled current sources, as shown in Fig. 1b. Thus, the currents of the switches i_{D1}, i_{D2}, and i_{T1} can be interpreted as a vector of input values $\mathbf{u}_{NL}$ and the switch voltages v_{D1}, v_{D2}, and v_{T1} as output values $\mathbf{y}_{NL}$ of a continuous state-space representation with matrices $\mathbf{A}$ (continuous system matrix), $\mathbf{B}$ (input matrix), $\mathbf{C}$ (output matrix), and $\mathbf{D}$ (feed-through matrix). The values G_{D1}, G_{D2}, and G_{T1} are the conductance values of the switching elements and $\mathbf{G} = \mathrm{diag}\left(G_{D1}, G_{D2}, G_{T1}\right)$ is the conductance matrix. While a conducting semiconductor is represented by a small resistance, $G_{on} = R_{on}^{-1}$, the off-state is represented by an infinite resistor, i.e., $G_{off} = 0$.

The resistors R_{sn1} and R_{sn2} are artificial snubbers. They were added to avoid derivatives of the input to enable the description of the linear part as state-space representation.
As depicted in Fig. 2a, the switch voltages $\mathbf{y}_{NL}$ are fed back as switch currents $\mathbf{u}_{NL}$ by the extended conductance matrix

$$\tilde{\mathbf{G}} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{G} \end{bmatrix}. \tag{1}$$

The switch state is determined by switching events, which are detected by the simple switching logics shown in figure 2b (as an example for a diode and a MOSFET). Combining the extended conductance matrix with the continuous state-space representation yields a switch-state-dependent state-space representation:

$$\dot{\mathbf{x}} = \mathbf{A}^{(\xi)}\mathbf{x} + \mathbf{B}^{(\xi)}\mathbf{u}, \quad \mathbf{y} = \mathbf{C}^{(\xi)}\mathbf{x} + \mathbf{D}^{(\xi)}\mathbf{u} \tag{2}$$

with

$$\mathbf{A}^{(\xi)} = \mathbf{A} + \mathbf{B}\mathbf{G}^{(\xi)}\mathbf{N}\mathbf{C}, \mathbf{B}^{(\xi)} = \mathbf{B} + \mathbf{B}\mathbf{G}^{(\xi)}\mathbf{N}\mathbf{D}, \quad \mathbf{C}^{(\xi)} = \mathbf{N}\mathbf{C}, \quad \mathbf{D}^{(\xi)} = \mathbf{N}\mathbf{D}, \quad \mathbf{N} = \left(\mathbf{I} - \mathbf{D}\tilde{\mathbf{G}}^{(\xi)}\right)^{-1}. \tag{3}$$

Therein, $\mathbf{I}$ represents the identity matrix and ξ indicates the ξ-th switch state. Due to the combination, the input $\mathbf{u}_{NL}$ is no longer necessary and the relevant entries in the feedthrough- and input-matrices can be omitted. Thus, the input vector $\mathbf{u}$ reduces to $\mathbf{u}_L$. After discretizing the state-space representation, e.g., by using the Tustin or Backward Euler method [7], the following matrices of a switch-state-dependent, discrete state-space representations are obtained:

$$\mathbf{x}_{k+1} = \mathbf{\Phi}^{(\xi)}\mathbf{x}_k + \mathbf{H}^{(\xi)}\mathbf{u}_{L,k} \tag{4}$$

$$\mathbf{y}_k = \mathbf{C}_\mathbf{d}^{(\xi)}\mathbf{x}_k + \mathbf{D}_\mathbf{d}^{(\xi)}\mathbf{u}_{L,k} \tag{5}$$

Here, $\mathbf{\Phi}^{(\xi)}$, $\mathbf{H}^{(\xi)}$, $\mathbf{C}_\mathbf{d}^{(\xi)}$, and $\mathbf{D}_\mathbf{d}^{(\xi)}$ are the discrete system, input, output, and feed-through matrix for the ξ-th switch state. These representations in combination with the switching conditions yield the overall simulation algorithm, shown in Fig. 2c. Because of the dependency between the output- and feed-through-matrix and the switch-state ξ, the output equation (5) must be iterated for detecting conditional switching events.

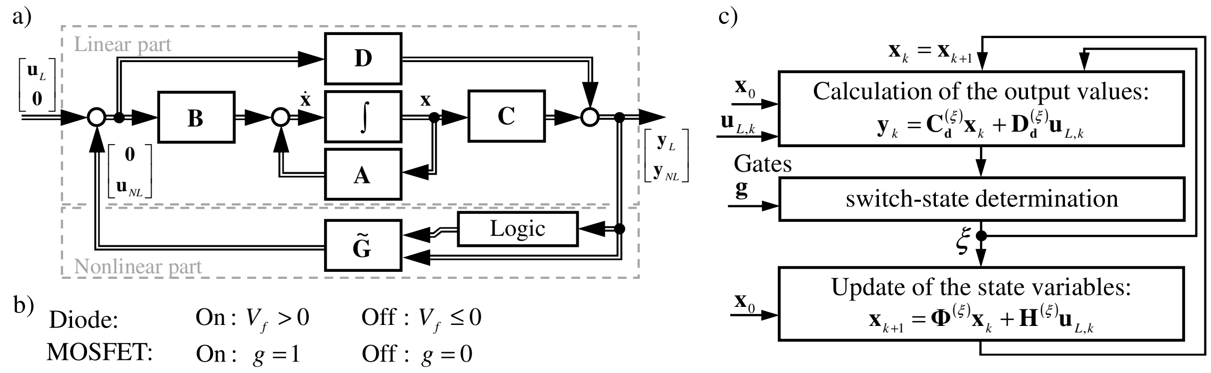

a)
Linear part
$\begin{bmatrix}\mathbf{u}_L\\ \mathbf{0}\end{bmatrix}$
D
B
$\dot{\mathbf{x}}$
$\int$
$\mathbf{x}$
C
A
$\begin{bmatrix}\mathbf{0}\\ \mathbf{u}_{NL}\end{bmatrix}$
$\begin{bmatrix}\mathbf{y}_L\\ \mathbf{y}_{NL}\end{bmatrix}$
Logic
$\tilde{\mathbf{G}}$
Nonlinear part
b)
Diode: On: $V_f > 0$ Off: $V_f \le 0$
MOSFET: On: $g = 1$ Off: $g = 0$
c)
$\mathbf{x}_k = \mathbf{x}_{k+1}$
$\mathbf{x}_0$
$\mathbf{u}_{L,k}$
Calculation of the output values:
$\mathbf{y}_k = \mathbf{C}_\mathbf{d}^{(\xi)}\mathbf{x}_k + \mathbf{D}_\mathbf{d}^{(\xi)}\mathbf{u}_{L,k}$
Gates
$\mathbf{g}$
switch-state determination
ξ
$\mathbf{x}_0$
Update of the state variables:
$\mathbf{x}_{k+1} = \mathbf{\Phi}^{(\xi)}\mathbf{x}_k + \mathbf{H}^{(\xi)}\mathbf{u}_{L,k}$

Fig. 2: a) Description of the overall circuit using voltage-controlled current sources.
b) Switch logic for diode and MOSFET.
c) Simulation algorithm of the described simulation approach.

3. Splitting of Power Electrical Circuits

Figure 3 shows an exemplary model splitting between the resistor R_1 and the inductor L_1 of the example in Fig. 1. The coupling values $i_{\mathbf{I,II}}$ and $v_{\mathbf{II,I}}$ are taken from the inductor current i_{L1} and the voltage of the corresponding measurement on the left part v_{RC} (Fig. 2b). The measurement values in one subsystem are source values in the other subsystem and vice versa. Both subsystems are represented by discretized switch-state-dependent state-space representations according to reference [3].

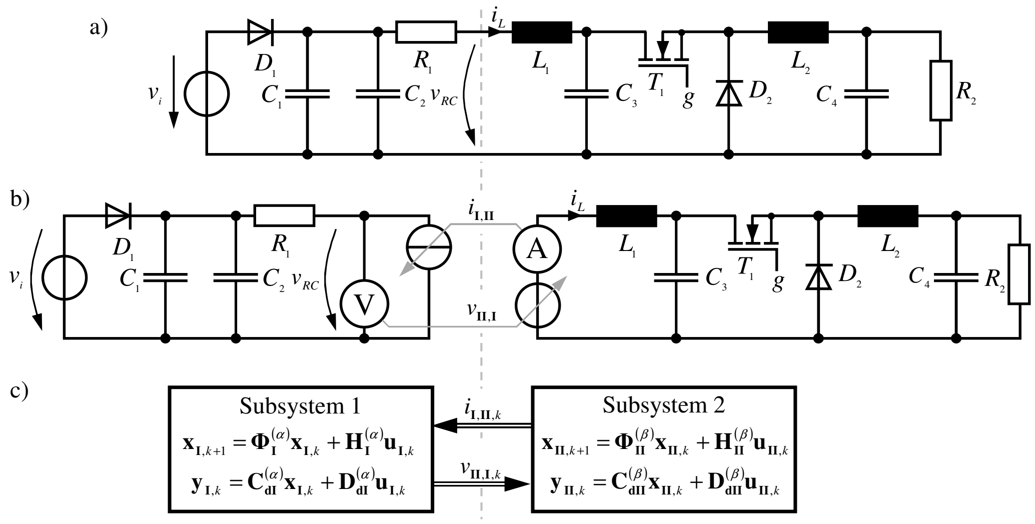

a)
b)
c)
v_i
D_1
C_1
R_1
C_2
v_{RC}
i_L
L_1
C_3
T_1
g
D_2
L_2
C_4
R_2
$i_{\mathrm{I,II}}$
$v_{\mathrm{II,I}}$
A
V
Subsystem 1
$\mathbf{x}_{\mathrm{I},k+1} = \mathbf{\Phi}_{\mathrm{I}}^{(\alpha)}\mathbf{x}_{\mathrm{I},k} + \mathbf{H}_{\mathrm{I}}^{(\alpha)}\mathbf{u}_{\mathrm{I},k}$
$\mathbf{y}_{\mathrm{I},k} = \mathbf{C}_{\mathrm{dI}}^{(\alpha)}\mathbf{x}_{\mathrm{I},k} + \mathbf{D}_{\mathrm{dI}}^{(\alpha)}\mathbf{u}_{\mathrm{I},k}$
$i_{\mathrm{I,II},k}$
$v_{\mathrm{II,I},k}$
Subsystem 2
$\mathbf{x}_{\mathrm{II},k+1} = \mathbf{\Phi}_{\mathrm{II}}^{(\beta)}\mathbf{x}_{\mathrm{II},k} + \mathbf{H}_{\mathrm{II}}^{(\beta)}\mathbf{u}_{\mathrm{II},k}$
$\mathbf{y}_{\mathrm{II},k} = \mathbf{C}_{\mathrm{dII}}^{(\beta)}\mathbf{x}_{\mathrm{II},k} + \mathbf{D}_{\mathrm{dII}}^{(\beta)}\mathbf{u}_{\mathrm{II},k}$

Fig. 3: Exemplary model splitting of the example in Fig. 1a.

As illustrated in Fig. 3c, $i_{\mathrm{I,II},k}$ is part of the input vector of the first subsystem ($\mathbf{u}_{\mathrm{I},k}$) and the output vector of the second subsystem ($\mathbf{y}_{\mathrm{II},k}$). The opposite applies for the voltage $\mathbf{v}_{\mathrm{II,I},k}$. This results in an implicit coherence (algebraic loop). A simple and pragmatic way to break these loops is to insert additional delays between the measurements and sources at the splitting position. They allow for the parallel computation of both parts. Regarding the computation effort and the ability to implement this on FPGAs, this approach is very efficient and does not increase the computation time significantly. However, splitting positions might exist where the delays influence the simulation quality, and, in a worst-case scenario, even yield an unstable simulation. This paper primarily deals with approaches to find stable and accurate splitting positions.
In general, the following effects have to be considered to find a suitable position for the model splitting interface:

Conditional switching events:

Switching events (known as forced switching events) are controlled by external values such as the gate signal g of the MOSFET, while others (known as natural switching events) evaluate internal values, such as the diode voltage. Furthermore, forced switching events can trigger other natural switching events, which must be dealt with immediately. Conditional switching must be limited to one subsystem to ensure a correct switch state detection and keep the computation effort low.

Stiff coupling:

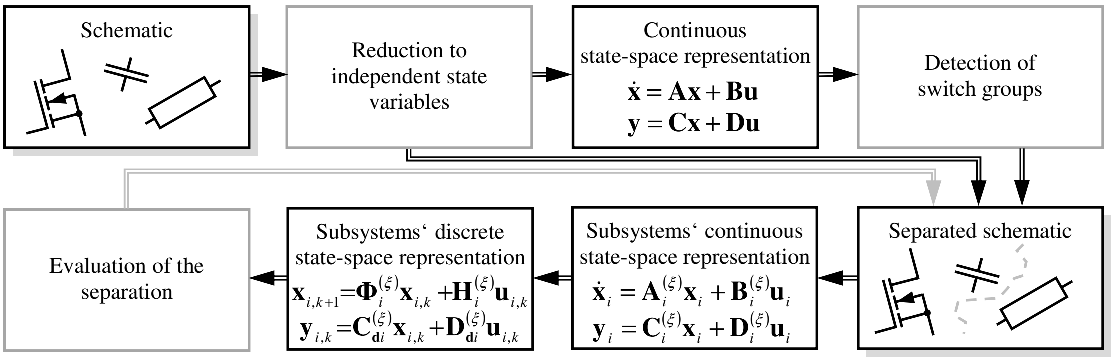
The reaction of one subsystem to another must be slow to simplify the solution of implicit equations and yield accurate simulation results.

Effective calculation of implicit coherence:

The splitting interface should be positioned in such a way that the implicit equations can be solved efficiently by the real-time system. For example, divisions on FPGA-based real-time systems should be avoided because they are very time-consuming.

4. A Systematic Model-Splitting Procedure

To consider the requirements mentioned at the end of the previous section, the procedure shown in Fig. 4 is the proposed approach. It starts with the overall circuit that is not split yet. Afterwards, components that lead to dependent state variables are detected and highlighted in the circuit. In these groups of dependent states, the circuit must not be divided. Based on the continuous state-space representation, which is determined in the third step and represents the continuous linear part of the described modeling approach, switch groups are detected. They are also highlighted in the circuit to illustrate the dependencies and make it easier for the user to insert possible separation interfaces.


Schematic
Reduction to independent state variables
Continuous state-space representation
$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$
$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}$
Detection of switch groups
Evaluation of the separation
Subsystems‘ discrete state-space representation
$\mathbf{x}_{i,k+1} = \mathbf{\Phi}_i^{(\xi)}\mathbf{x}_{i,k} + \mathbf{H}_i^{(\xi)}\mathbf{u}_{i,k}$
$\mathbf{y}_{i,k} = \mathbf{C}_{\mathbf{d}i}^{(\xi)}\mathbf{x}_{i,k} + \mathbf{D}_{\mathbf{d}i}^{(\xi)}\mathbf{u}_{i,k}$
Subsystems‘ continuous state-space representation
$\dot{\mathbf{x}}_i = \mathbf{A}_i^{(\xi)}\mathbf{x}_i + \mathbf{B}_i^{(\xi)}\mathbf{u}_i$
$\mathbf{y}_i = \mathbf{C}_i^{(\xi)}\mathbf{x}_i + \mathbf{D}_i^{(\xi)}\mathbf{u}_i$
Separated schematic
Fig. 4: Illustration of the systematic model splitting procedure.

Based on the information about dependent states and switches, the user inserts the separation interface at all remaining and interesting positions. Every candidate for the splitting is automatically evaluated successively regarding stability and accuracy properties as well as hardware limitations, such as memory consumption and maximum dimension of the state-space representations. Finally, a report is generated, which includes appropriate recommendations. The following subsections describe the different steps in detail, except for the circuit analysis and discretization. The descriptions for these steps can be found in references [7] and [9], respectively.

4.1 Dependent Components

Two types of dependent components can be directly detected and highlighted in the unsplit circuit to reduce the possible positions of the splitting interfaces and make the insertion of the splitting interfaces easier for the user.

Dependent state variables

Loops of capacitors such as the capacitors C_1 and C_2 in the example or nodes that connect branches of inductances indicate dependent state variables. In this case, one of the capacitor voltages of the loop or one of the inductor currents of the node are forced by the other. This represents a direct stiff coupling, which must not be separated. Thus, searching for appropriate loops or nodes in a netlist is mandatory.

Switch group detection

As explained in section 2, the switch-dependent conductance matrix $\mathbf{G}$ feeds back the switch voltages $\mathbf{y}_{NL}$ as switch currents $\mathbf{u}_{NL}$ while the feed-through submatrix $\mathbf{D}_{22}$ represents the direct influence without any delays from the switch current $\mathbf{u}_{NL}$ back to the switch voltages $\mathbf{y}_{NL}$:

$$\begin{bmatrix} \mathbf{y}_L \\ \mathbf{y}_{NL} \end{bmatrix} = \begin{bmatrix} \mathbf{C}_1 \\ \mathbf{C}_2 \end{bmatrix} \cdot \mathbf{x} + \begin{bmatrix} \mathbf{D}_{11} & \mathbf{D}_{12} \\ \mathbf{D}_{21} & \mathbf{D}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{u}_L \\ \mathbf{u}_{NL} \end{bmatrix}. \tag{6}$$

Therefore, $\mathbf{D}_{22}$ describes the direct coupling between different switches and enables the detection of switch groups that should not be divided. To detect switch groups, every row p of matrix

$$\mathbf{D}_{22} = \left(d_{22,pq}\right)_{p=1\ldots r,q=1\ldots r}, \mathbf{D}_{22} \in \mathbb{R}^{r\times r} \tag{7}$$

is evaluated row-by-row relative to column entries $d_{22,pq}$ that are unequal to zero. While the entries on the main diagonal represent the effects of the switch on itself, all other indicate a coupling between the q-th and the p-th switch. Regarding the splitting of a circuit into subsystems, the focus is on devices that change their switch state conditionally because of forced switching events. Thus, only those rows of $\mathbf{D}_{22}$ are evaluated that refer to devices whose switching is caused by natural events, e.g., diodes.

Additionally, to avoid multiple detection of switch groups because of the symmetry of $\mathbf{D}_{22}$ (cf. [4]), the entries unequal to zero, their mirrored entries on the main diagonal, and the entry of the main diagonal should be set to zero ($d_{22,pp} = d_{22,pq} = d_{22,qp} = 0$) after each evaluation of row p .

4.2 Evaluation of the Separation

As illustrated in Fig. 3 the separation of an electrical circuit leads to almost independent state-space systems, typically of lower dimensions. The only coupling between the subsystems is given through the coupling variables, more precisely, the currents and voltages at the splitting points. In this paper, the formulas are written down for only one splitting point to keep it simple. However, all the following considerations can be easily extended to the case of multiple splittings.

For the following considerations, the discrete state-space equations of two subsystems $\mathbf{I}$ and $\mathbf{II}$ are considered. The inputs of the subsystems are split into two categories: arbitrary inputs $\mathbf{u}_{\mathrm{I,int},k} / \mathbf{u}_{\mathrm{II,int},k}$ (typically splitting-independent sources within the subsystem) and inputs at the splitting point $\mathbf{u}_{\mathbf{I,MS},k}$ / $\mathbf{u}_{\mathbf{II,MS},k}$. Similarly, for each subsystem the output vector is also split into arbitrary outputs (typically splitting-independent measurements within the subsystem as well as switch voltages) $\mathbf{y}_{\mathbf{I,m},k} / \mathbf{y}_{\mathbf{II,m},k}$ and outputs at the splitting point $\mathbf{y}_{\mathbf{I,MS},k} / \mathbf{y}_{\mathbf{II,MS},k}$. The subscript (α) and (β) represent the switch states of subsystem $\mathbf{I}$ and $\mathbf{II}$, respectively. This leads to the following state-space equations:

Subsystem I

$$\mathbf{x}_{\mathrm{I},k+1} = \mathbf{\Phi}_{\mathbf{I}}^{(\alpha)} \mathbf{x}_{\mathrm{I},k} + \begin{bmatrix} \mathbf{H}_{\mathbf{I,u}}^{(\alpha)} & \mathbf{H}_{\mathbf{I,MS}}^{(\alpha)} \end{bmatrix} \begin{bmatrix} \mathbf{u}_{\mathrm{I,int},k} \\ \mathbf{u}_{\mathbf{I,MS},k} \end{bmatrix}, \quad \begin{bmatrix} \mathbf{y}_{\mathbf{I,m},k} \\ \mathbf{y}_{\mathbf{I,MS},k} \end{bmatrix} = \begin{bmatrix} \mathbf{C}_{\mathbf{dI,m}}^{(\alpha)} \\ \mathbf{C}_{\mathbf{dI,MS}}^{(\alpha)} \end{bmatrix} \mathbf{x}_{\mathrm{I},k} + \begin{bmatrix} \mathbf{D}_{\mathbf{dI,m,u}}^{(\alpha)} & \mathbf{D}_{\mathbf{dI,m,MS}}^{(\alpha)} \\ \mathbf{D}_{\mathbf{dI,MS,u}}^{(\alpha)} & \mathbf{D}_{\mathbf{dI,MS,MS}}^{(\alpha)} \end{bmatrix} \begin{bmatrix} \mathbf{u}_{\mathrm{I,int},k} \\ \mathbf{u}_{\mathbf{I,MS},k} \end{bmatrix} \quad (8)$$

Subsystem II

$$\mathbf{x}_{\mathrm{II},k+1} = \mathbf{\Phi}_{\mathbf{II}}^{(\beta)} \mathbf{x}_{\mathrm{II},k} + \begin{bmatrix} \mathbf{H}_{\mathbf{II,u}}^{(\beta)} & \mathbf{H}_{\mathbf{II,MS}}^{(\beta)} \end{bmatrix} \begin{bmatrix} \mathbf{u}_{\mathrm{II,int},k} \\ \mathbf{u}_{\mathbf{II,MS},k} \end{bmatrix}, \quad \begin{bmatrix} \mathbf{y}_{\mathbf{II,m},k} \\ \mathbf{y}_{\mathbf{II,MS},k} \end{bmatrix} = \begin{bmatrix} \mathbf{C}_{\mathbf{dII,m}}^{(\beta)} \\ \mathbf{C}_{\mathbf{dII,MS}}^{(\beta)} \end{bmatrix} \mathbf{x}_{\mathrm{II},k} + \begin{bmatrix} \mathbf{D}_{\mathbf{dII,m,u}}^{(\beta)} & \mathbf{D}_{\mathbf{dII,m,MS}}^{(\beta)} \\ \mathbf{D}_{\mathbf{dII,MS,u}}^{(\beta)} & \mathbf{D}_{\mathbf{dII,MS,MS}}^{(\beta)} \end{bmatrix} \begin{bmatrix} \mathbf{u}_{\mathrm{II,int},k} \\ \mathbf{u}_{\mathbf{II,MS},k} \end{bmatrix} \quad (9)$$

The two subsystems are coupled via the in- and outputs at the splitting points: $\mathbf{u}_{\mathbf{II,MS},k} = \mathbf{y}_{\mathbf{I,MS},k}$, $\mathbf{u}_{\mathbf{I,MS},k} = \mathbf{y}_{\mathbf{II,MS},k}$. Equation (8) and (9), together with the coupling equations can be transformed to the following representation:

Subsystem I

$$\mathbf{x}_{\mathrm{I},k+1} = \mathbf{\Phi}_{\mathbf{I}}^{(\alpha)} \mathbf{x}_{\mathrm{I},k} + \mathbf{H}_{\mathbf{I,u}}^{(\alpha)} \mathbf{u}_{\mathrm{I,int},k} + \mathbf{H}_{\mathbf{I},i,\mathbf{MS}}^{(\alpha)} \mathbf{u}_{\mathbf{I,MS},k}, \quad \mathbf{y}_{\mathbf{I,m},k} = \mathbf{C}_{\mathbf{dI,m}}^{(\alpha)} \mathbf{x}_{\mathrm{I},k} + \mathbf{D}_{\mathbf{dI,m,u}}^{(\alpha)} \mathbf{u}_{\mathrm{I,int},k} + \mathbf{D}_{\mathbf{dI,m,MS}}^{(\alpha)} \mathbf{u}_{\mathbf{I,MS},k} \quad (10)$$

$$\mathbf{y}_{\mathbf{I,MS},k} = \mathbf{u}_{\mathbf{II,MS},k} = \mathbf{C}_{\mathbf{dI,MS}}^{(\alpha)} \mathbf{x}_{\mathrm{I},k} + \mathbf{D}_{\mathbf{dI,MS,u}}^{(\alpha)} \mathbf{u}_{\mathrm{I,int},k} + \mathbf{D}_{\mathbf{dI,MS,MS}}^{(\alpha)} \mathbf{u}_{\mathbf{I,MS},k} \quad (11)$$

Subsystem II

$$\mathbf{x}_{\mathrm{II},k+1} = \mathbf{\Phi}_{\mathbf{II}}^{(\beta)} \mathbf{x}_{\mathrm{II},k} + \mathbf{H}_{\mathbf{II,u}}^{(\beta)} \mathbf{u}_{\mathrm{II,int},k} + \mathbf{H}_{\mathbf{II,MS}}^{(\beta)} \mathbf{u}_{\mathbf{II,MS},k}, \quad \mathbf{y}_{\mathbf{II,m},k} = \mathbf{C}_{\mathbf{dII,m}}^{(\beta)} \mathbf{x}_{\mathrm{II},k} + \mathbf{D}_{\mathbf{dII,m,u}}^{(\beta)} \mathbf{u}_{\mathrm{II,int},k} + \mathbf{D}_{\mathbf{dII,m,MS}}^{(\beta)} \mathbf{u}_{\mathbf{II,MS},k} \quad (12)$$

$$\mathbf{y}_{\mathbf{II,MS},k} = \mathbf{u}_{\mathbf{I,MS},k} = \mathbf{C}_{\mathbf{dII,MS}}^{(\beta)} \mathbf{x}_{\mathrm{II},k} + \mathbf{D}_{\mathbf{dII,MS,u}}^{(\beta)} \mathbf{u}_{\mathrm{II,int},k} + \mathbf{D}_{\mathbf{dII,MS,MS}}^{(\beta)} \mathbf{u}_{\mathbf{II,MS},k} \quad (13)$$

After some reorganization of equations (11) and (13), the coupling behavior can be concluded in a system of linear equations:

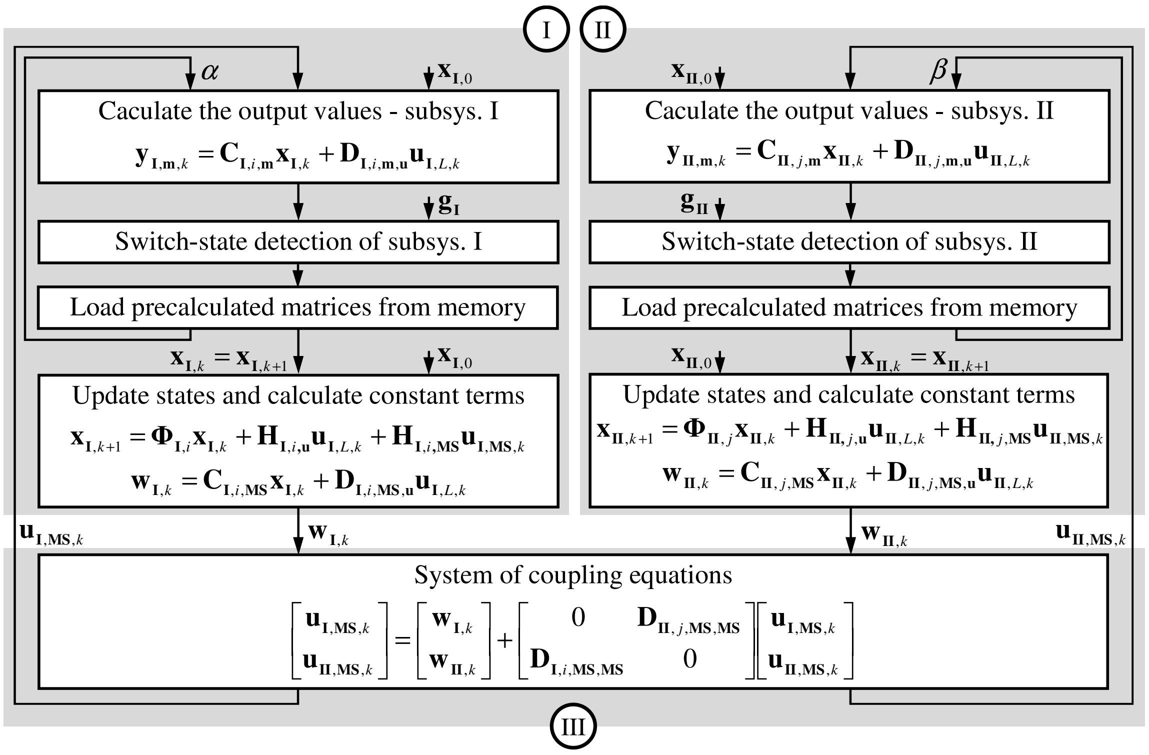
$$\begin{bmatrix} \mathbf{u}_{\mathbf{I,MS},k} \\ \mathbf{u}_{\mathbf{II,MS},k} \end{bmatrix} = \begin{bmatrix} \mathbf{w}_{\mathbf{II},k} \\ \mathbf{w}_{\mathbf{I},k} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{D}_{\mathbf{dII,MS,MS}}^{(\beta)} \\ \mathbf{D}_{\mathbf{dI,MS,MS}}^{(\alpha)} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u}_{\mathbf{I,MS},k} \\ \mathbf{u}_{\mathbf{II,MS},k} \end{bmatrix} \quad (14)$$

with

$$\mathbf{w}_{\mathbf{I},k} = \mathbf{C}_{\mathbf{dI,MS}}^{(\alpha)} \mathbf{x}_{\mathrm{I},k} + \mathbf{D}_{\mathbf{dI,MS,u}}^{(\alpha)} \mathbf{u}_{\mathrm{I,int},k} \text{ and } \mathbf{w}_{\mathbf{II},k} = \mathbf{C}_{\mathbf{dII,MS}}^{(\beta)} \mathbf{x}_{\mathrm{II},k} + \mathbf{D}_{\mathbf{dII,MS,u}}^{(\beta)} \mathbf{u}_{\mathrm{II,int},k} . \quad (15)$$

As can be seen, equation (15) depends only on the switch state of the relevant subsystem, while equation (14) depends on both switch states and yields $2^{n_\alpha + n_\beta}$ representations. Thus, this small system of equations can be solved numerically during simulation with low memory use. An overview of the deductive simulation algorithm is shown in Fig. 5. Starting with initial values $\mathbf{u}_{\mathbf{I,MS},0}$ and $\mathbf{u}_{\mathbf{II,MS},0}$ as well as $\mathbf{x}_{\mathrm{I},0}$ and $\mathbf{x}_{\mathrm{II},0}$, parts I and II are calculated in parallel (equations (10), (12), and (15)).

Afterwards, on the basis of on $\mathbf{w}_{\mathrm{I},k}$ and $\mathbf{w}_{\mathrm{II},k}$, the implicit system of equations (14) is solved and $\mathbf{u}_{\mathrm{I,MS},k}$ as well as $\mathbf{u}_{\mathrm{II,MS},k}$ are provided for the next simulation step. Methods for solving (14) in part III are discussed in the following subsections:



I
II
α
$\mathbf{x}_{\mathrm{I},0}$
Caculate the output values - subsys. I
$\mathbf{y}_{\mathrm{I,m},k} = \mathbf{C}_{\mathrm{I},i,\mathrm{m}}\mathbf{x}_{\mathrm{I},k} + \mathbf{D}_{\mathrm{I},i,\mathrm{m,u}}\mathbf{u}_{\mathrm{I},L,k}$
$\mathbf{g}_{\mathrm{I}}$
Switch-state detection of subsys. I
Load precalculated matrices from memory
$\mathbf{x}_{\mathrm{I},k} = \mathbf{x}_{\mathrm{I},k+1}$
$\mathbf{x}_{\mathrm{I},0}$
Update states and calculate constant terms
$\mathbf{x}_{\mathrm{I},k+1} = \mathbf{\Phi}_{\mathrm{I},i}\mathbf{x}_{\mathrm{I},k} + \mathbf{H}_{\mathrm{I},i,\mathrm{u}}\mathbf{u}_{\mathrm{I},L,k} + \mathbf{H}_{\mathrm{I},i,\mathrm{MS}}\mathbf{u}_{\mathrm{I,MS},k}$
$\mathbf{w}_{\mathrm{I},k} = \mathbf{C}_{\mathrm{I},i,\mathrm{MS}}\mathbf{x}_{\mathrm{I},k} + \mathbf{D}_{\mathrm{I},i,\mathrm{MS,u}}\mathbf{u}_{\mathrm{I},L,k}$
$\mathbf{x}_{\mathrm{II},0}$
β
Caculate the output values - subsys. II
$\mathbf{y}_{\mathrm{II,m},k} = \mathbf{C}_{\mathrm{II},j,\mathrm{m}}\mathbf{x}_{\mathrm{II},k} + \mathbf{D}_{\mathrm{II},j,\mathrm{m,u}}\mathbf{u}_{\mathrm{II},L,k}$
$\mathbf{g}_{\mathrm{II}}$
Switch-state detection of subsys. II
Load precalculated matrices from memory
$\mathbf{x}_{\mathrm{II},0}$
$\mathbf{x}_{\mathrm{II},k} = \mathbf{x}_{\mathrm{II},k+1}$
Update states and calculate constant terms
$\mathbf{x}_{\mathrm{II},k+1} = \mathbf{\Phi}_{\mathrm{II},j}\mathbf{x}_{\mathrm{II},k} + \mathbf{H}_{\mathrm{II},j,\mathrm{u}}\mathbf{u}_{\mathrm{II},L,k} + \mathbf{H}_{\mathrm{II},j,\mathrm{MS}}\mathbf{u}_{\mathrm{II,MS},k}$
$\mathbf{w}_{\mathrm{II},k} = \mathbf{C}_{\mathrm{II},j,\mathrm{MS}}\mathbf{x}_{\mathrm{II},k} + \mathbf{D}_{\mathrm{II},j,\mathrm{MS,u}}\mathbf{u}_{\mathrm{II},L,k}$
$\mathbf{u}_{\mathrm{I,MS},k}$
$\mathbf{w}_{\mathrm{I},k}$
$\mathbf{w}_{\mathrm{II},k}$
$\mathbf{u}_{\mathrm{II,MS},k}$
System of coupling equations
$\begin{bmatrix}\mathbf{u}_{\mathrm{I,MS},k}\\ \mathbf{u}_{\mathrm{II,MS},k}\end{bmatrix} = \begin{bmatrix}\mathbf{w}_{\mathrm{I},k}\\ \mathbf{w}_{\mathrm{II},k}\end{bmatrix} + \begin{bmatrix}0 & \mathbf{D}_{\mathrm{II},j,\mathrm{MS,MS}}\\ \mathbf{D}_{\mathrm{I},i,\mathrm{MS,MS}} & 0\end{bmatrix}\begin{bmatrix}\mathbf{u}_{\mathrm{I,MS},k}\\ \mathbf{u}_{\mathrm{II,MS},k}\end{bmatrix}$
III

Fig. 5: Overview of the simulation algorithm for a circuit, separated in two parts.

Direct solution:
The direct solution for solving (14), which is equivalent to (16), leads to a simulation algorithm similar to the State-Space Nodal Solver (cf. [6]).

$$\begin{bmatrix}\mathbf{I} & -\mathbf{D}^{(\beta)}_{\mathrm{dII,MS,MS}}\\ -\mathbf{D}^{(\alpha)}_{\mathrm{dI,MS,MS}} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{u}_{\mathrm{I,MS},k}\\ \mathbf{u}_{\mathrm{II,MS},k}\end{bmatrix} = \begin{bmatrix}\mathbf{w}_{\mathrm{II},k}\\ \mathbf{w}_{\mathrm{I},k}\end{bmatrix} \tag{16}$$

Regarding methods for solving systems of linear equations with constant coefficients, such as the LU decomposition, the condition number κ (cf. [12]) is a measure that represents both convergence speed and accuracy of the solution.

Fixed point iteration:
To avoid a high computation effort and getting a well-suited representation for FPGA-based simulation, the fixed point iteration that uses only a few steps is a good option and does not need divisions. In this case, equation (14) is reorganized to:

$$\begin{bmatrix}\mathbf{u}_{\mathrm{I,MS},k,p+1}\\ \mathbf{u}_{\mathrm{II,MS},k,p+1}\end{bmatrix} = \begin{bmatrix}\mathbf{w}_{\mathrm{II},k}\\ \mathbf{w}_{\mathrm{I},k}\end{bmatrix} + \begin{bmatrix}\mathbf{0} & \mathbf{D}^{(\beta)}_{\mathrm{dII,MS,MS}}\\ \mathbf{D}^{(\alpha)}_{\mathrm{dI,MS,MS}} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{u}_{\mathrm{I,MS},k,p}\\ \mathbf{u}_{\mathrm{II,MS},k,p}\end{bmatrix}, \qquad \begin{bmatrix}\mathbf{u}_{\mathrm{I,MS},k,0}\\ \mathbf{u}_{\mathrm{II,MS},k,0}\end{bmatrix} = \begin{bmatrix}\mathbf{u}_{\mathrm{I,MS},k-1}\\ \mathbf{u}_{\mathrm{II,MS},k-1}\end{bmatrix} \tag{17}$$

Therein p represents the iteration step. For the fixed-point iteration, the spectral radius ρ is a crucial criterion for stability and convergence speed [8]:

$$\rho = \max\{|\mathbf{eig}(\mathbf{J})|\} < 1, \qquad \mathbf{J} = \begin{bmatrix}\mathbf{0} & \mathbf{D}^{(\beta)}_{\mathrm{dII,MS,MS}}\\ \mathbf{D}^{(\alpha)}_{\mathrm{dI,MS,MS}} & \mathbf{0}\end{bmatrix}. \tag{18}$$

Delay insertion:
The result cannot be equation (14) if the implicit equation is disrupted by inserting additional delays between the measurements and sources at the splitting position (cf. section 3). This occurrence yields the following system of equations, with independent rows.

$$\begin{bmatrix}\mathbf{u}_{\mathbf{I},\mathbf{MS},k}\\ \mathbf{u}_{\mathbf{II},\mathbf{MS},k}\end{bmatrix}=\begin{bmatrix}\mathbf{w}_{\mathbf{II},k}\\ \mathbf{w}_{\mathbf{I},k}\end{bmatrix}+\left[\begin{array}{c|c}\mathbf{0} & \mathbf{D}^{(\beta)}_{\mathbf{dII},\mathbf{MS},\mathbf{MS}}\\ \hline \mathbf{D}^{(\alpha)}_{\mathbf{dI},\mathbf{MS},\mathbf{MS}} & \mathbf{0}\end{array}\right]\begin{bmatrix}\mathbf{u}_{\mathbf{I},\mathbf{MS},k-1}\\ \mathbf{u}_{\mathbf{II},\mathbf{MS},k-1}\end{bmatrix} \tag{19}$$

The equations can be assigned to part I and II of Fig. 5, enabling a full parallel calculation. The delays at the splitting point ($\mathbf{u}_{\mathbf{I},\mathbf{MS},k}=\mathbf{y}_{\mathbf{II},\mathbf{MS},k-1}$ and $\mathbf{u}_{\mathbf{II},\mathbf{MS},k}=\mathbf{y}_{\mathbf{I},\mathbf{MS},k-1}$) can be interpreted as discrete transfer functions, which can be considered in the discrete state space representation. This yields equations (20)-(23) with an increased number of state variables:

Subsystem I

$$\begin{bmatrix}\mathbf{x}_{\mathbf{I},k+1}\\ \mathbf{y}_{\mathbf{I},\mathbf{MS},k}\end{bmatrix}=\begin{bmatrix}\mathbf{\Phi}^{(\alpha)}_{\mathbf{I}} & \mathbf{0}\\ \mathbf{C}^{(\alpha)}_{\mathbf{dI},\mathbf{MS}} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathbf{I},k}\\ \mathbf{y}_{\mathbf{I},\mathbf{MS},k-1}\end{bmatrix}+\begin{bmatrix}\mathbf{H}^{(\alpha)}_{\mathbf{I},\mathbf{u}}\\ \mathbf{D}^{(\alpha)}_{\mathbf{dI},\mathbf{MS},\mathbf{u}}\end{bmatrix}\mathbf{u}_{\mathbf{I},\mathrm{int},k}+\begin{bmatrix}\mathbf{H}^{(\alpha)}_{\mathbf{I},i,\mathbf{MS}}\\ \mathbf{D}^{(\alpha)}_{\mathbf{dI},\mathbf{MS},\mathbf{MS}}\end{bmatrix}\mathbf{u}_{\mathbf{I},\mathbf{MS},k}\,, \tag{20}$$

$$\begin{bmatrix}\mathbf{y}_{\mathbf{I},\mathbf{m},k}\\ \mathbf{y}_{\mathbf{I},\mathbf{MS},k-1}\end{bmatrix}=\begin{bmatrix}\mathbf{C}^{(\alpha)}_{\mathbf{dI},\mathbf{m}} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathbf{I},k}\\ \mathbf{y}_{\mathbf{I},\mathbf{MS},k-1}\end{bmatrix}+\begin{bmatrix}\mathbf{D}^{(\alpha)}_{\mathbf{dI},\mathbf{m},\mathbf{u}}\\ \mathbf{0}\end{bmatrix}\mathbf{u}_{\mathbf{I},\mathrm{int},k}+\begin{bmatrix}\mathbf{D}^{(\alpha)}_{\mathbf{dI},\mathbf{m},\mathbf{MS}}\\ \mathbf{0}\end{bmatrix}\mathbf{u}_{\mathbf{I},\mathbf{MS},k}\,, \tag{21}$$

Subsystem II

$$\begin{bmatrix}\mathbf{x}_{\mathbf{II},k+1}\\ \mathbf{y}_{\mathbf{II},\mathbf{MS},k}\end{bmatrix}=\begin{bmatrix}\mathbf{\Phi}^{(\beta)}_{\mathbf{II}} & \mathbf{0}\\ \mathbf{C}^{(\beta)}_{\mathbf{dII},\mathbf{MS}} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathbf{II},k}\\ \mathbf{y}_{\mathbf{II},\mathbf{MS},k-1}\end{bmatrix}+\begin{bmatrix}\mathbf{H}^{(\beta)}_{\mathbf{II},\mathbf{u}}\\ \mathbf{D}^{(\beta)}_{\mathbf{dII},\mathbf{MS},\mathbf{u}}\end{bmatrix}\mathbf{u}_{\mathbf{II},\mathrm{int},k}+\begin{bmatrix}\mathbf{H}^{(\beta)}_{\mathbf{II},\mathbf{MS}}\\ \mathbf{D}^{(\beta)}_{\mathbf{dII},\mathbf{MS},\mathbf{MS}}\end{bmatrix}\mathbf{u}_{\mathbf{II},\mathbf{MS},k}\,, \tag{22}$$

$$\begin{bmatrix}\mathbf{y}_{\mathbf{II},\mathbf{m},k}\\ \mathbf{y}_{\mathbf{II},\mathbf{MS},k-1}\end{bmatrix}=\begin{bmatrix}\mathbf{C}^{(\beta)}_{\mathbf{dII},\mathbf{m}} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{x}_{\mathbf{II},k}\\ \mathbf{y}_{\mathbf{II},\mathbf{MS},k-1}\end{bmatrix}+\begin{bmatrix}\mathbf{D}^{(\beta)}_{\mathbf{dII},\mathbf{m},\mathbf{u}}\\ \mathbf{0}\end{bmatrix}\mathbf{u}_{\mathbf{II},\mathrm{int},k}+\begin{bmatrix}\mathbf{D}^{(\beta)}_{\mathbf{dII},\mathbf{m},\mathbf{MS}}\\ \mathbf{0}\end{bmatrix}\mathbf{u}_{\mathbf{II},\mathbf{MS},k}\,. \tag{23}$$

Furthermore, the subsystems' descriptions can be combined to one overall discrete state-space representation given in (24), which can be evaluated for stability and accuracy reasons:

$$\begin{bmatrix}\mathbf{x}_{\mathbf{I},k+1}\\ \mathbf{x}_{\mathbf{II},k+1}\\ \mathbf{y}_{\mathbf{I},\mathbf{MS},k}\\ \mathbf{y}_{\mathbf{II},\mathbf{MS},k}\end{bmatrix}=\underbrace{\begin{bmatrix}\mathbf{\Phi}^{(\alpha)}_{\mathbf{I}} & \mathbf{0} & \mathbf{0} & \mathbf{H}^{(\alpha)}_{\mathbf{I},\mathbf{MS}}\\ \mathbf{0} & \mathbf{\Phi}^{(\beta)}_{\mathbf{II}} & \mathbf{H}^{(\beta)}_{\mathbf{II},\mathbf{MS}} & \mathbf{0}\\ \mathbf{C}^{(\alpha)}_{\mathbf{dI},\mathbf{MS}} & \mathbf{0} & \mathbf{0} & \mathbf{D}^{(\alpha)}_{\mathbf{dI},\mathbf{MS},\mathbf{MS}}\\ \mathbf{0} & \mathbf{C}^{(\beta)}_{\mathbf{dII},\mathbf{MS}} & \mathbf{D}^{(\beta)}_{\mathbf{dII},\mathbf{MS},\mathbf{MS}} & \mathbf{0}\end{bmatrix}}_{\mathbf{\Phi}_G}\begin{bmatrix}\mathbf{x}_{\mathbf{I},k}\\ \mathbf{x}_{\mathbf{II},k}\\ \mathbf{y}_{\mathbf{I},\mathbf{MS},k-1}\\ \mathbf{y}_{\mathbf{II},\mathbf{MS},k-1}\end{bmatrix}+\underbrace{\begin{bmatrix}\mathbf{H}^{(\alpha)}_{\mathbf{I},\mathbf{u}} & \mathbf{0}\\ \mathbf{0} & \mathbf{H}^{(\beta)}_{\mathbf{II},\mathbf{u}}\\ \mathbf{D}^{(\alpha)}_{\mathbf{dI},\mathbf{MS},\mathbf{u}} & \mathbf{0}\\ \mathbf{0} & \mathbf{D}^{(\beta)}_{\mathbf{dII},\mathbf{MS},\mathbf{u}}\end{bmatrix}}_{\mathbf{H}_G}\begin{bmatrix}\mathbf{u}_{\mathbf{I},\mathrm{int},k}\\ \mathbf{u}_{\mathbf{II},\mathrm{int},k}\end{bmatrix} \tag{24}$$

A simulation is stable, if for all occurring switch states the absolute values of the eigenvalues $\tilde{\lambda}_j$ are lower than one (cf. [13]). The differences between the n eigenvalues of the overall unsplit discrete state-space representation (λ_i) and the m eigenvalues $\tilde{\lambda}_j$ of the system described by (24) lead to a measure for accuracy, such as:

$$\varepsilon=\sum_{\alpha,\beta}\sum_{j=1}^{m}\min_{i}\left\{\left|\tilde{\lambda}_j-\bar{\lambda}_i\right|\right\},\quad \bar{\lambda}_i=\begin{cases}\lambda_i & 0<i\le n\\ 0 & ,n<i\le m\end{cases} \tag{25}$$

Small values represent good splitting points, while higher values indicate less suitable choices.

The last paragraphs described different criteria that can be applied for rating a given splitting. With regard to a good splitting point choice, the LU decomposition can usually be applied to almost all arbitrary splitting points, but it needs divisions and high computation efforts for solving the implicit equation system. Thus, it is less suitable for fast simulations on FPGAs. The delay insertion has the opposite characteristics. The fixed-point iteration is a compromise between both methods, the implementation is more robust than the delay insertion method and does not need divisions for computation. Despite the methods' criteria for stability, convergence speed, and accuracy, the limitations of every computation platform, e.g., the maximum dimension of the inputs, outputs, and state variables must be considered as well. To find an appropriate candidate for splitting, the hardware limitations must be checked beforehand. Depending on the chosen methods for solving (14), the relevant criteria are evaluated.

5. Simulation Results

In the following section, the results for the exemplary circuit diagram regarding the splitting positions shown in Fig. 6 are outlined.

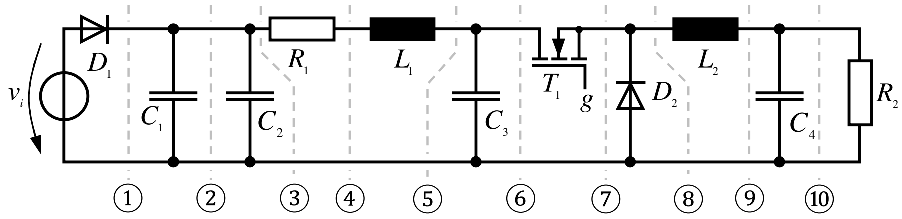


Fig. 6: Examples for possible splitting positions in the circuit of Fig. 1a.

The splitting point ② is not suitable, because C_1 and C_2 are linearly dependent. Splitting point ⑦ would separate the transistor T_1 and diode D_1, which represent a switch group. This splitting point must be avoided so that the correct conditional switching events of the diode can be computed. At first sight, all other illustrated splitting points in Fig. 6 might be suitable candidates. But stability and accuracy properties differ significantly. The result of the analysis is shown in Fig. 7.

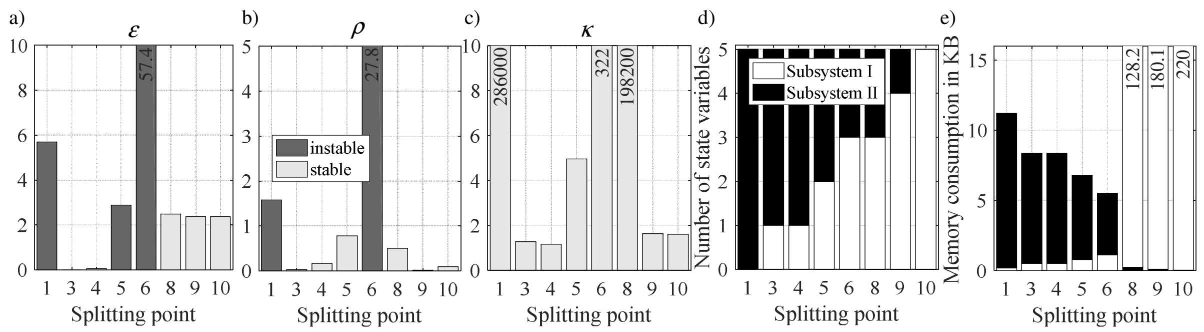


Fig. 7: Analysis results: a-c) Criteria; d) Number of states; e) Memory consumption in KB.

While the diagrams in Fig. 7a to Fig. 7c show the values of the criteria and mark the unstable splitting points in dark gray and the stable ones in light gray, Fig. 7d and 7e present the distribution of the state variables to the subsystems and the memory consumption per subsystem. For the memory consumption, all possible switch states are considered.

The values of the criteria ε, ρ, and κ are not directly comparable, because they have different meanings in general. For example, the spectral radius ρ of the fixed point iteration needs to be less than one to guarantee stability, while the condition number is always greater than or equal to one. However, all three criteria lead to statements about stability and accuracy, and thus about the suitability of the respective method for the splitting point under consideration. For the circuit of Fig. 6, the evaluation of the criteria plotted in Fig. 7 leads to the following interpretations:

- ① is not recommended, because delay insertion and fixed-point iteration are unstable and the direct solution is not well conditioned.
- ③ and ④ are well-suited independent of the chosen method. Thus, the delay insertion method is preferred because of the low computation effort.

- ⑤ is recommended only for applying the direct solution, because delay insertion is unstable and the spectral radius of the fixed-point approach is near one. The number of state variables and the memory consumption are well separated.
- ⑥ is desirable because of the well-separated state variables and the low memory consumption. However, it is not recommended for delay insertion and fixed-point iteration, because both are unstable. The direct solution may be used, but other splitting points are better conditioned.
- ⑧, ⑨, and ⑩ are stable independent of the used method. However, all switches are located in the first subsystem, which is not recommended, because of the high memory consumption.

6. Conclusion

This paper presented a mathematical formulation for splitting electrical circuit simulation models (state-space formulation) into almost independent subsystems that are coupled by voltages and currents at the splitting positions only. This mathematical formulation allows for different approaches to solve implicit coherences between different circuit parts. Three approaches were investigated in this document: the direct solution of the coupling equations (which leads to an approach similar to the state-space nodal method [6]), a fixed-point approximation of the coupling equations, and the simple and pragmatic delay insertion approach, which uses delays between the coupling voltage and current measurements (cf. [5]). Moreover, this paper proposed a process to support users in finding suitable splitting positions for their simulation models if the models must be computed on different hardware platforms due to model complexity. This includes the automatic detection of dependent states and switch groups, and the computation of the stability for given splitting positions and approaches. The rating of different splitting points was illustrated for a simple example.
In summary, in the future, the basic ideas of this paper will lead to a user-friendly splitting procedure that enables the user to distribute real-time computations to different hardware platforms for the specific electrical simulation model in a systematic way.

References

[1] Benigni, A.; Monti, A.; Dougal, R. A.: Latency-Based Approach to the Simulation of Large Power Electronics Systems, IEEE Transactions on Power Electronics, Vol. 29, No. 6, June, 2014.
[2] Burden, R. L.; Faires, J. D.: Numerical Analysis, Ninth Edition, Brooks Cole, 2010.
[3] Champagne, R.; Dessaint, L.-A.; Fortin-Blanchette, H.; Sybille, G.: Analysis and Validation of a Real-Time AC Drive Simulator, IEEE Transactions on Power Electronics, Vol. 19, No. 2, March, 2004.
[4] De Kelper, B.; Blanchette, H. F.; Dessaint, L.-A.: Switching Time Model Updating for the Real-Time Simulation of Power Electronic Circuits and Motor Drives, IEEE Transactions on Energy Conversion, Vol. 20, No. 1, March, 2005.
[5] dSPACE GmbH, Manual of the Electrical Power Systems Simulation Package, September, 2016.
[6] Dufour, C.; Mahseredjian, J.; Bélanger, J.: A Combined State-Space Nodal Method for the Simulation of Power System Transients, IEEE Transactions on Power Delivery, Vol. 26, No. 2, p. 928 – 935, May, 2011.
[7] Franklin, G.; Powell, J. D.; Workman, M.: Digital Control of Dynamic Systems, 3rd ed., Addison Wesley Longman Inc., 1998.
[8] Kelley, C. T.: Iterative Methods for Linear and Nonlinear Equations, Society for Industrial and Applied Mathematics, 1995.
[9] Kuh, E., S.; Rohrer, R., A.: The State-Variable Approach to Network Analysis, Proceedings of the IEEE, 1965.
[10] Majstorovic, D.; Celanovic, I.; Teslic, N. Dj., Celanovic, N.; Katic, V. A.: Ultralow-Latency Hardware-in-the-Loop Platform for Rapid Validation of Power Electronics Designs, IEEE Transactions on Industrial Electronics, Vol. 58, No. 10, October 2011.
[11] Ould-Bachir, T.; Blanchette, H. F.; Al-Haddad, K.: A Network Tearing Technique for FPGA-Based Real-Time Simulation of Power Converters, IEEE Transactions on Industrial Electronics, Vol. 62, No. 6, June 2015.
[12] Quarteroni A., Sacco R., Saleri F.: Numerical Mathematics, Springer-Verlag New York Inc., 2000.
[13] Zadeh, L. A., Desoer, C. A.: Linear System Theory: The State Space Approach, McGraw-Hill, 1963.