Chapter 8
COMSON Demonstrator Platform

Georg Denk, Tamara Bechtold, Massimiliano Culpo, Carlo de Falco, and Alexander Rusakov

Abstract This chapter describes the *Demonstrator Platform* (DP), a framework for simulation of devices, interconnects, circuits, electromagnetic fields, and thermal effects. This framework is used to develop and test new mathematical methods and algorithms. Section 8.1 describes the design of the DP and gives an overview of the available modules. Section 8.2 is a tutorial on how to use the DP focusing on model-order reduction. It shows for the example of a micro-hotplate model all the steps needed to apply model-order reduction, including postprocessing and error estimation. In a second part, a coupled simulation of a circuit combined with a reduced model of a transmission line is presented. Section 8.3 emphasizes the aspect of the DP as a development framework. After introducing the benchmark example of an n-channel power MOS-FET, it is shown how to combine and extend different modules of the DP to a fully coupled electro-thermal simulation of the device.

G. Denk (✉)
Infineon Technologies AG, 81726 München, Germany
e-mail: georg.denk@infineon.com

T. Bechtold
IMTEK – University of Freiburg, Georges-Koehler-Allee 103, 79110 Freiburg, Germany
e-mail: tamara.bechtold@imtek.uni-freiburg.de

M. Culpo
Chair of Applied Mathematics/Numerical Analysis, Bergische Universität Wuppertal, Gaußstraße 20, 42119 Wuppertal, Germany
e-mail: m.culpo@cineca.it

C. de Falco
MOX – Modeling and Scientific Computing, Dipartimento di Matematica, Politecnico di Milano, P.zza L. da Vinci 32, 20133 Milano, Italy

CEN – Centro Europeo di Nanomedicina, P.zza L. da Vinci 32, 201333 Milano, Italy
e-mail: carlo.defalco@polimi.it

A. Rusakov
Institute for Design Problems in Microelectronics of Russian Academy of Sciences (IPPM RAS), 3, Sovetskaya Street, Moscow 124365, Russian Federation
e-mail: rusakov@inm.ras.ru

M. Günther (ed.), *Coupled Multiscale Simulation and Optimization in Nanoelectronics*, Mathematics in Industry 21,
DOI 10.1007/978-3-662-46672-8_8

8.1 Introduction

The purpose of the COMSON project is to develop algorithms for coupled multiscale simulation and optimization in nanoelectronics. As several nodes are involved, a common platform for this development is needed. The main objective of the consortium is therefore to realize an experimental *Demonstrator Platform* in software code, which comprises simulation of devices, interconnects, circuits, electromagnetic fields, and thermal effects in one single framework. It connects each individual achievement, and offers an adequate simulation tool for optimization in a compound design space.

The *Demonstrator Platform* is used as a framework to test mathematical methods and approaches, so as to assess whether they are capable of addressing the industry's problems, and to adequately educate young researchers by hands-on experience on state-of-the-art problems, and beyond. The *Demonstrator Platform* does not aim at replacing existing industrial or commercial codes. However, it will be capable of analyzing medium sized coupled problems of industrial relevance, thus offering a chance to develop advanced mathematics for realistic problems. The second benefit of such a platform is to collect the knowledge of models and methods, which is widespread distributed over the different partners, giving a good opportunity for transfer of knowledge.

This section gives an introduction to the ideas and concepts of the *Demonstrator Platform*, followed by a tutorial section on how this platform can be used in the context of model-order reduction (Sect. 8.2). In Sect. 8.3, a research study is presented which uses the *Demonstrator Platform* for development of the coupled electro-thermal simulation of an n-channel power MOSFET. The *Demonstrator Platform* was also used for developing a coupled circuit-device simulation, see [21].

8.1.1 Design of the Demonstrator Platform

In order to allow an easy installation and application within different environments, the design of the *Demonstrator Platform* was influenced by the following assumptions:

– Provides a fast prototyping environment,
– Is not restricted to a particular operating system,
– Can easily be extended,
– Can easily be distributed to others,
– Allows different license conditions for different parts.

These conditions are especially important, if the *Demonstrator Platform* should be used both in an academic environment and in an industrial environment, as they are present within COMSON. To foster cooperations with other research groups, it is essential to share the *Demonstrator Platform* as a common development platform.

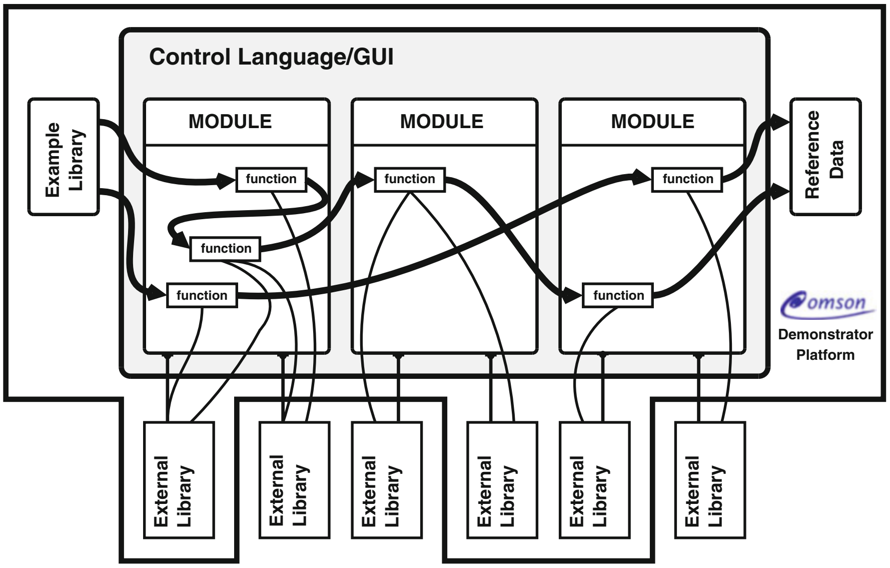


Fig. 8.1 Layout of the *Demonstrator Platform*

For fast prototyping of mathematical algorithms, interpreted languages like Matlab or Octave are widely used. To avoid license problems, we decided to use the free tool Octave [38] which is available for many operating systems and can be distributed without license issues. Octave allows both an interactive approach for development and a batch-oriented usage for long-running computations. It offers additionally a free API for integrating software written in other programming languages like C or Fortran. Octave builds the controlling language of the *Demonstrator Platform*.

The *Demonstrator Platform* uses a modular structure consisting of so-called *modules* and *external libraries*. Modules provide some functionality to the user of the *Demonstrator Platform*, e.g. model-order reduction techniques. To enable the re-use of already existing software, the modules may interface to external libraries, e.g. numerical libraries like Slicot. In Fig. 8.1, the structure of the *Demonstrator Platform* is depicted.

This flexible concept of external libraries provides a solution for incorporating software released under different license conditions. Libraries with a free license can be completely integrated to the *Demonstrator Platform* which is distributed as free software. Other libraries with a more restrictive license have to be kept separated, only the interfacing routines are part of the *Demonstrator Platform*. With this approach it is possible to call confidential software from the *Demonstrator Platform*, without making it a part of it. Especially in an industrial environment, this is of great importance. This is indicated in Fig. 8.1, where the boundary of the *Demonstrator Platform* includes some of the external libraries, while others are located outside.

For quality software, it is not enough to provide a set of functions. Also the documentation is an essential part of the *Demonstrator Platform*. This documentation contains research papers where the theoretic background of the routines is provided. In addition, a description of the available functions is automatically generated out of the sources.

The *Demonstrator Platform* contains an integrated self test to ensure the correctness of the modules. For this purpose every module is provided with some test examples for which a reference solution is known. During the self test, the computed solution is compared with the reference solution and an according message is printed. This feature is especially important during the joint development of new modules, as they might interfere with existing modules.

All examples, including the test examples, are categorized in so-called *class A*, *class B*, or *class C* examples. While *class A* examples are basic unit tests, *class B* examples are (simple) academic examples which already require a full-fledged algorithm to be solved. *Class C* examples are real-life examples which need the proper coupling of algorithms, advanced approaches, and in most cases longer computing times to be solved.

8.1.2 Modules

This section gives a short overview of the available modules of the *Demonstrator Platform*.

8.1.2.1 Generic Numerical Methods

DAEN Differential-Algebraic Equations Solver
DAEN is a BDF implementation of a differential-algebraic equations solver for problems with index 0, 1, and 2. It uses a variable step size, variable order.

RADAU Differential Algebraic Equation Solver
RADAU is a differential-algebraic equation solver for equations of index 0, 1 and 2. This code is a variable step size, variable order implementation of the RADAU methods.

GLIMDA Differential Algebraic Equations Solver
GLIMDA is for the numerical solution of differential equations of index 0, 1 and 2. It is a variable step size, variable order implementation of general linear methods.

BIM Finite-Element Box Integration Method
BIM is a PDE solver using a finite element/finite volume approach. It solves diffusion-advection-reaction (DAR) partial differential equations based on the finite volume Scharfetter-Gummel (FVSG) method also known as Box Integration Method (BIM).

8.1.2.2 Model-Order Reduction

AMOR Interface to MOR4ANSYS
AMOR provides an interface to the software package MOR for ANSYS which is part of the module MOR4ANSYS

MOR4ANSYS MOR for ANSYS
MOR4ANSYS is an adapted version of the software package MOR for ANSYS [44] which reduces dynamic systems provided in Matrix Market format.

ROM-WB Tools for migration between RomWork and OCS
ROM-WB provides some tools for converting input files to make them usuable for the module OCS and RomWork.

SLICOTINT Interface to SLICOT Model Reduction Routines
SLICOTINT provides an interface to the SLICOT library [49]. It supports balance and truncate model reduction (routine AB09AD), singular perturbation approximation based model reduction (routine AB09BD), Hankel norm approximation based model reduction (routine AB09CD). It also allows the efficient computation of Hankel Singular Values of the dynamical system.

MOR Collection of simple MOR tools
MOR provides a collection of simple projector matrix builders for DAE systems.

8.1.2.3 Circuit and Device Simulation

OCS Octave Circuit Simulator
OCS is a module for solving DC and transient MNA equations stemming from electrical circuits.

SKIF Interface to the SiMKit library
SKIF provides an interface to SiMKit library [37], a library of compact transistor models.

D4MEKAI Device Simulator based on Maximum Entropy Principle
D4MEKAI provides a device simulation of the hydrodynamical model of semiconductors in 1D in case of the Kane dispersion relation.

ETMEP1D Device Simulator of the MEP energy-transport model in 1D
ETMEP1D provides the 1D simulation of the MEP energy-transport model for semiconductors by using a Scharfetter-Gummel like scheme while the Poisson equation is solved with a false transient method.

ET_MEP_MOSFET Device Simulator for the 2D MEP energy-balanced model.
ET_MEP_MOSFET provides the 2D numerical simulation of the MEP energy-transport model for semiconductors by using the Scharfetter-Gummel scheme while the Poisson equation is solved with a false transient method [41–43].

FIDES Field Device Simulator
FIDES provides a package for monolithic or co-simulation of field-circuit coupled problems.

ROMI Reduced Order Model of the multiconductor interconnects
ROMI provides routines for extraction of the reduced order model of the multiconductor interconnects.

SECS1D, SECS2D, SECS3D Drift-Diffusion simulator for 1d, 2d, and 3d semiconductor devices
SECS1D, SECS2D, and SECS3D provides a device simulator for 1d, 2d, and 3d, resp., semiconductor devices based on drift-diffusion [22, 24, 25].

8.1.2.4 Optimization

CRS Optimization routine based on Controlled Random Search
CRS provides an optimization algorithm based on controlled random search plus some test functions.

OPTBOOK Optimization procedures
OPTBOOK provides a set of optimization procedures, described in the book "Optimization of the EM devices".

8.1.2.5 Auxiliary Routines

MSH Meshing Software Package
MSH is a package for creating and managing triangular and tetrahedral meshes for finite-element or finite-volume PDE solvers.

FPL FEM Plotting
FPL provides a collection of routines to plot data on unstructured triangular and tetrahedral meshes.

8.2 Tutorial on Working with the *Demonstrator Platform* with Emphasis on MOR

Principle and methods of model order reduction (MOR) are explained in Chaps. 4–6. Here we would like to demonstrate how the *Demonstrator Platform* can be used for testing new MOR algorithms and for coupling reduced order models with the surrounding/driving circuitry. Section 8.2.1 lists the implemented MOR modules and describes in more details those, which are relevant for this tutorial. Section 8.2.2 introduces the chosen nanoelectronic case studies. Section 8.2.3 demonstrates in a step-by-step-manner how to run model order reduction within the *Demonstrator Platform* by using two external libraries MOR for ANSYS [44] and SLICOT [49]. It also describes a provided framework for prototyping and testing new model reduction algorithms. Section 8.2.4 demonstrates how to use reduced order models (created by the mentioned libraries) within a circuit simulation with the OCS.

8.2.1 Overview and Structure of MOR Modules

At present, there are the following MOR related modules within the *Demonstrator Platform*:

– AMOR,
– MOR,
– MOR4ANSYS,
– MOR_UTILITIES,
– ROMI,
– ROM_WB,
– SLICOTINT,

each located in the subdirectory with the same name. For example, the module AMOR is located in *DP_DIR*/`AMOR`, where *DP_DIR* denotes the directory in which the demonstrator platform has been installed. In the following, we will shortly describe the structure and functionality of AMOR, MOR4ANSYS, MOR_Utilities and SLICOTINT modules, which are used in the tutorials in Sect. 8.2.3.

8.2.1.1 Interface Modules AMOR and SLICOTINT

AMOR and SLICOTINT are interfaces between the *Demonstrator Platform* and two external libraries, MOR for ANSYS (described below) and SLICOT (stands for **S**ubroutine **L**ibrary **in** **Co**ntrol **T**heory, [49]). Both libraries depend on subroutines from BLAS (**B**asic **L**inear **A**lgebra **S**ubroutines) and LAPACK (**L**inear **A**lgebra **Pack**age) [2] for numerical linear algebra and are interfaced to the *Demonstrator Platform* using dynamically linked C++ functions, contained in `amor.cc` and `slicot.cc`, as schematically represented in Fig. 8.2. The compilation of those two functions with `mkoctfile` results in `amor.oct` and `slicot.oct` which provide the Octave functions `amor` and `slicot`. The Fortran subroutines `AB09AD`, `AB09BD` and `AB09CD` from SLICOT implement three different MOR algorithms for first order linear dynamical systems, namely Balanced Truncation Approximation (BTA) [52], Singular Perturbation Approximation (SPA) [36], and Hankel-Norm Approximation (HNA) [27], respectively. Note that the SLICOT library also contains other mathematical tools such as discrete sine/cosine and Fourier transformations, which are however not available within the *Demonstrator Platform* at present. A full list of SLICOT subroutines, the documentation and examples are available at [49]. Input arguments for all functions displayed in Fig. 8.2 will be discussed in Sect. 8.2.3.

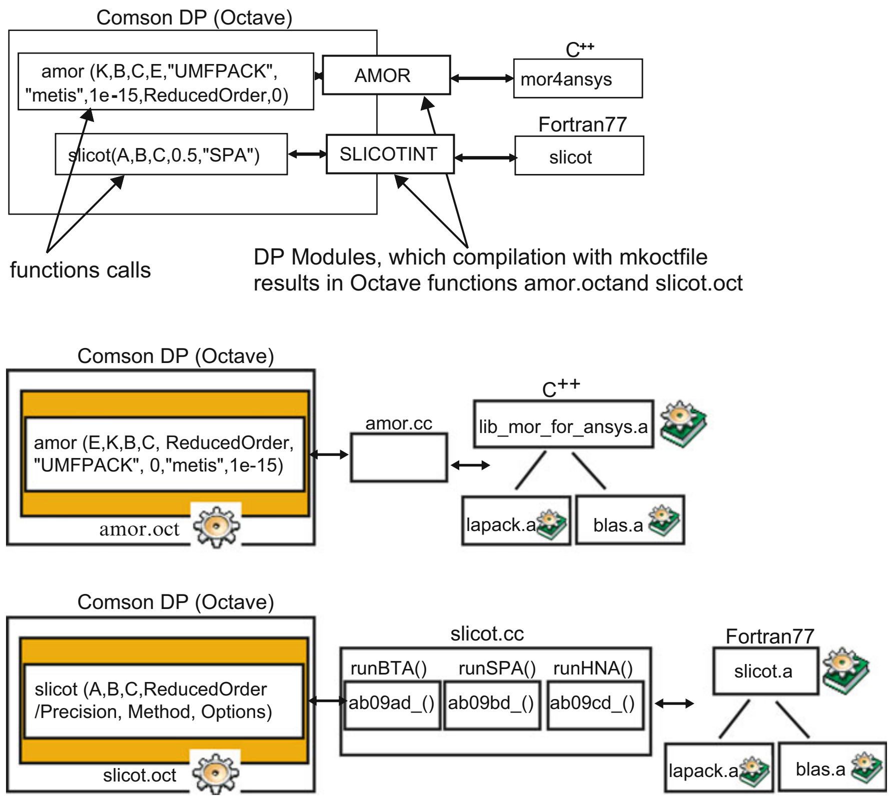


Fig. 8.2 *Demonstrator Platform* interfaces to MOR for ANSYS and SLICOT libraries

8.2.1.2 MOR4ANSYS

The *Demonstrator Platform* module MOR4ANSYS incorporates the C++ library MOR for ANSYS (stands for **mo**del **o**rder **r**eduction **for ANSYS**) [45]. In its original form, MOR for ANSYS is meant to build compact models directly from finite element models implemented in the ANSYS simulator.[1] It implements the first and second order Arnoldi algorithm [5, 26] for model reduction of the linear dynamical systems. In the current implementation, as a *Demonstrator Platform* module, ANSYS support has been disabled and only the reduction of the first order linear dynamical systems is possible (see Fig. 8.3), provided the system matrices are available in the MatrixMarket format [10].

[1]The last GPL licensed version, which has been adapted for the *Demonstrator Platform* is MOR for ANSYS 1.8. Current commercial version is MOR for ANSYS 2.5 (see http://modelreduction.com).

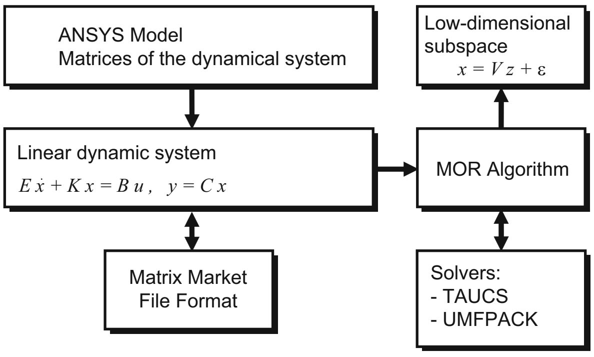


Fig. 8.3 MOR for ANSYS structure. ANSYS support has been disabled within the COMSON *Demonstrator Platform* and only system matrices of the first order linear dynamical system, in the Matrix Market format, can be passed

8.2.1.3 MOR_UTILITIES

The *Demonstrator Platform* module MOR_UTILITIES contains two Octave scripts, namely `MOR4ANSYS_Tutorial.m` and `SLICOT_Tutorial.m`, which demonstrate in a step-by-step manner how to perform model order reduction by interfacing the two libraries MOR for ANSYS and SLICOT. It further contains a script `Post4MOR.m` with a number of Octave functions for postprocessing a reduced order model, i.e. analyzing it in time and frequency domain, visualizing results, etc. The goal of this environment is to provide a framework for prototyping new model reduction algorithms. Our experience shows that it is more convenient to first perform research and prototyping in an interpreter environment, like Octave or Matlab, and only then to perform a compiled language implementation in e.g. C++.

8.2.2 MOR Case Studies

In this section we introduce two case studies to demonstrate the usage of linear model order reduction routines within the *Demonstrator Platform*. The first one is a model of a micro-machined silicon nitride membrane which is, mathematically speaking, a large-scale linear ordinary differential equation system (ODE). The second one is an academic model of a transmission line which is, mathematically speaking, a large-scale linear differential algebraic equation system (DAE).

8.2.2.1 Micro-hotplate

The first test model is a model of a MEMS (micro-electro-mechanical system) device, known as a micro-hotplate. This class of structures is employed in a variety of other microfabricated devices such as gas sensors [31] and infrared sources [50]. The device features sub-millimeter dimensions and is fabricated using technology originating from the semiconductor industry. Silicon is used as a substrate material

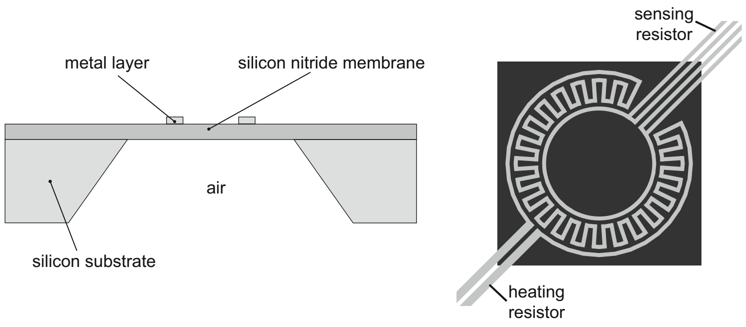


Fig. 8.4 A silicon-nitride membrane with integrated heater and sensing elements was fabricated by low-frequency plasma enhanced chemical vapor deposition. The square membrane is 500 nm thick with a side length of 550 μm, and the metal layer is made from 150 nm platinum with a 50 nm titanium adhesion layer

with a thickness of 525 μm. Silicon nitride with a thickness of 500 nm is deposited onto this substrate. A metal layer, consisting of 150 nm platinum with a 50 nm titanium adhesion layer is fabricated on top of the silicon nitride and shaped to form resistor structures for heating and temperature sensing. In order to achieve suitably high temperatures, the silicon substrate right below the resistor structures is removed by means of wet chemical etching. In this way, a tiny square membrane with 550 μm side length has been created (see Fig. 8.4 left). Such membrane configuration is favorable, as it increases the thermal isolation between the heat source (the heating resistor) and the heat sink (remaining silicon).

Different applications rely on the same working principle, of Joule heating a thin-film membrane. The membrane is being heated by applying an electrical voltage signal to the heating resistor. Temperature measurement on the membrane (necessary for the control) is done with a second resistor, called sensor. Both thin-film metal resistors are arranged as concentric circles (see Fig. 8.4 right) in order to achieve a homogeneous temperature distribution over the membrane.

The target temperature of the membrane is defined by the specific application. In thermo-optical application [33], the membrane temperature determines which wavelength of the focused light will be transmitted. The device works in the temperature range between room temperature and 400 °C. At gas sensing applications [54], the temperature determines the chemical reaction between the unknown gas and the membrane material. The working temperature is between 200 and 400 °C.

In all applications, the simulation goal is to consider important thermal issues such as, which electrical power should be applied in order to reach the target temperature at the membrane or to ensure the homogeneous temperature distribution over the membrane. The original model is the heat-transfer partial differential equation:

$$\nabla \bullet (\kappa(\mathbf{r})\nabla T(\mathbf{r},t)) + Q(\mathbf{r},t) - \rho(\mathbf{r})C_p(\mathbf{r})\frac{\partial T(\mathbf{r},t)}{\partial t} = 0 \tag{8.1}$$

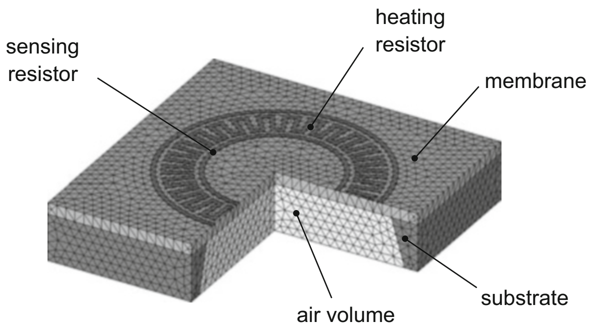


Fig. 8.5 FE mesh of three-dimensional model with 60020 nodes

where **r** is the position, t is the time, κ is the thermal conductivity of the material, C_p is the specific heat capacity, ρ is the mass density, Q is the heat generation rate that is different from zero only within the heater volume ($Q = \mathbf{j}^2 R(T)$, where **j** is the current density within the heater), and T is the unknown temperature distribution. Assuming that the heat distribution is uniformly distributed within the heater and that both material properties κ and C_p are temperature independent around the working point (such assumption can be made in some applications, see e.g. [6]), the finite element (FE) based spatial discretization of (8.1) leads to a large linear ODE system of the form:

$$E \cdot \dot{\mathbf{T}} + K \cdot \mathbf{T} = B \cdot \frac{U^2(t)}{R(T)} \tag{8.2}$$

where t is the time, $\mathbf{T}(t) \in \mathbf{R}^n$ is the state vector of unknown temperatures. E, $K \in \mathbf{R}^{n \times n}$ are heat capacity and heat conductivity which are symmetric, sparse and positive definite matrices.[2] $B \in \mathbf{R}^n$ is the input distribution array and n is the dimension of the system. $U(t)$ is the electrical voltage applied to the heating resistor with temperature-dependent resistance $R(T)$. In case that other sources than $U(t)$ (in total m sources) would be applied to the model, B would be a matrix, $B \in \mathbf{R}^{m \times n}$. Figure 8.5 shows the finite element model (FEM) of the three dimensional geometry, which consists of 60020 nodes. In terms of model (8.24), this means that the dimension of the ODE system is 60020. The solution of (8.24) can be done with a transient analysis within ANSYS, which results in temperature values at specified times in all nodes of the finite element mesh. However, in engineering applications, it is often not necessary to determine the complete temperature field. Mostly, temperature curves in a few nodes are of interest, as specified by the following output equation:

$$\mathbf{y} = C \cdot \mathbf{T} \tag{8.3}$$

[2]In engineering applications the heat capacity matrix is usually denoted with C and heat conductivity matrix with K. However, in the following we use the internal notation of MOR for ANSYS tool, which is used for reduction.

Table 8.1 Outputs for the micro-hotplate model

Name	Number	Comment
Sense_min	37577	Sensor node with minimal temperature
Sense_max	37179	Sensor node with maximal temperature

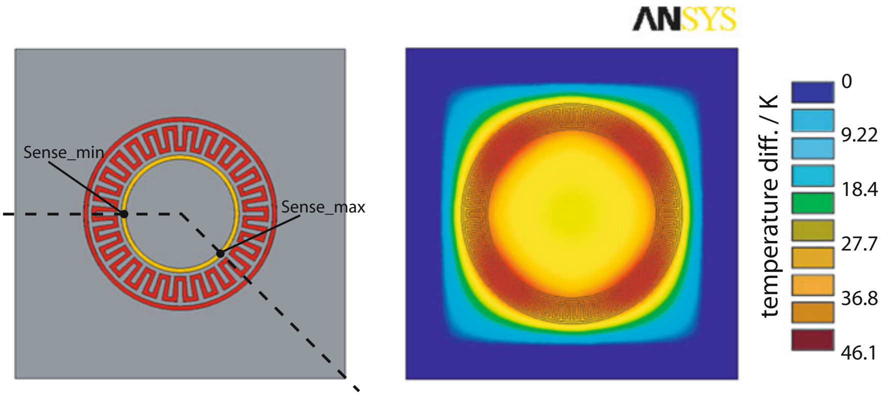


Fig. 8.6 Schematic position of the chosen output nodes (*left*). Temperature distribution after 0.025 s of heating with 2.49 mW (*right*)

where $C \in \mathbf{R}^{p\times n}$ is the user-defined output distribution array and p is the number of outputs of interest. In case that one seeks the complete temperature field, C is the unity matrix of dimension $n \times n$.

For the micro-hotplate, the output nodes of interest (defined by the matrix $C \in \mathbf{R}^{2\times 60020}$), are described in Table 8.1 and schematically displayed in Fig. 8.6 (left).

The output nodes have been selected in order to capture the average temperature of the sensing resistor. Experimentally, this temperature is obtained by measuring the resistance value of the sensor. Although local variations in the temperature result in local changes in the material's resistivity, this is not resolved by measuring the total resistance. Hence, in order to extract an equivalent temperature value from the FEM results we observe minimum and maximum temperature values in the sensor resistor. The arithmetic mean of these two temperatures is used as an equivalent to the measured temperature.

Equation (8.24) is the starting point for model order reduction, leading to a system of the same form but with smaller dimension. In Sect. 8.2.3 it is demonstrated how to apply MOR to this case study within the framework of the COSMON *Demonstrator Platform*.

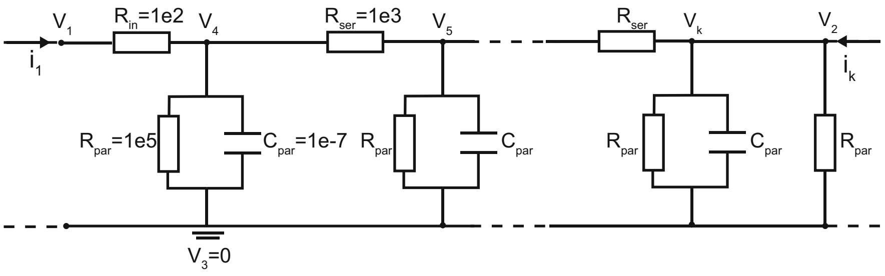


Fig. 8.7 Structure of a transmission line case study. The node numbering is according to the Intermediate File Format (IFF) of the *Demonstrator Platform*, in which external nodes must be numbered first

8.2.2.2 Transmission Line: An Academic Example

Figure 8.7 shows an academic model of a transmission line. This very simple model has been chosen, because it resembles the interconnect modeling and can also be effectively used for testing new MOR algorithms applicable to DAE systems. It consists of a scalable number of RC ladders and after performing a charge-oriented Modified Nodal Analysis (MNA) a linear DAE system of the form:

$$E \cdot \dot{\mathbf{x}} + K \cdot \mathbf{x} = B \cdot \mathbf{u}(t) \tag{8.4}$$

is obtained. In (8.4), t denotes the time, $\mathbf{x}(t) \in \mathbf{R}^n$ is the state vector containing in case of MNA nodal voltages and branch currents, i.e. in the *Demonstrator Platform* implementation also the charges of the capacitors, n is the dimension of the system and $\mathbf{u}(t) \in \mathbf{R}^m$ is the input excitation vector. E, $K \in \mathbf{R}^{n \times n}$ are constant and sparse system matrices,[3] which represent the contribution of capacitors and resistors, respectively. For the transmission line model E is singular as it contains only zero-valued entries in the rows corresponding to the resistor branches. $B \in \mathbf{R}^{n \times m}$ is the input distribution array and m is the number of inputs, i.e. sources. It is further possible to define the output measurement vector $\mathbf{y}(t) \in \mathbf{R}^p$, where p is the number of outputs of interest as:

$$\mathbf{y} = C \cdot \mathbf{x} + D \cdot \mathbf{u}(t) \tag{8.5}$$

with $C \in \mathbf{R}^{p \times n}$ and $D \in \mathbf{R}^{p \times m}$. Contrary to the previously described finite element model, in case of transmission line, it is necessary to specify a second term in the output equation, $D \cdot \mathbf{u}(t)$, which allows for the coupling of the transmission line with surrounding circuitry.

[3] In engineering applications the circuit matrices are usually denoted with C and G. However, here we follow the internal notation of MOR for ANSYS tool, which is used for reduction.

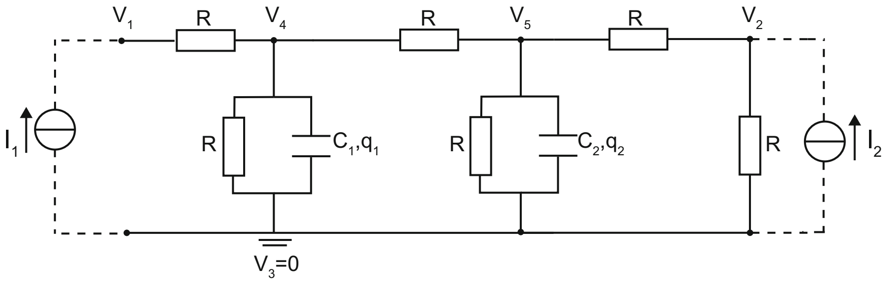


Fig. 8.8 Transmission line with six resistances and two capacitors. Coupling to the surrounding is modeled through two fictive current sources

In order to explain how the system matrices are assembled within the *Demonstrator Platform*, we observe the test model with six resistors and two capacitors, shown in Fig. 8.8, and write down the MNA equations. We model the coupling to the surrounding through two virtual current sources, I_1 and I_2. MNA for nodes N_1, N_2, N_4 and N_5 and for both charges C_1 and C_2, leads to:

$$\text{node } N_1 : \frac{V_1 - V_4}{R} = I_1 \tag{8.6}$$

$$\text{node } N_2 : \frac{V_2 - V_5}{R} + \frac{V_2}{R} = I_2 \tag{8.7}$$

$$\text{node } N_4 : \frac{V_4 - V_1}{R} + \frac{V_4}{R} + \dot{q}_1 + \frac{V_4 - V_5}{R} = 0 \tag{8.8}$$

$$\text{node } N_5 : \frac{V_5 - V_4}{R} + \frac{V_5}{R} + \dot{q}_2 + \frac{V_5 - V_2}{R} = 0 \tag{8.9}$$

$$\text{charge } C_1 : C_1 \cdot V_4 - q_1 = 0 \tag{8.10}$$

$$\text{charge } C_2 : C_2 \cdot V_5 - q_2 = 0 \tag{8.11}$$

or in matrix form:

$$\begin{bmatrix} 0&0&0&0&0&0 \\ 0&0&0&0&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&0&0&1 \\ 0&0&0&0&0&0 \\ 0&0&0&0&0&0 \end{bmatrix} \cdot \begin{bmatrix} \dot{V}_1 \\ \dot{V}_2 \\ \dot{V}_4 \\ \dot{V}_5 \\ \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} \frac{1}{R} & 0 & -\frac{1}{R} & 0 & 0 & 0 \\ 0 & \frac{2}{R} & 0 & -\frac{1}{R} & 0 & 0 \\ -\frac{1}{R} & 0 & \frac{3}{R} & -\frac{1}{R} & 0 & 0 \\ 0 & -\frac{1}{R} & -\frac{1}{R} & \frac{3}{R} & 0 & 0 \\ 0 & 0 & C_1 & 0 & -1 & 0 \\ 0 & 0 & 0 & C_2 & 0 & -1 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_4 \\ V_5 \\ q_1 \\ q_2 \end{bmatrix} = \begin{bmatrix} I_1 \\ I_2 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{8.12}$$

This resembles (8.4). If we consider (8.12) as a stand-alone system, we can define C in such a way that an arbitrary nodal voltage V_k is considered to be an output of interest. In such case $D = 0$ and the output equation has the form (8.3).

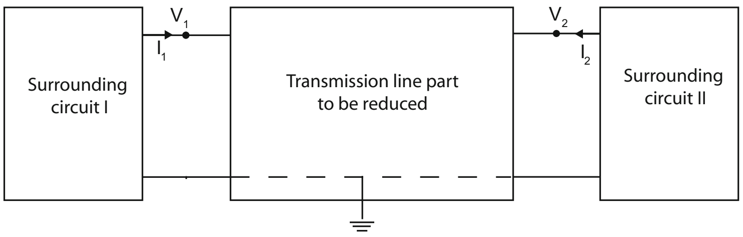


Fig. 8.9 Inputs and outputs of the transmission line, when coupled to the surrounding circuitry

However, if the transmission line is a part of a broader circuit, as shown in Fig. 8.9, it is necessary to introduce terminal-voltages and terminal-currents (V_1, V_2, I_1 and I_2) as inputs/outputs of the model. If we define e.g. terminal voltages as inputs, $u(t) = [V_1\ V_2]^T$, terminal currents as outputs, $y = [I_1\ I_2]^T$ and $x = [V_4\ V_5\ q_1\ q_2]^T$ as the new state-vector, (8.12) can be interpreted as:

$$\left.\begin{aligned} E \cdot \dot{\mathbf{x}} + K \cdot \mathbf{x} &= B \cdot \mathbf{u}(t) \\ \mathbf{y} &= C \cdot \mathbf{x} + D \cdot \mathbf{u}(t) \end{aligned}\right\} \quad \begin{bmatrix} 0 & 0 \\ 0 & E \end{bmatrix} \cdot \begin{bmatrix} \dot{u} \\ \dot{x} \end{bmatrix} + \begin{bmatrix} D & C \\ -B & K \end{bmatrix} \cdot \begin{bmatrix} u \\ x \end{bmatrix} = \begin{bmatrix} y \\ 0 \end{bmatrix} \tag{8.13}$$

with matrices E, K, B, C and D defined as schematically displayed in (8.14).

$$\left[\begin{array}{cc|cccc} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \underbrace{0}_{} & 0 & 0 & 0 \end{array}\right]_{E} \cdot \overset{\dot{\mathbf{u}}}{\underset{\dot{\mathbf{x}}}{\begin{bmatrix} \dot{V}_1 \\ \dot{V}_2 \\ \dot{V}_4 \\ \dot{V}_5 \\ \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}}} + \left[\begin{array}{cc|cccc} \multicolumn{2}{c|}{D} & \multicolumn{4}{c}{C} \\ \frac{1}{R} & 0 & -\frac{1}{R} & 0 & 0 & 0 \\ 0 & \frac{2}{R} & 0 & -\frac{1}{R} & 0 & 0 \\ \hline -\frac{1}{R} & 0 & \frac{3}{R} & -\frac{1}{R} & 0 & 0 \\ 0 & -\frac{1}{R} & -\frac{1}{R} & \frac{3}{R} & 0 & 0 \\ 0 & 0 & c_1 & 0 & -1 & 0 \\ 0 & 0 & 0 & c_2 & 0 & -1 \\ \multicolumn{2}{c|}{\textbf{-}B} & \multicolumn{4}{c}{K} \end{array}\right] \cdot \overset{\mathbf{u}}{\underset{\mathbf{x}}{\begin{bmatrix} V_1 \\ V_2 \\ V_4 \\ V_5 \\ q_1 \\ q_2 \end{bmatrix}}} = \overset{\mathbf{y}}{\begin{bmatrix} I_1 \\ I_2 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}} \tag{8.14}$$

System (8.13) can be subjected to model order reduction, which leads to a system of the same form, but with reduced system matrices, as follows:

$$\begin{bmatrix} 0 & 0 \\ 0 & E_r \end{bmatrix} \cdot \begin{bmatrix} \dot{\mathbf{u}} \\ \dot{\mathbf{z}} \end{bmatrix} + \begin{bmatrix} D & C_r \\ -B_r & K_r \end{bmatrix} \cdot \begin{bmatrix} \mathbf{u} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} \mathbf{y_r} \\ 0 \end{bmatrix} \tag{8.15}$$

(8.15) can be coupled to the surrounding circuitry. Note that the dimension of D is the same before and after reduction (2×2 in case of transmission line), because the inputs/outputs must stay preserved. The reduction of the transmission line case study is demonstrated in Sect. 8.2.2.2.

8.2.3 A Step by Step Model Reduction Tutorial

In the following we demonstrate model order reduction of the micro-hotplate model in a step-by-step tutorial. We use the following Octave scripts from the module MOR_Utilities:

- `MOR4ANSYS_Tutorial.m`
- `SLICOT_Tutorial.m`
- `Post4MOR.m`

8.2.3.1 Preparing the Model

It is the responsibility of the user to supply the system in the form:

$$\begin{aligned} E \cdot \dot{\mathbf{x}} &= A \cdot \mathbf{x} + B \cdot \mathbf{u} \\ \mathbf{y} &= C \cdot \mathbf{x} \end{aligned} \tag{8.16}$$

where the system matrices E, A, B and C are written in the Matrix Market Format [10]. The naming convection is *ModelName*`.E`, *ModelName*`.A`, *ModelName*`.B` and *ModelName*`.C` where *ModelName* is a user-defined string. It is further convenient to prepare a text file, which contains names of the output nodes, as listed in Table 8.1. These names are used by the postprocessing functions in the `Post4MOR.m` script to e.g. label the plots for the specific outputs of the full-scale and reduced order models. The microhotplate model (Sect. 8.2.2.1) is defined through the following files, all located within the module MOR_Utilities:

- `MicroHotplate.E`,
- `MicroHotplate.A`,
- `MicroHotplate.B`,
- `MicroHotplate.C`,
- `MicroHotplate.C.names`,

where the first four are are written in Matrix Market Format and the last one is a text file in which each line contains the name of the output node, as:

```
Sense_min
Sense_max
```

Prior to actual reduction, it is necessary to load the functions from `Post4MOR.m` and the test model into the Octave environment, as follows:

```
Post4MOR;
[E,A,B,C] = ReadInTestModel("MicroHotplate");
```

`Post4MOR.m` contains Octave functions for integrating the model (8.24), computing the transfer functions of the full and reduced models, computing and plotting different reduction errors in either time or frequency domain, etc. The function `ReadInTestModel` loads the system matrices of the micro-hotplate model from the above files into Octave.

8.2.3.2 Model Reduction with MOR for ANSYS via AMOR

After all four matrices from (8.16) are available in Matrix Market Format and the text file with outputs names is prepared, we can proceed to model order reduction of the dynamical system (8.16). For reduction of the large-scale systems (a few thousand degrees of freedom) within the *Demonstrator Platform*, we propose to use the C++ library, MOR for ANSYS. From the *Demonstrator Platform* it can be called via its Octave interface AMOR.

The implemented Arnoldi reduction algorithm [26] can be applied for ODE and DAE systems equally, i.e. regardless if E in (8.16) is regular or singular. The passivity and stability of the reduced model are granted, as long as both system matrices E and A are positive and semi-definite, as shown in [48]. The basic idea of model reduction with the Arnoldi algorithm is to find a low-dimensional subspace that approximates the transient behavior of the state vector $\mathbf{x}$:

$$\mathbf{x} = V \cdot \mathbf{z} + \epsilon \tag{8.17}$$

and the approximation error ϵ is assumed to be small even though the number of columns of projection matrix V (i.e. the dimension of $\mathbf{z}$) is much less than the number of rows (i.e. the dimension of $\mathbf{x}$). The compact model of the linear first order dynamical system is obtained by the projection of (8.16) as follows:

$$\begin{aligned} V^T EV \cdot \mathbf{z} &= V^T AV \cdot \mathbf{z} + V^T B \cdot \mathbf{u} \\ \mathbf{y_r} &= CV \cdot \mathbf{z} \end{aligned} \tag{8.18}$$

The projection matrix V is iteratively obtained by the MOR algorithm, i.e. column by column. This means that if we produced the reduced model of order r, we can obtain all reduced models of any lower order just by discarding the last columns in the project matrix. This can be used in the recommended strategy [7] to find an optimum dimension of the reduced model, by observing the convergence of the relative error between two "neighbored" reduced order models, with orders r and $r + 1$. The reduced order model is valid for an arbitrary input function. Furthermore, the transient and harmonic simulation of (8.18) is much faster than those of the original high-order system (8.16). However, the physical meaning of the original state vector x is lost in (8.18). The new state vector z can be considered as a vector of generalized coordinates, which requires a certain level of abstraction in the engineering applications. Even so, the inputs and outputs defined by the matrices B and C, will stay preserved after the reduction, i.e. y_r from (8.18) approximates y from (8.16) with high accuracy. It is also possible to recover the complete original state vector x by back-projecting z, as indicated in (8.17), while neglecting ϵ. The functionality of the Arnoldi algorithm is that the transfer function of the full-scale model, defined as:

$$H(s) = C(sE - A)^{-1}B \tag{8.19}$$

when developed into a Taylor series around some value of the Laplace variable $s = s_0$:

$$H(s) = \sum_{i=0}^{\infty} m_i(s_0)(s - s_0)^i \tag{8.20}$$

where $m_i(s_0) = C(-(s_0E - A)^{-1}E)^i \cdot (s_0E - A)^{-1}B$ is called the i-th moment around s_0 will have the same moments as the transfer function $H_r(s)$ of the reduced model, up to the degree r. With other words, it approximates the input/output behavior.

The AMOR interface between MOR for ANSYS and the *Demonstrator Platform* should be called with

```
V = amor(E, K, B, C, r , solver , s_0, \
         ReorderingScheme , tol );
```

where, E, $K = -A$, B and C are the system matrices[4] of the dynamic system, as defined in (8.16) and other parameters (which are optional) are meant to control the model reduction process, as follows:

– `r` specifies the dimension of the reduced model. By default it is 30.
– `solver` is to choose a solver. By default it is `TAUCSllmf` (suitable for positive definite matrices).
– `s_0` specifies which expansion point should be used for the transfer function. By default it is 0.
– `ReorderingScheme` is the solver parameter. By default it is `metis`.
– `tol` sets up the tolerance to deflate the next column vector of V. By default it is 10^{-15}.

The default values have been chosen based on experience with electro-thermal models of MEMS devices [8]. The output of AMOR is a projection matrix V from (8.17), which is constructed vector by vector. When a new vector is generated, MOR for ANSYS checks its norm. If it is less then the tolerance specified with `tol` option,

[4]Note that the internal system representation from of MOR for ANSYS V. 1.8 differs from (8.16), as it uses matrix $K = -A$.

the vector is deflated (removed), as it is assumed to represent a zero vector within rounding errors.

There is a vast number of solvers to solve a system of linear algebraic equations. MOR for ANSYS 1.8 uses the TAUCS [51] and UMFPACK [19] libraries with following solver choices:

- `TAUCSlltmf`: Multifrontal supernodal Cholesky decomposition.
- `TAUCSlltll`: Left-looking supernodal Cholesky decomposition.
- `TAUCSlltooc`: Out-of-core sparse Cholesky decomposition.
- `TAUCSllt`: Cholesky decomposition column by column (slow).
- `TAUCSldlt`: LDLT factorization.
- `TAUCSlu`: Out-of-core sparse pivoting LU decomposition.
- `UMFPACK`: Multifrontal LU decomposition.

TAUCS solvers for symmetric matrices can take a reordering-scheme parameter, which specifies the reordering method. Allowable values for `ReorderingScheme` parameter are:

- `metis`: hybrid nested-dissection minimum degree ordering.
- `genmmd`: multiple minimum degree ordering.
- `md`: minimum degree ordering.
- `mmd`: multiple minimum degree ordering.
- `amd`: approximate minimum degree ordering.
- `treeorder`: no-fill ordering code for matrices whose graphs are trees.

Our recommendations are as follows. For symmetric and positive definite matrices `TAUCSlltmf` with `metis` is the best choice. If the matrix is symmetric but indefinite `TAUCSldlt` with `metis` is a good choice, although `UMFPACK` may be faster in this case. For non-symmetric matrices one must use `UMFPACK`.

By default, MOR for ANSYS uses zero as an expansion point of the transfer function. This means that the reduced order model will approximate the original model accurately at low frequencies. If, however, the expansion point is different from zero, the reduced model will not preserve the stationary state. The choice of the expansion point depends on the application. For more information about methods and more details on input parameters, please consult the MOR for ANSYS 1.8 manual within the *Demonstrator Platform* documentation.

We run the model order reduction of the micro-hotplate model as follows:

```
K = -A;
V = amor(E, K, B,C, 30, "UMFPACK", 0, "metis", 1e-15);
```

As the output of the call to `amor` is a projection matrix, it is necessary to actually construct the reduced order model (8.18). This can be done by projection:

```
Er = V' * E * V;
Kr = V' * K * V;
Br = V' * B;
Cr = C * V;
```

or in the more compact form, by calling the `build_reduced_system` routine from `Post4MOR.m`:

```
[Er,Kr,Br,Cr] = build_reduced_system(E, K, B, C, V);
Ar = -Kr;
```

It is further possible to save the reduced system in the Matrix Market form by setting a base name for the reduced model. The base name can be same or different than for the original model. In either case, an extention `.MOR` will be attached. For example:

```
write_reduced_system("MicroHotplate", Er, Ar, Br, Cr);
```

will produce the following files:

– `MicroHotplate.MOR.E`,
– `MicroHotplate.MOR.A`,
– `MicroHotplate.MOR.B`,
– `MicroHotplate.MOR.C`.

8.2.3.3 Postprocessing of the Reduced Model and Error Estimation

Once the reduced order model has been created, it is necessary to integrate it, compare it with the full-scale model or with measurements, plot it in time and/or frequency domain etc. For the micro-hotplate we define the integration time of 0.04 s and the constant input function $u(t) = 1$, which corresponds to the constant input power of $Q = 2.49$ mW. Note, that it is also possible to introduce the non-linearity of the input function, as indicated in (8.24) directly on the level of the reduced model, by defining u to be a function of the reduced state-vector z. The following code sequence performs the time integration of the full-scale and reduced order models:

Define the input function and the time range for the time integration:

```
global u = 1;
t = linspace(0, 0.04, 100);
```

Now we need the initial values of the full and reduced state vector:

```
x(:,1) = zeros(rows(K), 1);
z(:,1) = zeros(rows(Kr), 1);
```

The time integration of the full-scale model might take quite some time:

```
y = TimeIntegration(E, K, B, C, u, x, t);
```

The time integration of the reduced model should be fast:

```
yr = TimeIntegration(Er, Kr, Br, Cr, u, z, t);
```

It is advisable to save solutions for further use:

```
save("FullSolution", "y");
save("ReducedSolution", "yr");
```

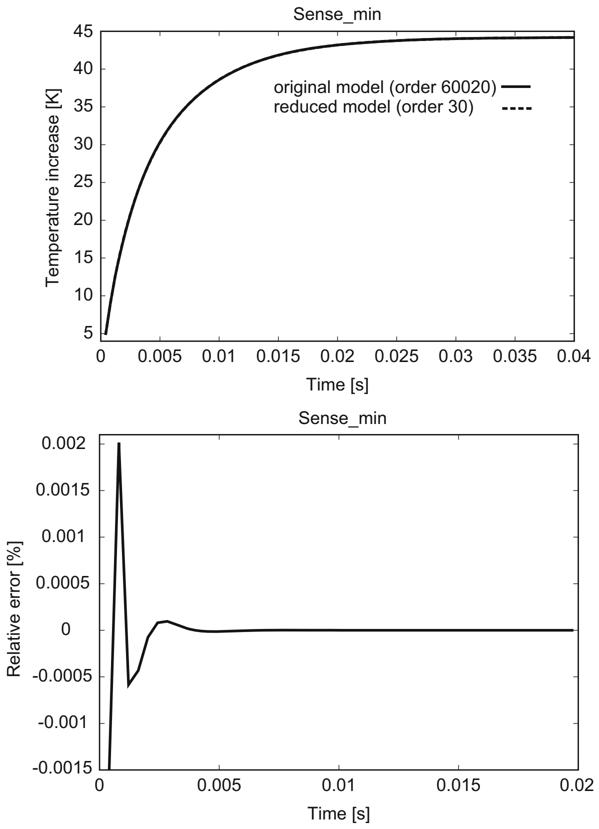


Fig. 8.10 Time response of the full model (order 60000) and reduced model (order 30) (*top*) and relative error between the both (*bottom*)

One can further plot all outputs of the full and reduced system in time domain with:

```
PlotAllOutputs(y, yr, t, "MicroHotplate.C.names");
```

and plot the relative and absolute errors between the full and the reduced model in each node:

```
PlotRelativeErrorTimeDomain(y, yr, t, \
                            "MicroHotplate.C.names");
PlotAbsoluteErrorTimeDomain(y, yr, t, \
                            "MicroHotplate.C.names");
```

Figure 8.10 shows the temperature response in time and the relative error between the full-scale and the reduced order model in `Sense_min` node.

It is further possible to compare the dynamics of the reduced and original system in frequency domain as well. The following functions compute and plot the transfer

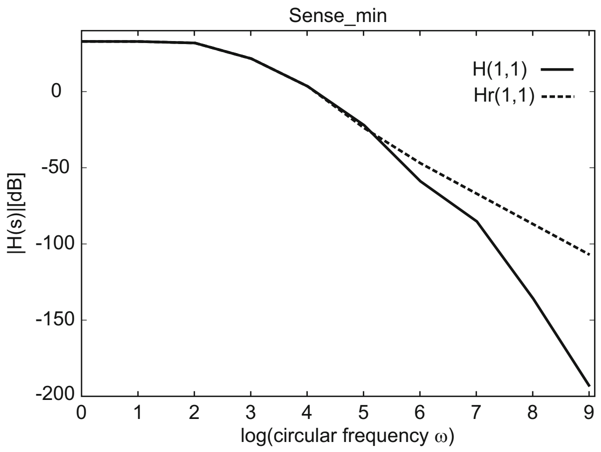


Fig. 8.11 Frequency response H(1,1) of the full model (order 60000) and frequency response Hr(1,1) of the reduced model (order 30)

functions of the full and reduced models for 10 frequency values in the range from 1 to 10^9 rad/s:

Specify the number of frequencies (the frequency range is from 10^0 rad/s to $10^{\text{NrOfFreq}-1}$ rad/s):

```
NrOfFreq = 10;
```

Now compute the transfer function of the full-scale system:

```
G = ComputeTransferFunction(A, B, C, E, NrOfFreq);
```

Compute the transfer function of the reduced system:

```
Gr = ComputeTransferFunction(Ar, Br, Cr, Er, NrOfFreq);
```

Plot the transfer functions between each input/output:

```
PlotAllTransferFunctions(G, Gr, NrOfFreq, \
                         "MicroHotplate.C.names");
```

Figure 8.11 shows both transfer functions in output node Sense_min. The transfer functions match well for low frequencies, which is due to the fact that we have chosen zero as an expansion point for the Taylor series in (8.20).

The main draw-back of the Arnoldi algorithm is that there is no mathematical theory to estimate the reduction error. Hence, for the user it is difficult to predict which dimension of the reduced model will provide the required accuracy. In [7] we have shown that it is possible to estimate the reduction error by observing the relative error in frequency domain between two "neighbored" reduced models of order r and $r + 1$. If we define a relative frequency response error as:

$$E_r(s) = \frac{|H(s) - H_r(s)|}{|H(s)|} \tag{8.21}$$

where $H(s)$ and $H_r(s)$ are the transfer functions of the original and of the reduced order model (as defined in (8.19)), respectively, and a relative frequency-response error between two successive reduced order models as:

$$\hat{E}_r(s) = \frac{|H_r(s) - H_{r+1}(s)|}{|H_r(s)|} \tag{8.22}$$

it turns out that for the micro-hotplate case studies it holds:

$$E_r(s) \approx \hat{E}_r(s) \tag{8.23}$$

For benchmarks from [35] (8.23) holds for a wide range of frequencies around the expansion point $s_0 = 0$. To observe the convergence of the relative error within the *Demonstrator Platform* run:

```
ErrorEstimate(E, A, B, C, V, Freq);
```

where `Freq` is the circular frequency of interest. Figure 8.12 shows the convergence of relative error at 10^3 rad/s and at 10^6 rad/s. We observe that E_r and $\hat{E}_r$ match well at lower frequency 10^3 rad/s, which is near the expansion point $s_0 = 0$. The system order necessary to reach the convergence at $\omega = 10^3$ rad/s is 16, which means that it is not possible to approximate the system better with higher order. The convergence occurs presumably because the machine's numerical precision has been reached. At high frequencies convergence disappears. Instead, we observe fluctuations, due to being too far away from the expansion point.

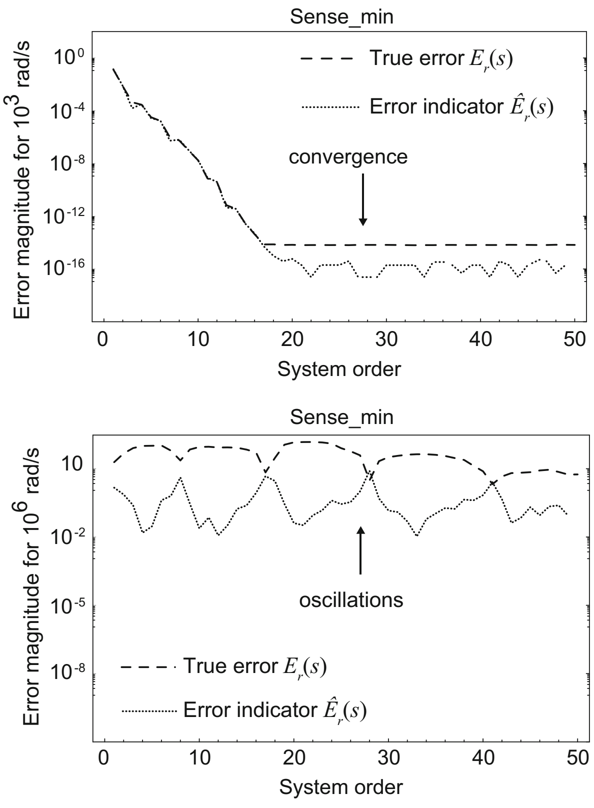
8.2.3.4 Running SLICOT via SLICOTINT for Further Reduction of the Compact Model

The SLICOT subroutines can be used for reduction of moderate size models (order few thousands) and for ODE systems of the form:

$$\begin{aligned} \dot{\mathbf{x}} &= A \cdot \mathbf{x} + B \cdot \mathbf{u} \\ \mathbf{y} &= C \cdot \mathbf{x} \end{aligned} \tag{8.24}$$

only. It is possible to use them to further compact the model that has been obtained by MOR for ANSYS. We have developed the Octave interface SLICOTINT to the model reduction subroutines of the SLICOT library. The provided script `SLICOT_Tutorial.m` gives the guidelines on how to use it and demonstrates how the reduction is performed with the micro-hotplate example.

Fig. 8.12 Error indicators in the frequency domain in output node Sense_min of the micro-hotplate model at $\omega = 10^3$ rad/s (*top*) and at $\omega = 10^6$ rad/s (*bottom*)



We invoke Octave, load the reduced-order model (it is of order 30) into the Octave environment and convert it to the state-space (single-matrix) form, which is necessary for MOR methods implemented in SLICOT:

Load the functions for the MOR postprocessing and the test model:

```
Post4MOR;
[E1,A1,B1,C1] = ReadInTestModel("MicroHotplate.MOR");
```

Convert the model into the state-space (single matrix) form, which is necessary for control-theory routines implemented in SLICOT:

```
A = E1\A1;
B = E1\B1;
C = C1;
```

We call the Balanced Truncation Approximation algorithm, implemented in SLICOT with:

```
[Ar,Br,Cr,HSV] = slicot(A, B, C, 5, "BTA");
```

or with:

```
[Ar,Br,Cr,HSV] = slicot(A, B, C, 0.01, "BTA");
```

The first call invokes reduction with the BTA method to order five (resulting ODE system will have five equations) and the second one, a reduction to the smaller order system with the accuracy of 1 %. The latest is possible, because the MOR methods implemented in SLICOT (also known as control theory methods), provide a global error bound between the transfer function $H(s)$ of the original and $H_r(s)$ of the reduced system as follows:

$$\|H(s) - H_r(s)\|_\infty \leq 2(\sigma_{r+1} + \ldots + \sigma_n) \tag{8.25}$$

where the infinity norm $\|.\|_\infty$ denotes the largest magnitude of the difference of the transfer functions and $\sigma_{r+1}, \ldots, \sigma_n$ are the smallest Hankel singular values (HSV) of the dynamical system under consideration. Hankel singular values are the property of the dynamical system, which reflect the contributions of the different entries of the state vector to system responses (see [3] for theoretical explanation).

For calling one of the other two MOR algorithms for linear ODE systems implemented in SLICOT, that is Hankel Norm Approximation or Singular Perturbation Approximation, it is necessary to replace `BTA` with `HNA` or `SPA`. Each function returns the system matrices of the reduced ODE system, A_r, B_r and C_r, as well as the list of Hankel singular values sorted in descending order.

After reduction, it is possible to display the responses of the full and the reduced order systems in time and/or frequency domain and to plot the relative and absolute errors, by using the same commands as described in Sect. 8.2.3.3. Figure 8.13 shows the responses of two different reduced models produced with the BTA algorithm for the same input function, initial conditions and time- and frequency ranges, as in the previous step.

It is further interesting to observe (see Fig. 8.14), how the target order five model can be reached with smaller error, in transient phase, if sequential MOR is used (reduction of the original model with order 60020 down to order 30 with Arnoldi algorithm and further to order five with BTA) than the Arnoldi algorithm alone. This is due to the fact that the resulting reduced model of order five (gained by sequential MOR) includes information of 30 Arnoldi vectors. The steady-state error increases however, which is typical for the BTA. Better approximation of the steady-state can be reached by using SPA (see [18] for more explanation). With

```
plot(log(HSV),".");
```

we can plot the Hankel singular values of the system. The following commands compute and plot the error bound as defined by (8.25). First, we compute the frequency domain error of reduction (infinity norm):

```
Eps_jw = InfinityNormError(G, Gr, NrOfFreq);
```

Then, we plot the frequency domain error of reduction (infinity norm):

```
PlotInfinityNormError(Eps_jw, NrOfFreq);
```

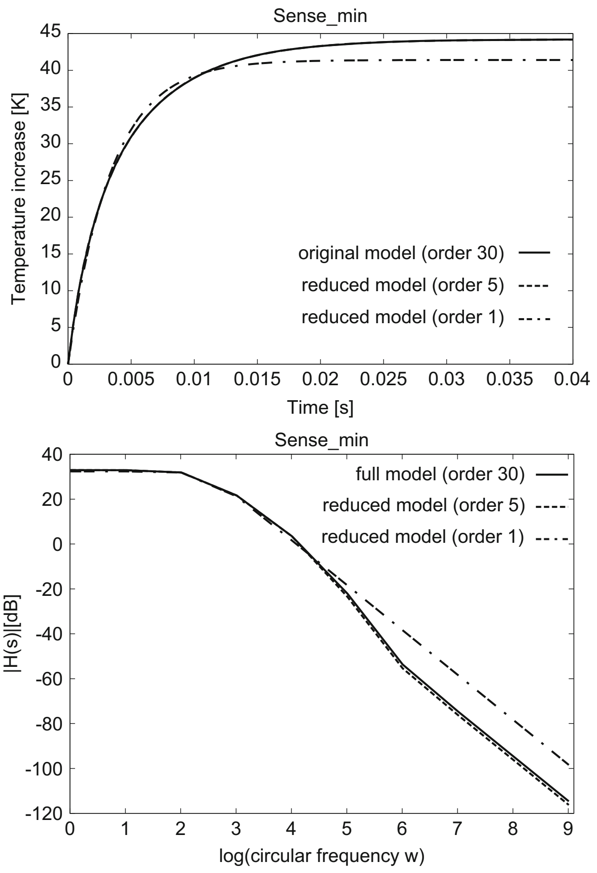


Fig. 8.13 Time response (*top*) and frequency response (*bottom*) in Sense_min output node of the micro-hotplate model of order 30 (gained through the reduction with MOR for ANSYS) and of reduced orders five and one (both gained through further reduction with BTA algorithm from SLICOT)

Figure 8.15 shows the rapid decay of Hankel singular values for the micro-hotplate model of order 30, which indicates further reducibility of the dynamical system. Figure 8.16 shows the difference between the transfer function of the full (order 30) model of the micro-hotplate and of the reduced (order 6) model.

8.2.4 Coupled Simulation (MOR + Circuit Simulation)

The main goal of implementing model order reduction feature into the COMSON *Demonstrator Platform* is to speed up the simulation of complex electronic circuits.

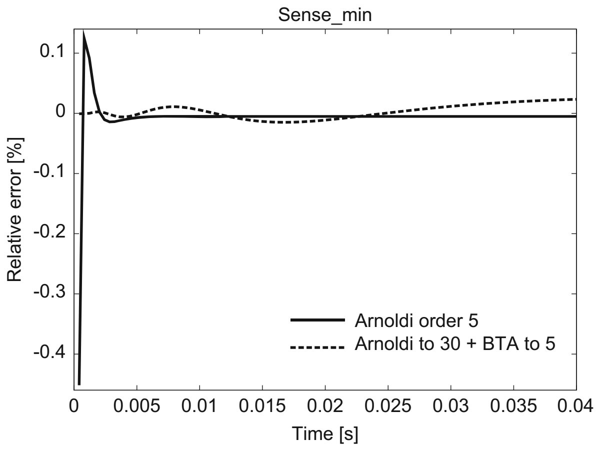


Fig. 8.14 Relative error of the single-step and of the sequential reduction of the micro-hotplate model with Arnoldi algorithm (MOR for ANSYS implementation) and with BTA algorithm (SLICOT implementation)

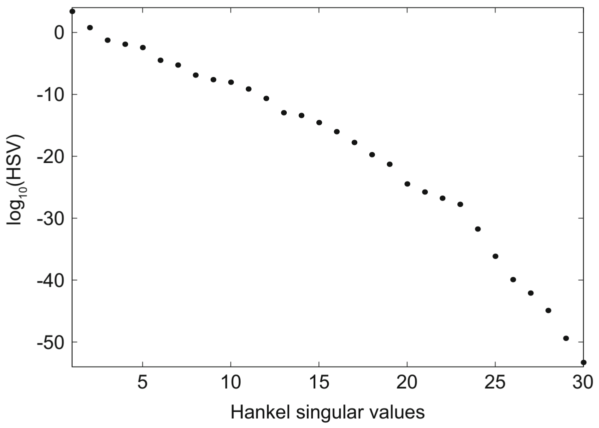


Fig. 8.15 Logarithm of Hankel singular values for the micro-hotplate model of order 30. Original model (order 60020) was reduced with MOR for ANSYS

In this section we demonstrate how MOR can be efficiently applied to build a compact model of interconnects and couple it with the surrounding circuitry.

We simulate the transmission line model from Sect. 8.2.2.2 with 20 resistors and 9 capacitors, which is coupled to the simple non-linear circuit, as shown in Fig. 8.17. The transmission line model is a linear DAE system of form (8.13) which can be reduced within the *Demonstrator Platform* with the MOR for ANSYS library. We would like to emphasize once again, that the entries of the reduced state vector (z in (8.15)) are without physical meaning, i.e. they do not represent internal voltages and charges of the transmission line circuit. However, as during

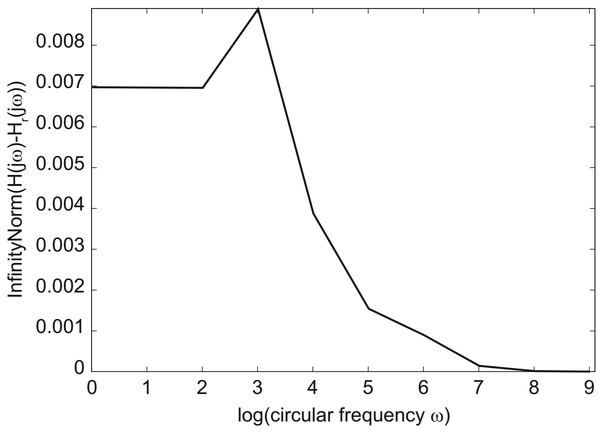


Fig. 8.16 Infinity norm of the difference between the transfer function of the full (order 30) model of the micro-hotplate and of the reduced (order six) model. Reduction was done with SLICOT (BTA) by specifying the error of (8.25) to be 0.01

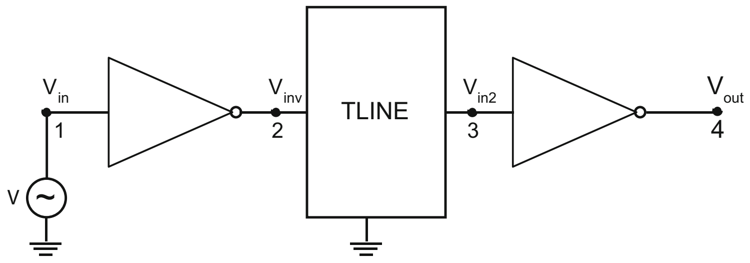


Fig. 8.17 Transmission line model coupled to non-linear circuity

the reduction the coupling states (those entries of the MNA state vector which are common for the transmission line and the surrounding circuit, as V_{inv} and V_{in2} in Fig. 8.17), are preserved, it is possible to couple the reduced order model to both inverters. This coupling is done in the same manner as without reduction, i.e. by stamping the matrices of (8.15) into the global system matrices (system matrices of the whole inverter circuit) at proper positions.

The transmission line model and the inverter circuit are parts of OCS (Octave Circuit Simulator) module. They are described in the following files:

– `MTransLine.cir` (located in the *DP_DIR*/`OCS/SBN`)
– `MTransLine.m` (located in the *DP_DIR*/`OCS/SBN`)
– `TLine2Inv.cir` (located in *DP_DIR*/`OCS/Examples/TLINE`)
– `TLine2Inv.nms` (located in *DP_DIR*/`OCS/Examples/TLINE`)

`MTransLine.cir` describes the circuit from Fig. 8.7 with 9 capacitors and 20 resistors. It has two external variables (dimension of **u** in (8.13)) and 18 internal variables (dimension of **x** in (8.13)).

MTransLine.m builds the local system matrices. It takes as input parameter a string NonReduced or Reduced. If former, the system (8.13) is built. If latter, first the system (8.13) is built and then there is a call to MOR for ANSYS with:

```
ReducedOrder = 3;
V = amor(E, K, B, C, ReducedOrder, "UMFPACK", 0,\
        "metis", 1e-15);
```

reducing the internal states down to 3. Reduced matrices are computed by projecting the original ones as follows:

```
Er = V'*E*V;
Kr = V'*K*V;
Br = V'*B;
Cr = C*V;
```

and the system (8.15) is built.

TLine2Inv.cir describes the circuit from Fig. 8.17. It contains a sinusoidal voltage source with an DC offset of 1.5 V, two inverters and a single transmission line from the template MTransLine.m. The call to transmission line is as follows:

```
% Tranmission line from MTransLine.m
MTransLine NonReduced 2 4
1 4
Rin    Rser    Rpar    Cpar
1e2    1e3     1e5     1e-7
2 3
```

where the string NonReduced can be replaced with Reduced.

TLine2Inv.nms contains the names of the circuit outputs of interest, as follows:

```
%0.1b1
1 Vin
2 Vinv
3 Vin2
4 Vout
```

One can run the simulation of TLine2Inv.cir (either with or without the reduction of the transmission line) by running the Octave script runme.m which is located in the directory *DP_DIR*/OCS/Examples/. It is first necessary to manually set the path to AMOR interface by executing

```
Usetpath(pwd);
```

within the *DP_DIR*/`AMOR` directory. Calling `runme` within Octave yields:

```
Chose an example to run:

  [ 1] rcs
  [ 2] DIODEEXAMPLE
  [ 3] MOR
  [ 4] nmos
  [ 5] pmos
  [ 6] inverter
  [ 7] and
  [ 8] and2
  [ 9] tl
  [10] rect
  [11] MOSIV
  [12] TLINE
  [13] RLC
  [14] RCSPDE

pick a number, any number:
```

and it is necessary to chose the *TLINE* example (number 12). The transient simulation is performed and the function `UTLplotbyname` from *DP_DIR*/`OCS`/`UTL` is used to plot the variables specified in the `TLine2Inv.nms`. Figure 8.18 shows the four node potentials over 0.2 s of transient simulation. If we now change the parameter within the `TLine2Inv.cir` into “Reduced”, we can observe and compare the outputs (nodal voltages) with and without reduction. It is possible to use the MOR post-processing functions from the *MOR_Utilities* module, which were

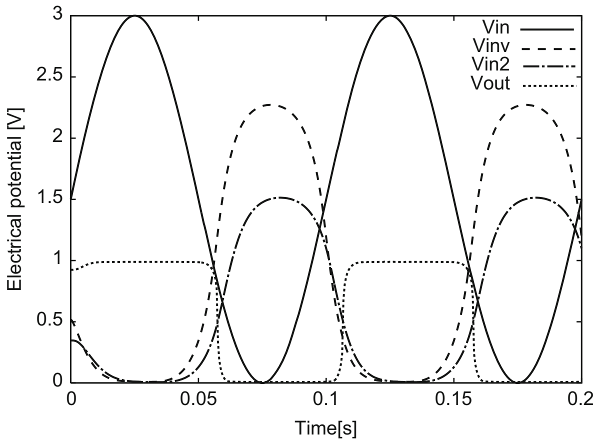


Fig. 8.18 Outputs of the circuit from Fig. 8.17. Transient simulation was done with the OCS module of the COMSON *Demonstrator Platform*

described in the previous section. We observe the nodal voltage V_{out}. The following code saves the solution with and without reduction of the transmission line:

```
if (strcmp(exmpl, "TLINE"))
  if (strcmp(outstruct.LCR(2).section , "NonReduced"))
    FullOutputVoltage = out(:,4);
    save("FullOutputVoltage", FullOutputVoltage);
  elseif(strcmp(outstruct.LCR(2).section , "Reduced"))
    ReducedOutputVoltage = out(:,4);
    save("ReducedOutputVoltage", ReducedOutputVoltage);
  endif
endif
```

We plot full and reduced output and the relative error between the both with:

```
PlotAllOutputs(FullOutputVoltage, \
    ReducedOutputVoltage, t, "TransLineOutputs.names");

PlotRelativeErrorTimeDomain(FullOutputVoltage, \
    ReducedOutputVoltage, t, "TransLineOutputs.names");
```

where in text file `TransLineOutputs.names` we just write the name `V_out`. The resulting plots are displayed in Fig. 8.19. As expected, the change in the output voltage is negligible. As the reduction was performed with MOR for ANSYS, which implements Arnoldi algorithm, and as we have chosen 0 Hz as an expansion point, the steady-state phases are better approximated than the transient steps (see Fig. 8.19, bottom).

8.3 The *Demonstrator Platform* as a Development Tool for Research

In the following it will be shown how the CoMSON *Demonstrator Platform* can be used as an effective tool to prototype new algorithms handling different physical effects in one single framework. In particular the development of a method that allows a self consistent electro-thermal simulation of a n-channel power MOS-FET will be taken into account as a case study.

A description of the physical features of this device is therefore given in Sect. 8.3.1 where the main challenges arising during its design phase are also highlighted. The state-of-the-art modeling procedures actually adopted in industry to cope with these challenges will be then introduced, showing the lack of a method that permits to simulate consistently both the thermal and electrical behavior of the MOS-FET. To overcome this limit an extension of the model is proposed that makes use of a PDE-based thermal element to account for heat-diffusion at the system level (see [1, 14–16]). The reference implementation of the novel method inside the CoMSON *Demonstrator Platform* framework will be analyzed into the details in Sect. 8.3.2. It will be shown first how the modular high-level design allows for an extreme flexibility in defining methods and solution procedures to

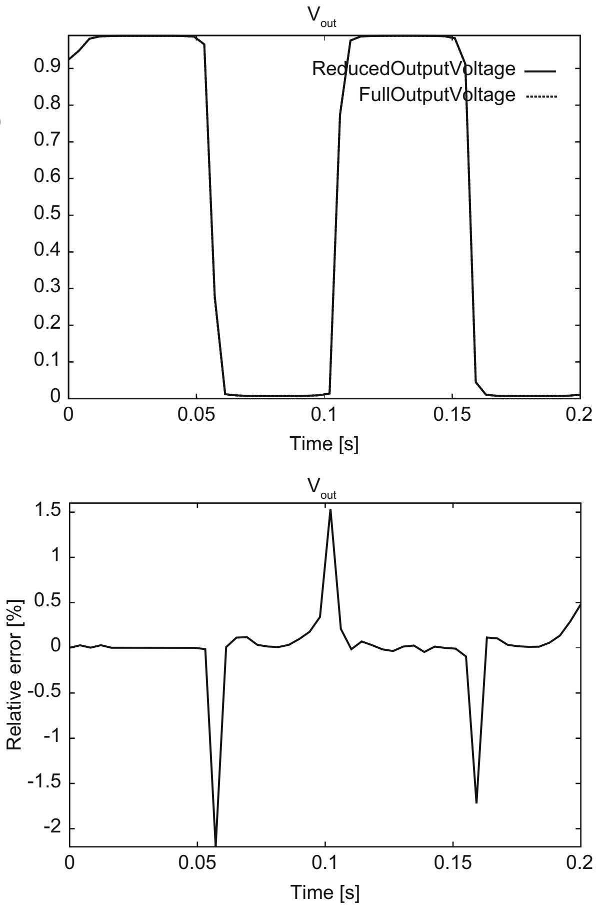


Fig. 8.19 V_{out} from circuit in Fig. 8.17 (*top*) in case when the transient simulation was run without the reduction of the TLINE (FullOutputVolatage) and with the reduction of the TLINE (ReducedOutputVoltage). Relative error between the both (*bottom*)

be used. Then the test-driven development (TDD) philosophy adopted during the implementation will be thoroughly exemplified, showing the progression from basic unit tests (class "A") to simple academic examples requiring the full-functionality of the algorithm (class "B") reaching finally the height of a real-life benchmark (class "C"). Simulation results obtained on this final problem will be then adequately illustrated and commented in Sect. 8.3.3.

8.3.1 Class "C" Benchmark: n-Channel Power MOS-FET

In this section is presented the "real-life" application upon which the algorithm proposed in [1, 16] will be tested to ensure its effectiveness. The choice to introduce the final benchmark at this stage reflects the actual development process of the CoMSON *Demonstrator Platform*, where a combination of top-down and bottom-up strategies was employed to correctly structure the modules and dimension

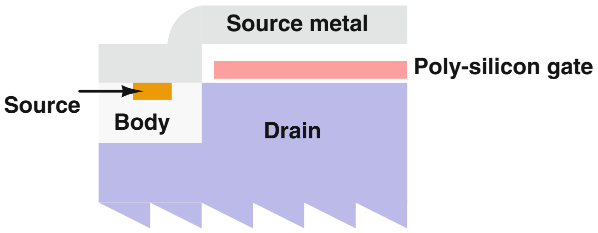


Fig. 8.20 Sketch of the cross-section of the power n-channel MOS-FET: only one metal layer and one polysilicon layer are employed during the fabrication process

the *Demonstrator Platform* itself. The correct identification of a final benchmark constitutes, within this framework, the starting point. This full-size problem will be then analyzed and divided into smaller parts in Sect. 8.3.2 where it will be also shown how already existing modules may be possibly re-used to provide specific functionalities.

8.3.1.1 Physical Features of the n-Channel Power MOS-FET

The device taken into account in the following is a vertical n-channel MOS-FET, mainly used for power applications [9]. As it can be seen in Fig. 8.20 (where a sketch of its cross-section is depicted) the drain contact is placed at the bottom of the die. To maintain the lowest possible production cost, the technology employed for the fabrication of the power MOS-FET exploits only one metal layer and one polysilicon layer. Source and body are thus short-circuited through the source metal layer (to avoid the turn-on of the parasitic npn bipolar transistor), while gate interconnects are laid-out using polysilicon. It should be stressed that the device surface is almost completely covered by source metal, with the only exception of a few regions where the metal layer is exploited to provide low-resistance gate connections.

The device layout (Fig. 8.21) is constituted by several elementary transistors cells connected in parallel to achieve the high current handling capability typical of power devices. The *active device regions* are organized in rows (each of which is a single wide-channel MOS-FET) as the polysilicon interconnects follow horizontal paths from the external gate metal. Due to the poor conductive property of the polysilicon layer, gate metal fingers are used to provide an alternative path between the gate pad and the elementary cells. However, as space has to be left toward the center to allow connection of the source bond wires, these fingers cannot extend from the upper to the lower part of the layout: some elementary cells are thus left without a direct metal connection.

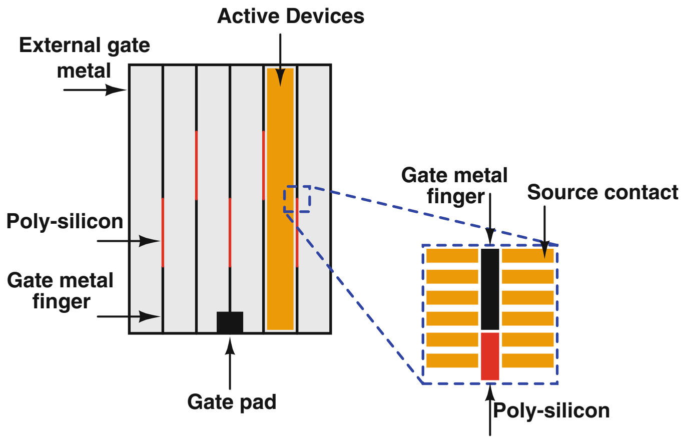


Fig. 8.21 Schematic layout of the power n-channel MOS-FET. Several active cells are connected in parallel. The external gate signal reaches every cell passing through metal fingers (low resistance) or polysilicon interconnects (high resistance)

When the power device operates at low frequencies, the effect on the elementary cells of a polysilicon gate connection instead of a metal one can be safely neglected as in this case the delay of the signal travelling through the polysilicon layer is small compared to the rise and fall times of the input. Anyhow, this condition does not hold when the switching frequencies get higher (as it is the case of many power application). In fact, due to polysilicon high electrical resistance, the signal given at the gate pad reaches some elementary cells with a delay that causes a non-uniform current distribution and the presence of a temperature gradient across the device surface.

8.3.1.2 State-of-the-Art Modeling and Simulation Procedures

The development of new models for power electron devices has been an active area of research in the last years, due mainly to the increasing use of these type of devices in many applications [11, 34, 40]. As the main interest of end users is in optimizing the performance of the circuitry driving the power stage rather than improving the device itself, the most part of these new models is "only" able to reproduce with a reasonable degree of accuracy the static or switching characteristic of the device as observed from external pins [4, 39, 46]. Anyhow they do not provide information on what happens inside the device, and therefore they are not suitable to be used in computer aided tools to improve the layout design of power devices themselves.

To overcome this weakness in [9] a *lumped-element distributed modeling approach* was introduced, which allowed to observe local maxima in the current density distribution. The main idea was to exploit the concepts employed for high frequency modeling of microstrips to describe the electrical behavior of polysilicon and metal interconnects of a power device. Hence the layout design information is used to generate a scalable electrical network feasible to be analyzed by any spice-like circuit simulator. The resulting netlist has a hierarchical structure based upon three basic building blocks, representing respectively:

– Metal over passive area,
– Polysilicon over passive area,
– Polysilicon over active area.

A thorough description of the model is reported in [9].

In [32] the model proposed in [9] was extended to account also for the self-heating of the cells. Still the dependence of the electrical characteristics of each cell on the dissipated power remains purely local, and thus mutual-heating effects are not being caught.

8.3.1.3 Extension: PDE-Based Electro-thermal Circuit Element

To allow the description of non-local heating effects the model presented in [9, 32] will be complemented in the following by the introduction of a PDE-based electro-thermal circuit element [13]. The general idea of this further extension is presented in Fig. 8.22. Starting from available layout and/or package geometry information, a thermal element model is derived directly from PDEs describing heat-diffusion

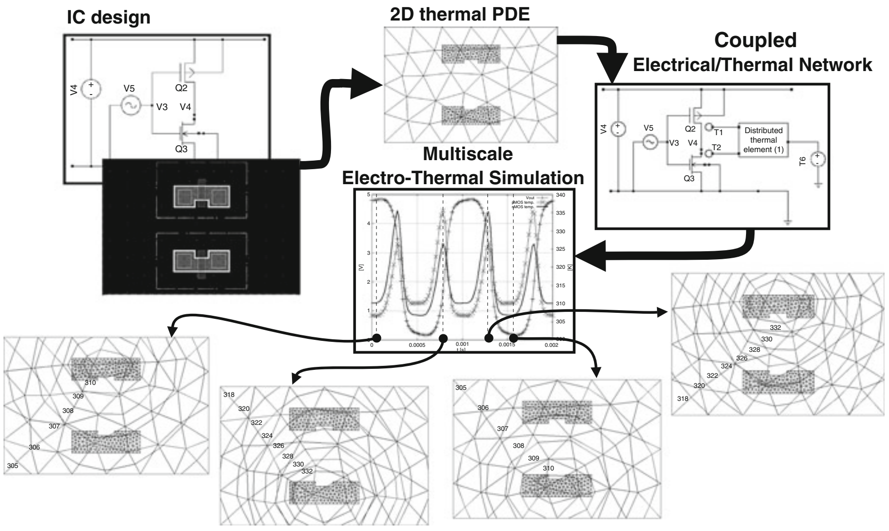

IC design
2D thermal PDE
Coupled
Electrical/Thermal Network
Multiscale
Electro-Thermal Simulation
V4
V5
V3
Q2
Q3
T1
T2
Distributed thermal element (1)
T6

Fig. 8.22 Automated design flow for the electro-thermal simulation of ICs. A thermal element model is automatically constructed from available circuit schematic and design layout, permitting the set-up and simulation of an electro-thermal network that accounts for heat diffusion at the system level

at the system level. By imposing suitable integral conditions this element is casted in a form analogous to that of usual electrical circuit elements, so that its use in a standard circuit simulator requires only the implementation of a new element evaluator. This permits to describe possibly non-linear heat-diffusion phenomena on 2D/3D domains without modifying the main structure of the circuit solver. The mathematical model standing at the base of this approach is thoroughly treated in Chap. 2.

A particular spatial discretization scheme [17, 23, 28–30, 53], based on the use of completely overlapping non-nested meshes, was chosen to cope with multiscale issues. This method has two main advantages for the application at hand:

1. It allows to cover the whole thermal domain with a uniform triangulation without having to excessively refine the mesh to capture small geometrical features,
2. It allows to generate a mesh for each circuit element only once and deploy it at different positions on the IC with a significant time improvement during the mesh generation phase.

The latter feature may also give performance gain if an optimization of the relative device placement is to be performed. In the end the adopted algorithm resembles what it is known in literature as a *brute-force* approach [12], the only difference being that in this case no a-priori interpretation in terms of a circuit netlist is necessary for the discretized PDEs.

8.3.2 Implementation and Development Procedure

In a research project it is normal and desirable that software requirements undergo extensive modifications as development proceeds. In fact, rigidly defining even such basic things as data structures and interfaces at initial stage could severely limit possible future extensions. Nevertheless developers need clear indications to structure a software and start coding without loosing focus on the application. The definition of appropriate benchmarks constitutes an effective way to provide these indications, as it enables a unique and non controversial way of assessing validity of design choices while it allows for an extreme flexibility in implementation. In the following it will be shown how this theoretical principle was applied to prototype the algorithm briefly introduced in Sect. 8.3.1.3 within the CoMSON *Demonstrator Platform* framework.

8.3.2.1 High-Level Design

It is possible to see from the definition of the mathematical model presented in Chap. 2 that the implementation of the algorithm of interest requires functionalities that are typical either of circuit simulators, or of finite-element solvers. The `Octave Circuit Simulator` module (OCS) provides CoMSON *Demonstrator Plat-*

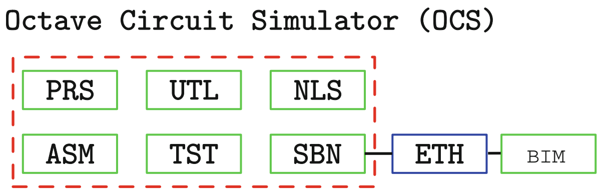


Fig. 8.23 High-level design for the implementation of the algorithm presented in Sect. 8.3.1.3. The basic features required by a circuit simulator are provided by OCS. To implement the PDE-based electro-thermal element a new module (ETH) will be designed with the same interface as OCS element evaluators (SBN)

form with the former features, while `Box Integration Method` module (BIM) provides the latter.

To implement the PDE-based electro-thermal element a new module (ETH) has therefore been devised to provide the necessary link between these different capabilities (see Fig. 8.23). The only constraint imposed by this decision is that the element evaluator should fit the structure:

```
[a,b,c] = M<elname>(string,prms,prmnms,extvar,intvar,t)
```

thoroughly described in the Intermediate File Format specifications [20].

8.3.2.2 Early Stage: Class "A" Test Cases

In the early stage of the implementation it is important to ensure the correctness of each module subroutine as soon as it is coded. This can be done exploiting `octave` testing capabilities that permits to insert test at the end of the function source code [38]. Notice that this feature provides the *Demonstrator Platform* with a simple but effective strategy to perform regression tests.

8.3.2.3 Intermediate Stage: A Class "B" Test Case

Once the designed module has been implemented and its basic functionalities have been assessed through class "A" test cases, a preliminary validation of the method is obtained applying it to a simple problem of academic size. In the case at hand the choice was to simulate the response of the CMOS-inverter circuit depicted in Fig. 8.24 to a 1 kHz sinusoidal input signal [14, 15]. The two MOS-FETs appearing in the schematic are modeled by a simplified version of the classic Shichman-Hodges model [47] with an added temperature pin (the actual parameters used in the simulations are collected in Table 8.2).

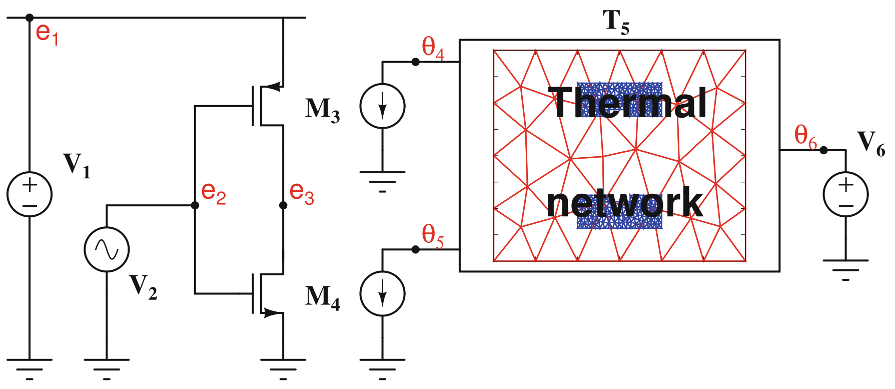


Fig. 8.24 CMOS-inverter electro-thermal network. Inside the thermal element the 2D mesh used for the approximation of heat diffusion on a distributed domain is shown

Table 8.2 Shichman-Hodges MOS-FET model parameters for the CMOS-inverter simulation

	W/L	μ_0	θ_0	V_{th}	r_d	C_{gb}	C_{gd}	C_{gs}	C_{sb}	C_{db}
nMOS	5	10^5	300	0.1	10^6	10^{-11}	10^{-12}	10^{-12}	10^{-12}	10^{-12}
pMOS	5	10^5	300	-0.1	10^6	10^{-11}	10^{-12}	10^{-12}	10^{-12}	10^{-12}

The impact of temperature is represented by a temperature dependent carrier mobility:

$$\mu(\theta) = \mu_0 \left(\frac{\theta}{\theta_0}\right)^{-3/2}, \tag{8.26}$$

where μ denotes the electron mobility for the n-channel transistor M_4 and the hole mobility for the p-channel transistor M_3 and μ_0 is the value of μ at the reference temperature θ_0. The total dissipated Joule-power is given by the simple expression:

$$P = i_{ds} v_{ds} , \tag{8.27}$$

where i_{ds} denotes the current flowing in the controlled current source appearing in the transistor model and v_{ds} is the drain-to-source voltage.

The thermally active regions on the IC substrate are taken to roughly correspond to the channel region of the transistors. Comparing Figs. 8.25 and 8.26 it can be seen that, while maintaining the same mesh refinement in the channel regions, the patched mesh greatly reduces the number of unknowns with respect to a standard conforming triangulation. Linear heat-diffusion is supposed to properly describe thermal effects on the layout (Table 8.3).

The plot in Fig. 8.27 shows the voltage waveforms and the corresponding values of the device temperatures. As it is expected the junction temperatures θ_4 and θ_5 are close to the ambient temperature value $\theta_6 = 300\,\text{K}$ when the output is either in the ON or in the OFF state, as in such situation only small leakage currents flow in the

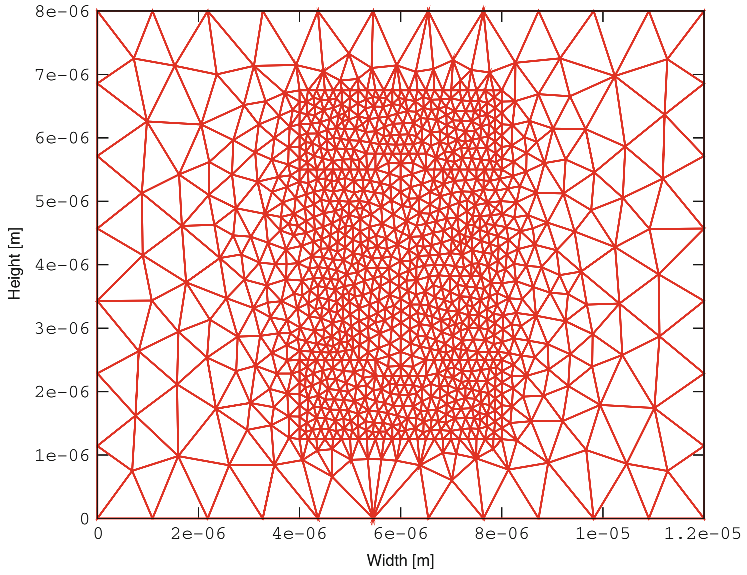


Fig. 8.25 Globally conforming triangulation of 2D chip-layout: 1,052 nodes, 2,066 elements and 1,052 unknowns

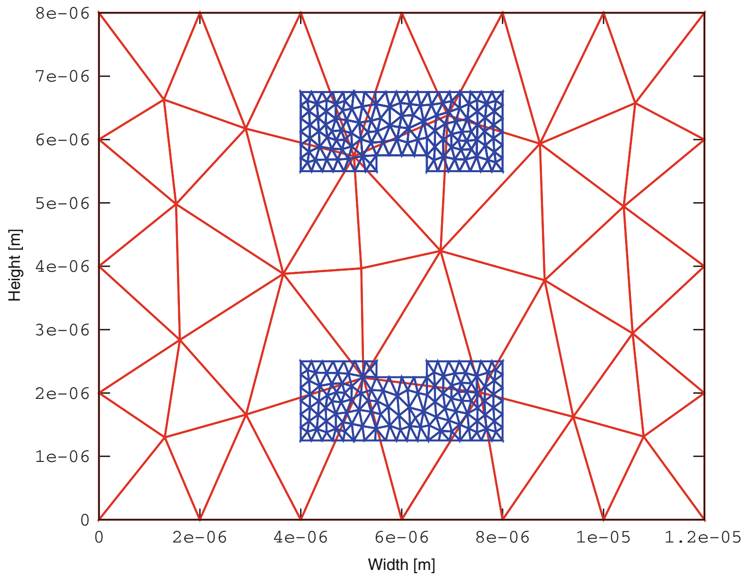


Fig. 8.26 Patched triangulation of 2D chip-layout: 339 nodes, 550 elements and 237 unknowns

Table 8.3 Heat diffusion equation parameters for the CMOS-inverter simulation

$\hat{c}_v$	$\hat{\kappa}$	$\hat{c}$	$\hat{\alpha}$
$1.5 \cdot 10^{-6}$	$1.5 \cdot 10^{-6}$	1	10

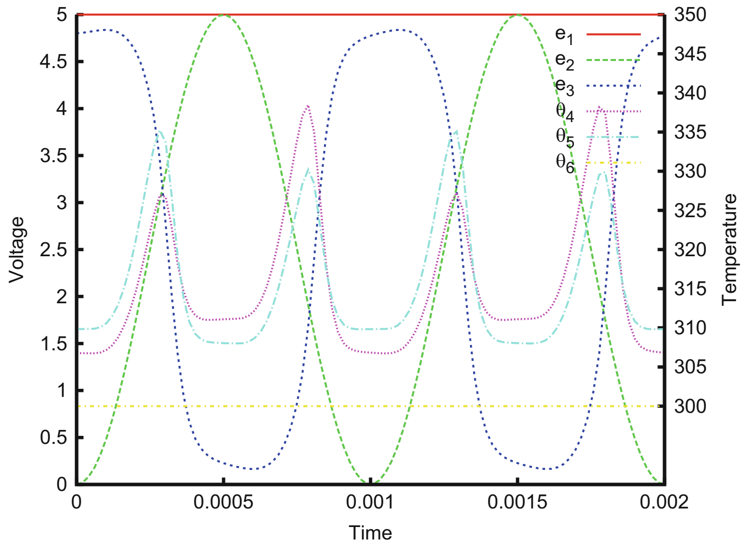


Fig. 8.27 Node voltages and junction temperatures plotted against time for two periods of an input sine voltage at the frequency of 1 kHz

devices, while the current flowing during the ON-to-OFF or OFF-to-ON transitions generates a relatively more significant heating. Figure 8.28 depicts the temperature distribution in the IC substrate at different instants during an OFF-to-ON transition. It can be noted that the heat produced mainly by the p-channel device (above), diffuses through the substrate and affects the n-channel device (below).

8.3.3 Simulation Set-Up and Results on the Class "C" Benchmark

A transient simulation of the turn-off switching of the device introduced in Sect. 8.3.1 is performed as a final validation (Fig. 8.29). This benchmark constitutes a major step toward a real industrial test case, due to its complexity that greatly exceeds the one of usual academic problems. The regularity of the n-channel MOSFET layout permits easily to show an important characteristic of the method, that is to say the possibility to replicate a fine mesh associated with a thermally active area at different positions in the die (Fig. 8.30). This feature allows for the creation of a *library* of electro-thermal devices in which a pre-computed mesh of their active region is included, diminishing thus the computational effort during the mesh

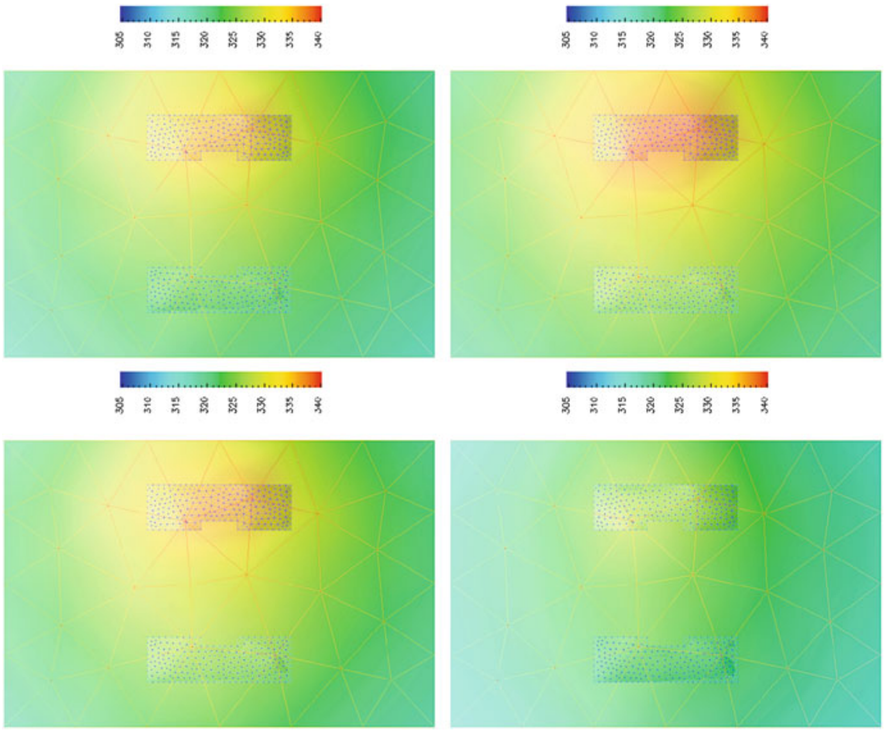


Fig. 8.28 Subsequent snapshots of the distributed temperature field taken during the first switching phase. The heat produced mainly by the p-channel device (above), diffuses through the substrate and affects the n-channel device (below)

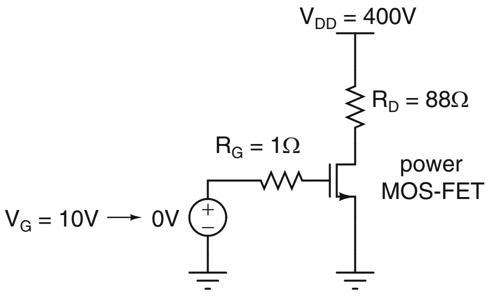


Fig. 8.29 Circuit used to simulate the turn-off switching of the power n-channel MOS-FET. The input signal switches at $t = 3 \times 10^{-9}$ s

generation phase and possibly enabling a performance gain if an optimization of the relative device placement is to be performed.

The electrical behavior of the power n-channel MOS-FET is described by the same *lumped-element distributed approach* presented in [9]. The electrical network

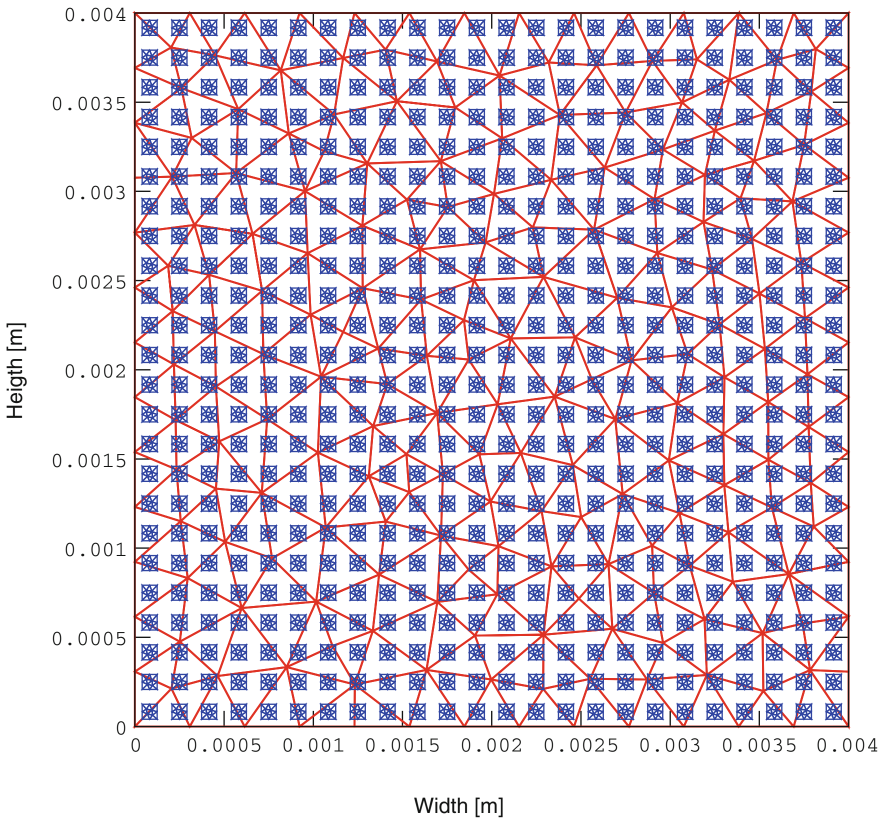


Fig. 8.30 Non-conforming meshes for the whole CHIP and the basic cells. The nMOS-FET model is here scaled to 576 active cells. A coarse grid (in *red*) covers the 4 × 4 mm die, while a fine one (in *blue*) is replicated at each active region position

Table 8.4 Basic cell parameters employed in the simulation of the n-channel MOS-FET turn-off transient. See [9] for more details

	R (series)	L (series)	R (ground)	C (ground)
Metal (passive)	10	10^{-15}	10^{12}	10^{-13}
PolySi (passive)	100	10^{-6}	10^{12}	10^{-12}
PolySi (active)	100	10^{-6}	–	–

is scaled to contain 24×24 active cells and 6 metal fingers. A simplified Shichman-Hodges model with an added temperature pin is used for each elementary transistor cell. Notice that the values of the parameters gathered in Tables 8.4 and 8.5, though fitted to provide realistic results, do not stem from any existing technology.

Thermal effects are supposed to be adequately described by a linear heat-diffusion equation, whose parameters are given in Table 8.6.

Table 8.5 Parameters of the simplified Shichman-Hodges MOS-FET model used to describe the behavior of each active cell

W/L	μ_0	θ_0	V_{th}	r_d	C_{gb}	C_{gd}	C_{gs}	C_{sb}	C_{db}
2.2	10^6	300	0.5	10^9	10^{-12}	10^{-15}	10^{-15}	10^{-15}	10^{-15}

Table 8.6 Parameters of the thermal element employed in the simulation of the power n-channel MOS-FET turn-off

$\hat{c}_v$	$\hat{\kappa}$	$\hat{c}$	$\hat{\alpha}$
10^{-4}	0.02	1,000	$4 \cdot 10^4$

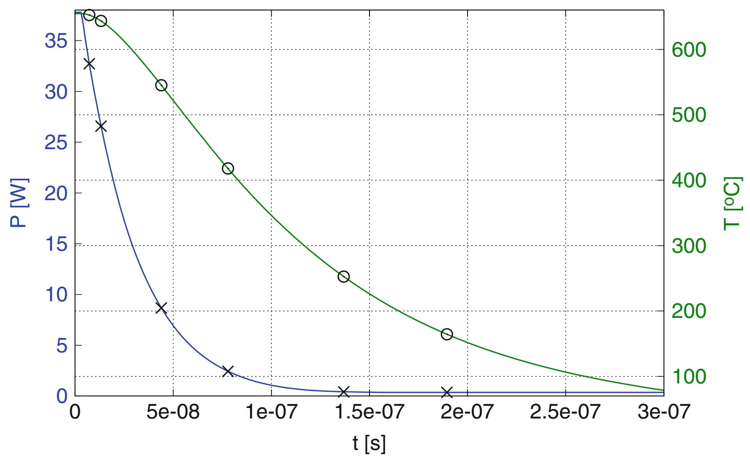


Fig. 8.31 Total dissipated power and mean temperature plotted against time for a turn-off transient. The sampled points refer to the snapshots presented in Figs. 8.32 and 8.33. A simple backward Euler scheme was adopted to time discretize the coupled system

Figure 8.31 shows the total dissipated power and the mean temperature of the device plotted against time. As expected to a lowering of the power corresponds a cooling of the device; however these two effects exhibit different relaxation times. The power densities and junction temperatures of the cells are shown respectively in Figs. 8.32 and 8.33 for six different time-points defined in Fig. 8.31. It can be clearly seen a delay in the propagation of the signal from the gate-pad in the lower part of the die to the single cells, and the presence of an hot-spot in the central upper part of the die for $t = t_2$ and $t = t_3$. Moreover the presence of a non-negligible temperature gradient over the device area is detected at times $t = t_1$ and $t = t_2$. Furthermore the different spatial distribution of heat density and temperature are an indication that non-local effects may not be negligible in estimating the device performance.

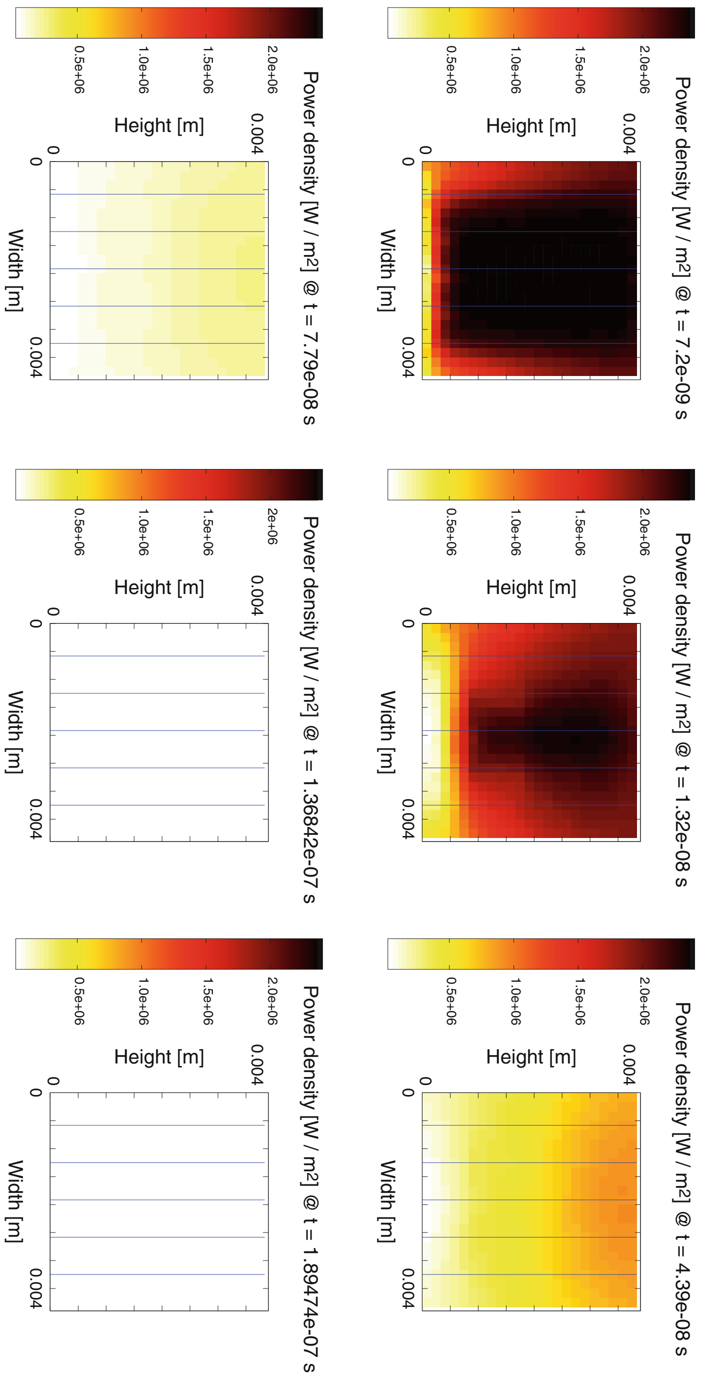


Fig. 8.32 Snapshots of the active cell power densities at the time points t_1, t_2, t_3, t_4, t_5, and t_6 defined in Fig. 8.31

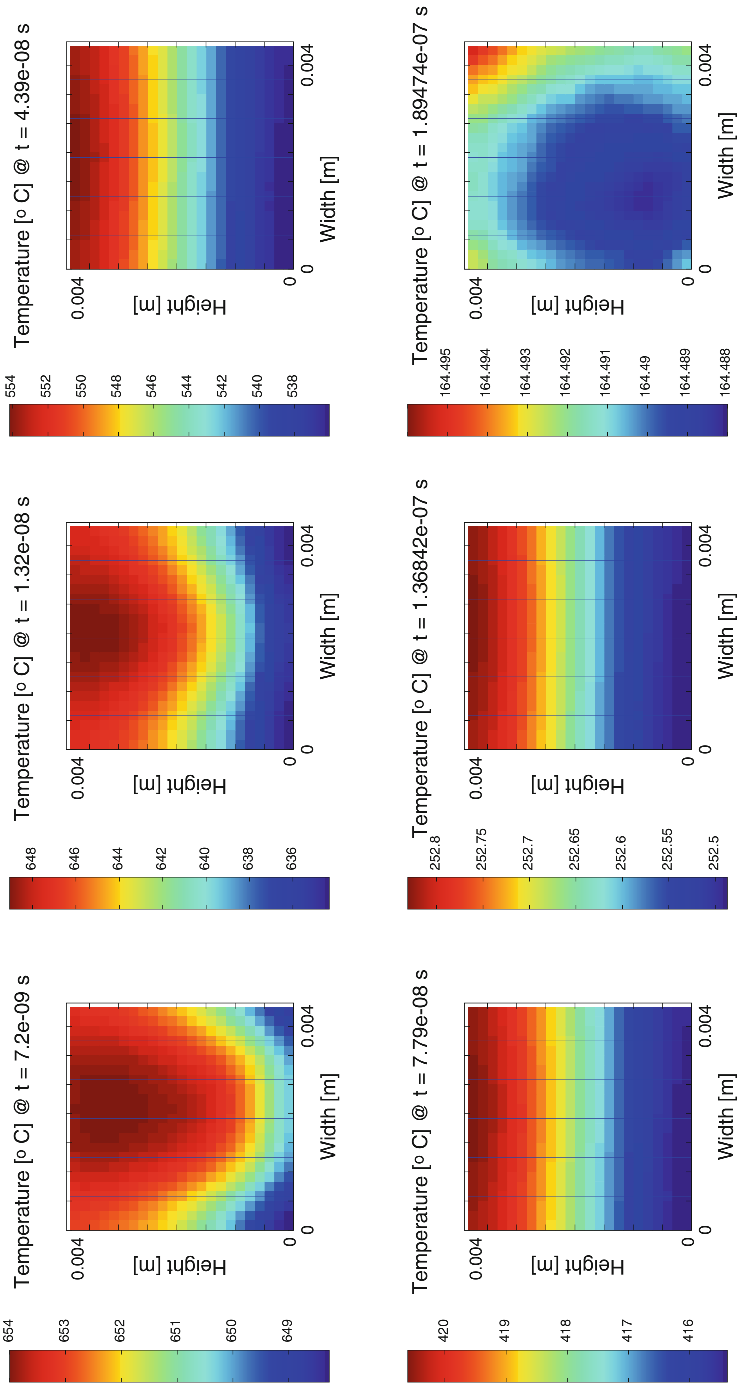


Fig. 8.33 Snapshots of the active cell junction temperatures at the time points t_1, t_2, t_3, t_4, t_5, and t_6 defined in Fig. 8.31

References

1. Alì, G., Bartel, A., Culpo, M., de Falco, C.: Analysis of a PDE thermal element model for electrothermal circuit simulation. In: Roos, J., Costa, L. (eds.) Proceedings of Scientific Computing in Electrical Engineering (SCEE) 2008, Espoo. Mathematics in Industry, vol. 14, pp. 273–280. Springer (2010)
2. Anderson, E., Bai, Z., Bischof, C., Demmel, J., Dongarra, J., Croz, J.D., Greenbaum, A., Hammarling, S., McKenney, A., Ostrouchov, S., Sorensen, D.: LAPACK Users' Guide, 2nd edn. Technical report, SIAM, Philadelphia (1995)
3. Antoulas, A.C.: Approximation of Large-Scale Dynamical Systems. SIAM, Philadelphia (2005)
4. Aubard, L., Verneau, G., Crebier, J., Schaeffer, C., Avenas, Y.: Power MOSFET switching waveforms: an empirical model based on a physical analysis of charge locations. In: IEEE 33rd Annual Power Electronics Specialists Conference, PESC 02, Cairns, vol. 3, pp. 1305–1310 (2002)
5. Bai, Z.J., Meerbergen, K., Su, Y.F.: Arnoldi methods for structure-preserving dimension reduction of second-order dynamical systems. In: Benner, P., Mehrmann, V., Sorensen, D. (eds.) Dimension Reduction of Large-Scale Systems, Oberwolfach. Lecture Notes in Computational Science and Engineering, vol. 45, pp. 173–189 (2005)
6. Bechtold, T., Hohlfeld, D., Rudnyi, E.B., Guenther, M.: Efficient extraction of thin film thermal parameters from numerical models via parametric model order reduction. J. Micromech. Microeng. **20**(4), 045030 (2010)
7. Bechtold, T., Rudnyi, E.B., Korvink, J.G.: Error indicators for fully automatic extraction of heat-transfer macromodels for MEMS. J. Micromech. Microeng. **15**(3), 430–440 (2005)
8. Bechtold, T., Rudnyi, E.B., Korvink, J.G.: Fast Simulation of Electro-thermal MEMS. Springer, Berlin/Heidelberg (2006)
9. Biondi, T., Greco, G., Allia, M., Liotta, S., Bazzano, G., Rinaudo, S.: Distributed modeling of layout parasitics in large-area high-speed silicon power devices. IEEE Trans. Power Electron. **22**(5), 1847–1856 (2007)
10. Boisvert, R.F., Pozo, R., Remington, K.A.: The Matrix Market exchange formats – initial design. http://math.nist.gov/MatrixMarket/formats.html
11. Chen, Y., Lee, F., Amoroso, L., Wu, H.P.: A resonant MOSFET gate driver with complete energy recovery. In: Proceedings IPEMC 2000 the Third International Power Electronics and Motion Control Conference, Beijing, vol. 1, pp. 402–406 (2000)
12. Codecasa, L., D'Amore, D., Maffezzoni, P.: Compact modeling of electrical devices for electrothermal analysis. IEEE Trans. Circuits Syst. I: Fundam. Theory Appl. **50**(4), 465–476 (2003)
13. Culpo, M.: Numerical algorithms for system-level electro-thermal simulation. Ph.D. thesis, Bergische Universität Wuppertal (2009)
14. Culpo, M., de Falco, C.: Dynamical iteration schemes for coupled simulation in nanoelectronics. In: Proceedings in Applied Mathematics and Mechanics, PAMM 2008. Wiley (2009)
15. Culpo, M., de Falco, C.: Dynamical iteration schemes for multiscale simulation in nanoelectronics. In: Proceedings in Applied Mathematics and Mechanics, PAMM 2008. Wiley (2009)
16. Culpo, M., de Falco, C., Denk, G., Voigtmann, S.: Automatic thermal network extraction and multiscale electro-thermal simulation. In: Roos, J., Costa, L.R.J. (eds.) Scientific Computing in Electrical Engineering SCEE 2008, Espoo. Mathematics in Industry. Springer, Berlin/Heidelberg (2010)
17. Culpo, M., de Falco, C., O'Riordan, E.: Patches of finite elements for singularly-perturbed diffusion reaction equations with discontinuous coefficients. In: Fitt, A., Norbury, J., Ockendon, H. (eds.) Proceedings of the 2008 ECMI Conference, London. Mathematics in Industry. Springer (2009)
18. Datta, B.N.: Numerical Methods for Linear Control Systems. Elsevier Incorporation, Amsterdam/Boston (2004)

19. Davis, T.A.: UMFPACK. http://www.cise.ufl.edu/research/sparse/umfpack
20. de Falco, C.: Specification of an intermediate file format for the CoMSON demonstrator platform. Technical report, Bergische Universität Wuppertal (2006)
21. de Falco, C., Denk, G., Schultz, R.: A demonstrator platform for coupled multiscale simulation. In: Ciuprina, G., Ioan, D. (eds.) Scientific Computing in Electrical Engineering (SCEE) 2006, Sinaia. Mathematics in Industry, pp. 63–72. Springer (2007)
22. de Falco, C., Gatti, E., Lacaita, A., Sacco, R.: Quantum-corrected drift-diffusion models for transport in semiconductor devices. J. Comput. Phys. **204**(2), 533–561 (2005)
23. de Falco, C., O'Riordan, E.: A patched mesh method for singularly perturbed reaction-diffusion equations. In: Hegarty, A., O'Riordan, N.K.E., Stynes, M. (eds.) Proceedings of the International Conference on Boundary and Interior Layers – Computational and Asymptotic Methods, Limerick. Lecture Notes in Computational Science and Engineering, vol. 69. Springer (2009)
24. de Falco, C., O'Riordan, E.: Interior layers in a reaction-diffusion equation with a discontinuous diffusion coefficient. Int. J. Numer. Anal. Model. **7**(3), 444–461 (2010)
25. de Falco, C., Sacco, R., Jerome, J.: Quantum corrected drift-diffusion models: solution fixed point map and finite element approximation. J. Comput. Phys. **228**, 1770–1789 (2009)
26. Freund, R.: Krylov-subspace methods for reduced-order modeling in circuit simulation. J. Comput. Appl. Math. **123**, 395–421 (2000)
27. Glover, K.: All optimal Hankel norm approximation of linear multivariable systems and their L-infinity error bounds. Int. J. Control **36**, 1145–1193 (1984)
28. Glowinski, R., He, J., Lozinski, A., Rappaz, J., Wagner, J.: Finite element approximation of multi-scale elliptic problems using patches of elements. Numer. Math. **101**(4), 663–687 (2005)
29. Glowinski, R., He, J., Rappaz, J., Wagner, J.: Approximation of multi-scale elliptic problems using patches of finite elements. C. R. Math. Acad. Sci. Paris **337**(10), 679–684 (2003)
30. Glowinski, R., He, J., Rappaz, J., Wagner, J.: A multi-domain method for solving numerically multi-scale elliptic problems. C. R. Math. Acad. Sci. Paris **338**(9), 741–746 (2004)
31. Graf, M., Barrettino, D., Taschini, S., Hagleitner, C., Hierlemann, A., Baltes, H.: Metal oxide-based monolithic complementary metal oxide semiconductor gas sensor microsystem. Anal. Chem. **76**, 4437–4445 (2004)
32. Greco, G., Rallo, C.: XA integration in custom power MOSFET analysis flow. In: SNUG 2008 Proceedings, Bangalore (2008)
33. Hohlfeld, D., Zappe, H.: Thermal and optical characterization of silicon-based tunable optical thin-film filters. J. Microelectromech. Syst. **16**(3), 500–510 (2007)
34. Hong, S., Lee, Y.G.: Active gate control strategy of series connected IGBTs for high power PWM inverter. In: Proceedings of the IEEE 1999 International Conference on Power Electronics and Drive Systems, PEDS '99, Hong Kong, vol. 2, pp. 646–652 (1999)
35. IMTEK: Oberwolfach model reduction benchmark collection. http://portal.uni-freiburg.de/imteksimulation/downloads/benchmark
36. Liu, Y., Anderson, B.: Singular perturbation approximation of balanced systems. Int. J. Control **50**, 1379–1405 (1989)
37. NXP: SiMKit library. http://www.nxp.com/models/source/
38. Octave: homepage. http://www.gnu.org/software/octave/
39. Pagano, R.: Characterization, parameter identification, and modeling of a new monolithic emitter-switching bipolar transistor. IEEE Trans. Electron Devices **53**(5), 1235–1244 (2006)
40. Raciti, A., Belverde, G., Galluzzo, A., Greco, G., Melito, M., Musumeci, S.: Control of the switching transients of IGBT series strings by high-performance drive units. IEEE Trans. Ind. Electron. **48**(3), 482–490 (2001)
41. Romano, V.: Non-parabolic band hydrodynamical model of silicon semiconductors and simulation of electron devices. Math. Methods Appl. Sci. **24**(7), 439–471 (2001)
42. Romano, V., Rusakov, A.: 2D numerical simulation of electron-phonon MEP based model for semiconductors. In: ICTT-21, Torino (2009)

43. Romano, V., Rusakov, A.: Numerical simulation of semiconductor devices by the MEP energy-transport model with crystal heating. In: Michielsen, B., Poirier, J.R. (eds.) Scientific Computing in Electrical Engineering (SCEE) 2010, Toulouse. Mathematics in Industry, pp. 357–363. Springer, Berlin/New York (2012)
44. Rudnyi, E.: Model reduction software. http://modelreduction.com/Software.html
45. Rudnyi, E.B., Korvink, J.G.: Model order reduction for large scale engineering models developed in ansys. Lect. Notes Comput. Sci. **3732**, 349–356 (2006)
46. Schroder, S., De Doncker, R.: Physically based models of high power semiconductors including transient thermal behavior. IEEE Trans. Power Electron. **18**(1), 231–235 (2003)
47. Shichman, H., Hodges, D.: Modeling and simulation of insulated-gate field-effect transistor switching circuits. IEEE J. Solid-State Circuits **3**(3), 285–289 (1968)
48. Silveira, L.M., Kamon, M., Elfadel, I., White, J.: A coordinate-transformed Arnoldi algorithm for generating guaranteed stable reduced-order models of RLC circuits. In: Technical Digest of the 1996 IEEE/ACM International Conference on Computer-Aided Design, pp. 288–294. IEEE Computer Society, Los Alamitos (1996)
49. SLICOT: The control and systems library. http://www.slicot.org
50. Spannhake, J., Schulz, O., Helwig, A., Müller, G., Doll, T.: Design, development and operational concept of an advanced MEMS IR source for miniaturized gas sensor systems. In: Proceedings of the IEEE Sensors Conference, Irrine, California, pp. 762–765 (2005)
51. Toledo, S., Chen, D., Rotkin, V.: TAUCS – a library of sparse linear solvers. http://www.tau.ac.il/~stoledo/taucs
52. Tombs, M.S., Postlethwaite, I.: Truncated balanced realization of stable, non-minimal state-space systems. Int. J. Control **46**, 1319–1330 (1987)
53. Wagner, J.: Finite element methods with patches and applications. Ph.D. thesis, EPFL, Lausanne (2006)
54. Woellenstein, J., Boettner, H., Plaza, J.A., Cane, C., Min, Y., Tuller, H.L.: A novel single chip thin film metal oxide array. Sens. Actuators B: Chem. **93**(1–3), 350–355 (2003)