Weierstraß-Institut
für Angewandte Analysis und Stochastik
Leibniz-Institut im Forschungsverbund Berlin e. V.

Technical Report ISSN 1618 – 7776

WIAS–TeSCA — Two-dimensional semi-conductor analysis package

Herbert Gajewski, Matthias Liero, Reiner Nürnberg, Holger Stephan

submitted: May 27, 2016

Weierstrass Institute
Mohrenstr. 39
10117 Berlin
Germany
email: Herbert.Gajewski@wias-berlin.de
Matthias.Liero@wias-berlin.de
Reiner.Nürnberg@wias-berlin.de
Holger.Stephan@wias-berlin.de

No. 14
Berlin 2016

2010 *Mathematics Subject Classification.* 65M08, 65N08.

Key words and phrases. semiconductor analysis package, semiconductor simulation, semiconductor heterostructures, van Roosbroeck drift-diffusion model, bipolar transistor simulation, MOS transistor simulation, laser diode simulation, solar cell simulation, silicon semiconductor devices, III-V compounds semiconductor devices, silicon carbide semiconductor devices.

We thank Uwe Bandelow, Bernd Heinemann, Hartmut Langmach, Norbert Strecker, Hans-Jürgen Wünsche and Hans Wenzel for their contribution to earlier versions of this documentation and Jens Griepentrog for technical assistance with the layout.

Edited by
Weierstraß-Institut für Angewandte Analysis und Stochastik (WIAS)
Leibniz-Institut im Forschungsverbund Berlin e. V.
Mohrenstraße 39
10117 Berlin
Germany

Fax: +49 30 20372-303
E-Mail: preprint@wias-berlin.de
World Wide Web: http://www.wias-berlin.de/

Abstract

WIAS-TeSCA (Two-dimensional semiconductor analysis package) is a sim-ulation tool for the numerical simulation of charge transfer processes in semiconductor structures, especially in semiconductor lasers. It is based on the drift-diffusion model and considers a multitude of additional physical effects, like optical radiation, temperature influences and the kinetics of deep impurities. Its efficiency is based on the analytic study of the strongly nonlinear system of partial differential equations – the van Roosbroeck system – which describes the electron and hole currents. Very efficient numerical procedures for both the stationary and transient simulation have been implemented.

WIAS-TeSCA has been successfully used in the research and industrial development of new electronic and optoelectronic semiconductor devices such as transistors, diodes, sensors, detectors and lasers and has already proved its worth many times in the planning and optimization of these devices. It covers a broad spectrum of applications, from heterobipolar transistor (mobile telephone systems, computer networks) through high-voltage transistors (power electronics) and semiconductor laser diodes (fiber optic communication systems, medical technology) to radiation detectors (space research, high energy physics).

WIAS-TeSCA is an efficient simulation tool for analyzing and designing modern semiconductor devices with a broad range of performance that has proved successful in solving many practical problems. Particularly, it offers the possibility to calculate self-consistently the interplay of electronic, optical and thermic effects.

Contents

About this manual

WIAS-TeSCA (Two-dimensional semiconductor analysis package) is a simulation tool for the numerical simulation of charge transfer processes in semiconductor structures, especially in semiconductor lasers. It is based on the drift-diffusion model and considers a multitude of additional physical effects, like optical radiation, temperature influences and the kinetics of deep impurities. Its efficiency is based on the analytic study of the strongly nonlinear system of partial differential equations – the van Roosbroeck system – which describes the electron and hole currents. Very efficient numerical procedures for both the stationary and transient simulation have been implemented.

WIAS-TeSCA has been successfully used in research and industrial development of new electronic and optoelectronic semiconductor devices such as transistors, diodes, sensors, detectors and lasers and has already proved its worth many times in the planning and optimization of these devices. It covers a broad spectrum of applications, from hetero-bipolar transistor (mobile telephone systems, computer networks) through high-voltage transistors (power electronics) and semiconductor laser diodes (fiber optic communication systems, medical technology) to radiation detectors (space research, high energy physics).

WIAS-TeSCA is an efficient simulation tool for analyzing and designing modern semiconductor devices with a broad range of performance that has proved successful in solving many practical problems. Particularly, it offers the possibility to calculate self-consistently the interplay of electronic, optical and thermic effects.

This user manual describes how to use WIAS-TeSCA. It is divided into the following parts:

- Chapter 1 presents the physical models that are implemented in WIAS-TeSCA.
- Chapter 2 describes the script language used in WIAS-TeSCA and how to do simulations.
- In Chapter 3 information on the numerical schemes and implementations is given.

1 Physics in TeSCA

The aim of this chapter is to sketch the mathematical and physical model which is used by the simulation tool `WIAS-TeSCA`. Instructions and references for the related `WIAS-TeSCA` commands, which are descibed in detail in Chapter 2, are given.

1.1 Fundamental system of equations

1.1.1 Drift-diffusion model

`WIAS-TeSCA` is a simulation tool which is designed for numerically solving the fundamental system of equations of charge carrier transport in semiconductors in the two-dimensional case (cross sections or rotational symmetry, see Fig. 1.1 and description of `DEVICE` command). The widely accepted phenomenological system of equations for modeling semiconductor devices was deduced by van Roosbroeck [vR50] in 1950. It is based on Boltzmann statistics and is governed by the Poisson equation and continuity equations for electrons and holes, respectively, that read

$$-\nabla \cdot (\varepsilon_0 \varepsilon_\mathrm{r} \nabla \varphi) = q(C_\mathrm{net} + p - n), \tag{1.1a}$$

$$q\frac{\partial}{\partial t} n - \nabla \cdot \mathbf{J}_n = q(G - R), \tag{1.1b}$$

$$q\frac{\partial}{\partial t} p + \nabla \cdot \mathbf{J}_p = q(G - R). \tag{1.1c}$$

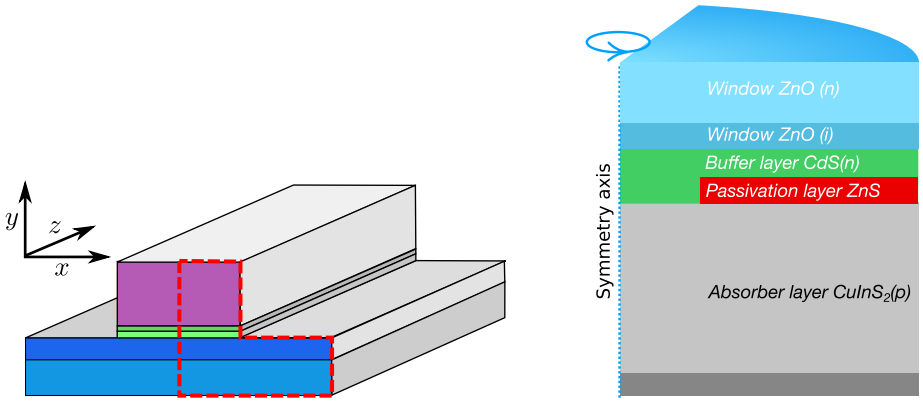

Figure 1.1: Left: Sketch of ridge waveguide laser, Right: Sketch of a cylindric solar cell

The three calculated physical quantities are the electron density n, the hole density p, and the electrostatic potential φ. The latter gives the electric field $\mathbf{E} = -\nabla\varphi$. The electron and hole densities and the electrostatic potential are functions of time t and of two spatial coordinates x and y. Further variables are

ε_0	vacuum permittivity $\approx 8.854 \cdot 10^{-12}$ C/(Vm),
ε_r	relative permittivity of the material,
q	elementary charge $\approx 1.6021 \cdot 10^{-19}$As,
C_net	net doping density of donators and acceptors $= N_\mathrm{D} - N_\mathrm{A}$,
$\mathbf{J}_n$, $\mathbf{J}_p$	vectorial current density of electrons and holes, respectively,
$G-R$	generation-recombination rate.

The vectorial electron and hole current densities $\mathbf{J}_n$ and $\mathbf{J}_p$, respectively, are calculated from φ, n, and p as follows

$$\mathbf{J}_n = -qn\mu_n\nabla\varphi_n, \quad \mathbf{J}_p = -qp\mu_p\nabla\varphi_p, \tag{1.2}$$

where μ_n, μ_p denoted the mobility of electrons and holes, respectively, and φ_n and φ_p are the quasi-Fermi potentials of electrons and holes. The quasi-Fermi potentials φ_n and φ_p are related to the electron and hole densities n and p by

$$n = N_\mathrm{c}\mathcal{F}\Big[\frac{q(\varphi-\varphi_n) - E_\mathrm{c}}{k_\mathrm{B}T}\Big], \quad \text{and} \quad p = N_\mathrm{v}\mathcal{F}\Big[\frac{q(\varphi_p-\varphi) + E_\mathrm{v}}{k_\mathrm{B}T}\Big], \tag{1.3}$$

where

T	lattice temperature,
k_B	Boltzmann constant $\approx 1.380662 \cdot 10^{-23}$ VA/K,
$N_\mathrm{c}, N_\mathrm{v}$	effective density of states of electrons and holes, respectively,
$E_\mathrm{c}, E_\mathrm{v}$	conduction and valence band edge, respectively,
$\mathcal{F}$	Boltzmann or Fermi statistic.

`WIAS-TeSCA` is suited to solve the van Roosbroeck system (1.1)–(1.3) numerically in practically any two-dimensional area (including three-dimensional domains with rotational symmetry). Heterostructures are modeled by dividing the computational domain into several subareas (material regions). For each subarea individual material parameters (ε_r, $E_\mathrm{c}, E_\mathrm{g}, N_\mathrm{c}, N_\mathrm{v}$, etc.) can be defined.

For transient calculations, in `WIAS-TeSCA` the continuity equations for the electrons and holes (1.1b) and (1.1c) as well as the total current balance instead of the Poisson equation are used, namely

$$\nabla\cdot\mathbf{J} = 0, \quad \mathbf{J} = \mathbf{J}_n + \mathbf{J}_p - \varepsilon_0\varepsilon_\mathrm{r}\nabla\frac{\partial\varphi}{\partial t}. \tag{1.4}$$

Here, the total current $\mathbf{J}$ is the sum of electron hole and displacement current.[1]

[1] The continuity equation in (1.4) can be obtained by differentiating the Poisson equation in (1.1a) with respect to time

$$\nabla\cdot\Big(\varepsilon_0\varepsilon_\mathrm{r}\frac{\partial}{\partial t}\mathbf{E}\Big) = q\Big(\frac{\partial p}{\partial t} - \frac{\partial n}{\partial t}\Big)$$

WIAS-TeSCA is designed for the numerical treatment of both the stationary and the transient case. To limit the numerical complexity, defect calculations are used to decide dynamically whether the full system can be reduced temporary by suppressing one equation.

1.1.2 Equations of state

In WIAS-TeSCA, either Boltzmann statistics or Fermi-Dirac statistics can be used. In general, the following relationship between the charge carrier densities n, p, the electrostatic potential φ, and the quasi-Fermi potentials φ_n, φ_p is assumed

$$n = N_{\mathrm{c}}\,\mathcal{F}(\eta_n), \quad \eta_n = \frac{q(\varphi - \varphi_n) - E_{\mathrm{c}}}{k_{\mathrm{B}}T}, \tag{1.5a}$$

$$p = N_{\mathrm{v}}\,\mathcal{F}(\eta_p), \quad \eta_p = \frac{q(\varphi_p - \varphi) + E_{\mathrm{v}}}{k_{\mathrm{B}}T}. \tag{1.5b}$$

The function $\mathcal{F}$ is given via

$$\mathcal{F}(\eta) = \begin{cases} F_{1/2}(\eta) & \Leftrightarrow \text{ Fermi-Dirac statistics,} \\ \exp(\eta) & \Leftrightarrow \text{ Boltzmann statistics.} \end{cases} \tag{1.6}$$

The Fermi integral $F_{1/2}(s)$, which holds for Fermi-Dirac statistics for free particles, is defined in the following way:

$$F_{1/2}(\eta) = \frac{2}{\sqrt{\pi}} \int_0^\infty \frac{\sqrt{y}}{1 + \exp(y - \eta)} \mathrm{d}y. \tag{1.7}$$

If Boltzmann statistics holds, the relations in (1.5) can be written in terms of the intrinsic charge carrier density n_{i} and intrinsic Fermi level E_{i}

$$n = n_{\mathrm{i}} \exp\left[\frac{q(\varphi - \varphi_n) - E_{\mathrm{i}}}{k_{\mathrm{B}}T}\right], \tag{1.8a}$$

$$p = n_{\mathrm{i}} \exp\left[\frac{q(\varphi_p - \varphi) + E_{\mathrm{i}}}{k_{\mathrm{B}}T}\right], \tag{1.8b}$$

$$n_{\mathrm{i}} = \sqrt{N_{\mathrm{c}} N_{\mathrm{v}}} \exp\left[\frac{E_{\mathrm{v}} - E_{\mathrm{c}}}{2k_{\mathrm{B}}T}\right], \tag{1.8c}$$

$$E_{\mathrm{i}} = \frac{E_{\mathrm{c}} + E_{\mathrm{v}}}{2} + \frac{k_{\mathrm{B}}T}{2} \ln\left[\frac{N_{\mathrm{v}}}{N_{\mathrm{c}}}\right]. \tag{1.8d}$$

---

and by substituting $\frac{\partial}{\partial t}p$ and $\frac{\partial}{\partial t}n$ using the continuity equations in (1.1c) and (1.1b)

$$\begin{aligned} q\Big(\frac{\partial p}{\partial t} - \frac{\partial n}{\partial t}\Big) &= \big(q(G - R) - \nabla \cdot \mathbf{J}_p\big) - \big(q(G - R) + \nabla \cdot \mathbf{J}_n\big) \\ &= \nabla \cdot (\mathbf{J}_n + \mathbf{J}_p). \end{aligned}$$

By canceling the derivatives with respect to time $\frac{\partial}{\partial t}n$ and $\frac{\partial}{\partial t}p$ the transient case becomes stationary.

In the case of Fermi-Dirac statistics, the actual calculation of the densities is realized by replacing the exponential function of the Boltzmann case with the Fermi integral (1.7) to the index 1/2. However, these relations are traced back to the Boltzmann relations in the intern implementation by iteratively calculated correction variables, which read

$$\gamma_n = \frac{F_{1/2}(\eta_n)}{\exp(\eta_n)}, \quad \text{and} \quad \gamma_p = \frac{F_{1/2}(\eta_p)}{\exp(\eta_p)}. \tag{1.9a}$$

With this, we can write the Fermi case as a corrected Boltzmann statistic via

$$n = n_{\mathrm{i}}' \exp\left[\frac{q(\varphi - \varphi_n) - E_{\mathrm{i}}'}{k_{\mathrm{B}}T}\right], \tag{1.9b}$$

$$p = n_{\mathrm{i}}' \exp\left[\frac{q(\varphi_p - \varphi) + E_{\mathrm{i}}'}{k_{\mathrm{B}}T}\right], \tag{1.9c}$$

$$n_{\mathrm{i}}' = n_{\mathrm{i}}\sqrt{\gamma_n \gamma_p}, \tag{1.9d}$$

$$E_{\mathrm{i}}' = E_{\mathrm{i}} + \frac{k_{\mathrm{B}}T}{2} \ln\left[\frac{\gamma_p}{\gamma_n}\right]. \tag{1.9e}$$

For optoelectronic applications (see Section 1.5), Fermi-Dirac statistics are automatically employed. In `WIAS-TeSCA` the command `FERMI` is used to describe the state equations (see Section 2.8).

1.1.3 Mobility models

Various models for the electron and hole mobilities μ_n and μ_p in (1.2) are implemented in `WIAS-TeSCA`. They describe the dependence of the mobilities on temperature, doping, and electric field. The models and the related parameters are set via the `MOBILITY` command, which is discussed in Section 2.11.

1.1.4 Generation and recombination

The generation-recombination term $G - R$ in the continuity equations for electrons and holes in (1.1b) and (1.1c) is additively split into various effects, viz.

$$G - R = G_{\mathrm{Ava}} - R_{\mathrm{rad}} - R_{\mathrm{Aug}} - R_{\mathrm{SRH}} - R_{\mathrm{surf}}\,\delta_{\Gamma_{\mathrm{Gate}}} \pm \ldots,$$

where $\delta_{\Gamma_{\mathrm{Gate}}}$ denotes the Dirac distribution concentrated on a Gate contact, where surface recombination takes place.

In `WIAS-TeSCA` the following recombination models are implemented

Radiative recombination	$R_{\mathrm{rad}} = a_{\mathrm{b}}(np - n_{\mathrm{i}}^2),$
Auger recombination	$R_{\mathrm{Aug}} = (a_n n + a_p p)(np - n_{\mathrm{i}}^2),$
Shockley–Read–Hall recombination	$R_{\mathrm{SRH}} = \dfrac{np - n_{\mathrm{i}}^2}{\tau_n(n + r_n) + \tau_p(p + r_p)},$
Surface recombination at gate contacts	$R_{\mathrm{Surf}} = \dfrac{np - n_{\mathrm{i}}^2}{\frac{n+r_n}{v_n} + \frac{p+r_p}{v_p}}.$

The Avalanche generation is given via

$$G_{\mathrm{Ava}} = a_1|\mathbf{J}_n| \exp\Big[-\frac{a_2}{\beta_n}\Big] + b_1|\mathbf{J}_p| \exp\Big[-\frac{b_2(\beta_p)}{\beta_p}\Big],$$

where $\beta_n = |\mathbf{E}\cdot\mathbf{J}_n|/|\mathbf{J}_n|$ and $\beta_p = |\mathbf{E}\cdot\mathbf{J}_p|/|\mathbf{J}_p|$, respectively, see [Sel84, p. 110].

The command `RECOMBINATION` (described in Section 2.18) allows the input of data for the generation-recombination model. In addition to the recombination terms above, trap dynamics can be included as well. They are described in detail in Section 1.3. Moreover, the `SPECIAL` allows to include a generation rate G_{ext} that describes the generation of electron-hole pairs e.g. due to Lambert–Beer absorption or by the trace of an incident particle in a sensor device.

1.1.5 Doping

In `WIAS-TeSCA`, several different analytically specified doping profiles are implemented, which enter the Poisson equation in (1.1a). They can be superposed with each other and with one-dimensional profiles. Other analytically given or in files defined two-dimensional doping profiles can be included as well. The doping profile can be selected by entering the command `DOPING`.

1.1.6 Including external magnetic fields

The effects of an external magnetic field perpendicular to the xy-plane can be taken into account in the simulation. We define

$$b_n = \mu_n|\mathbf{B}|, \quad \text{and} \quad b_p = \mu_p|\mathbf{B}|, \tag{1.10}$$

where

μ_n, μ_p	electron and hole Hall mobility, respectively,
$\mathbf{B}$	vectorial magnetic field.

In this case, the current densities in the continuity equations (1.1b) and (1.1c) are replaced by

$$\mathbf{J}_n(\mathbf{B}) = \frac{\mathbf{J}_n(0) + \mu_n \mathbf{B} \times \mathbf{J}_n(0)}{1 + b_n^2}, \tag{1.11a}$$

$$\mathbf{J}_p(\mathbf{B}) = \frac{\mathbf{J}_p(0) + \mu_p \mathbf{B} \times \mathbf{J}_p(0)}{1 + b_p^2}. \tag{1.11b}$$

Here $\mathbf{J}_n(0)$ and $\mathbf{J}_p(0)$ are the vectorial current densities without magnetic field given in (1.2). Note that $\mu_n \mathbf{B} = \pm b_n \mathbf{e}_z$ and $\mu_p \mathbf{B} = \pm b_p \mathbf{e}_z$, where $\mathbf{e}_z$ is perpendicular to the xy-plane.

The influence of the magnetic field is set by providing values for b_n (parameter `BMUEN`) and b_p (parameter `BMUEP`) in the `DEVICE` command.

1.1.7 Boundary conditions

The system of differential equations in (1.1) is complemented by boundary conditions, that model the interaction of the device with its vicinity. In `WIAS-TeSCA` the following types of boundary conditions are implemented.

Ohmic contacts

At ohmic contacts the program merely needs the applied potential φ_a as input (in the `STEP` command). The following Dirichlet boundary values at the contact are assumed:

$$\varphi = \varphi_\mathrm{a} + U_0, \quad \text{and} \quad n = n_0, \quad p = p_0. \tag{1.12}$$

Here $U_0 = U_T \ln(n_0/N_i^{\mathrm{eff}})$, and the boundary values n_0 and p_0 are determined as positive solutions of the equilibrium and charge neutrality condition, namely

$$n_0 p_0 = (N_i^{\mathrm{eff}})^2, \quad C_0 + p_0 - n_0 = 0 \tag{1.13}$$

with C_0 denoting the doping at the contact.

Bulk contact

`WIAS-TeSCA` offers the possibility to characterize an ohmic contact as bulk contact (by assigning the respective contact number to the integer parameter `IBULK` in the `DEVICE` command). At the bulk contact, n and p are treated like in the case of ohmic contacts. However, the ohmic boundary condition for the electrostatic potential φ is modified as follows:

$$\varphi = \varphi_\mathrm{a} + U_0 + R_{\mathrm{AB}} J_\mathrm{b}. \tag{1.14}$$

Here, U_0 is as above and R_{AB} is the bulk resistance (in Ω) that must be entered in the `STEP` command (parameter name `RAB`). Furthermore, J_b is the calculated current through the contact (for example caused by avalanche generation cf. Schütz-Selberherr-Pötzl [SSP82]).

Schottky contacts

For Schottky contacts, WIAS-TeSCA needs the applied potential φ_a as well as the Dirichlet value for the electron density n_0 (in the STEP command). Then, the following boundary conditions are assumed at the contact

$$\varphi = \varphi_a + U_T \ln(n_0/N_i^{\text{eff}}), \quad n = n_0, \quad p_0 = (N_i^{\text{eff}})^2/n_0 \tag{1.15}$$

Note: In the literature, it is usually set

$$n_0 = N_i^{\text{eff}} \exp\Big[\frac{E_g - 2\Phi_S}{2U_T}\Big]. \tag{1.16}$$

Here, $E_g = E_c - E_v$ is the band gap of the semiconductor material at the contact and Φ_S is the Schottky barrier.

Gate contacts

For gate contacts, the following boundary conditions hold:

$$\varepsilon_s \nabla\varphi \cdot \mathbf{n} + \frac{\varepsilon_{ox}}{d_{ox}}(\varphi - \varphi_a - \varphi_k) = Q_{SS}, \tag{1.17}$$

$$\mathbf{J}_n \cdot \mathbf{n} = \mathbf{J}_p \cdot \mathbf{n} = 0. \tag{1.18}$$

The used variables are:

$\mathbf{n}$	unit vector perpendicular to contact pointing outwards,
ε_{ox}	dielectric coefficient of oxide,
d_{ox}	thickness of the oxide,
φ_a	applied voltage at gate,
φ_k	contact voltage at gate,
Q_{SS}	density of states at boundary surfaces.

The quantities ε_{ox}, d_{ox} and φ_k are entered in the DEVICE command, φ_a is entered in the STEP command.

Inductivity, capacity, and resistance

For transient calculations, it is possible to attach an external circuit at each contact. This circuit contains an inductance, a resistance and a parallelly connected capacitance. For this purpose, the parameter ICLR must be set to 1 or 2 in the DEVICE command and for each of the parameters A_C (capacitance), A_L (inductance) and A_R (resistance) and each of the contacts a value has to be assigned. Then, at contacts with $\max(A_C, A_L, A_R) > 0$ the dynamic boundary condition

$$A_L \frac{\mathrm{d}^2}{\mathrm{d}t^2} J + A_R \frac{\mathrm{d}}{\mathrm{d}t} J + \frac{1}{A_C} J = \frac{\mathrm{d}}{\mathrm{d}t} U \tag{1.19}$$

is realized as (natural) boundary condition for the total current equation.

For ICLR $= 1$ we have

$U = U_i - U_a$	U_i inner and U_a outer electrostatic potential at the contact,
J	total current through the contact,

otherwise, for `ICLR` $= 2$ we have

$U = U_i(x) - U_a(x)$	$U_i(x)$ inner and $U_a(x)$ outer electrostatic potential in the boundary point x,
$J = J(x)$	current component pointing outwards in the boundary point x.

If an ohmic contact is marked (by assigning the corresponding contact number to the parameter `ISTROM` or `ICAP` in the `DEVICE` command), the equation

$$J - J_S = A_C \frac{dU}{dt} \tag{1.20}$$

will be realized as a (natural) boundary condition for the total current equation (instead of `AC` in the `DEVICE` command, the parameter `CAP` in the `STEP` command can be used).[2] The quantities are:

$U = U_i(x) - U_a(x)$	$U_i(x)$ inner and $U_a(x)$ outer electrostatic potential in the boundary point x,
$J = J(x)$	current component pointing outwards in the boundary point x.

The parameters for the current J_S (`STROM`) and if necessary for the capacity A_S (`CAP`) have to be entered in the `STEP` command for each working point, respectively.

The electron and hole densities n and p are fixed by the Dirichlet conditions in the Ohmic case in (1.12).

Note: This type of boundary condition can be used to calculate the potential at the current contact (as asymptotic value in the time domain), that corresponds to a given current J_S (current controlled simulations).

Isolating and symmetry conditions

Boundaries of the device that are not contacts are treated with ideal Neumann boundary conditions, namely,

$$\nabla\varphi \cdot \mathbf{n} = \mathbf{J}_n \cdot \mathbf{n} = \mathbf{J}_p \cdot \mathbf{n} = 0. \tag{1.21}$$

[2] In particular, with $A_C \to 0$ the boundary condition $J = J_S =$ `const` can be realized, see below.

1.1.8 Initial conditions

WIAS-TeSCA is able to execute alternately stationary and transient calculations. As initial value of a transient calculation always the solution of the last calculated stationary problem is chosen (if not an interrupted calculation is continued with saved values). If there is no preceding stationary calculation, the program automatically starts from the thermodynamic equilibrium.

For modulation experiments, in the transient case it is e.g. possible to apply a timely varying external voltage $U(t)$.

1.1.9 Oxide and passivation layers

As limiting cases, isolating and conducting areas are allowed. Oxide areas are characterized by an intrinsic charge carrier density of zero. Conduction areas can be modeled as oxide areas with a very large dielectric constant. In oxide areas only the linear Poisson equation is solved:

$$-\nabla\cdot(\varepsilon_0\varepsilon_{\rm ox}\nabla\varphi) = qC_{\rm ox}, \quad C_{\rm ox} = \gamma_{\rm ox}(N_{\rm D}-N_{\rm A}), \tag{1.22}$$

The factor $\gamma_{\rm ox}$ can be entered in the DEVICE command (parameter name FADOOX). If $\gamma_{\rm ox}$ is zero, the Laplace equation will be solved in the oxide area. At the boundary surface between the semiconductor and the oxide area the relation

$$\varepsilon_0(\varepsilon_{\rm r}\nabla\varphi - \varepsilon_{\rm ox}\nabla\varphi)\cdot\mathbf{n} = qQ_{\rm SS}, \quad \mathbf{J}_n\cdot\mathbf{n} = \mathbf{J}_p\cdot\mathbf{n} = 0. \tag{1.23}$$

is realized as natural transition condition. Here the quantities are :

$\mathbf{n}$	unit vector perpendicular to boundary surface and pointing into oxide,
$\varepsilon_{\rm ox}$	relative dielectric constant of oxide,
$Q_{\rm SS}$	density of states at boundary surface.

With the simulation of floating gate transistors in mind a model for the injection of hot charge carriers into the oxide area is implemented.

1.2 Thermodynamic model

In WIAS-TeSCA the heat equation

$$\frac{\partial}{\partial t}\left(\left[C\rho + \frac{3}{2}(n+p)k_{\rm B}\right]T\right) - \nabla\cdot\left[\kappa(T)\nabla T\right] = Q, \tag{1.24}$$

is implemented. It can be solved simultaneously with the continuity equation and the optical equation. In (1.24) C is the heat capacity, ρ is the material density, and κ is the thermal conductivity.

The source term Q accounts for the following components (energy dissipation):

$$Q = Q_{\rm joule} + Q_{\rm rec} + Q_{\rm rad}, \tag{1.25}$$

where

$$Q_{\text{joule}} = \frac{\mathbf{J}_n^2}{e\mu_n n} + \frac{\mathbf{J}_p^2}{e\mu_p p} \tag{1.26}$$

accounts for joule heat, and

$$Q_{\text{rec}} = qR_{\text{nr}}(F_n - F_p + T(P_n + P_p)) \tag{1.27}$$

for recombination heat [3] (R_{nr} is the non-radiative SRH and Auger recombination, P_n and P_p are thermoelectric powers).

The last heat source Q_{rad} in (1.24) is due to the absorption of spontaneous and stimulated emission of radiation:

$$Q_{\text{rad}} = Q_{\text{rad}}^{\text{spont}} + Q_{\text{rad}}^{\text{stim}}. \tag{1.28}$$

Below the lasing threshold the first term dominates, above it the second contribution becomes decisive. The spatial distribution of *spontaneously* emitted radiation is difficult to determine, moreover the absorption itself is a non-local process. Usually, it is assumed that the heating contribution through absorption of spontaneously emitted radiance is small. Thus, in `WIAS-TeSCA` it is neglected for modeling. Therefore, the total power equation (UI= coupled out power + dissipated power) is not fulfilled anymore. However, above the lasing threshold this should cause only a small error as the second term clearly dominates. [4]

The heat source which draws its energy from absorption of coherent radiation is governed by the equation

$$Q_{rad}^{\text{stim}} = \frac{\omega\varepsilon_0}{2}\Im m[\varepsilon_{\text{intra}}(n,p)]|\mathbf{E}(\mathbf{r})|^2. \tag{1.29}$$

Here the imaginary part of the intra band part of the relative dielectric function $\varepsilon_{\text{intra}}$ accounts for all absorption processes, that change the number of charge carriers within the valence band and the conduction band, respectively. This is possible through *free charge carrier absorption* and *inter valence band absorption*. The distribution of the electric field $\mathbf{E}$ corresponds with the laser mode, plus other possibly considered modes.

Therefore, in (1.24) the heat source is estimated to be the following:

$$Q_{\text{rad}}^{\text{stim}} = (\alpha_{\text{fc}} + \alpha_{\text{bg}}) \cdot (P_1|\Phi_1|^2 + P_2|\Phi_2|^2) \tag{1.30}$$

where α_{fc} is the free charge carrier absorption

$$\alpha_{\text{fc}} = f_{\text{cn}}n + f_{\text{cp}}p \tag{1.31}$$

and α_{bg} describes the background absorption.

[3] The expression for recombination heat is only exact in the stationary case.
[4] The origin of Q_{rad}^{spont} is the term R_{rad} in the total charge carrier equation.

1.3 Trap model and incomplete ionization

The Shockley-Read-Hall recombination was extended to incorporate deep traps in volume and on interfaces. These trap levels can take different states (neutral, negatively charged, positively charged) that are governed by additional equations. Therefore the basic equations in (1.1) were extended as follows. The partly ionized traps enter the Poisson equation (1.1a)

$$-\nabla\cdot(\varepsilon_0\varepsilon_\mathrm{r}\nabla\varphi) = q(C_\mathrm{net}+p-n)+\sum_{k=1}^{K} q_k N_k f_k \tag{1.32a}$$

Here, $q_k = \pm q$, $N_k > 0$, and $0 \le f_k \le 1$ denote the charge number (donor type $q_k = +q$ or acceptor type $q_k = -q$), the trap density, and defect occupancy, respectively. In particular, $f_k = 1$ means that the trap is completely ionized.

The continuity equations for electrons and holes are augmented by the trap recombination rates

$$\frac{\partial}{\partial t}n - \frac{1}{q}\nabla\cdot\mathbf{J}_n = G - R - \sum_{k=1}^{K} R_{n,k}^\mathrm{trap}, \tag{1.32b}$$

$$\frac{\partial}{\partial t}p + \frac{1}{q}\nabla\cdot\mathbf{J}_p = G - R - \sum_{k=1}^{K} R_{p,k}^\mathrm{trap}. \tag{1.32c}$$

We drop the index k from now on, then, the trap recombination rates are given via

$$R_n^\mathrm{trap} = \begin{cases} N\big(s_n n(1-f) - e_n f\big) & \text{acceptor-type traps,} \\ N\big(s_n n f - e_n(1-f)\big) & \text{donor-type traps,} \end{cases} \tag{1.33a}$$

$$R_p^\mathrm{trap} = \begin{cases} N\big(s_p p f - e_p(1-f)\big) & \text{acceptor-type traps,} \\ N\big(s_p p(1-f) - e_p f\big) & \text{donor-type traps,} \end{cases} \tag{1.33b}$$

where s_n and s_p are the capture coefficients and e_n and e_p the emission coefficients. The latter are given via

$$\frac{e_n}{s_n} = n_\mathrm{i}\exp\left[\frac{E_\mathrm{r}}{k_\mathrm{B}T}\right] = N_\mathrm{c}\exp\left[\frac{E_\mathrm{trap}-E_\mathrm{c}}{k_\mathrm{B}T}\right], \tag{1.33c}$$

$$\frac{e_p}{s_p} = n_\mathrm{i}\exp\left[-\frac{E_\mathrm{r}}{k_\mathrm{B}T}\right] = N_\mathrm{v}\exp\left[\frac{E_\mathrm{v}-E_\mathrm{trap}}{k_\mathrm{B}T}\right] \tag{1.33d}$$

with trap energy level $E_\mathrm{trap} = E_\mathrm{i} + E_\mathrm{r}$ (E_i is the intrinsic Fermi level, cf. (1.8), and E_r the trap level relative to E_i).

In addition to (1.32), evolution equations for the trap occupancy functions f have to be solved, namely,

$$N\frac{\mathrm{d}}{\mathrm{d}t}f = \begin{cases} R_n^{\mathrm{trap}} - R_p^{\mathrm{trap}} & \text{acceptor-type traps,} \\ R_p^{\mathrm{trap}} - R_n^{\mathrm{trap}} & \text{donor-type traps.} \end{cases} \tag{1.34}$$

In particular, in the stationary case the left-hand side in (1.34) is equal to zero. Thus, an algebraic equation for f_{stat} is obtained and we find

$$f_{\mathrm{stat}} = \begin{cases} \dfrac{e_p + s_n n}{s_n n + e_n + s_p p + e_p} & \text{for acceptor-type traps,} \\ \dfrac{e_n + s_p p}{s_n n + e_n + s_p p + e_p} & \text{for donor-type traps.} \end{cases}$$

Using this expression in (1.33) leads to

$$R_n^{\mathrm{trap}} = R_p^{\mathrm{trap}} = \frac{np - n_{\mathrm{i}}^2}{\tau_n(n + r_n) + \tau_p(p + r_p)},$$

where $\tau_{n/p} = \frac{1}{s_{p/n}N}$ and $r_{n,p} = \frac{e_{n/p}}{s_{n/p}}$. In particular, this corresponds to the classical Shockley–Read–Hall recombination (cf. Subsection 1.1.4). Note, however, that the trapped charges contribute to the space charge via the Poisson equation (1.32a).

The parameters for the trap model are set in the `RECOMBINATION` command.

1.4 Small signal analysis

After the calculation of stationary solutions of (1.1), small signal analysis can be carried out. Here, the consequences of disturbances of the form

$$a\exp(\mathtt{i}\omega t) \quad \text{with } a \text{ small,} \tag{1.35}$$

of the contact potential φ_{a}, see (1.14), are analyzed and used for the calculation of conductance and capacitance matrices A and B resp. according to the following formulas (cf. [Lau85]):

$$A_{kj} = \mathsf{Re}\Big[\frac{\mathrm{d}J_j}{\mathrm{d}U_{\mathrm{a},k}}\Big], \quad \text{and} \quad B_{kj} = \mathsf{Im}\Big[\frac{\mathrm{d}J_j}{\mathrm{d}U_{\mathrm{a},k}}\Big], \tag{1.36}$$

where

Re, Im	real and imaginary part,
$\mathtt{i}$	imaginary unit,
ω	excitation frequency,
t	time variable,
$\mathrm{d}J_j$	the change of current through contact j,
$\mathrm{d}U_{\mathrm{a}k}$	the change of potential at contact k.

The system of equations that arises from small signal analysis couples real and imaginary parts of φ, n and p. It is solved by block iteration that combines a linearized Gummel method with a SOR method [GG92]. The latter decouples real and imaginary parts and was introduced by Laux [Lau85]. Small signal analysis is activated by entering the excitation frequency ω in the `STEP` command.

1.5 Optoelectronic model

The mathematical modeling of optoelectronic devices has some essentially new additional possibilities compared to the pure electronic simulation.

- By default, in the case of optoelectronic applications in `WIAS-TeSCA` the *Fermi-Dirac statistics* is active, in order to describe heterostuctures with degenerate semiconductor components.
- The van Roosbroeck system is extended by a (scalar) waveguide equations for TE- or TM- modes. in the following section 1.5.1 are treated.
- Different models for the local optical gain g_i in the active region are implemented (details in section 2.9.1).
- To the continuity equations (1.1b), (1.1c) a term is added, modeling the stimulated recombination R^{stim}. It has the form

$$R^{stim} = g_{net}(\hbar\omega) * P|\Phi|^2/(\hbar\omega). \tag{1.37}$$

  $\hbar\omega$ is the photon energy, P the total power and $|\Phi|^2$, the transversal intensity distribution. The net gain g_{net} is the local gain $G(\hbar\omega)$ minus the local losses α, as later described in section 1.5.1.
- In `WIAS-TeSCA` two variants are implemented, considering the optical power P
  1. self-consistently (see section 1.5.2) assuming spatial homogeneity in the remaining spatial direction, or
  2. parametrically, see section 1.5.3.

1.5.1 Helmholtz equation

A quasi-planar layered waveguide structure is supposed. Most epitaxial laser structures belong to this type, including ridge-waveguide lasers. The growth direction is y, the layer plane is x.

The optical field $\mathbf{E}(\mathbf{r}, t)$ is prescribed in the following way:

$$\mathbf{E}(\mathbf{r}, t) = \sum_i a_i \sqrt{P_i}\mathbf{e}_i\Phi_i(x, y) \cdot \Re e\left[e^{i\omega t}\left(e^{i\beta_i z} + e^{-i\beta_i z}\right)\right] \tag{1.38}$$

where i is a mode index. The values β_i and $\Phi_i(x,y)$ are the eigenvalues and eigenfunctions of the corresponding Helmholtz equation, respectively. The norm of $\Phi_i(x,y)$ is normalized to unity. The constant a_i is choosen such that P_i is the corresponding modal power. The central frequency

$$\omega = 2\pi c/\lambda \tag{1.39}$$

can be specified by the user via the wavelength λ.

For TE-modes ($\mathbf{e}_i \| \mathbf{x}$), the following Helmholtz equation is solved:

$$\left[\Delta_{x,y} + \frac{\omega^2}{c^2}\left(\bar{n}^2 - \bar{n}_{\mathrm{TE}}^2\right)\right]\Phi_{\mathrm{TE}}(x,y) = 0 \qquad \text{where} \quad \bar{n} = \bar{n}(x,y) \tag{1.40}$$

is the local refractive index varying in the transvers (x,y) plane. Φ_{TE} and its derivatives are continuous at material boundaries. Available temperature dependent models for the local refractive index $\bar{n}$ are described under the command `FERMI` in section 2.9.2.

For TM modes ($\mathbf{e}_i \| \mathbf{y}$) the Helmholtz equation for the "generating" magnetic field component H_x^{TM} is solved.

$$\left[\bar{n}^2 \nabla_{x,y} \frac{1}{\bar{n}^2} \nabla_{x,y} + \frac{\omega^2}{c^2}\left(\bar{n}^2 - \bar{n}_{\mathrm{TM}}^2\right)\right] H_x^{\mathrm{TM}}(x,y) = 0. \tag{1.41}$$

$H_x^{\mathrm{TM}}(x,y)$ and $\bar{n}^{-2}(\vec{e}_n \nabla_{x,y}) H_x^{\mathrm{TM}}(x,y)$ are continuous at material boundaries with normal unit vector $\vec{e}_n$. The corresponding dominant electric field component ($\perp$ to the layer level) is calculated according to the rule

$$\Phi_{\mathrm{TM}}(x,y) = \frac{-\beta_{\mathrm{TM}}}{\omega \varepsilon_0 \bar{n}^2} H_x^{\mathrm{TM}}(x,y). \tag{1.42}$$

`WIAS-TeSCA` currently allows the self-consistent consideration of up to 2 modes. The corresponding behavior can be select by the switch `ISpec`. An extension to more modes is under preparation.

1.5.2 Self-Consistent photon balance

The longitudinally averaged internal optical power P_i of transverse mode i is determined from the rate equation

$$\frac{d}{dt} P_i = v_{gi}(G_i - \alpha_i - \gamma_i) P_i + \dot{P}_i^{spont}. \tag{1.43}$$

The modal group velocity $v_{gi} = c/n_{gi}$ can be specified by the user via the modal group index n_{gi}. The net gain of the mode traveling along the cavity is calculated as

$$G_i = \int (g - \alpha_b)\ |\Phi_i|^2 \ \mathrm{d}x\mathrm{d}y. \tag{1.44}$$

Models for the optical background losses α_b are described in Section 2.8 on page 81. α_i are possible additional losses (for example by scattering into radiation modes), that do not contribute to heating. In addition, outcoupling losses of a Fabry-Perot (FP) laser with facet reflectivities $R_i(0)$ on the left facet ($z = 0$) and $R_i(L)$ on the right facet ($z = L$) are included,

$$\gamma_i = -\frac{1}{L} \log \left(R_i(0) R_i(L) \right). \tag{1.45}$$

DFB lasers can be treated as FP lasers with appropriate reflectivities. The rate of spontaneous emission into transverse mode i is modeled as

$$\dot{P}_i^{spont} = K_i \ \hbar\omega_i \ \frac{v_{gi}}{L} \int \left[1 - \exp\left(\frac{\hbar\omega - eU_F}{kT} \right) \right]^{-1} v_{gi} \ g(\hbar\omega) \ |\Phi_i|^2 \ \mathrm{d}x \ \mathrm{d}y. \tag{1.46}$$

K_i is a correcture (e.g. Petermann factor), that can be specified by the user (parameter `PEFA`). L is the laser length.

The following calculated powers are written to the terminal in mW: total internal power $P_1 + P_2$ (named Power), internal power ratio $\eta = \frac{P_1 - P_2}{P_1 + P_2}$ (named eta), output power Pout(i,0) of each mode i at facet $z = 0$, modal output powers Pout(i,L) at facet $z = L$, where

$$\text{Pout}(i,0) = \frac{\rho}{1+\xi} P_i \quad \text{and} \quad \text{Pout}(i,\text{L}) = \frac{\xi\rho}{1+\xi} P_i. \quad \begin{cases} \rho = -\ln(\sqrt{R_i(0) R_i(L)}) \\ \xi = \frac{1-R_i(L)}{1-R_i(0)} \sqrt{\frac{R_i(0)}{R_l}} \end{cases} \tag{1.47}$$

1.5.3 Treating Powers as Parameters (TPP)

Another approach enabled in `WIAS-TeSCA` provides data for a quasi-3D treatment of edge-emitting semiconductor lasers in the stationary case (see e.g. [WBW93]) Suppose the currents flow only transversely, the status of a transverse cross section does not explicitely depend on the longitudinal position z, but only implicitely via the powers $P_i(z)$. The letter ones can be used as transverse-longitudinal separation parameters. To this purpose, `WIAS-TeSCA` solves the transverse transport and wave equations for an externally given series of powers and stores quantities like modal gain G_i in tabular form. Postprocessing programs for the longitudinal propagation (not incorporated in `WIAS-TeSCA`) can determine e.g. the local modal gain without rerunning `WIAS-TeSCA` by interpolation in these tables. More details and the input parameters will be described further below on page 82.

2 Simulations with TeSCA

Simulations with WIAS-TeSCA are either controlled by a script file (DIO file) or/and directly by the user on a command line.

A simulation consists of several commands that define e.g. the material properties, control the graphic output, or start the computation. All WIAS-TeSCA commands are listed in Table 2.1.

Command	Short description
break	Sets break point in script execution to enter interactive mode
device	Fundamental values for the semiconductor device, such as temperature, scaling and symmetry factors, relative dielectric permittivity, etc., can be entered
energy	Sets parameters for energy transport model described in Section 1.2
fermi	Sets parameters for carrier statistics, e.g. conduction and valence band edges for Boltzmann or Fermi case, and optical parameters for the simulation of optoelectronic devices
graphic	Specifies and controls graphical output
grid	Defines computational domain, i.e. triangulation and boundary conditions
load	Reads WIAS-TeSCA save files from previous simulations and loads analytical profiles or interpolates profiles from external meshes
mobility	Sets parameters for intrinsic density and carrier mobility models
numeric	Used to define the accuracy and termination parameters required for the numerical calculations
recombination	Defines the parameters for the generation-recombination processes
save	Used to write output files for subsequent evaluation or continuation of the simulation or offline coupling to other simulation tools
special	Describes physical effects not covered by the standard model
step	Defines external bias, step control parameters, and time intervals for transient simulations. Solution of drift-diffusion system is calculated, printed (to the terminal) and saved
substrate	Used to initialize the layer system
title	Sets title of simulation and allocates memory for computation
use	Switch from DIOS process simulation to WIAS-TeSCA device simulation.

Table 2.1: Main WIAS-TeSCA commands in alphabetical order

In the subsequent sections all commands are thoroughly discussed. Parameters without comments are special ones, useable by the developers, only.

2.1 DIO script files

Comments in DIO files have to be preceded by an exclamation mark

```
! This is a comment
```

It is possible to split the DIO file into many separate files, which are then included into the main DIO file using the command, viz.

```
! main.dio
@file1.dio !include file1.dio
@file2.dio !include file2.dio
```

2.2 Device command

With the use of the `DEvice`-command fundamental values for the semiconductor device can be entered. Moreover, using the `DEvice`-command, an internal scaling of all physical parameters is done.

2.2.1 Cylindrical symmetry

`WIAS-TeSCA` is able to treat problems that have cylindrical symmetry. If the parameter `IZYLIN1` is set, the y-axis will be interpreted as cylindrical axis and the x-coordinate will be utilized as radius. In this case, the parameter `ZAUS` which usually specifies the extension of the device in the third dimension has no impact.

2.2.2 Comments

During the execution of the `DEvice`-command, the validity of all parameters is tested.

These physical parameters are: The temperature `TEMP`, the intrinsic density `ENI`, the dielectric constant for the substrate `EPSSI`, the extension of the device in z-direction `ZAUS` and the explicit specification of the time scaling `TSkal`. All given parameters have default values.

For the treatment of heterostructures (devices with different materials), the device can be partitioned in up to 55 zones (sub-regions) using the `DOMAIN`-command. This is a possibility to consider spatial varying values of intrinsic density, dielectric constant and basic mobilities. Moreover, the Fermi level – constant in every zone – can be given. For this purpose values for `ENIFA`, `EPSSI`, `AMUNFA`, `AMUPFA`, `PHIN` and `PHIP` have to be entered. In this case the calculation will be performed, with the parameters `ENIFA(i)*ENI`, `EPSSI(i)`, `AMUNFA(i)*AMUNO`, `AMUPFA(i)*AMUPO`, `PHIN(i)` and `PHIP(i)` in the zone with index i.

If `ENIFA(i)=0`, the i-th zone is understood as oxid. In such zones, the Laplace equation (Poisson equation with vanishing right hand side) is solved. However,

if FADOOX=1, in oxid-zones the doping is considered, whereas the mobile charge carriers are set to zero.

A zonally constant interface charge QSSIOX(i) between the semiconducter and oxid layers can be taken into account.

More general, it is possible to consider surface charges along interfaces.

2.2.3 Scaling

Units of Measurement and Scaling:

All physical values, entered as a parameter and can be changed, subject to an internal scaling in the following way:

```
entered value = internal value * scale factor
internal value = entered value / scale factor
```

The scaling factors are calulated internally during the execution of the DEvice-command and depend on temperature, intrinsic density and dielectric constant. The user has to note the unities of the physical parameters used in WIAS-TeSCA. This is the CGS system. By way of derogation from the CGS system in the mobility model, incoming energies are specified in eV (electron volts).

The following constants are used:

vacuum permittivity	$\varepsilon_0 = 8.85419 \cdot 10^{-14} \frac{As}{Vcm}$
Boltzmann constant	$k_{\mathrm{B}} = 1.380662 \cdot 10^{-23} \frac{VAs}{K}$
elementary charge	$q = 1.6021 \cdot 10^{-19} As$

A display of the values of the scaling factors is possible by using the commands PRINT.

2.2.4 Parameters

name	unit *type* option	default	comment
AC()	*real* $[s/\Omega]$	0.d0	capacity on contact, Length≤mdiri
AL()	*real* $[\Omega \cdot s]$	0.d0	inductivity on contact, Length≤mdiri
AR()	*real* $[\Omega]$	0.d0	resistance on contact, Length≤mdiri

continued on next page

name	unit *type* option	default	comment
BMUEN	*real*	0.d0	product of magnitude of the magnetic field strength and electron Hall velocity
BMUEP	*real*	0.d0	product of magnitude of the magnetic field strength and hole Hall velocity
a11	*real*	0.d0	matrix component for anisotropy
a12	*real*	0.d0	matrix component for anisotropy
a21	*real*	0.d0	matrix component for anisotropy
a22	*real*	0.d0	matrix component for anisotropy
BVN()	*real* [cm^{-3}]	-one	electronic density at the contact (Dirichlet boundary condition) Length≤mdiri
BVP()	*real* [cm^{-3}]	-one	hole density at the contact (Dirichlet boundary condition) Length≤mdiri
CURMAX	*real*	1d7	Max. contact current (if the current exceeds CURMAX, then the iv-characteristic calculation is canceled.)
DIbez	AN	SOURCE DRAIN BULK	Names of contacts with Dirichlet boundary conditions (ohmic contacts and Schottky contacts)
Dox()	*real* [cm]	35.d-7	oxide thickness at the gate contacts, or the oxid domain (cf. DOMAIN command, type-4) Length≤mnatur
ELEM()	*integer*	1	Length≤ mreg

continued on next page

name	unit *type* option	default	comment
ENi	*real* [cm^{-3}]	rundef	intrinsic density (constant part)
ENIFA()	*real*	1.d0	factor for the space-dependent intrinsic density Length≤ mreg
EPSOx()	*real*	3.8d0	relative permittivity constant of the oxide at the gate contacts Length≤mnatur
EPSSi()	*real*	11.67d0	relative permittivity constant of the substrate (Si) Length≤ mreg
Execute	*real*	0.d0	toggle on/off the execution and, therefore, the tests of the values and the automatic calculation of the scaling.
FADOOX		0.	cancellation factor for the oxide doping
GAbez	AN	GATE	names of the gate contacts (natural boundary conditions)
Ger()	*real* [V]	0.d0	Length≤10
GNR()	*real* [cm^{-2}]	0.d0	Length≤10
GSN()	*real*	0.d0	Length≤10
GSP()	*real*	0.d0	Length≤10
GEN()	*real*	1.d0	Length≤10
GEP()	*real*	1.d0	Length≤10
IAZPQ	*integer*	0	number of zone pairs for interface charge
IBulk	*integer*	0	number of the contact with resistance
ICap	*integer*	0	number of the contact with capacity
ICLR	*integer*	0	outer current control

continued on next page

name	unit *type* option	default	comment
IDiri	*integer*	0	number of Dirichlet-contacts
IGRenz	*integer*	0	traps on boundary
INatur	*integer*	0	number of Gate-contacts
IPERio	*integer*	0	periodic boundary conditions
ISPAN()	*integer*	0	Length≤5
ISTROM()	*integer*	0	Length≤5
IZPQ()	*integer*	0	indices of zone pairs for interface charge, Length≤2*IAZPQ
IZYlin	*integer*	0	switch for cylindrical symmetry
KAC()	*integer*	1	Length≤15
KADI	*integer*	0	number of Dirichlet boundary parts
KANA	*integer*	0	number of Gate boundary parts
MDiri	*integer*	20	max. number of Dirichlet boundary parts
MNatur	*integer*	10	max. number of Gate boundary parts
Odi()	*real* [cm]	35.d-7	Length≤mnatur
PHIN()	*real* [V]	0.d0	quasi-Fermi potential for electron, Length≤ mreg
PHIP()	*real* [V]	0.d0	quasi-Fermi potential for holes, Length≤ mreg
Qss()	*real* [cm^{-2}]	0.d0	surface state density at the gate contacts, Length≤mnatur
QSSIox()	*real* [cm^{-2}]	0.d0	surface state density at the gate contacts, Length≤ mreg
QZP()	*real* [cm^{-2}]	0.d0	interface charge, Length≤IAZPQ

continued on next page

name	unit *type* option	default	comment
RAB	*real* [Ω]	0.d0	resistance at the contact IBULK
SPULE	*real* [Ω]	0.d0	inductivity on contact IBULK
TEmp	*real* [K]	300	temperature
TSkal	*real* [s]	rundef	timescale
Ukonga()	*real* [V]	0.55d0	contact voltages at the gate contacts, Length≤mnatur
Xskal	*real* [cm]	rundef	internal length scale
Zaus	*real* [cm]	1d-4	length of the device in the z-direction
ZAUSFA()	*real*	1.d0	Length≤ mreg
SYMfak	*real*	1.d0	symmetry factor

2.3 Break command

The break command is used to define a break point in the processing of a DIO script file. At this point the execution of the script is stopped and the user can enter commands in the interactive mode e.g. to change the graphical output. The execution of the script can be resumed by typing GO in the interactive mode.

2.4 Title command

The command TITle has to precede each WIAS-TeSCA simulation. In the command the problem size can be specified. Memory is allocated and deallocated during the simulation using the specified maximum values.

By default no memory limitations are defined, i.e. the program automatically reallocates memory if this is required by refinement of the mesh of the layer system or during the delaunization of the mesh. It might be useful to restrict the number of nodes in the grid adaptation or to allocate a fixed small amount of memory on machines with small main memory resources or to initially allocate a large amount of memory if the problem is known to be huge.

2.4.1 Parameters

parameter name	unit *type* options	default value	comment
Title	*string*80*	blank	title line for the simulation.
MAXV	*integer*	undefined	Maximum node number of vertices in the triangle tree. Refinement is stopped if the new mesh would have more mesh points. MAXV can be specified to limit the memory growth on small machines.
MAXVDelaunay	*integer*	undefined	Maximum total number of nodes. Specifying this value might turn o the delaunization of the mesh This is not recommended. It is recommended to specify MAXV only.
MPOINTS	*integer*	10000	Initially allocated number of points in the layer system. This number is increased internally if required.
MXT	*integer*	2000	Initially allocated number of triangles in the user triangulation. Increased internally if necessary.
MAXT	*integer*	0	Maximum number of triangles in the triangle tree. Refinement is stopped if the new tree would exceed MAXT. Internal default 4/3MAXV.
MAXUTR	*integer*	0	Initial maximum number of triangles and boundaries in the final UTRI-grid.
MAXL	*integer*	0	Internal Switch. The max. number of multigrid levels in the ITRI grid.

continued on next page

name	unit *type* option	default	comment
INFO	*integer*	0	controls the default printed output on the terminal (0, 1, 2, ...).
PRInt	*integer*	0	controls the default printed output into the protocol file (0, 1, 2, ...).

2.5 Energy command

In WIAS-TeSCA the heat equation is implemented. It can be solved simultaneously with the continuity equations and the optical equation. In (1.24) C is the heat capacity and κ is the thermal conductivity.

As usual, temperature-depending parameters are normalized with respect to the room temperature $T_0 = 300K$. In some formulas, the reduced temperature $t = T/T_0$ is used.

The heat capacity C=HEATFA (default=1, zonewise) can be specified by the user.

The thermal conductivity depends on temperature according to the formula

$$\kappa(T) = \kappa \cdot T^{\gamma} \tag{2.1}$$

The avalanche generation is temperature dependent according to the formulas

$$R_{ava} = -\alpha_n \frac{j_n}{q} - \alpha_p \frac{j_p}{q} \tag{2.2}$$

$$\alpha_{n,p} = \alpha^0_{n,p} \exp\left(-\frac{\beta_{n,p}}{E}\right) \tag{2.3}$$

$$\alpha^0_{n,p} = \alpha^0_{n,p}(1 + \alpha^1_{n,p}(t-1)(t+1)) \tag{2.4}$$

$$\beta_{n,p} = \beta^0_{n,p}(1 + \beta^1_{n,p}(t-1)(t+1)) \tag{2.5}$$

The parameters α and β depend on energy for the holes and differ for $E < E_0$ and $E > E_0$ for a user given E_0.

variable	TeSCA-name	Proc	default	Stift
α_n^0	AVA1	RECOMB	1.00d6	7.00d5
β_n^0	AVA2	RECOMB	1.66d6	1.23d6
$\alpha_p^0,\ E < E_0$	AVA3	RECOMB	1.582d6	1.58d6
$\beta_p^0,\ E < E_0$	AVA4	RECOMB	2.036d6	2.04d6
$\alpha_p^0,\ E > E_0$	AVA5	RECOMB	6.71d5	1.58d6
$\beta_p^0,\ E > E_0$	AVA6	RECOMB	1.693d6	2.04d6
E_0	AVA7	RECOMB	4.d5	4.d5
α_n^1	AVAT1	ENERGY	0.0d0	0.43d0
β_n^1	AVAT2	ENERGY	0.0d0	0.375d0
$\alpha_p^1,\ E < E_0$	AVAT3	ENERGY	0.0d0	0.42d0
$\beta_p^1,\ E < E_0$	AVAT4	ENERGY	0.0d0	0.33d0
$\alpha_p^1,\ E > E_0$	AVAT5	ENERGY	0.0d0	0.42d0
$\beta_p^1,\ E > E_0$	AVAT6	ENERGY	0.0d0	0.33d0

Stift are the values used in the thesis of Martin Stiftinger. They differ even for $T = T_0$ from the `WIAS-TeSCA` defaults.

The Auger recombination is temperature dependent according to the formulas

$$R_{aug} = (C_n^{aug} n + C_p^{aug} p)(np - n_i^2) \tag{2.6}$$

$$C_n^{aug} = C_n^0 (T/E_n)^{\gamma_n} \left(e^{E_n/T_0 - E_n/T} \right) \tag{2.7}$$

$$C_p^{aug} = C_p^0 (T/E_p)^{\gamma_p} \left(e^{E_p/T_0 - E_p/T} \right) \tag{2.8}$$

variable	TeSCA-name	Proc	default
C_n^0	AUGN	RECOMB	2.8d-31
γ_n	AUGNGAM	ENERGY	0.0d0
E_n	EAN	ENERGY	0.1d0
C_p^0	AUGP	RECOMB	9.9d-32
γ_p	AUGPGAM	ENERGY	0.0d0
E_p	EAP	ENERGY	0.1d0

The radiant or optical recombination is temperature dependent according to the formulas

$$R_{opt} = C^0 T^{\gamma} \tag{2.9}$$

variable	TeSCA-name	Proc	default
C^0	AUGB	RECOMB	0.d0
γ	AUGBGAM	ENERGY	1.5d0

The Shockley - Read - Hall - recombination is temperature dependent according to the formulas

$$R_{srh} = \frac{np - n_i^2}{\tau_p(n+n_1) + \tau_n(p+p_1)} \tag{2.10}$$

$$\frac{1}{\tau_n} = \frac{1}{t^{\gamma_n}}\left(\frac{1}{\tau_n^0\tau_n^1} + C_n^{srh}D\right) + \frac{1}{t^{\delta_n}}C_n p^2 \tag{2.11}$$

$$\frac{1}{\tau_p} = \frac{1}{t^{\gamma_p}}\left(\frac{1}{\tau_p^0\tau_p^1} + C_p^{srh}D\right) + \frac{1}{t^{\delta_p}}C_p n^2 \tag{2.12}$$

variable	TeSCA-name	Proc	default	remark
n_1	REN	RECOMB	1.09d10	
τ_n^0	TAUN0	RECOMB	2d-4	
τ_n^1	TAUNFA	RECOMB	1.d0	zone-depend.
C_n^{srh}	CSRHN	RECOMB	0.d0	
C_n	CAUGN	RECOMB	0.d0	
γ_n	GAMMAN	RECOMB	0.d0	
δ_n	DELTAN	RECOMB	0.d0	
p_1	REP	RECOMB	1.09d10	
τ_p^0	TAUP0	RECOMB	2d-6	
τ_p^1	TAUPFA	RECOMB	1.d0	zone-depend.
C_p^{srh}	CSRHP	RECOMB	0.d0	
C_p	CAUGP	RECOMB	0.d0	
γ_p	GAMMAP	RECOMB	0.d0	
δ_p	DELTAP	RECOMB	0.d0	

The refractive index is temperature dependent according to the formulas

$$n(T) = \Big(n(T_0) - n_d(n+p)/2\Big) + n_T(T-T_0)\ , \quad typ = 0 \tag{2.13}$$

$$n(T) = \Big(n(T_0) - n_d(n+p-|D_{net}|)/2\Big) + n_T(T-T_0)\ , \quad typ = 1 \tag{2.14}$$

$$n(T) = n(T_0) + n_T(T-T_0)\ , \quad typ = ... \tag{2.15}$$

variable	TeSCA-name	Proc	default	remark
$n(T_0)$	BRE	FERMI	1.d0	
n_d	BREFAK	FERMI	1d-19	zone-depend.
n_T	BREA	ENERGY	6.8d-4	
typ	BRETYP	FERMI	0	model type

The transverse optical losses α_t in the bulk laser are composed of the free carriers absorption (f_{cn} and f_{cp}) and the inter valence band absorption α. We have

$$\alpha_t = \alpha(T) + f_{cn}(T)n + f_{cp}(T)p \tag{2.16}$$

$$\alpha(T) = \alpha_0\alpha_1\left(e^{E_0/T_0 - E_0/T}\right) \tag{2.17}$$

$$f_{cn}(T) = f_{cn0}T^{\gamma_n} \tag{2.18}$$

$$f_{cp}(T) = f_{cp0}T^{\gamma_p} \tag{2.19}$$

variable	TeSCA-name	Proc	default	remark
α_0	AALPHA	FERMI	1d4	
α_1	AALPHF	FERMI	1d0	zone-depend.
E_0	EA	FERMI	0.1d0	
f_{cn0}	FCNALF	FERMI	0d0	
γ_n	GN	MOBILITY	2.33d0	
f_{cp0}	FCPALF	FERMI	0d0	
γ_p	GP	MOBILITY	2.33d0	

2.5.1 Parameters

`name`	`unit`
	type
	`option`
		`default`	comment

`IEnergy`	*integer*	`-1`	
`BOUnd`	*real*	`5.d0`	
`CONDuc()`	*real* [cm^6/s]	`0.d0`	Length≤20
`EPS1`	*real*	`1d-4`	
`EPS`	*real*	`1d-5`	
`EPTemp`	*real*	`1d-2`	
`EREL`	*real* [cm]	`9d-7`	
`ERLO`	*real* [cm]	`9d-7`	
`HEATFA()`	*real*	`1.d0`	heat capacity, Length≤ `mreg`
`HETERO`	*integer*	`0`	
`IHO`	*integer*	`3`	
`ITAU`	*integer*	`2`	
`ITEMO`	*integer*	`0`	=5 temperatur dependence

continued on next page

name	unit *type* option	default	comment
ITEMP	*integer*	0	
ITIN	*integer*	3	
Joule	*integer*	0	
KATE	*integer*	0	
MODel	*integer*	1	
MTemp	*integer*	20	
RANDDI()	*real* [cm]	1d-4	Length≤20
SIK0	*real*	0.03d0	
SIK1	*real*	0.00156d0	
SIK2	*real*	1.65d-6	
TKN	*real*	2.5d0	
TKP	*real*	2.5d0	
SIW	*real*	1.636584d0	
TEMPScal	*real*	300.d0	
EGA	*real*	2.73d-4	
EA	*real*	0.1d0	
BREA	*real*	6.8d-4	n_T in formulas (2.13–2.15)
AUGBGam	*real*	1.5d0	γ in formula (2.9)
AUGNGam	*real*	0.0d0	γ_n in formula (2.7)
AUGPGam	*real*	0.0d0	γ_p in formula (2.8)
EAN	*real*	0.1d0	E_n in formula (2.7)
EAP	*real*	0.1d0	E_p in formula (2.8)
AVAT1	*real*	0.d0	See formulas (2.2–2.5)
AVAT2	*real*	0.d0	See formulas (2.2–2.5)
AVAT3	*real*	0.d0	See formulas (2.2–2.5)
AVAT4	*real*	0.d0	See formulas (2.2–2.5)
AVAT5	*real*	0.d0	See formulas (2.2–2.5)
AVAT6	*real*	0.d0	See formulas (2.2–2.5)

continued on next page

name	unit *type* option	default	comment
TEBIas()	*real*	300.d0	Length≤100
TLOwboun	*real*	0.25d0	
TUPbound	*real*	5.d0	
WLEIFA()	*real* [cm^6/s]	1.d0	κ in formula (2.1) Length≤ mreg
WLEX	*real*	-1.33d0	γ in formula (2.1)
ZUSatz	*real*	5.d-2	

2.6 Grid command

The GRID command is used to generate an initial triangulation and to do a first refinement of this grid. The GRID command has to be executed before defining the substrate area. In WIAS-TeSCA there are different triangulation types, that can be used to triangulate simple geometrical areas (rectangles, trapezoids, etc.) Two triangulation types are preferred: TYPe=Default and TYPe=1D.

2.6.1 Default Grid

```
GRID(XLeft=..., XRight=..., YBottom=..., YTop=..., TYPe=Default,
              NX=..., DX=..., CONTrol(MAXTRl=...))
```

The rectangle [XLeft, XRight] × [YBottom, YTop] is tessellated into equilateral triangles each having a horizontal edge. Exceptionaly, at the right and the left side of the rectangle right-angled triangles are used.

The number NX of the triangles along the surface (XLeft, XRight) can be prescribed. Otherwise it is computed from the minimum edge length in the finest grid DX and the number of the refinement levels MAXTRl(from the parameter record CONTrol, see 2.13, p 98). YBottom is eventually reduced in order to choose equilateral triangles. If a new user grid is generated after a few processing steps, by default the extensions of the layer system are used for expanding the grid.

2.6.2 1D-Grid

```
GRID(TYPe=1D,X=(...), Y=(...), NX=(...), DX=(...), NY=(...),
                                DY=(...),)
```

A triangulation is generated, which might have only one vertical stripe of rectangles inhomogeneously subdivided in vertical direction. Each of the rectangles is

split into two triangles. On these grids 1D-simulations can be done with a minimum overhead. The mesh refinement usually applied in WIAS-TeSCA is inappropriate for a 1D simulation. One may either turn off any refinement

```
REPLace(CONTrol(MAXTRl=1))
```

or do a special 1D-refinement, using:

```
REPLace(CONTrol(1D=on))
```

The vectors X and Y define reference points in the grid. Equidistant subdivisions between the reference values are applied. Either the number of subdivisions NX, NY or the step sizes DX DY can be prescribed. The sign of NX and NY defines, which of the diagonals is used to split the rectangles into two triangles. This allows to construct a symmetric initial grid. The vertical spacing can also be read from a 1D cross section file:

```
GRID(TYPe=1D,X=(...), NX=(...),DX=(...),Filein‘...‘.plx))
```

If contained in the file, also the 1D layer structure is read. It is saved temporarily and can be used in the SUBStrate command. Saving the structure in TESIM-4 provides the required file content. If a 1D grid adaptation is desired, one can use the default grid and the switch `REPLace(CONTrol(1D=on))`. First the usual grid adaptation is done and after that a 1D-grid is constructed, using only the nodes at the left side of the domain.

2.6.3 Boundary condition types

The GRID-command is used also, to define the types of boundary conditions (contacts) on the outer boundary of the triangulation domain. This is used to define boundary conditions for the monolayer diffusion NEWDIFf=0. The main application is the definition of the contacts for the device simulation with WIAS-TeSCA.

The following conditions are used for process simulation in WIAS-TeSCA: typ. . . =1 homogeneous Neumann conditions (default type), > 1 inhomogeneous Neumann conditions (default at substrate surface), < 0 Dirichlet conditions.

To define the contacts for a device simulation with WIAS-TeSCA a new default type of boundary conditions must be defined TYPEO=999(symmetry conditions, homogeneous Neumann condition).

The types of boundary conditions is prescribed by the WIAS-TeSCA-conventions:

typ . . . =999 symmetry conditions (default type) > 0 Gate contact (increasing from 1) < 0 Metal contact (decreasing from -1)

In addition, optical or thermal contacts can be defined on the outer contour of the grid. The following rule is applied: $type = sign(el) \cdot 100 \cdot opt + \|el\|$ where el denotes the index of the electrical contact(or 0) and $opt \geq 0$ denotes the index of the thermal or optical contact.

For `FERMI(ISPec=0)` the optical contacts are not used and for `ENergy(IEnergy=-1)` the thermal contacts are not used. There is no way of specifying different optical and thermal contacts in the same device.

All contacts of the device have to be numbered contiguously, i.e. -1,-2, $\cdots$,-10 for the metal contacts,-1,-2, $\cdots$,-5 for the gate contacts and 1,2, $\cdots$,20 for optical and thermal contacts.

The names of the contacts can not be specified in the `Grid` command. They can be defined in the Use or `Device` commands.

If the initial triangulation is of complicate shape, the definition of the boundary conditions should (must) be supported by the (approximate) location of the starting and end points on the boundary of the triangulation. The node in the `User`-grid, closest to one of the given points, is used as start or end point of the contact.

`(XRT,YRT), (XLT,YLT) ,(XLB,YLB) (XRB,YRB)`

GRID(...BCtyp(TYPE0=...,NAME1=...,TYPE1=..., XB1=..., YB1=..., XE=..., YE=...,)

Note! The first and second end point of the contacts are defined such that moving from the first to the last, the aouter boundary of the simulation domain is traversed in positive direction (i.e. **COUNTERclockwise!!!!**).

Note! Instead of using the `GRID` command the `USE` command should be preferred for the definition of boundary conditions (see 2.20, p. 135).

2.6.4 Parameters

parameter name	unit *type* options	default value	comment
Y	*real* [μm]	0.	Vector of coordinates for the definition of the vertical discretization of a "1D" grid. If more than 2 elements of the vector are specified, TYPe=1D is assumed automatically. For TYPe=Default the vertical extension of the grid can be defined by the first two values of this vector. Unlike YBottom the exact values are used but only nearly equilateral triangles are produced.
DY	Distance	undefined	Vector of stepsizes between each pair of Y-values for TYPe=1D.
NY	*integer*	1	Vector of the numbers of subintervals between each pair of Y-values for TYPe=1D.
X	*real*	0.	Vector of coordinates for the definition of the lateral extension of a "1D" grid. For TYPe=Default the lateral extension of the grid can be defined by the first two values of this vector.
NX	*integer*	1	number of user triangles at the top edge for TYpe=Default vector of the numbers of subintervals between each pair of X-values for TYPe=1D. number of nodes for TYPe=4Triangle,2Triangle.

continued on next page

name	unit *type* option	default	comment
DX	*real*	undefined	Vector of stepsizes between each pair of X-values for TYPe=1D Minimum edge length for TYPe=Default DX and CONTrol MAXTRl define the number of user triangles NX at the top side of the user grid.
XLeft	*real* [μm]	-1.	left end of the top line
XRight	*real* [μm]	1.0	right end of the top line
YBottom	*real* [μm]	-2.4	Approximate location of the bottom edge of the simulation domain. The exact position of the bottom edge is modified, if necessary, in order to obtain equilateral triangles.
YTop	*real*	undefined	Position of the top of the triangulation domain. If the value is not specified, YTop=0 is chosen for NEWDIFf=0 and YTop=2 is chosen for NEWDIFf=1.
TYPe	*option*	Default	triangulation type: Default, 1D, Material, Finest, MOS, Netz, ROOF, TUB, Top, Bottom, TB, LEft, RIght, Allfiles, 4Triangle, 2Triangle, DOM, DUplicate, Mdraw, LEFTRIGHT
GRid		New	selection of grid manipulation type
	New		A new grid is made, replacing the old one.
	ADD		A second grid is stored.
	GLUE		Two grids are merged.

continued on next page

name	unit *type* option	default	comment
GLUE1	*integer*	0	Boundary type of the first grid, which is used for manipulations. GRid=Add:select all nodes for the given boundary type GRid=Glue:merge the nodes on the given boundary type with nodes from the second grid.
GLUE2	*integer*	0	boundary type of the second grid, which is used for manipulations. GRid=Glue: merge the nodes on the given boundary type with nodes from the first grid.
Filein	*string*80*	undefined	name of the input file, if one is required
DOMfile	*boolean*	off	controls the output of the triangulation into a file
FILEOut	*string*80*	undefined	name of output file
SIDEfac	*real*	0.5	Width of a boundary triangle for comTYPe=Default compared to internal triangles. For SIDEfac=0.5, right angled triangles at the left and right sides are assumed. For SIDEFAC=1, all triangles in a row have the same size.
MATerial	*record*	undefined	List of material names for the selection of triangles
BC(			data record to define boundary conditions (contacts) ≤ 15
TYPE1	*integer*		type of the first contact
XB1	*real* [μm]		lateral position of the start point of the first contact.

continued on next page

name	unit *type* option	default	comment
YB1	*real* [μm]		vertical position of the start point of the first contact.
XE1	*real* [μm]		lateral position of the end point of the first contact.
YE1	*real* [μm]		vertical position of the end point of the first contact.
TYPEO	*integer*	1	default type of the boundary conditions. **Note!** For WIAS-TeSCA , TYPO=999.
XRT	*real* [μm]		approximate lateral position of the right top corner of the simulation domain. If not specified, the maximum lateral coordinate of all nodes *xmax* is used.
YRT	*real* [μm]		approximate vertical position of the right top corner of the simulation domain. If not specified, the maximum vertical coordinate of all nodes *ymax* is used.
XLT	*real* [μm]		approximate lateral position of the left top corner of the simulation domain. If not specified, the minimum lateral coordinate of all nodes *xmin* is used.
YLT	*real* [μm]		approximate vertical position of the left top corner of the simulation domain. If not specified, then set to *ymax* .

continued on next page

name	unit *type* option	default	comment
XLB	*real* [μm]		approximate lateral position of the left bottom corner of the simulation domain. If not specified, then set to *xmin* .
YLB	*real* [μm]		approximate vertical position of the left bottom corner of the simulation domain. If not specified, the minimum vertical coordinate of all nodes *ymin* is used.
XRB	*real* [μm]		approximate lateral position of the right bottom corner of the simulation domain. If not specified, then set to *xmax* .
YRB	*real* [μm]		approximate vertical position of the right bottom corner of the simulation domain. If not specified, then set to *ymin* .
)			end of record BC
BCLeft	*integer*	1	type of boundary condition at the left side of the triangulation. In WIAS-TeSCA: 1:Hom.Neumann, -2:1D-D-continuation.
BCRight	*integer*	1	type of boundary condition at the right side of the triangulation. In WIAS-TeSCA:1 Hom.Neumann, -2:1 D-continuation.
BCBottom	*integer*	1	type of boundary condition at the bottom side of the triangulation. In WIAS-TeSCA: 1:Hom.Neumann, -3 no modification during diffusion

continued on next page

name	unit *type* option	default	comment
BCTop	*integer*	2	type of boundary condition at the top side of the triangulation. If BCTop=-999 the boundary conditions that are defined in the triangulation program (e.g. ITEDGE in the input file) are used.
DCHAN	*real* [μm]	0.2	For TYPe=MOS, channel depth.
DDEV	*real* [μm]	6	For TYPe=MOS, depth of the device.
DDOT	*real* [μm]	0.6	For TYPe=MOS, depth of the doping region
DFG	*real* [μm]	0.046	For TYPe=MOS, thickness of the floating gate (for IYFG >0).
DFG1	*real* [μm]	0.046	For TYPe=MOS, thickness of the oxide on top of the floating gate (for IYFG1 > 0).
DOXI	*real* [μm]	0.046	For TYPe=MOS, oxide thickness (for IYOXI>0).
DSUB	*real* [μm]	2.0	For TYPe=MOS, depth of the transition region.
IXAVA	*integer*	3	For TYPe=MOS, number of columns in the avalanche region.
IXDRAI	*integer*	3	For TYPe=MOS, number of columns in the drain region.
IXEFF	*integer*	3	For TYPe=MOS, number of columns in the channel region.
IXSOUR	*integer*	3	For TYPe=MOS, number of columns in the source region.
IYBULK	*integer*	3	For TYPe=MOS, number of rows in the bulk region.

continued on next page

name	unit *type* option	default	comment
IYCAN	*integer*	3	For TYPe=MOS, number of rows in the channel region.
IYDOT	*integer*	3	For TYPe=MOS, number of rows between channel and substrate.
IYFG	*integer*	0	For TYPe=MOS, number of rows in the floating gate.
IYFG1	*integer*	0	For TYPe=MOS, number of rows in the oxide on top of the floating gate.
IYOxi	*integer*	0	For TYPe=MOS, number of rows in the oxide.
IYSub	*integer*	3	For TYPe=MOS, number of rows in the substrate.
IZONEN	*integer*	1	For TYPe=MOS, number of zones in the simulation domain.
LAVA	*real* [μm]	0.4	For TYPe=MOS, length of the transition region.
LDEV	*real* [μm]	10.0	For TYPe=MOS, length of the device.
LEFF	*real* [μm]	6.0	For TYPe=MOS, effective gate length.
LGAT	*real* [μm]	8.0	For TYPe=MOS, gate length.
LISO	*real* [μm]	0.0	For TYPe=MOS, length of the isolator at source and drain.
XUnits()	*record*	1.	Vector of 2 scaling factors in lateral and vertical directions. Parameter for the MOS- and the Netz-type. laterally: $x_{dios} = x - tria \cdot$ XUnits (X1), vertically: $y_{dios} = y - tria \cdot$ XUnits (X2)

continued on next page

name	unit *type* option	default	comment
YO	*real* [μm]	0.	For TYPe=MOS, vertical position of the substrate surface.
TEST	*integer*	0	check the user triangulation with respect to internal edges, boundary conditions and overlapping triangles. 1:check 0:no check
CONTrol()			2.13, p. 98

2.7 Graphic command

The Graphic command should be used for checking the simulation results. REPLace (CONTrol(NGraphic=10)) can be used, to force WIAS-TeSCA to repeat the selected graphical output every 10 time steps and at the end of each processing step. REPLace CONTrol NGraphic turns off this mode.

In the interactive mode the command Graphic(calls a local command loop, where graphical output can be done. If the closing parenthesis is entered, the simulator leaves this local command loop ...

Replace(Control(NGraphic=10)) can be used, to force WIAS-TeSCA to redraw a picture every 10 time steps and at the end of each process step.

Replace(Control(NGraphic=0)) turns off this mode and is the default. A complete list of all parameters of the Graphics command is given at the end of this section.

By default the layer system and the net profile are shown:

```
Graphic (Plot).
```

The pictures are drawn (in X11) into a separate window. The DISPLAY variable is checked. In batch mode no X11-output is done.

The execution of input files continues, if the picture is drawn. The parameter WAit (default 1) defines a waiting time between finishing the output and continuation of the simulation. For WAit=0 each picture has to be confirmed by pressing the return key. The CTRL-C cancels the output of the current picture. The selected switches and modified parameters remain unchanged even if the Graphic command is left, so the next time a short command is sufficient: Graphic (Plot), Graphic (SUrface) or Graphic (Cross).

WIAS-TeSCA is now reading and handling some events in the X11 window. Moving the pointer with pressed left mouse button selects a zoom region. If the button

is released, a zoomed picture is drawn. If the pointer is moved to a certain position and then the left mouse button is clicked, the pointer position is taken as the new center of the zoomed picture for `Sample=off`. For `Sample=on` the pointer coordinates, the function values and the index of the closest vertex and triangle are printed.

If the middle mouse button is clicked a zoom-out is done, and if the right mouse button is clicked, the unzoomed picture is redrawn.

Moving the pointer with pressed middle mouse button selects the cutting line for a 1D cross section in the 2D picture. A 1D cross section along the selected line is drawn for the selected species. The length of the cutting line is printed. If the cutting line is outside the grid, it is moved. The first point of the cutting line or the first intersection of the cutting line and the grid are used as origin of the 1D plot. The distance along the cutline is displayed on the x-axis of the 1D plot.

The graphical output is repeated, if the window size of the (X11) window has been modified, or if the window had been hidden (partially) and is now visible completely.

The event queue is checked frequently during program execution and before a new command is read from standard input. A new picture can be drawn only at the end of a time step, where it does not disturb the simulation.

During the simulation the `CTRL-C` key can be used to interrupt the computation. `WIAS-TeSCA` enters the `TControl` mode. In this mode graphic and print commands can be used. The simulation is continue, when the closing parenthesis is entered.

The variables, that are displayed can be selected by `SPecies`. `WIAS-TeSCA` variable names have to be used. The selection of variables can be done even before they are introduced in the simulation. Of course they can not be drawn before defined. Before any user selection, `Net` is drawn.

The displayed region of the simulation domain can be selected by:

```
SCale(Xmid=..., YMid=..., Factor=...)
```

or

```
SCale(XLeft=..., XRight=..., YBottom=..., YTop=...)
```

`SCale (Equal=on)` is the default for equal scaling of the X- and Y-axes in the picture. The selected region is used for the 1D-, 2D- and 3D-pictures and for the output in the `Print` and `LControl` commands. The region, selected by the user is kept unchanged, otherwise the displayed region is adapted to the current size of layer system and grid.

```
SCale(Factor=1)
```

resets user selected region. For `SCale(Rescale=off)` a fixed region is kept during the entire simulation (movies!). If ABS is specified, the absolute value of the

selected functions is drawn in the 1D-, 2D- and 3D-pictures, resp. If the two components of a vector are selected, and if `ABS` is specified, isolines or isoareas of the norm of the vector field are displayed, rather than isoareas or isolines of the components.

If `MIN` or `MAX` are specified, the drawn functions are cut at the specified values.

`CUT` prescribes the minimum absolute value, displayed in the pictures. For each of the variables a logarithmic, hyperbolic or linear transformation can be selected: `LOGswitch Net Flog, Atot=Ashsur, POF=Linear`. The used color map is affected by the selected transformation.

To define 1D cross sections, the lateral or vertical positions have to be specified: `XSection(...), YSection(...)`. Vertical cross sections are shifted (by default to the current local substrate surface) and scaled.

In `XYSectio(...)` the starting and end points of arbitrary straight cross sections can be prescribed. The cross section is displayed as a function of the distance along the specified line.

All intersection points of the specified 1D cross section line and triangle edges are used. Some of the variations of the 1D profiles might result just from interpolation. The cross sections are computed and drawn, using the `Cross` command.

I-V-characteristics of a device simulation with `WIAS-TeSCA` can be displayed, using the `IVCurve` command. The parameters, displayed at the axes can be defined interactively, or using the parameters `XName, YName, BBIas` and `EBIas`. I-V-characteristics can be read from a protocol or spool file from a separate `WIAS-TeSCA` simulation, if a file name and the switch `READ=on` are specified before the `IVCurve` command. `BBIas` and `EBIas` denote numbers of user specified `BIAS` points. All `BIAS` points are numbered increasingly.

Arbitrary x-y plots can be read and displayed with the `LIneplot` command. A quite general (ASCII) file format is supported: all input lines, not containing exactly 2 numbers, are interpreted as comment lines. A dataset is defined by a comment line and several data lines. The first value of a data line is used as abscissa and the second as ordinate.

The `NLinplot` command can be used to redisplay the curves with different parameter settings.

If DELete=off has been specified also 1D cross sections and I-V-curves are not deleted after they have been displayed. Using `(... DELete=on, NLinplot ...)` all curves are deleted.

If SAVE is specified, 1D cross sections, I-V-curves and externally read curves are written to a `FIle=...`, by default in `xgraph` format.

2D pictures are drawn with the `Plot` command. The elements of the picture have to be selected with switches before drawing: `Isoline, Layer, Vector, Triangle, GLayer, BORder, Junction`.

Material, areas and boundary lines, in which grid and doping are shown, can be selected. `MATerial(...) ISOMaterial(...) AREA(...) LINE(...)`

For BORder=on the outer contour line of the grid is drawn.

Isoline=No Onebyone, Allinone, SFill, Linked, Fill selects the drawing style of isolines or isoareas.

The levels for the isoline plot are chosen automatically, but the ($\leq$) levels can be overwritten by the user. LEvel(Species=..., Nlevel=..., L1=..., L2=...).

The chosen levels are kept until the extremal values of the variable in the selected regions and the selected window are changed by more than LEvel REselect percent. In that case, using the new extremal values, the levels are reselected automatically.

Layer=No, Contour, Material, Area, Lines, Sort, SOMat and LIArea select different representations of the layer system. GLayer does the same for the approximation of the layer system in the grid. If Junction=off, the p-n-junctions are not highlighted. PNStyle=solid | bold defines the line style for the p-n-junctions.

Vector valued functions are drawn as a vector field, if Vector=on is specified, and if at least one of the components is selected. First, the x-component has to be selected. If VSw=on a vector is drawn in each grid point, otherwise VNX and VNY define a tensor product mesh in the currently displayed regions for which the values are interpolated and the vectors are drawn. For Grid=Itri internally VSw=on is assumed always.

The lengths of the vectors is defined by the norm. By default the lengths are scaled with respect to the maximum of the norms of the displayed vectors, such that the drawn vectors do not overlap. A minimum relative length VMIN independent of the norm is used to display small vectors. The drawn vectors are enlarged if VFactor > 1.

VNOrm prescribes a global maximum of the norm used in all vector plots. Vectors are not drawn, if their norm is smaller that VSuppress.

The parameter VSHape = -11, 0, 11, 12, 13, 14 ... selects the drawing style of the arrows.

The parameter Numbers=No, On, Diffgrid, Polygrid, OXidgrid selects the output of point, triangle and edge numbers. The various grids known in WIAS-TeSCA are selected by Grid=Itri, Utri, Diff, the default is Utri. For the ITRI grid the various hierarchical (multigrid) levels can be displayed: GRId=Itri, MLevel=... If MLevel=0 the user grid is displayed.

The boundary conditions and contacts for the device simulation with WIAS-TeSCA are drawn, if Contacts= on. For Contacts=Pieces the different connectivity components of the contacts are displayed. By default no contacts are drawn (Contacts=No).

The command Value(x= y=) prints the (interpolated) doping values at the given point.

The command SUrface is used to draw 3D-surface plots of the selected species, by default the functions are interpolated on a rectangular mesh. NST, MAXX and

MAXY define the maximum number of grid points and the limits in the coordinate directions of that grid.

The rotation and the tilt angle can be incremented Rotate, Tilt or prescribed exactly RAngle, TAngle. Alternatively the coordinates of a view point VIEWX, VIEWY, VIEWZ can be specified. The finest triangle grid can also be drawn as surface plot: 3DSwitch=Triangle.

XName, YName and ZName are the names of the coordinate axes that are displayed.

Additional text, markers, lines or arrows can be drawn. The displayed coordinate system in a 1D or 2D picture is used to position them. For the 3D pictures and for the LABel command the positions are to be defined in the internally used picture coordinates: $(0\ldots29.7)\times(0\ldots21)$.

A vector of text strings, lateral and vertical positions and colors has to be defined for the text. For markers and lines the style also has to be defined.

```
TEXt(T1='...',...,T10='...'), XText(...), YText(...), CTExt (...),
STarrow(s1=arrow) XArrow(x1=10, y1=12) YArrow(y1=0 2) CArrow(c1=1).
```

The LABel command just draws the specified text, markers and lines.

The parameters in the data record WIndow(...) control the subdivision of the graphical window. The lengths are interpreted in the internal plot rectangle $(0\ldots29.7)\times(0\ldots21)$. Left= Right= Bottom= Top= define the size and position of the next picture. LSHIFT and VSHIFT define the offset, reserved for scales. By default WORDs=off and the entire window is used for the picture. Text is drawn into the picture. The content MBox, IBox, NBox position XBox, YBox and orientation Box=No, Lateral, UPward, Horizont, Downward of the palette can be specified. (MBox=on material list, NBox=on species name and unit, IBox=on doping levels).

If WORDs=on, the drawing region is subdivided into picture and text part. PLeft, PRight, PBottom and PTop define the picture. 2D pictures are drawn in the right or bottom part. The text is drawn in the left or upper side. PLeft and PTop prescribe the used picture range, DLeft and DTop are used as defaults. TTop and TLeft locate the text window, LPos and VPos locate the simulation domain in the picture region; (Centered, Left, Right resp. Centered, Bottom, Top) For WORDs=off these parameters are ignored.

In WIAS-TeSCA only a single text font can be used. The X11 font can be predefined by an environment variable:

```
setenv GMSFONT "adobe-courier-bold-r-normal--25-*-*-*-*-*"
```

or specified in the input file as WIndow(FONT='screen.b.16') ZLine defines the line spacing and ZSize the character height and the offset of an exponent with respect to the internal plot range $(0\ldots21)$. ZDist defines the lateral spacing of the characters with respect to $(0\ldots29.7)$ HIgh, DOtlow, LInelow and BOLd define the lengths and the thicknesses for the various line styles. If the font or the size

of the graphical window has been modified, these parameters should be adapted.

The colors can be defined in the data record `COLors`. The switches `Substrate` and `GAS` control the filling of the polygon in the substrate and the gas regions when drawing the layer system. Color indices can be given for `SCales, TExt, TRiangle`, triangle and node numbers `TriangleNumber, NodeNumber`, contours in the layer system `LayerSystem`, the outer contour of the simulation domain `Border`,vectors `Vector`, and `Vector1`, p-n-junctions `PNcolor`, and the top and bottom side of a 3D plot `SUTop` and `SUBottom`.

The color indices for each `Material(...)`, the triangles in the material `TRIMaterial(...)` and the bounary type `SOrt(...)` can be defined too.

`Area(Area=.., Color..)` and `Line(Line=.., Color=..)` can be used to highlight a certain region or line.

The colors of the contacts are selected by `DIrichlet(..)` and `Gate (..)`

The parameter vectors Index(...), Red(...), Green(...) and Blue(...) define the RGB color values for the various color indices. Index 0 is used for the background color (white: `Red=255 Green=255 Blue=255`). Index 1 defines the inverse (black: `Red=0 Green=0 Blue=0`). The color indices 1...7 (black, red, green, blue, yellow, magenta, cyan) are used for grid, scales, surfaces plots etc. and if necessary, repeatedly used. The indices 8...18 are preserved for the materials in the layer system and the remaining colors are used as a rainbow to display the doping. To modify a color, the indices and the RGB values have to be specified.

`STeps` defines the number of color levels in the isoline plot. If `STeps=2` in the net doping, only n- and p-regions are distinguished. If `STeps` is larger than the number of allocated colors all colors from the rainbow are used. The levels in the palette are drawn each in a single color, otherwise the line of change of the color defines the isoline to the specified level.

`WIAS-TeSCA` pictures can be saved as HPGL plotter files `.hpgl`, encapsulated postscript files `.eps`, Sun raster files `.ras` or GIF-files `.gif`. This can be selected by `TERminal=PS HPGL ras ras.Z ras.gz gif gif.Z`. Postscript and HPGL files are written explicitly, for the other formats the following commands are used: `xwd -name ...|xwdtopnm | pnmtorast xwd -name ...|xwdtopnm| ppmtogif`.

`WIAS-TeSCA` can also be used, to save a series of similar pictures as a "movie". (`MOVIE=ras ras.Z ras.gz gif gif.Z gif.gz`).

The command sequence to display the "movie" is written into a file with the extension `.xmovie`. After the `WIAS-TeSCA` simulation the movies can be displayed using: `xmovie xxx.xmovie`.

A second command sequence to display the "movie" using `screenload` commands is written into an executable shell script with the extension `.movie`.

To illustrate the total processing time an analog watch can be drawn:
`CLock(Xmid=..., Ymid =..., XDiameter=..., YDiameter=...)`

The temperature can be displayed:
`TEMperature(Xleft= Ybottom= )`.

Alternatively a diagram can be used to show the current process step time (and the temperature profile):
`CLock(STGone=Difbar)`.

In `WIAS-TeSCA` several pictures can be displayed on the screen. First the picture size has to be defined `WIndow(Left= Right= Bottom= Top= )`. Next the picture is drawn and then the current graphics parameter set is saved using: `NEXTpicture()`. The largest non-overlapping rectangle is used as default for the next picture. This procedure may be repeated. The command `DRAWpicture()` restores the saved parameter sets one after the other and redraws the pictures.

The command `ERASepicture()` deletes the saved data sets. These functions can be called with a data set number (default 0), and work then with a single data set `DRAWpicture(1)`.

Note! Only the graphics parameters are saved and restored, not the grid, layer system, doping etc. If the simulation continues or after loading a file, a redraw changes all pictures.

2.7.1 Parameters shared by all plots

In this section parameters are listed, which are not specified in 1D, 2D or 3D plots. They can be specified as `Graphic(Name=value...)`

`Graphic(`			
`parameter name`	type [unit] *type* `options`	`default value`	comment
`Compare`	*string* `WordAndLength` `Word` `FullComment` `Length` `OrderInFile` `FullAndLength`	`WordAndLength`	Select the curves to be compared, if several curves are present. Used is the comment string per curve or the x-coordinate. The parameters specify several modes for comparison.
`ABS`	*boolean*	`undefined`	Draw the absolute value of each of the specified functions. If both vector components have been selected, isolines or isoareas of the norm of the vector field are drawn.
`CArrow`		`1`	List of (15) color indices of markers and lines.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
CCUTN	*real*	undefined	Shift the negative data range of the logarithm for LOGswitch=Flog
CHAracters	*boolean*	on	Do text output.
CLEar		PlotArea	Clean (part of) X11 window, before drawing.
	No		Do not clear at all. Draw all on top of each other.
	PlotArea		Fill the plot area for the next picture with background color.
	Window		Clear the entire graphical window. GKS: call clrwk
	OnceWindow		Clear once the window and switch back to plotArea
	Border		Fill scales, text, palette in background color. Draw on the old picture.
	NewWindow		Delete the X11 Window and build a new one.
	Destroy		Delete the X11 Window.
Species()	*task*		List of species to be drawn ($\leq$12).
LOGswitch()	Linear Flog Ashsur		Transformation for each of the variables: Net=Flog, Btot=Ashsur.
ABS	*boolean*	undefined	Draw the absolute value of each of the specified functions. If both vector components have been selected, isolines or isoareas of the norm of the vector field are drawn.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
MIN	*real*	-1e32	Minimum cut off value for the plot.
MAX	*real*	1.e32	Maximum cut off value for the plot.
CUT	*real*	undefined	Minimum absolute value for the plot.
Grid		Utri	Selection of the displayed grid and node numbers: Itri, Utri, Diff, USER.
SCale(	*Procedure*		Parameters for the definition of the zoom window in the simulation domain.
Xmid	*real*	undefined	Input value for the lateral position of the midpoint.
Ymid	*real*	undefined	Input value for the vertical position of the midpoint.
Factor	*real*	1	Input value for the zoom factor.
Equal	*boolean*	on	selects equal scaling of x- and y-axis.
XMID	*real*		Currently used lateral position of the midpoint.
YMID	*real*		Currently used vertical position of the midpoint.
FACTOR	*real*		Currently used zoom factor.
XLeft	*real*	undefined	Input value for the left boundary of the zoom window.
XRight	*real*	undefined	Input value for the right boundary of the zoom window.
YBottom	*real*	undefined	Input value for the bottom boundary of the zoom window.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
YTop	*real*	undefined	Input value for the top boundary of the zoom window.
XLEFT	*real*		Currently used left boundary of the zoom window.
XRIGHT	*real*		Currently used right boundary of the zoom window.
XBOTTOM	*real*		Currently used bottom boundary of the zoom window.
YTOP	*real*		Currently used top boundary of the zoom window.
GXLeft	*real*		Left boundary of the simulation domain.
GXRight	*real*		Right boundary of the simulation domain.
GYBottom	*real*		Bottom boundary of the simulation domain.
GYTop	*real*		Top boundary of the simulation domain.
FX	*real*	undefined	Input value for the lateral zoom factor.
FY	*real*	undefined	Input value for the vertical zoom factor.
Zyx	*real*	undefined	Input value for the ratio FY/FX.
FXDef	*real*	1	Currently used lateral zoom factor.
FYDef	*real*	1	Currently used vertical zoom factor.
ZYX	*real*	1	Currently used ratio FY/FX.
Rescale	*boolean*	on	Automatical rescale of the picture if the simulation domain is changed.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
)			End of the procedure Scale.
Window(	*record*		Position of picture and text in the graphical window: x:0...29.7 y:0...21.
Left	*real*	0	Left boundary.
Right	*real*	29.7	Right boundary.
Top	*real*	21	Top boundary.
Bottom	*real*	0.	Bottom boundary.
PLeft	*real*	undefined	Left boundary of the picture for WORDs=on.
PRight	*real*	29.5	Right boundary of the picture for WORDs=on.
PBottom	*real*	0.8	Bottom boundary of the picture for WORDs=on.
PTop	*real*	undefined	Top boundary of the picture for WORDs=on.
DLeft	*real*	9.5	Default left boundary of the picture, if picture is at the right, for WORDs=on.
DTop	*real*	17.	Default upper boundary of the picture, if picture is at the bottom for WORDs=on.
TLeft	*real*	0.5	Left text boundary for WORDs=on.
TTop	*real*	20.8	Upper text boundary for WORDs=on.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
Maxword	*integer*	0	Number of characters reserved for scales, names and logo at the left side of the picture (internally increased for the next picture).
LShift	*real*	1.5	Lateral offset between text and picture.
VShift	*real*	0	Vertical offset between text and picture.
LPos	Centered Left Right	Centered	Lateral position of the simulation domain in the picture for WORDs=on.
VPos	Centered Bottom Top	Centered	Vertical position of the simulation domain in the picture for WORDs=on.
SCWindow	*record*		Additional text window for WORDs=on. Used only if much text is displayed. Default: inside the picture.
Left	*real*	undefined	Left boundary.
Right	*real*	29.	Right boundary.
Bottom	*real*	22.	Bottom boundary.
Top	*real*	undefined	Top boundary.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
)			End of record SCWindow.
XLogo	*real*	undefined	Lateral position of the WIAS-TeSCA logo.
YLogo	*real*	undefined	Vertical position of the WIAS-TeSCA logo.
FONT	*string*80*		Name of an X11 text font.
DISPLAY	*string*32*		Overwrites DISPLAY variable.
ZSize	*real*		Character size relative to the (0...21).
ZDist	*real*		Character spacing relative to (0...29.7).
ZLine	*real*		Line spacing relative to (0...21).
MSize	*real*		Marker width relative to (0...29.7).
MHeight	*real*		Marker height relative to (0...21).
PRIVateColormap	*boolean*	off	switch between shared and private colormap.
WIdth	*integer*	2	X11 line width in pixels.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
WSize	*real*		X11 window width relative to 0…1.
WHeight	*real*		X11 window height relative to 0…1.
HIgh	*real*	0.15	Length of a gap in dashed or dotted lines.
DOtlow	*real*	0.1	Length of a dot in dotted lines.
LInelow	*real*	0.3	Length of a dash in dashed lines.
BOLd	*real*	0.15	Width of a bold line.
BOX		UpOrLateral	Orientation of the palette No Lateral Up Horizontal Down UpOrLateral.
XBox	Length	undefined	Lateral position of the palette in the displayed coordinate system.
YBox	Length	undefined	Vertical position of the palette in the displayed coordinate system.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
IBox	*boolean*	on	Display doping levels in the palette. For isoareas and small STeps the color change represents the isoline level. At most 40 data items with each ≤14 characters are displayed in the palette.
MBox	*boolean*	on	Display materials in the palette.
NBox	*boolean*	on	Display variable names and units in the palette.
LOGO	*string*32*	WIAS-TeSCA	Text of WIAS-TeSCA logo. Used if COLors(LOgo=...) is defined.
)			End of record Window.
COLors(	*Procedure*		Color definitions.
Reset	*Procedure*		Return to default colors.
Substrate	*boolean*	on	fill polygon for the substrate.
GAS	*boolean*	off	if on, fill the gas layer.
BAckground	*integer*	0	Index of the background color.
SCales	*integer*	undefined	Color index of the scales.
TExt	*integer*	undefined	

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
TRiangle	*integer*	undefined	
TriangleNumber	*integer*	undefined	color index of triangle numbers.
NodeNumber	*integer*	undefined	color index of node numbers.
LayerSystem	*integer*	1	Only for Layer=Contour.
Border	*integer*	undefined	Outer contour of the simulation grid.
LOgo	*integer*	undefined	By default a ball, containing all WIAS-TeSCA colors is used as a WIAS-TeSCA-logo. If LOgo=..., then the text string WIndow(LOGO='=DIOS=') is drawn. For LOgo=0 the text is drawn in background color.
Vector	*integer*	undefined	
VPoint	*integer*	undefined	Vector point for VSHape=0.
SUTop	*integer*	undefined	"Up" side in 3D-plots.
SUBottom	*integer*	undefined	"Down" side in 3D-plots.
Material()	*record*	undefined	Color indices for materials in Layer plot. If not specified, the internally reserved colors are used.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
TRIMaterial()	*record*	undefined	Color indices of triangles for the materials.
STeps	*integer*	3	Number of color steps.
SOrt	*record*		Color indices of the boundary types.
Area()	*task*		Color index for areas, defined as Area=...Color=...
Line()	*task*		Color index for lines, defined as Area=...Color=...
DIrichlet()	*record*		Color indices of the Dirichlet contacts.
Gate()	*record*		Color indices of the Gate contacts.
Index()	*integer*	undefined	Vector of color indices. 0:background, 1:inverse of 0. Currently (in X11) the indices 8...18 are reserved for the layer materials and the indices ≥ 19 are used for the rainbow. The colors 1...7 are used for everything else.
Red()	*real*	undefined	Red values 0...255 corresponding to Index().
Green()	*real*	undefined	Green values 0...255.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
Blue()	*real*	undefined	Blue values 0...255.
XBLUE(	*record*		Position and RGB values of "blue" in the rainbow. Similar records are defined for XCYAN, XGREEN, XYELLLOW, XRED
POSition	*real*		XBLUE:0 XCYAN:0.35 XGREEN:0.5 XYELLOW 0.65 XRED:1 position in the rainbow:0...1.
Red	*real*		XBLUE:0.01 XCYAN:0.01 XGREEN:0.01 XYELLOW 0.8 XRED:0.8 Red value 0...1.
Green	*real*		XBLUE:0.01 XCYAN:0.8 XGREEN:0.8 XYELLOW 0.8 XRED:0.01 Green value: 0...1.
Blue	*real*		XBLUE:0.8 XCYAN:0.8 XGREEN:0.01 XYELLOW 0.01 XRED:0.01 Blue value: 0...1.
)			end of record XBLUE, XCYAN, XGREEN, XYELLOW, XRED
)			End of the procedure COLors.
WAit	*integer*	-1	Waiting time in csec. If 0 each picture has to be confirmed by pressing Return.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
Terminal	Typ	X11	0 Regis 4ColorRegis 8ColorRegis 16ColorRegis 16FreeColorRegis TEK4014 TEK4014Emulation TEK42xx X11 PS BGI SUNVIEW GMSHPGL HPGL Off ras ras.Z ras.gz gif gif.Z gif.gz The default type can be predefined by the environment variable GMSDEFAULT. Only one terminal type can be defined at a time. For the types ras* gif* the X11 window is required and a system command is used to generate the files. The file name can be specified. xwd -name ...|xwdtopnm|pnmtorast xwd -name ...|xwdtopnm|ppmtogif.
NCutp	integer	undefined	Number of orders of magnitudes used in the positive data range for LOGswitch=Flog.
NCUTN	*integer*	undefined	Number of orders of magnitudes used in the negative data range for LOGswitch=Flog.
CCutp	*real*	undefined	Shift of the positive data range of the logarithm for LOGswitch=Flog.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
CCUTN	*real*	undefined	Shift of the negative data range of the logarithm for LOGswitch=Flog.
TRAnsformed	*boolean*	off	Plot of transformed functions. For concentrations in polysilicon: off: c^g, c^{gb} on: $cc^g + F \cdot c^{gb}, F \cdot c^{gb}$ off: silicon consumption for O2Dif=Zone on:silicon density for O2Dif=Zone
SCARrow	*boolean*	off	Select scale arrows or scale rectangle.
SECscale	Tic Default Grid No	Default	Tic: Only one set of scale tics. Default: Second set of scale tics at the opposite side. Grid: Rectangular grid in the entire picture.
CHAracters	*boolean*	on	Do text output.
WORDs	*boolean*	off	on: Picture and text separated. off: Text inside the picture.
XName	*strin*8*	undefined	Name of the x-axis. Used also to select I-V-curves.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
XScale	*record*	undefined	Vector of scale values, used for the x-axis.
YName	*record*	undefined	Name of the y-axis. Up to 5 names can be specified to select I-V-curves.
YScale	*real*	undefined	Vector of scale values, used for the y-axis.
ZName	*string*8*	undefined	Name of the z-axis in 3D-plots.
ZScale	*real*	undefined	Vector of scale values, used for the z-axis. For 2D plots this overrides the level definition for all species.
PLOtter	*integer*	0	Pen velocity, when writing a HPGL file. TERminal=HPGL
RESET	*procedure*		Reset the graphics command to its initial state.
EXponent	*boolean*	on	Representation of real numbers on: 10^{20}, off:1E20
TEXt	*string*80*	undefined	Vector of ≤ 15 text lines.
XTEXT	*real*	undefined	Lateral positions for TEXt. In 1D- and 2D-pictures, with respect to the displayed coordinate system. In 3D-pictures with respect to 0 ... 29.7.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
YTEXT	*real*	undefined	Vertical positions for TEXt. In 1D- and 2D-pictures, with respect to the displayed coordinate system. In 3D-pictures with respect to 0 ... 21.
CTExt	*integer*	undefined	Vector of color indices for TEXt.
STExt		Border	Style of text representation.
	Plain		Draw text in the given color.
	Background		Draw text in a rectangle of background color.
	Border		Draw text in a rectangle of background color, surrounded by a line in text color.
STarrow		Arrow	Vector of marker or line styles: No Arrow Solid Dotted DAshed DASDotte DASH2Dot Bold BDotted BDAshed BDASDott BDASH2Do Plus Asterisk X Square Rhomb TTriangl BTriang RTriangl LTriangl Circle REctangl FSquare FRhomb FTTriang FBTriang FRTriang FLTriang BUllit For a line or an arrow two values have to be specified in XArrow, YArrow, for a marker one.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
XArrow	*real*	undefined	Vector of lateral positions of markers and lines.
YArrow	*real*	undefined	Vector of vertical positions of markers and lines.
YArrow	*real*	undefined	Vector of vertical positions of markers and lines.
CArrow		undefined	Vector of color indices of markers and lines.
LABEL()	*procedure*		Display only text, markers and lines. The graphics window is not erased and subwindows are not respected.
IPLOtu	*integer*	0	Graphical output channel. Internally defined.
IWT	*integer*		Workstation type for GKS output. Internally defined.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
MOVIE	No ras ras.Z gif ras.gz gif.Z gif.gz	No	After each picture is drawn, an image file is created: xwd -name ...|xwdtopnm|pnmtorast ...|xwdtopnm|ppmtogif The names of the image files are written into a script file with extension .xmovie After the WIAS-TeSCA simulation the movie can be displayed using xmovie xxx.xmovie At the same time a shell script with extension .movie is written that uses screenload to display the movie files. (Works on Sun only). Internally defined.
MOVPIC	*integer*	0	Number of the first picture, incremented after a picture is dumped.
MOVCMD	*string*80*	undefined	User defined command for dumping pictures. Internally only the name of the picture file and the number are appended to the string.
FILe	*string*80*	undefined	File name, used to save 1D cross sections, to read curves, to save pictures for movies.
CLEar	Action	Window	Erase (parts of) the graphical window.
	No		Do not clear at all.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
	PlotArea		Fill the plot area for the next picture with background color.
	Window		Clear the entire graphical window. GKS: call clrwk
	OnceWindow		Clear once the window and switch back to PlotArea
	Border		Fill scales, text, palette in background color. Draw on the old picture.
	NewWindow		Delete the X11 window and build a new one.
	Destroy		Delete the X11 window .
NEXTpicture	*procedure*		Store the plot parameter set of the just drawn picture, define the maximum new window size. A picture number can be specified to overwrite a parameter set. Default: 0 i.e. append a new parameter set.
DRAWpicture	*procedure*		Read the saved parameter set, redraw the picture(s) and define the maximum new window size. A picture number can be specified. Default 0: redraw one by one all stored data sets.
ERASpicture	*procedure*		Delete the saved parameter sets and define the maximum new window size. A picture number can be specified to erase a specific parameter set. Default 0: rase all data sets.
CLock	*record*		Parameters for displaying the processing time.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
Xmid	Length	undefined	Lateral position of the midpoint and the left boundary of a time bar, resp.
Ymid	Length	undefined	Vertical position of the midpoint and the bottom boundary of a time bar, resp.
XDiameter	Length	undefined	Width of the clock and time bar, resp.
YDiameter	Length	undefined	Height of the clock and time bar, resp.
BackColor		0	Background color index.
BOrderColor		1	Border color index.
GoneColor		0	Color index for already simulated time.
STBorder		Solid	Style of the borderline: Solid, Bold.
STGone		Minutes	Style of the clock or already simulated time: Minutes Hours Bar Tempbar Diffbar. If Tempbar, a time-temperature bar is used. If Diffbar only during diffusion steps.
DATeShift		undefined	Shift of the displayed date. If undefined, no date is displayed.
XLeftDate	length	undefined	Left boundary of the date.
XRightDate	length	undefined	Right boundary of the date.
YBottomDate	Length	undefined	Bottom boundary of the date.
YTopDate	Length	undefined	Top boundary of the date.
Name	*string*8*	blank	Name of the time axis for STGone=Bar.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
Time	Time	undefined	Start time of the analog clock.
TIMElist	Time	undefined	List of parameters for the time axis for STGone=Bar. TEMPList Parameters, displayed as a piecewise linear function of TIMEList. If undefined during diffusion steps, the temperature ramps are displayed.
Scale	Time	undefined	Vector of scale values for the time scale, if STGone=Bar.
)			End of record CLock.
TEMperature	*record*		Parameters for the temperature display.
Xleft	Length	undefined	Left boundary.
Ybottom	Length	undefined	Bottom boundary.
XRight	Length	undefined	Right boundary.
BackColor		0	Background color index.
BOrderColor		1	Borderline color index.
GoneColor		2	Color of the temperature.
STBorder		Solid	Style of the borderline: Solid Bold.
Minimum	Temperature	undefined	Minimum value.
MAximum	Temperature	undefined	Maximum value.
Name	*string*8*	blank	Name of the temperature axis.
Temperature	Temperature	undefined	value to be displayed.

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
TEMPList	Temperature	undefined	List of parameters for the temperature axis for STGone=Bar. TEMPList is displayed as a piecewise linear function of TIMEList. If undefined during diffusion steps, the temperature ramps are displayed.
Scale	Temperature	undefined	Vector of scale values for the temperature scale.
)			End of record TEMperature
XCoordinate		XCoord	Variable used as x-coordinate.
YCoordinate		YCoord	Variable used as y-coordinate.
MLevel	*integer*	10	Multigrid level, used for Grid =Itri.
XSecond(	*record*		Parameters for a second x-scale
Xminimum	*real*	undefined	minimum value for the 2nd x-axis in x-y-plots.
Xmaximum	*real*	undefined	maximum value for the 2nd x-axis in x-y-plots.
Logswitch	Linear Flog Ashsur	Linear	Interpolation of the second x-scale
Name	*string*8*	undefined	name of the second x-axis
)			End of record XSecond
YSecond(	*record*		Parameters for a second y-scale
Yminimum	*real*	undefined	minimum value for the 2nd y-axis in x-y-plots.
Ymaximum	*real*	undefined	maximum value for the 2nd y-axis in x-y-plots.
Logswitch	Linear Flog Ashsur	Linear	Interpolation of the second y-scale

continued on next page

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
Name	*string*8*	undefined	name of the second y-axis
)			End of record YSecond
VARiable()	*record*		see section 20, p.224

2.7.2 Parameters for 1D Plots

name	unit *type* option	default	comment
Cross	*procedure*		Command to compute and display 1D cross sections, see XSection, YSection, XYSection
LIneplot	*procedure*		Command to read and display an x-y plot file.
IVCurve	*procedure*		Command to display an I-V-curve, after the device simulation with WIAS-TeSCA, or read from a WIAS-TeSCA protocol or spool file.
NLineplot	*procedure*		Command to redisplay curves (with modified selected curves, parameters, colors, line styles, curve text, delete flag...).
XSection	*procedure*		Lateral positions of $\leq$12 vertical 1D cross sections.
C1	Length	0	1st position (a WIAS-TeSCA x-coordinate).
MIN	*real*	undefined	Starting point (a WIAS-TeSCA y-coordinate).
MAX	*real*	undefined	End point (a WIAS-TeSCA y-coordinate).

continued on next page

name	unit *type* option	default	comment
)			End of procedure XSection.
YSection	*procedure*		Vertical position of the lateral cross sections.
C1	Length	---um	1st position (a WIAS-TeSCA y-coordinate).
MIN	*real*	undefined	Starting point (a WIAS-TeSCA x-coordinate).
MAX	*real*	undefined	End point (a WIAS-TeSCA x-coordinate).
)			End of procedure YSection.
XYsection(	*procedure*		Define ($\leq$3) arbitrary straight cross sections by specifying start and end points.
Xb1	Length	0um	X-coordinate of the first starting point.
Yb1	Length	0um	Y-coordinate of the first starting point.
Xe1	Length	0um	X-coordinate of the first end point.
Ye1	Length	0um	Y-coordinate of the first end point.
)			End of procedure XYSection.
SHift	*real*	1.e10	Shift of vertical cross sections. displayed_ value := (DIOS_ value-SHift)· FACtor For SHift>1E9 the local position of the substrate surface is used.

continued on next page

name	unit *type* option	default	comment
FACtor	*real*	-1000	Scaling factor for the coordinates of a vertical cross section. (-1000 to invert the direction and to scale from WIAS-TeSCA μm into TESIM-4's nm).
Append	*boolean*	on	Append or replace when saving curves into a file.
BBIas	*integer*	undefined	First BIAS point, selected for an I-V- curve.
EBIas	*integer*	undefined	Last BIAS point, selected for an I-V- curve.
READ	*boolean*	off	on: Read I-V-curves from a file, resp. read all curves from the file when using LIneplot. off: Select the curves from the file interactively when using LIneplot.
FOrmat	Tesim Xgraph All plx plt	All	File format for the input/output file of x-y plots.
MErge	*boolean*	off	Linear combination of two selected curves.
XMInimum	*real*	-1.e32	Minimum value of the displayed x-axis.
XMAximum	*real*	1.e32	Maximum value of the displayed x-axis.
INTegral	*real*	1	Integral of the normed curves for INTNorm.
XX	*real*	1	Parameter for the transformation $x := XX \cdot x + XY \cdot yX0$.
XY	*real*	0	

continued on next page

name	unit *type* option	default	comment
X0	*real*	0.	
YX	*real*	1	Parameter for the transformation $y := YX \cdot x + YY \cdot y + Y0$.
YY	*real*	-1	
Y0	*real*	0	
DX	*real*	0	Spacing for an interpolation of the curves to an equidistant grid.
NX	*integer*	0	Number of grid point for an equidistant grid.
Y1merge	*real*	1	Scaling factor of the first curve for MErge.
Y2merge	*real*	1	Scaling factor of the second curve for MErge.
COmpress	*boolean*	on	Compress the curve list.
PERMUTation	*record*	0	Vector of curve indices for permutation of curves.
LIStyle	*record*	1	Vector of line styles for each of the curves: No Solid Dotted DAshed DASHDotted DASH2Dot Bold BDotted BDAshed BDASHDott BDASH2Do
INStyle	*boolean*	on	Increment line style (after each curve, resp. if all colors have been used).
LIColor	*record*	0	Color indices of the curves.
INColor	*boolean*	on	Increment the color index.

continued on next page

name	unit *type* option	default	comment
LIMarker	*record*	0	Marker style for each of the curves: No Plus Asterisk X Square Rhomb TTriangl BTriang RTriangl LTriangl Circle FSquare FRhomb FTTriang FBTriang FRTriang FLTriang BUllit
INMarker	*boolean*	off	Increment the marker style.
LIText	*record*	undefined	Comment text for each of the curves.
LILogsw	Linear Flog Ashsur	Flog	Transformation of the y-values of all curves. For general plots or if different LOGswitch values are defined for the displayed species.
LIEQual	*boolean*	off	Equal scaling of x- and y-axis in plots.
Select	*record*	0	List of curve indices. If possible, the curves are selected from the previously displayed curves, otherwise from all defined curves.
INTNorm	*boolean*	off	Scale the curves to fit INTegral.
MAXNorm	*boolean*	off	Scale the curves to fit MIN and MAX.
XYchange	*boolean*	off	Toggle the axes.
Invert	*boolean*	off	Invert the order of the points in the curves.
LICLear	*boolean*	on	Delete unnecessary curve points with the same y-values.

continued on next page

name	unit *type* option	default	comment
IPU	*integer*	0	Additional print channel for x-y plot comments, extrema, values, sheet resistance.
PCOmment	*boolean*	off	Print the curve comments.
PExtrema	*boolean*	off	Print the extremal values of the curves.
PVALues	*boolean*	off	Print all curve values.
RS	*boolean*	off	Print the sheet resistance between the p-n-junctions of the 1D cross sections.
PIntegral	*boolean*	off	Print the integrals.
PPlot	*boolean*	on	Plot the curves.
SAVE	*boolean*	off	Save the curves into a file. **Note!** For xgraph the maximum of the absolute values and 1e-32 are written into the file.
SAVTrans	*boolean*	off	Save the transformed curves, i.e. log(...), into the file.
DELete	*boolean*	on	Delete the (selected) curves.

2.7.3 Parameters for 2D Plots

name	unit *type* option	default	comment
Plot	*procedure*		Command to display 2D cross pictures.
LEvel(	*task*		Selection of isoconcentration levels.
NCon	*integer*	undefined	Minimum number of levels for Set

continued on next page

name	unit *type* option	default	comment
REselect	Percent	100%	Reselect levels if extremal values changed by given percentage.
Set	procedure		define new levels.
Species	*string*8*	S1	Variable, for which the levels should be chosen.
Nlevel	*integer*		Currently used number of levels.
L1	*real*		$\leq$20 concentration values.
)			End of task LEvel.
MATerial()		undefined	List of WIAS-TeSCA materials in which the grid should be displayed.
ISOMaterial()		undefined	List of WIAS-TeSCA materials, in which the doping is displayed.
AREA()	*record*	undefined	List of Area numbers to be plotted.
LINE()	*record*	undefined	List of LINE numbers to be plotted.
ISoline	*Isolines*	Fill	Draw style for the doping.
	Onebyone		One doping level at a time.
	Allinone		All doping levels at the same time.
	SFill		Fill each triangle.
	Linked		Define a linked list of isolines. Cannot be used for GRID=ITRI.
	Fill		Fill isoareas.
Triangle	*boolean*	off	Display the grid.
Border	*boolean*	off	Display the outer contour of the grid.

continued on next page

name	unit *type* option	default	comment
Vector	*boolean*	off	display vector valued functions as vector field.
Layer		Material	Draw style for the layer system.
	No		
	Contour		Only contour lines.
	Material		Fillarea with the material colors.
	Area		Fillarea for the areas.
	Lines		Draw the contour Lines. When using this together with a marker, LIMarker the points in the layer system are marked
	Sort		Draw boundary sorts.
	SOMat		Do Material and Contour.
	Zone		Fill WIAS-TeSCA zones.
Contacts		No	Display style for boundary conditions and contacts.
	on		Draw contact types.
	BC		Draw boundary condition types.
	Pieces		Draw connectivity components of the contacts.
GLayer		No	Similar to Layer but for the regions of the UTRI-grid.
Junction	*boolean*	off	Draw thick isoline at th p-n-junctions.
PNStyle		Bold	Linestyle for the p-n-junctions. See LIStyle.
PNcolor	*integer*	1	Color index for the p-n-junctions.

continued on next page

name	unit *type* option	default	comment
PNMarker		No	Marker style for the p-n-junctions. See LIMarker.
Number		No	Display triangle and node numbers.
	On		numbers in the grid
	Diff		Node numbers, used in the matrix (for the total concentrations).
	Poly		Node numbers used in the matrix (for the grain boundary concentrations).
	Oxid		Node numbers used in the matrix (for the oxidant diffusion).
	All		debugging. draw all UTRI triangles.
VNX	*integer*	40	Number of lateral discretization points for VSW=off.
VNY	*integer*	35	Number of vertical discretization points for VSW=off.
VFactor	*real*	1.5	Scaling factor for vector plot. By default the vectors are scaled with respect to their maximum value or VNOrm and the minimum grid spacing, such that they do not overlap, and then multiplied by VFactor.
VNOrm	*real*	undefined	Maximum norm. If specified, this value is used to scale all vector fields.
VSuppress	*real*	undefined	Minimum norm. Vectors with smaller norm are not displayed.

continued on next page

name	unit *type* option	default	comment
VMIN	*real*	0.3	Minimum relative length. The smallest, displayed vector is drawn at this length compared to the largest vector. If VMIN=1 all vectors are drawn with the same length.
VSHape	*integer*	-11	Arrow style. For ±11 and ±12 the tip is defined relative to the body. For ±11 and ±13, the tip is filled. For 0, the tip is displayed in a different color. For values < 0, a bold vector body is drawn. For > 0, a line, for > 10 a different line style is used.
VSW	*boolean*	off	on: Display vectors in the nodes of the grid. off: Use a VNX×VNY tensor product grid to display the vector field.
VLength	*real*	0.4	Relative length of the vector tip.
VWidth	*real*	0.23	Relative width of the vector tip.
MVLength	*real*	0.5	Absolute length of the vector tip.
MVWidth	*real*	0.4	Absolute width of the vector tip.
XFill	*integer*	1000	Number of lateral lines for the fiillpolygon function.
YFill	*integer*	0	Number of vertical lines for the fiillpolygon function.
RFill	*integer*	0	Display borderline in the fillpolygon function.

continued on next page

name	unit *type* option	default	comment
IMAGline	*boolean*	off	highlight the imaginary lines. LSWitch=Contour
LI1	*integer*	undefined	First boundary type, displayed with Contacts.
LI2	*integer*	undefined	Last boundary type, displayed with Contacts.
CTHickness	*real*	0.3	Thickness of the contacts.
Value(	*procedure*		Print the interpolated values in the specified points.
X	Length	undefined	Position.
Y	Length	undefined	Position.
)			End of procedure Value.
LOngtext	*boolean*	off	Length of text when displaying grid, contacts, boundaries.

name	unit *type* option	default	comment

2.7.4 Parameters for 3D plots

In this section parameters are listed, which control the 3D surface plots. They can be specified as Graphic(Name=value...)

Graphic(			
parameter name	type [unit] *type* options	default value	Comment
3Switch	Rectangle Triangle SRectangle	Rectangle	Selection of displayed grid.

continued on next page

Graphic(			
parameter name	*type* options	default value	Comment
MAXX	*integer*	126	Maximum number of discretization points in x direction for 3Switch=Rectangle,SRectangle.
MAXY	*integer*	126	Maximum number of discretization points in y direction for 3Switch=Rectangle,SRectangle.
NEW	*boolean*	on	Repeat the interpolation to the rectangular mesh before each 3D plot.

2.8 Fermi command

WIAS-TeSCA is based on the Boltzmann statistics. For the simulation of opto-electronic heterostructures and degenerated semiconductor materials it is possible to use Fermi-Dirac statistics. The FERmi-command is used to enter the required parameters. Using Fermi-Dirac statistics (IFERMI=1), the exp-function is replaced by the Fermi integral $F_{1/2}(s)$.

Moreover, the FERmi-command is used to enter the optical parameters for the simulation of optical devices like semiconductor lasers with the models described in section 1.5 on page 13.

In command DOMAIN the parameter TYPe = 3 (analogously for other values of TYPe).

ILASER> 0 has to be set (number of "laser contacts") and the number IANSEI of the sides and, for each of these sites, the zone number IZ and the local number of a side in a zone has to be specified.

2.9 Models and their parameters

The model parameters of following quantities are set within FERMI.

2.9.1 Gain g

The optical gain (amplification coefficient) is always set nonzero only in the active material, i.e. the material with smallest band gap in the device. Parameter GTYP chooses one of the following gain models.

Default Model for Maximum Gain, GTYP=0

$$g = \kappa \cdot \left[\exp\left(\frac{eU_F - \hbar\omega}{kT}\right) - 1\right] \cdot \frac{np}{N_i^2 e^{eU_F/kT}}; \qquad eU_F = F_n - F_p. \tag{2.20}$$

T is the current local temperature. g is the density-dependent gain at a fixed user-set lasing wavelength $\lambda =$ `ALAM` cm. $\kappa =$ `AKAPPA` m^{-1} is the absorption coefficient at this wavelength in equilibrium. g is negative at small densities, goes through zero at $eU_F = \hbar\omega$ and increases proportional to np at much higher densities. Only a small interval above the zero is needed in lasers, where the slope can be fitted by κ.

Maximum Gain Model, GTYP=1 Source: H. Wenzel and G. Erbert, in Physics and Simulation of optoelectronic devices IV, SPIE vol. 2693 (1996).

$$\begin{aligned} g &= \kappa \cdot \left[\exp\left(\frac{eU_F - \hbar\omega}{kT}\right) - 1\right] f\left(\frac{F_n - E_c}{kT}\right) f\left(\frac{E_v - F_p}{kT}\right) \\ &\text{with } f(x) = \frac{1}{1+e^x}. \end{aligned} \tag{2.21}$$

Like `GTYP=0`, but with saturating asymptotics above crossing zero.

Spectral Gain Model, GTYP=2 Source: H.-J. Wünsche et al., IEEE Journ. Quant. Electron. 29, no. 6, pp 1751-61 (1993).

$$g = \kappa\sqrt{\frac{\max(\Delta, 0)}{kT}} \cdot \left[f\left(\frac{E_c + \frac{m_h}{M}\Delta - F_n}{kT}\right) - f\left(\frac{E_v - \frac{m_c}{M}\Delta - F_p}{kT}\right)\right] \tag{2.22}$$

with $\Delta = \hbar\omega - E_g$ being the photon energy in excess of the band gap and $M = m_e + m_h$. g is the textbook gain formula for parabolic bands and no multiparticle effects. It depends correctly on densities and wavelength within this approximation. κ is the band-band absorption coefficient in equilibrium at $\Delta = kT$. Note: the difference between the arguments of the two occupation factors is $(\hbar\omega - eU_F)/kT$. Hence, g crosses also zero at $eU_F = \hbar\omega$.

2.9.2 Refractive index $\bar{n}$

$\bar{n}$ is temperature dependent according to the formulas

$$\bar{n}(T) = \Big(\bar{n}(T_0) - \bar{n}_d \cdot (n+p)/2\Big) + \bar{n}_T \cdot (T - T_0), \qquad typ = 0 \tag{2.23}$$

$$\bar{n}(T) = \Big(\bar{n}(T_0) - \bar{n}_d \cdot (n+p-|D_{net}|)/2\Big) + \bar{n}_T \cdot (T - T_0), \qquad typ = 1 \tag{2.24}$$

$$\bar{n}(T) = \Big(\bar{n}(T_0) - \theta_A \cdot \sqrt{\bar{n}_d \cdot (n+p)/2}\Big) + \bar{n}_T \cdot (T - T_0), \qquad typ = 3 \tag{2.25}$$

$\theta_A = 1$ in active layer and zero elsewhere.

Variable	Name in TeSCA	Proc	remark
$\bar{n}(T_0)$	BRE	FERMI	
$\bar{n}_d$	BREFAK	FERMI	material factor
$\bar{n}_T$	BREA	ENERGY	
T_0			fixed temperature
typ	BRETYP	FERMI	

2.9.3 Internal optical loss α_b

α_b is composed of the free carriers absorption (f_{cn} and f_{cp}) and the inter valence band absorption α. We have

$$\alpha_b = \alpha(T) + f_{cn}(T)n + f_{cp}(T)p \tag{2.26}$$

$$\alpha(T) = \alpha_0\alpha_1\left(e^{E_0/T_0 - E_0/T}\right) \tag{2.27}$$

$$f_{cn}(T) = f_{cn0}T^{\gamma_n} \tag{2.28}$$

$$f_{cp}(T) = f_{cp0}T^{\gamma_p} \tag{2.29}$$

Variable	Name in TeSCA	Proc	Bemerkung
α_0	AALPHA	FERMI	
α_1	AALPHF	FERMI	material factor
E_0	EA	FERMI	
f_{cn0}	FCNALF	FERMI	
f_{cp0}	FCPALF	FERMI	
γ_n	GN	MOBILITY	
γ_p	GP	MOBILITY	

2.9.4 Photon balance

Concerns the model equations (1.43–1.46) on page 14.

Variable	Name in TeSCA	Proc	Bemerkung
α_1	AALPH1	FERMI	scattering losses mode 1
α_2	AALPH2	FERMI	↑ mode 2
$\bar{n}_{g1}$	GRUP1	FERMI	group index mode 1
$\bar{n}_{g2}$	GRUP2	FERMI	↑ mode 2
K_1	PEFA1	FERMI	Petermann factor mode 1
K_2	PEFA2	FERMI	↑ mode 2
$R_1(0)$	R01	FERMI	reflectivity left facet mode 1
$R_1(L)$	RL1	FERMI	dto. right facet
$R_2(0)$	R02	FERMI	↑ mode 2
$R_2(L)$	RL2	FERMI	dto. right facet
	SRELAX	FERMI	numerical parameter

2.9.5 Treat Powers as Parameters (TPP)

The basics of this approach are sketched on page 15. The folowing parameters can be used to steer the content of the calculated tables.

Variable	Name in TeSCA	Proc	Bemerkung
	NPower	FERMI	number of power values
$P_1 + P_2$	POwers	FERMI	power values (mW)
	IEtam	FERMI	number of η values
	NLAM1$\leq$ 20	FERMI	number of wavelengths mode 1
	NLAM2$\leq$ 20	FERMI	ditto mode 2
λ_1	LAM1	FERMI	wavelengths mode 1 (cm)
$\lambda_2 - \lambda_1$	LAM2	FERMI	wavel. mode 2 relative to mode 1 (cm)

The model powers are calculated according to $P_{1,2} =$ `POwers` $\cdot\,(1 \pm \eta)/2$, where η runs over `IEtaem` values equally spaced between -1 and +1.

2.10 Parameters

`name`	`unit`
	type
	`option`
		`default`	comment

`AALPHA`	*real* m^{-1}	`1d4`	α_0 (2.27) p. 81
`AALPHF()`	*real*	`1.d0`	α_1 material factors for `AALPHA`
`AKAPPA`	*real* m^{-1}	`1d3`	κ (2.20) to (2.22) p. 80
`AKAPPF()`	*real*	`1.d0`	κ_1 material factors for `AKAPPA`
`ALAM`	*real* [cm]	`1.3d-4`	Lasing Wavelength, Sec. 1.5
`ALPHA`	*real*	`0.1d0`	strain field
`BRE()`	*real*	`1.d0`	$\bar{n}(T_0)$ for all materials (2.13) p. 27
`BREFAK`	*real* cm^3	`1d-19`	$\bar{n}_d$ (2.13) p. 27
`BENRADI`	*real* [cm]	`1d-4`	
`BCENTER`	*real* [cm]	`0d0`	
`BRETyp`	*integer*	`0`	type of refr. index (2.13) ff. p. 27

continued on next page

name	unit *type* option	default	comment
CMO1()	*real* [1/cm]	0.d0	constant for optical boundary condition for mode 1, CMO1=0: natural bc, CMO1$\gg$ 1: Dirichlet bc, Length$\leq$20
CMO2()	*real*	0.d0	Length$\leq$20, see $\uparrow$
COMPOS	*real*	0.22d0	Material Composition
DEFKON	*real* [cm^{-3}]	1d18	Defect concentration
EC()	*real* [V]	0.562d0	Conduction band edge Length$\leq$ mreg
EG()	*real* [V]	1.124d0	Energy band gap Length$\leq$ mreg
EIGANF()	*real*	3.6d0	initial values for $c\Re e\beta_i/\omega$
EIGMAX	*real*	3.6d0	upper bound for $c\Re e(\beta_i)/\omega$
RO1	*real*	0.36d0	facet reflectivity at $z = 0$, mode 1
RL1	*real*	0.36d0	facet reflectivity at $z = L$, mode 1
RO2	*real*	0.36d0	facet reflectivity at $z = 0$, mode 2
RL2	*real*	0.36d0	facet reflectivity at $z = L$, mode 2
PEFA1	*real*	1.d0	Petermann factor K_1 for mode 1, (1.46) p. 15
PEFA2	*real*	1.d0	$\uparrow$ mode 2
GRUP1	*real*	3.6d0	group index $\bar{n}_{gi}$ mode 1, (1.43) p. 14
GRUP2	*real*	3.6d0	$\uparrow$ mode 2
AALPH1	*real*	0.d0	scatt. losses α_i mode 1, (1.43) p. 14
AALPH2	*real*	0.d0	$\uparrow$ mode 2
EPSP1	*real*	0.d0	nonlin. gain saturation

continued on next page

name	unit *type* option	default	comment
EPSP2	*real*	0.d0	↑ mode 2
SRELAX	*real*	1.d0	numerical parameter
EA	*real*	0.1d0	optical loss (2.27) p. 81
FCNalf	*real*	0.d0	optical loss (2.28) p. 81
FCPalf	*real*	0.d0	optical loss (2.29) p. 81
GTYP	*integer*	0	which gain model, cf. Sec. 2.9.1
HAOM	*real*	0.d0	Parameters (photogen.)
HHMAS	*real*	0.44d0	Parameters (photogen.)
IEtam	*integer*	0	Step number (variation of ETA) (TPP-method p. 82)
IFERMI	*integer*	0	IFERMI=0 (Boltzmann statistics), IFERMI=1 (Fermi-Dirac statistics)
INCNV	*integer*	0	
IFOTO	*integer*	0	switch on the photogeneration
IPOL1	*integer*	1	polarization mode 1, IPOL1=1 (TE polarization), IPOL1=2 (TM polarization)
IPOL2	*integer*	2	polarization mode 2, see ↑
IFARfld	*integer*	0	
IPtotm	*integer*	0	switch on the self consistent calculation of the optical gain, step number (variation of PTOT)
ISPec	*integer*		switch eigenmode calculation, ISPec < 0 activate the simple gain function
		0	optics is switched off
		1	one mode (TE or TM)

continued on next page

name	unit *type* option	default	comment
		2	two modes (TE or TM)
ITUN	*integer*	0	switch, tunnel generation
LAM1()	*real* [cm]	1.5d-4	wave-lengths Length≤ 20 (TPP-method p. 82)
LAM2()	*real* [cm]	0.0d0	differences of wave-lengths Length≤ 20 (TPP-method p. 82)
EFMN()	*real* [V/cm]	1.18d0	Length≤ mreg
EFMP()	*real* [V/cm]	0.5d0	Length≤ mreg
NC()	*real* [cm^{-3}]	2.86d19	Density of states (electrons) Length≤ mreg
NV()	*real* [cm^{-3}]	3.10d19	Density of states (holes) Length≤ mreg
NLAM1	*integer*	1	Number of wave-lengths (TPP-method p. 82)
NLAM2	*integer*	0	Number of observed wave-lengths (TPP-method p. 82)
NPower	*integer*	0	Number of powers (TPP-method p. 82)
POwers()	*real*	0.0d0	powers Length≤ 100 (TPP-method p. 82)
PTOTM	*real*	20.d0	Total output (laser)

2.10.1 Parameters for photogeneration

name	unit *type* option	default	comment
RICH	*real*	0.d0	
R0	*real*	7.d-8	Localization radius
TS	*real*	300.d0	radiation temperature

continued on next page

name	unit *type* option	default	comment
VERL	*real*	1.d0	Loss factor
VMA	*real*	0.d0	
XF1	*real* [cm]	0.d0	Window coordinate
XF2	*real* [cm]	0.d0	Window coordinate
YF1	*real* [cm]	0.d0	Window coordinate
YF2	*real* [cm]	0.d0	Window coordinate

2.11 Mobility command

2.11.1 Models

The MObility-command defines the parameters for the intrinsic density, the carrier mobilities and the diffusion.

The intrinsic density and the carrier mobilities can depend on the temperature T (the temperature is always normalized $T = \frac{\text{Temp in } K}{300K}$), the dopant concentration D and the electrical field E. Moreover, the dependence on the material M can be considered by a constant factor. Thus, the quantities have the general form

$$N_i^{eff} = N_i(T, M) = N_i(T)N^{\text{mat}}(M) \tag{2.30}$$
$$\mu_n^{eff} = \mu_n(T, D, E, M) = \mu_n(T, D, E)\mu_n^{\text{mat}}(M) \tag{2.31}$$
$$\mu_p^{eff} = \mu_p(T, D, E, M) = \mu_p(T, D, E)\mu_p^{\text{mat}}(M) \tag{2.32}$$

The parameters N_i(ENi), $N^{\text{mat}}(M)$(ENIFA) and T(TEmp) can be set in the command DEvice.

Six models (parameter Model) are offered to handel different dependings of the mobilities on the temperature, doping, and the field strength.

2.11.2 General description of the models

Model = 1
Use the basic quantities for the intrinsic density (N_i^{eff}) and the mobilities (μ_n^{eff}, μ_p^{eff}).

Model = 2
Mobilities and intrinsic carrier density depend on temperature and doping. No dependency on field strength.

Model = 3,5,6
Like Model = 2 but a saturation of the carrier velocity is taken into account. The formulas (2.52, 2.36, 2.39, 2.60, 2.62, 2.40) are used.

The models `Model = 1,2,3,5,6` are based on Einstein's relation between diffusion coefficient and mobility.

`Model = 4`

The same as `Model = 3` but a modified mobility for electrons and a generalized Einstein relation is used. This model is suitable for devices with electrons as majority-carriers like GaAS-MESFET devices.

The general dependence of the carrier mobilities $\mu_{n,p}(T, D, E, M)$ on temperature T, carrier concentration D, electrical field E and material M, can be turned off, successively:

```
mobility( modell=5 amun4=0 amun5=0 amup4=0 amup5=0 egap=0 )
```

(no temperature dependence)

```
mobility( modell=5 gn=0 gp=0 evn=0 evp=0 amun4=0 amun5=0 amup4=0 amup5=0
egap=0 )
```

(completely no temperature dependence)

```
mobility( modell=5 ealph=1 vgrn=1e+30 vgrp=1e+30 )
```

(no dependence on electrical field)

```
mobility( modell=5 amun2=1e+30 amun3=0 amup2=1e+30 amup3=0 )
```

(no dependence on carrier concentration and/or dopants)

The dependence on the material M can be turned off, setting

```
mobility( modell=5 amunfa(1 1 1 1 1 1 1) amunfa(1 1 1 1 1 1 1) )
```

with `mreg` times the parameter 1 (here `mreg`= 7).

2.11.3 Mobility dependence on the temperature

The following formulas and the defaults are taken from [Sel84, S. 86, 4.1-20] formulas (2.35,2.38), [Sel84, S. 87/88, 4.1-24/25] formulas (2.52,2.36,2.39), [Sel84, S. 95, 4.1-48] formulas (2.34,2.37), and [Sel84, S. 96, 4.1-54] formulas (2.60,2.62).

$$N_i(T) = N_i \, T^{\frac{3}{2}} \, e^{-\frac{1}{2}E_g(1-T)} \tag{2.33}$$

$$\mu_n(T, D, E) \;=\; \frac{\mu_n(T, D)}{\left(1 + \left(\frac{\mu_n(T,D)\; E_n^{||}}{v_n^{\mathrm{sat}}}\right)^{\beta_n}\right)^{\frac{1}{\beta_n}}} \tag{2.34}$$

$$\mu_n(T, D) \;=\; \mu_n^{\min}(T) + \frac{\mu_n(T) - \mu_n^{\min}}{1 + \left(\frac{N_A + N_D}{c_n^{\mathrm{ref}}(T)}\right)^{\alpha_n}} \tag{2.35}$$

$$\mu_n(T) = \mu_n \, T^{-\gamma_n}, \;\; c_n^{\mathrm{ref}}(T) = c_n^{\mathrm{ref}} \, T^{-\kappa_n}, \;\; \mu_n^{\min}(T) = \mu_n^{\min} \, T^{-\delta_n} \tag{2.36}$$

$$\mu_p(T,D,E) = \frac{\mu_p^{\min}(T,D)}{\left(1+\left(\frac{\mu_p^{\min}(T,D)\ E_p^{||}}{v_p^{\text{sat}}}\right)^{\beta_p}\right)^{\frac{1}{\beta_p}}} \tag{2.37}$$

$$\mu_p(T,D) = \mu_p^{\min}(T) + \frac{\mu_p(T)-\mu_p^{\min}}{1+\left(\frac{N_A+N_D}{c_p^{\text{ref}}(T)}\right)^{\alpha_p}} \tag{2.38}$$

$$\mu_p(T) = \mu_p\ T^{-\gamma_p},\ c_p^{\text{ref}}(T) = c_p^{\text{ref}}\ T^{-\kappa_p},\ \mu_p^{\min}(T) = \mu_p^{\min}\ T^{-\delta_p} \tag{2.39}$$

Some values are fixed: $\beta_n = 2$, $\beta_p = 1$.

The saturation can depend on temperature:

$$v_n^{\text{sat}}(T) = v_n^{\text{sat}} \cdot T^{-e_n},\ v_p^{\text{sat}}(T) = v_p^{\text{sat}} \cdot T^{-e_p} \tag{2.40}$$

Model: `Model = 2`

$$N_i^{eff}(T) = \sqrt{N_c * N_v} * \exp(-E_g/(2kT)) \tag{2.41}$$

The densities in conduction N_c and valence N_v bands and the band gap E_g are calculated according to the formulas (see [Sel84], S.24-29):

$$N_c = 2*(6.28*kT*m_n/\hbar^2)^{3/2},\quad N_v = 2*(6.28*kT*m_p/\hbar^2)^{3/2}, \tag{2.42}$$

$$E_g = E_{g1} - E_{g2}\cdot T - E_{g3}\cdot T^2, \tag{2.43}$$

$$m_n = m_0\cdot(c_{n1}+c_{n2}\cdot T) \tag{2.44}$$

$$m_p = m_0\cdot(c_{p1}+c_{p2}\cdot T - c_{p3}\cdot T^2) \tag{2.45}$$

Here, m_n, m_p and m_0 are the effective electron mass, effective hole mass and electron rest mass. $\hbar$ is the Planck constant.

The dependence on temperature is considered according to the formulas (see [SCW+81] and [Sel84], S.82, 4.1-5/6).

$$\mu_n(T) = \left(\frac{T^{e_{n1}}}{f_{n1}} + \frac{T^{e_{n2}}}{f_{n2}}\right)^{-1} \tag{2.46}$$

$$\mu_p(T) = \left(\frac{T^{e_{p1}}}{f_{p1}} + \frac{T^{e_{p2}}}{f_{p2}}\right)^{-1} \tag{2.47}$$

2.11.4 Mobility dependence on dopants

N_A, N_D are the dopant concentrations.

Model: `Model = 2,3`

The formulas are taken from [Sel84], S.37, (2.4-65).

$$N_i^{eff}(T,D) = N_i^{eff}(T) \cdot \exp\left(\frac{v_1 \cdot a + \sqrt{a^2 + C}}{U_T}\right) \tag{2.48}$$

$$a = \log\frac{C_i}{x_{n0}} \tag{2.49}$$

$$C_i = N_D + N_A \tag{2.50}$$

$N_i^{eff}(T)$ depends on temperature like for `Model = 2`.

For the mobility, the following formulas, introduced from Arora et al (see [Sel84], S. 87/88, (4.1-24/25/29)), are used.

$$\mu_n(T,D) = \mu_n^{\min}(T) + \frac{\mu_n(T)}{1 + \frac{C_i}{c_n^{\text{ref}}(T)}} \tag{2.51}$$

$$\mu_p(T,D) = \mu_p^{\min}(T) + \frac{\mu_p(T)}{1 + \frac{C_i}{c_p^{\text{ref}}(T)}}, \tag{2.52}$$

$$C_i = D \cdot (N_D + N_A) + (1 - D) \cdot (n + p) \tag{2.53}$$

$$C_i = 0.667 \cdot (N_D + N_A) + 0.333 \cdot (n + p) \tag{2.54}$$

The last formulae is used in `Model = 6`

Model: `Model = 4,5,6`

Formulas introduced from Caughey and Thomas are used (see [Sel84] S.95, (4.1-48) and S.86, (4.1-20))

$$\mu_n(T,D) = \mu_n^{\min} + \frac{\mu_n - \mu_n^{\min}}{1 + \left(\frac{C_i}{c_n^{\text{ref}}}\right)^{\alpha_n}} \tag{2.55}$$

$$\mu_p(T,D) = \mu_p^{\min} + \frac{\mu_p - \mu_p^{\min}}{1 + \left(\frac{C_i}{c_p^{\text{ref}}}\right)^{\alpha_p}} \tag{2.56}$$

2.11.5 Mobility dependence on the electric field

$E_{n,p}^{||}$ and $E_{n,p}^{\perp}$ are the transversal and parallel projections of the electrical field of the electron (hole) current vector.

Model: `Model = 3,5`

The following formulas are used (see [Sel84], S.95, 4.1-48):

$$\mu_n(T,D,E) = \frac{\mu_n}{\sqrt{1+\left(\frac{\mu_n \cdot E_n^{||}}{v_n^{\text{sat}}}\right)^2}} \tag{2.57}$$

$$\mu_n(T,D,E) = \frac{\mu_p}{1+\mu_p * E_p^{||}/v_p^{\text{sat}}} \tag{2.58}$$

For `EALPH`> 0 (`Model = 3,5`), the dependence on the electrical field is considered in a different way, due to [Yam83]

$$\mu_n(T,D,E) = \frac{\mu_n(T,D)G_n}{\left(1+\frac{(a_n^{\text{c}})^2}{a_n^{\text{c}}+y_n}+(a_n^{\text{s}})^2\right)^{\frac{1}{2}}} \tag{2.59}$$

$$a_n^{\text{c}} = \frac{v_n^{\text{c}}}{y_n^{\text{c}}},\ a_n^{\text{s}} = \frac{v_n^{\text{c}}}{v_n^{\text{sat}}},\ v_n^{\text{c}} = \mu_n(T,D)\ G_n\ E_n^{||},\ G_n = \frac{1}{\left(1+\frac{E_n^{\perp}}{y_n^{\text{o}}}\right)^{\frac{1}{2}}} \tag{2.60}$$

$$\mu_p(T,D,E) = \frac{\mu_p(T,D)G_p}{\left(1+\frac{(a_p^{\text{c}})^2}{a_p^{\text{c}}+y_p}+(a_p^{\text{s}})^2\right)^{\frac{1}{2}}} \tag{2.61}$$

$$a_p^{\text{c}} = \frac{v_p^{\text{c}}}{y_p^{\text{c}}},\ a_p^{\text{s}} = \frac{v_p^{\text{c}}}{v_p^{\text{sat}}},\ v_p^{\text{c}} = \mu_p(T,D)\ G_p\ E_p^{||},\ G_p = \frac{1}{\left(1+\frac{E_p^{\perp}}{y_p^{\text{o}}}\right)^{\frac{1}{2}}} \tag{2.62}$$

Model: `Model = 4`

Electron mobility and diffusion coefficient D are calculated in the following way (see [YTK75]). The hole mobility does not depend on the electrical field. The Einstein condition is used. E is the absolute value of the electic field.

$$\mu_n(T,D,E) = \frac{\mu_n(T,D)+v_n^{\text{sat}}\frac{E^3}{E_{\text{crit}}^4}}{\left(1+\frac{E}{E_{\text{crit}}}\right)^4} \tag{2.63}$$

$$D = U_T \cdot \mu_n(T,D),\ \text{if } E < D_0, \tag{2.64}$$

$$D = \mu_n(T,D)\cdot(U_T + 2/3 * D_1 * E^2 \cdot \mu_n(T,D)),\ \text{if } E \geq D_0, \tag{2.65}$$

Model: `Model = 6`

A reduction of the mobility on the surface of the device is taken into account (see Selberherr, Schütz und Pützl, in Process and Device Simulation for MOS-VLSI Circuits, edi. by Antognetti et.al., The Hagü 1983, Martinus Nijhoff Publishers.)

$$\mu_n(T,D,E) = A_n/(1+(A_n * E_n^{||}/v_n^{\text{sat}}(T))^2)^{1/2}, \tag{2.66}$$
$$\mu_p(T,D,E) = A_p/(1+A_p * E_p^{||}/v_p^{\text{sat}}(T)). \tag{2.67}$$

with

$$v_n^{\text{sat}}(T) = v_n^{\text{sat}} T^{-e_n} \tag{2.68}$$
$$v_p^{\text{sat}}(T) = v_p^{\text{sat}} T^{-e_p} \tag{2.69}$$
$$A_n = B_n \cdot \mu_n(T,D) \tag{2.70}$$
$$A_p = B_p \cdot \mu_p(T,D) \tag{2.71}$$
$$B_n = (Y+Q_n)/(Y+(2+E_n^{\perp}/E_{n0}^{\perp})\cdot Q_n) \tag{2.72}$$
$$Q_n = y_{n0}/(1+E_n^{||}/E_{n0}^{||}), \tag{2.73}$$
$$B_p = (Y+Q_p)/(Y+(2+E_p^{\perp}/E_{p0}^{\perp})\cdot Q_p) \tag{2.74}$$
$$Q_p = y_{p0}/(1+E_p^{||}/E_{p0}^{||}), \tag{2.75}$$

2.11.6 Parameters

name	unit *type* option	default	comment
A1	*real*	1.35d-20	
A2	*real*	3.59d-18	
A3	*real*	2.86d-17	
ALN	*real*	0.125d0	
ALP	*real*	0.0317d0	
AMUN0	*real* [cm^2/(Vs)]	1030.d0	μ_n in (2.36)
AMUN1	*real*	0.72d0	α_n in (2.36) $\longrightarrow$ (2.35)
AMUN2	*real* [cm^{-3}]	8.5d16	c_n^{ref} in (2.36) $\longrightarrow$ (2.35)
AMUN3	*real* [cm^2/(Vs)]	65.d0	μ_n^{min} in (2.36) $\longrightarrow$ (2.35)
AMUN4	*real*	2.546d0	κ_n in (2.36)
AMUN5	*real*	0.57d0	δ_n in (2.36)
AMUNFA()	*real*	1.d0	$\mu_n^{\mathrm{mat}}(M)$ in (2.31), $M = 1, ...,$ mreg
AMUP0	*real* [cm^2/(Vs)]	495.d0	μ_p in (2.39)
AMUP1	*real*	0.76d0	α_p in (2.39) $\longrightarrow$ (2.38)
AMUP2	*real* [cm^{-3}]	6.3d16	c_p^{ref} in (2.39) $\longrightarrow$ (2.38)
AMUP3	*real* [cm^2/(Vs)]	47.7d0	μ_p^{min} in (2.39) $\longrightarrow$ (2.38)
AMUP4	*real*	2.546d0	κ_p in (2.39)
AMUP5	*real*	0.57d0	δ_p in (2.39)
AMUPFA()	*real*	1.d0	$\mu_p^{\mathrm{mat}}(M)$ in (2.32), $M = 1, ...,$ mreg
BETA	*real*	1.d0	
Bn	*real* [cm/s]	4.75d7	
Bp	*real* [cm/s]	9.925d6	

continued on next page

name	unit *type* option	default	comment
C	*real*	0.5d0	formula (2.48)
CN	*real*	1.74d5	
CN1	*real*	1.045d0	c_{n1} in formula (2.44)
CN2	*real*	4.5d-4	c_{n2} in formula (2.44)
CONstant	*integer*	1	
CP	*real*	8.842d5	
CP1	*real*	0.523d0	c_{p1} in formula (2.45)
CP2	*real*	1.4d-3	c_{p2} in formula (2.45)
CP3	*real*	1.48d-6	c_{p3} in formula (2.45)
DLN	*real*	5.82d14	
DLP	*real*	2.05d14	
DIFE0	*real*	0.	D_0 in formula (2.64)
DIFTau	*real* [s]	1E-13	D_1 in formulas (2.64, 2.65)
DOTKOM	*real*	1.d0	D in formula (2.53)
Ealph	*real*	1.d0	toggel `Model = 3,5` see formulas (2.57 – 2.62)
EG1	*real* [V]	1.1785d0	e_{g1} in formula (2.43)
EG2	*real* [V]	9.025d-5	e_{g2} in formula (2.43)
EG3	*real* [V]	3.05d-7	e_{g3} in formula (2.43)
EGAP	*real* [V]	1.12d0	E_g in (2.52)
EKrit	*real* [V/cm]	4d3	E_{crit} in formula (2.63)
EN1	*real*	1.5d0	e_{n1} in formula (2.46)
EN2	*real*	3.13d0	e_{n2} in formula (2.46)
EP0N	*real* [V/cm]	1.E4	$E_{n0}^{\parallel}$ in formula (2.73)
EP0P	*real* [V/cm]	8.E3	$E_{p0}^{\parallel}$ in formula (2.75)
EP1	*real*	1.5d0	e_{p1} in formula (2.47)
EP2	*real*	3.25d0	e_{p2} in formula (2.47)
ETON	*real* [V/cm]	1.8E5	$E_{n0}^{\perp}$ in formula (2.72)

continued on next page

name	unit *type* option	default	comment
ETOP	*real* [V/cm]	3.8E5	$E_{p0}^{\perp}$ in formula (2.74)
EVN	*real*	0.87d0	e_n in formula (2.40)
EVP	*real*	0.52d0	e_p in formula (2.40)
FN1	*real* [cm^2/(Vs)]	4195.d0	f_{n1} in formula (2.46)
FN2	*real* [cm^2/(Vs)]	2153.d0	f_{n2} in formula (2.46)
FP1	*real* [cm^2/(Vs)]	2502.d0	f_{p1} in formula (2.47)
FP2	*real* [cm^2/(Vs)]	591.d0	f_{p2} in formula (2.47)
GAMMA	*real*	0.d0	
General	*integer*	3	
GN	*real*	2.33d0	γ_n in (2.36)
GP	*real*	2.23d0	γ_p in (2.39)
IFELD	*integer*	1	
MESFET	*integer*	4	
Modell	*integer*	5	number of model
Temp	*integer*	2	
T300K	*integer*	5	
V1	*real* [V]	9d-3	v_1 in formula (2.48)
VGRN	*real* [cm/s]	1.d7	v_n^{sat} in (2.60) $\longrightarrow$ (2.59)
VGRP	*real* [cm/s]	8.37d6	v_p^{sat} in (2.62) $\longrightarrow$ (2.61)
XN0	*real* [cm^{-3}]	1.d17	x_{n0} in formula (2.49)
YON	*real* [cm]	5.E-7	y_{n0} in formula (2.73)
YOP	*real* [cm]	4.E-7	y_{p0} in formula (2.75)
YETON	*real* [V/cm]	6.493d4	y_n° in (2.60) $\longrightarrow$ (2.59)
YETOP	*real* [V/cm]	1.869d4	y_p° in (2.62) $\longrightarrow$ (2.61)
YGN	*real*	8.8d0	y_n in (2.60) $\longrightarrow$ (2.59)

continued on next page

name	unit *type* option	default	comment
YGP	*real*	1.6d0	y_p in (2.62) $\longrightarrow$ (2.61)
YVCN	*real* [cm/s]	4.9d6	y_n^c in (2.60) $\longrightarrow$ (2.59)
YVCP	*real* [cm/s]	2.928d6	y_p^c in (2.62) $\longrightarrow$ (2.61)

2.12 Numeric command

The NUmeric-command is used to define the accuracy and termination parameters required for the numerical calculations.

2.12.1 Some comments on the numerical methods

A Gummel-Iteration U_{i+1} with a current J_{i+1} is accepted as solution, if the following conditions are fullfilled:

$$\begin{aligned} \mathrm{dist}\big(U_{i-1}, U_i\big) + \mathrm{dist}\big(U_i, U_{i+1}\big) &\leq 2 \cdot \texttt{EPGUAB}, && (2.76)\\ \mathrm{dist}\big(J_i, J_{i+1}\big) &\leq |J_{i+1}| \cdot \texttt{EPCURE}. && (2.77) \end{aligned}$$

A Newton-Iteration U_i is accepted as solution of the nonlinear Poisson equation, if the following condition is fullfilled:

$$\mathrm{dist}\big(U_{i-1}, U_i\big) \leq \texttt{EPPOAB}. \qquad (2.78)$$

A CG-Iteration U_{jk} is accepted as $(j+1)$-th approximation of the solution of the nonlinear Poisson equation, if the following condition is fullfilled:

$$\mathrm{dist}\big(U_{jk-1}, U_{jk}\big) \leq \mathrm{dist}\big(U_{i-1}, U_i\big) \cdot \texttt{EPPORE}. \qquad (2.79)$$

Moreover, EPPORE is the relative termination constant for the Jacobi-Iteration of the continuity equations.

For transient calculations EPPOAB and EPPORE are used as termination constant if MOCKEU > 1.

A Jacobi-Iteration N_{jk} is accepted as $(j+1)$-th Gummel-approximation of the electron density N_{j+1} if

$$\mathrm{def}_n\big(N_{jk}\big) \leq \mathrm{def}_n\big(N_j\big) \cdot \texttt{EPCURE}. \qquad (2.80)$$

Here $\mathrm{def}_n(\cdot)$ is the defect of the continuity equation for electrons (similar for the hole equation).

The embedding method for the calculation of a UI-characteristic-line is controlled in the following way: The euclidian norm $|\cdot|$ of a new working point A_{i+1} has to satisfy

$$|A_{i+1} - A_i| \leq \frac{|A_{i+1} - A_i|}{F(x_i, x_{i-1})} \cdot \texttt{FISTEP} \tag{2.81}$$

Here $x_i = (U_i, N_i, P_i)$ is the solution vector and F the free energy functional

$$F(X,x) = \frac{1}{2}\int\Big(\varepsilon|\nabla(U-u)|^2 + kT\Big((N-n)\log N/n + (P-p)\log P/p\Big)\Big)\,dx \tag{2.82}$$

For transient calculations the new time step δ_n is calculated from the old time step δ_o according to the relation

$$\delta_n \leq \delta_o \sqrt{\frac{\texttt{EPSTEP}}{F(X(t), X(t-\delta_o))}} \tag{2.83}$$

Here $X(t) = (U(t), N(t), P(t))$ is the solution vector at time t and F the free energy functional.

A calculation is switched from Gummel- to Newton-iterations, if the following conditions are fullfilled:

$$test_j \leq \texttt{SNEWT}, \tag{2.84}$$
$$test_j \cdot \texttt{GUMNEW} \leq test_{j-1}, \tag{2.85}$$
$$def_j \cdot \texttt{GUMNEW} \leq def_{j-1}. \tag{2.86}$$

We have a cancellation threshold `OMItnp` for the continuity equations:

$$\mathrm{def}_n \leq \mathrm{def}_n \cdot \texttt{OMItnp}, \tag{2.87}$$
$$\mathrm{def}_p \leq \mathrm{def}_p \cdot \texttt{OMItnp}. \tag{2.88}$$

2.12.2 Parameters

name	unit *type* option	default	comment
SFActo	*real*	1.d0	
AZEr	*real*	1.d0	current splines at boundary
CUitre	*real*	10.d0	
CURnul	*real*	1.d-16	zero current threshold

continued on next page

name	unit *type* option	default	comment
Damp0	*real*	1.d-10	Initial value for the regularisation parameter Damp (improvement of the conditioning of the current matrices)
DELzer	*real*	2.0d0	current splines at boundary
EPCure	*real*	0.001d0	termination constant, see (2.77,2.80)
EPGUAB	*real*	0.01d0	termination constant, see (2.76)
EPPOAB	*real*	0.01d0	termination constant, see (2.78)
EPPORE	*real*	0.01d0	termination constant, see (2.79)
EPSIL	*real*		ε in formula (2.82)
EPStep	*real*	0.5d0	Control value for the instationary (transient) case, see (2.83)
FISTEP	*real*	1.d10	Control value for the stationary case, see (2.81)
GUMnew	*real*	1.5d0	Threshold value for the transition from Gummel- to Newton method, see (2.85,2.86)
IANEW	*integer*	0	
IGUMAX	*integer*	100	
IRAND	*integer*	0	
IRed	*integer*	5	Maximum number of bisection for the Newton method.
ITnew	*integer*	20	Maximum number of block iterations for the Newton method and the AC analysis.
OMItnp	*real*	0.001d0	Threshold value for the cancellation of the n- and p equations, see (2.87,2.88)

continued on next page

name	unit *type* option	default	comment
PARdiso	*integer*	0	switch on Pardiso
POFAK	*real*	1.d0	
RELax	*real*	1.d0	SOR parameter for the AC analysis.
SNEwt	*real*	3.0d0	Threshold value for the transition to Newton's method, see (2.84)
TEPot	*real*	2.d0	

2.13 Control and Replace command

The record parameter CONTrol is used for general control purposes in particular for the grid adaptation. The parameters can be specified in the REPLace command in the normal command input mode and in the TControl mode

```
REPLace(CONTrol(name=value))
```

They can be specified in most of the process steps locally

```
GRID(CONTrol(MAXtrl=2)).
```

First all parameters declared in the REPLace command are reset to their default values. Then the required modifcations are done and the modified values are stored as new default values. Note! When using the REPLace command in the TControl mode local changes in the process step are lost In the REPLace-command, the CONTrol record and the variable parameters VARiable(...) can be specified.

The command ADAPtation() can be used to force a readaptation of the grid In the TControl mode a readaptation can be required at the end of the current step by REPLace(CONTrol(LADA=1)). The ADAPtation command can not be used in the TControl mode.

2.13.1 Parameters

parameter name	unit *type* options	default value	comment
VAroutswitch	*integer*		Selects the variables for the default print of integrals and extremal values. For each variable ivar there is defined a parameter varout(ivar). For LPRot=2 the extremal values and integrals of variables with varout(iva)≥ varoutswitch are printed each time step.
NPrint	*integer*	undefined	Number of time steps after which the last Print command(s) are repeated. If undefined or < 0, no output is done, otherwise also at the end of each process step.
NGraphic	*integer*	undefined	Number of simulation steps after which a new picture is drawn. If undefined or < 0, no output is done, otherwise also at the end of each simulation step.
NSAve	*integer*	undefined	Number of simulation steps after which a save file is written. If undefined, no automatical saves are done for < 0. Even explicit save commands are ignored. For ≥ 0 a save file is written at the end of each simulation step.
Saveeach	Time	undefined	Defines a time (human time, neither CPU nor simulated processing time) after which a save file is written.

continued on next page

name	unit *type* option	default	comment
SName	*string*80*		Save file name. Used for the automatically saved files. A counter is added to the file name, and the file name is written to terminal and protocol.
SIndex	*integer*	−2	Defines the index of the first automatically saved file if ≥ 0. If undefined no index is added. If −1, the index of a loaded file is incremented by 1 and used for the first automatically saved file. If −2, the automatical save toggles between indices −1 and −2. This is the default.
MAXV	*integer*	undefined	Maximum number of nodes during grid adaptation. Specify this value rather than MAXVDelaunay to limit the mesh size.
MPOINTS	*integer*	0	MPOINTS integer Maximum number of points in the layer system. Internally increased, if necessary.
MAXVDelaunay	*integer*	undefined	Maximum number of nodes in the final simulation grid. Contains all mesh points in triangles and line segments. If specified, delaunization of the mesh might be incomplete.
IOU	*integer*	6	Terminal output channel. In batch mode the only output channel.

continued on next page

name	unit *type* option	default	comment
IPU	*integer*	4	Output channel for the protocol. For IPU no protocol file is written.
PNV1	*integer*	10000	First node to be printed in a list.
PNV2	*integer*	1	Last node to be printed in a list.
PNT1	*integer*	10000	First triangle to be printed in a list.
PNT2	*integer*	1	Last triangle to be printed in a list.
AR1	*integer*	10000	First area in a Print command.
AR2	*integer*	1	Last area in a Print command.
LI1	*integer*	10000	First line in a Print command.
LI2	*integer*	1	Last line in a Print command.
ITYPhc	*integer*		Type of "honeycomb" for each of the nodes.
	1		"Honeycomb" defined by lines connecting triangle vertex and midpoint of opposite edge.
	2		Voronoi cells.
	3		Voronoi cells in cylindrical coordinates, for rotational symmetric 3D situations. X-axis: radius. $X = 0$ origin of cylin der coordinates. Y-axis height.
LPRot	*integer*	0	Length of the protocol file (0/1/2)

continued on next page

name	unit *type* option	default	comment
XX	*real*	1	Coefficient for coordinate transformations (input and output) e.g. load of external doping and grid. $x_{foreign} =$ xx $\cdot\ x_{dios} +$ xy $\cdot\ y_{dios} +$ x0 $y_{foreign} =$ yx $\cdot\, x_{dios} +$ yy $\cdot\, y_{dios} +$ y0
XY	*real*	0	
X0	Length	undefined	
YX	*real*	0	
YY	*real*	-1 (!!!)	
Y0	Length	undefined	
MASS	*integer*	2	Selection of triangle measure in trmas.f
	1		Arithmetic average of the gradients of the logarithm of the concentrations along the triangle edges.
	2		Maximum of the gradients of the logarithm of the concentrations along the triangle edges This is the default.
	6		Difference of the logarithms (ashsur) of the concentrations along the triangle edges.
INFO	*integer*	0	General control of the terminal output. In particular, used to print model parameters and for test prints.
MAXVFL	*integer*	0	Maximum number of refinement loops per refinement criterion. Internal default 2·MAXTRl.

continued on next page

name	unit *type* option	default	comment
IPGRID	*integer*	0	Draw the grid during the refinement. 1: Draw the nested ITRI-grid at the end of adaptation. 2: Draw all ITRI-grids during adaptation. 3: Draw each UTRI-grid during adaptation and at any change of the layer system. >3: Draw each ITRI- and each UTRI- grid.
IVERADA	Percent	10%	Fraction of "bad" triangles that is allowed without readaptation. If more triangles ar found to be "bad" the grid is readapted.
AUTOada	*integer*	1	
	−1		Turns off the automatic readaptation of the grid.
	1		Turns on the automatic readaptation of the grid.
	0		Automatic readaptation of the grid is turned off internally since the refinement with respect to MARKM was not finished.
MAXTRl	*integer*	4	Maximum triangle level (only triangles of the MAXTRl-1 generation or a lower generation can be marked for refinement).

continued on next page

name	unit *type* option	default	comment
TRl()	*record*		For each of the materials a separate MAXTRl value can be defined. TRL(SI=3,OX=5) All materials that are not specified receive the global value. **Note!** The refinement criteria are limited by the maximum of the global value and the values for the materials.
DX	Length	μ	The smallest triangle might be defined by a length. From the length MAXTRl is determined.
DXL()	*record*		For each of the materials a separate DX can be prescribed. From this TRl is determined.
MARKH	*integer*	0	Parameter for the homogeneous refinement.
	> 0		Number of refinement loops.
	< 0		Refine only triangles with a triangle level $< -$ MARKH.
MARKM	*integer*	-4	Parameter for the refinement of doping gradients
	> 0		Number of refinement loops.
	< 0		Refine only triangles with a triangle level $< -$ MARKM.
MARKGS	*integer*	1	parameter for the refinement of "green stars". 0: off, 1:on **Note!** 1 may require a lot of refinement loops.
REC1(	*record*		First rectangle used for the refinement.

continued on next page

name	unit *type* option	default	comment
Markr	*integer*	-4	>0 Number of refinement loops. <0 Refine only triangles with a triangle level $\leq$ Markr.
XLeft	Length [μm]	0.	Left boundary.
XRight	Length [μm]	-1	Right boundary.
YBottom	Length [μm]	0.	Bottom boundary.
YTop	Length [μm]	-1	Top boundary.
Icoswitch	Geometrical Physical	Geometrical	Type of coordinates.
MAXtrl			Maximum triangle level in the rectangle. The grid remains coarse even if a refinement criterion is met.
)			End of record REC.
			The rectangles 2,3,...,10 can be specified similarly. By default for these rectangles Icoswitch = Physical.
MARKG	*integer*	-5	Parameter for the refinement at material interfaces
	>0		Number of refinement loops.
	<0		Refine only triangles with a triangle level $<-$MARKG.
			Note! For a very coarse grid several interfaces may intersect a triangle edge. The boundary type might not be defined properly from the materials of the nodes.

continued on next page

name	unit *type* option	default	comment
MARKP	*integer*	-6	Parameter used for the refinement in the vicinity of certain points. Vertices and steps in lines of the layer system are defined internally and used in the refinement.
	> 0		Number of refinement loops.
	< 0		Refine only triangles with a triangle level $< -$ MARKP.
POInts()	*record* [μm]		Vector of $x-$ and $y-$ coordinates of $\leq$ 100 points to be marked in MARKP. Vertices and points with large curvature are defined in the layer system and added to the user-defined list.
MARKGNR	*record*	2	List of boundary types used for the boundary refinement. For the readaptation before a WIAS-TeSCA simulation, the contact numbers have to be specified: -1...-ndiri for metal contacts and 1...nnatur for gate contacts. The level of refinement is controlled by MARKG.
MARKJ	*integer*	-5	Refinement at p-n-junctions.
	> 0		Number of refinement loops.
	< 0		Refine only triangles with a triangle level $< -$MARKJ.
MARKL	*integer*	1	Refinement of triangles in front of doping fronts.
	> 0		Number of refinement loops.
	< 0		Refine only triangles with a triangle level MARKL.

continued on next page

name	unit *type* option	default	comment
MARKMAX	*integer*	-4	Refinement at local dopant maximum. To prevent a homogeneous refinement, only functions with minimum variation are handled: varmax > CMAMI · varmin
	> 0		Number of refinement loops.
	< 0		Refine only triangles with a triangle level $< -$MARKMAX.
MARKI	*integer*	0	Refinement of triangles with change of the curvature of the doping (concave-convex).
	> 0		Number of refinement loops.
	< 0		Refine only triangles with a triangle level $< -$ MARKI.
1D	*boolean*	off	Automatical grid adaptation during a 1D simulation for 1D=on. The grid is adapted as usual according to the refinement criteria. After refinement the $y-$coordinates at the left side of the refined grid are used to construct a 1D grid.
ANGstroem	Length [μm]	1e-4	Internal test length.
IVER()	*record*		For each of the variables var a critical triangle measure IVER(var) (in %) is defined such that the triangle is refined if the limit is exceeded *measure*(var,triangle)> IVER(var)

continued on next page

name	unit *type* option	default	comment
ASHdiff	*real*	1.	Reference value for the differences of ashsur(conc) from MASS=6. A triangle is refined if $\frac{\text{ashsur(c1)}-\text{ashsur(c2)}}{\text{ASHdiff}} >$ IVER/100.
RESM	*real*	1.	"Reserve" factor for the triangle size. Prevents from frequently readaptation. If the area of a triangle k exceeds RESM$\cdot a_m$ the triangle k has to be checked for the grid quality. Here a_m denotes the area of the largest triangle that would be refined by MARKM right after the grid adaptation.
RESJ	*real*	1.5	"Reserve" factor for the triangle size for MARKJ.
RESI	*real*	1.5	"Reserve" factor for the triangle size for MARKI.
CMAMI	*real*	1000.	MARKMAX is applied only to functions with a minimum variation: varmax > CMAMI$\cdot$ varmin.
IMAX()	Percent [%]	90	For each of the variables var a maximum value IMAX(var) in % is defined. If the concentration exceeds the value for one of the nodes in the triangle, i.e., value(var,node(triangle)) > varmin$+\frac{\text{IMAX(var)}}{100}$(varmax – varmin), the triangle is refined.

continued on next page

name	unit *type* option	default	comment
CMAX	*record* $[cm^3]$	1e23	For each of the variables var a maximum value CMAX(var) is defined. If the concentration exceeds the value for one of the nodes in the triangle, i.e., value(var,node(triangle)) > CMAX(var) the triangle is refined.
LADA	*integer*	1	Turns on the grid readaptation. May be defined in the TControl mode. At the end of the current simulation step the grid is readapted.
REAdapt	*integer*	1	Controls complete readaptation (= 1) or further refinement (= 0) of the grid.
RETriangulate	*integer*	1	Selection of the user grid before a readaptation.
	1		Use the previous coarse grid.
	2		Use the old geometrical grid.
	3		Construct a new user grid using the last defined GRID parameters.
MOvtrans	No Extrema Integral EPIintegral	EPIintegral	Selection of the rescaling type of the dopants after a vertical grid transformation, i.e. compute integral and extrema before the vertical transformation and rescale the profiles to restore minimum/maximum or the minimum/integral after the transformation. By default restore minimum/integral only during epitaxy.

continued on next page

name	unit *type* option	default	comment
STCenter	*integer*	5	Number of simulation steps after which the grid is centered.
BFLip	*integer*	20	Number of edge flipping loops before a grid centering.
CEnter	*integer*	2	Number of grid centering loops.
EFLip	*integer*	20	Number of edge flipping loops after a grid centering.
AFLip	Angle [°]	115	Maximum angle that is allowed after the edges are flipped. If the edge flipping would lead to larger (but compensated) angles the edges are not flipped since otherwise a "hole" in the grid would be created.
EPSFLip	*real*	0.99	Factor to multiply the radii of surrounding spheres for detecting non-Delaunay situations. Must be ≤ 1. Used to avoid instabilities arising from rounding errors.
CTriangle	Edges Wedges Triangle WTriangl No	No	selection of centering method, recommended No or Triangle i.e. move the grid points into the center of masses of the adjacent triangles.
CBoundary	No WEdges Mid Xloc WXloc	No	Selection of a centering method for boundary triangles.
LCSTeps	*integer*	3	Number of local corrections after a grid transformation.
LCEnter	*integer*	1	Number of local centering steps in each of the correction loops.

continued on next page

name	unit *type* option	default	comment
LAtriangle	*integer*	2	Number of neighboring triangle shells, added for the next centering step, if the local correction failed.
LACenter	*integer*	3	Number of added centering loops, if neighboring triangle shells have to be added.
LCTriangle	Edges Wedges Triangle WTriangl No	Triangle	Selection of the local centering method.
CBulk	*integer*	2000	Maximum number of edges incident into a bulk node of the mesh. If the number is exceeded, an additional subdivision is done.
CInterface	*integer*	2000	Maximum number of edges incident into an interface node of the mesh. If the number is exceeded, an additional subdivision is done.

2.14 Substrate command

The SUBStrate command is used to initialize the layer system.

The location of the layer system in the X- and Y-direction can be defined (XLeft,XRight,YBottom,YTop). In addition also the initial position of the substrate surface YSubs can be prescribed. If these values are not specified the values are copied from the GRID command.

2.14.1 Parameters

name	unit *type* option	default	comment
YSubs	Length [μm]	0.0d0	initial position of the substrate surface
XLeft	Length [μm]	0.0d0	left end of the substrat region
XRight	Length [μm]	0.0d0	right end of the substrat region
YBottom	Length [μm]	0.0d0	bottom of the substrat region
YTop	Length [μm]	0.0d0	Initial top position of the entire layer system

2.15 Special command

The SPECIAL command is used for physical effects not covered by the standard model. This includes hot carrier injection models for gate leakage and the generation of charge by the trace of an incident particle. The latter can be used as a simple model describing the Lambert–Beer absorption in solar cells.

2.15.1 Parameters

name	unit *type* option	default	comment
Aisf	*real*	-0.0287d0	
AE	*real*	1.d0	
ALpha	*real*	3d4	
ALPHA1	*real* [cm]	1d3	
ALPHA2	*real* [cm]	0.d0	
ALPHAL	*real* [cm]	0.d0	
Bisf	*real*	8.4933d0	
ELAM	*real* [V]	0.2d0	
EMN	*real*	0.5d0	Eff. electron mass of hot electrons
EMP	*real*	0.5d0	Eff. holes mass of hot electron

continued on next page

name	unit *type* option	default	comment
EREDOX	*real* [V]	0.d0	
FIN1	*real* [V]	3.1d0	hot electron, hight of barrier
FIN2	*real*	2.6d-4	hot electron, fitting parameter
FIN3	*real*	1.d-5	hot electron, fitting parameter
FIP1	*real* [V]	4.8d0	hot electron, hight of barrier
HOTAUN	*real* [s]	0.d0	hot electron, relaxation time
HOTAUP	*real* [s]	8.d-14	hot electron, relaxation time
INJE	*integer*	0	switch, hot electron
IISFET	*integer*	0	
IOPTical	*integer*	0	optical radiation
ISTOER	*integer*	0	switch, Schöll model
ITRack	*integer*	0	switch, track model
OXLAM	*real* [cm]	3.4d-7	hot electron, free path length in oxid
PHisf	*real*	0.0d0	
PHOisf	*real*	5.2d0	
PKN	*real*	0.d0	
PKP	*real*	0.d0	
QTRack	*real*	0.d0	generated charge
Sisf	*real* [V]	0.045d0	
SIGMA	*real*	0.d0	capture cross section
STreu	*real* [s]	1.d-9	
T1	*real*	-1.26d-5	
T2	*real*	0.2d0	
T3	*real*	-0.254d0	
T4	*real*	1.82d-5	
T5	*real*	0.421d0	
T6	*real*	-0.887d0	

continued on next page

name	unit *type* option	default	comment
TMean	*real* [s]	1.d-9	
TN0	*real* [cm^{-3}]	3.125d18	track density
TS	*real*	7.21d7	
WS	*real* [cm]	1.d-3	
WTRack	*real* [cm]	0.1d-4	track width
X1	*real*	8.26d-6	
X1S	*real*	4.40d-4	
X2	*real*	11.3d0	
X2S	*real*	3.72d0	
X3	*real*	-0.745d0	
X3S	*real*	-0.66d0	
XF1	*real* [cm]	0.d0	track limiting
XF2	*real* [cm]	0.d0	track limiting
XN1S	*real* [cm^{-3}]	9.48d13	
XN2S	*real* [cm^{-3}]	3.01d10	
XNS	*real* [cm^{-3}]	1.24d11	
XS	*real*	1d-3	
XS1	*real*	1.4d-6	
Y0	*real* [cm]	0.0d-7	track model
Y1	*real* [cm]	3.0d-7	track model
YF1	*real* [cm]	0.d0	track limiting
YF2	*real* [cm]	0.d0	track limiting

2.16 Save command

The SAVE command is used, to write output files for subsequent evaluation, continuation of the simulation (.dmp*) and for offline coupling to other simulation tools, respectively.

```
SAVE (File=xxx, TYPe=dmp, exp, prf, plf, dmp.gz, bound, dp, cmd,
 geb, mdraw, dmp.Z, dom, USer, ITri, Picasso, MESHDp, lay, lai,
```

KPIF))

By default a binary WIAS-TeSCA save file is written. All the other supported file types can be derived from a .dmp file, after loading it into WIAS-TeSCA.

Geometry Description

The geometry description in the xxx.rand file contains the polygons that define regions and contacts. A MATerial() list or an Arealist() can be specified to select regions. By default all WIAS-TeSCA regions are selected.

The WIAS-TeSCA material names are "translated" into DATEX material names.

Contact Definition

Up to 20 contacts for device simulation can be defined in the data record Contacts(Contact1(name= ,x= ,y= ,xe= ,ye=)...).

If all parameters for a contact are specified, a list of line segments is defined and added into the .rand file as a contact region. The line in the WIAS-TeSCA layer system, closest to the two specified points is determined. The two WIAS-TeSCA points closest to the specified start and end of the contact are kept but all line segments in between are smoothed in the usual way. After smoothing the line segments between the two points are defined as contact. All line segments of one contact are on the same DIOS line.

If xe, ye remain undefined for a contact, one of the WIAS-TeSCA regions is renamed. The approximate position x, y of the midpoint of the bounding box of the region can be specified to select the region.

If no midpoint is specified regions are renamed from right to left. Renaming is applied only to regions, which consist of a material, that has been redefined in Synonyms(...) to appear as Metal in the file.

The contact definition is assumed to support simple standard cases only.

The defined line contacts cannot be displayed in WIAS-TeSCA. The modified region names are kept in WIAS-TeSCA. The WIAS-TeSCA material of a region is not affected by Synonyms(...).

Command File

A command file xxx.cmd for mdraw is written, which refers to the WIAS-TeSCA simulation grid and doping file and which contains refinement data.

Global refinement parameters MaxElementSize, MinElementSize, MaxTransDiff and MaxAspectRatio can be specified in the WIAS-TeSCA inputfile.

Polysilicon regions (in contact to oxide but not to silicon) are treated as "gate" contacts. The parameters in the Gate record are used to select and define the refinement at these "gates". The parameters VerticalSmooth and LateralSmooth

are used to select the relevant parts of the material interfaces. No smoothing of the boundaries is applied near the gates.

In addition, there are defined refinement regions in the silicon underneath the "gates". These lateral extension of regions can be modified with the parameter `LateralRefine`. The `MaxAspectRatio` and a series of vertical stepsizes `MaxElementHeight` and `NumberOfIntervals` can be specified to generate a graded fine grid in the silicon. If `VerticalSmooth=undefined` is specified, "gate" refinement is turned off.

Grid And Doping

The `WIAS-TeSCA` simulation grid is saved in `xxx_dios.geo` and the doping functions are saved in `xxx_dios.dop` file. Both files are compressed by default. The variables in the `.dop` file can be selected with `SPecies(...)`. By default, the net doping and the total doping of the several dopants are saved.

name	unit *type* option	default	comment
FILE	*strin*80*		Name of the save file, the default file extensions are added internally. If the default extension is specified, the file type is defined from the extension and can be omitted.
Type	*record*	dmp.z	Type of output file, that has to be written. Several files are saved by specifying more than one type.
SPecies()	*record*		Names of `WIAS-TeSCA` variables, that have to be written into the file.
MATerial()	*record*		Names of `WIAS-TeSCA` materials which should be used in the output.
FNET	*real*	1	Net doping is divided by `FNET` if it is written into the file. In `DIOS : Net = donator-acceptor`

continued on next page

name	unit *type* option	default	comment
FORMat	*integer*	0	For TYPe=user, itri, picasso, dmp, formatted or unformatted files can be written. For TYPe=mdraw, the following values of FORmat can be used: 0: DF-ISE, 1: DATEX binary, 2: DATEX text compressed, 3: DATEX portable 5: DATEX text gzip
APPend	*boolean*	on	Append a snapshot to an existing file or replace the file. Only for TYPe=plf.
DXproeth	*real* [μm]	0.05	Lateral step size for PROETH doping file.
DYproeth	*real* [μm]	0.05	Vertical step size for PROETH doping file.
Xproeth	*integer*	undefined	Number of lateral discretization points for PROETH doping file.
Yproeth	*integer*	undefined	Number of vertical discretization points for PROETH doping file.
EPSEq	*real* [μm]	2.e-3	Minimum distance of two points in .rand
EPSLoc	*real* [μm]	1.5e-3	Minimum local y-coordinate in .rand
DISTmin	*real* [μm]	3.e-3	Minimum distance for subdivision of edges with small slope.
EPSAngle	Angle	3.degree	Smallest slope in .rand. Edges with smaller slopes are subdivided or moved.
MINAngle	Angle	5.degree	Angle, achieved in the subdivision.

continued on next page

name	unit *type* option	default	comment
EPSX	*real* [μm]	0	Minimum lateral distance of any 2 points in .rand
EPSY	*real* [μm]	0	Minimum vertical distance of any 2 points in .rand
Cutline(	*record*		Start and end point of the simulation area in the layout. This is used for the correct placement of the WIAS-TeSCA simulation area in the DF-ISE coordinate system.
AREAlist(	*record*		List of selected area numbers.
IMAG	*boolean*	off	Recompute imaginary lines in the WIAS-TeSCA layer system before saving them to .rand or .bound.
SYnonyms(	*record*		List of names for each WIAS-TeSCA material.
MaxElementSize	*real* [μm]	1	MaxElementSize in .cmd.
MinElementSize	*real* [μm]	0.02	MinElementSize in .cmd.
MaxTransDiff	*real*	1.	AsinhDifference resp. MaxTransDiff in .cmd.
MaxAspectRatio	*real*	30	MaxAspectRatio in .cmd
Gate	*record*		data record for defining gate regions.
VerticalSmooth	*real* [μm]	0.1	additional vertical range to detect gate.
LateralSmooth	*real* [μm]	0.01	additional lateral range to detect gate.
LateralRefine	*real* [nm]	0	offset of the refinement region with respect to the gate contact.

continued on next page

name	unit *type* option	default	comment
MaxElementHeight	*record* [μm]		record of required element heights underneath the gates
NumberOfIntervals	*integer*		Number of small intervals for each of the MaxElementHeight values
MaxAspectRatio	*real*	60	MaxAspectRatio underneath the gate.
)			End of record Gate.
DontSmooth	*record* [μm]		Record of up to 20 user defined bounding boxes, where no smoothing of the interfaces is applied. DontSmooth(Box1(X=,Y= ,XE= ,YE=),...)
Contacts	*record*		data record for defining contacts ($\leq$ 20)
Contact1	*record*		data record for the first contact
Name	*string*24*		Contact name.
X	*real* [μm]		Lateral position of the begin of the contact. If X, Y, XE, YE are specified, line segments are defined for the contact. If X, Y are specified, the Metal region, containing the point is renamed. If X, Y, XE, YE are undefined, the regions are renamed from right to left.
Y	*real* [μm]		Vertical position of the begin of the contact
XE	*real* [μm]		Lateral position of the end of the contact
YE	*real* [μm]		Vertical position of the end of the contact

continued on next page

name	unit *type* option	default	comment
)			End of record Contact1.
)			End of record Contacts.
DOTrect	*boolean*	on	Switches between doping on a tensor product grid or isolines in .geb.
LMAX	*real* [μm]		Maximum triangle edge in .geb.
LMIN	*real* [μm]		Minimum triangle edge in .geb.
Levels()	*record*		Up to 10 levels for isolines of net doping in .geb.
BC(			Data record for the definition of ≤ 15 contacts in .geb.
TYPE1	*integer*		Boundary condition type of the first contact in .geb.
XB1	*real* [μm]		Lateral position of the start of the first contact in .geb.
YB1	*real* [μm]		Vertical position of the start of the first contact in .geb.
XE1	*real* [μm]		Lateral position of the end of the first contact in .geb.
YE1	*real* [μm]		Vertical position of the end of the first contact in .geb.
)			End of record BC.

2.17 Step command

The STep-command is used to define the bias and the step control parameters.

For transient (time depending) calculations it is possible to enter time intervalls.

Executing the STep-command the solution of the equations are calculated, printed (to the terminal) and saved.

2.17.1 Comments

We call a "working point" the user given voltages on the Dirichlet (i.e., ohmic and Schottky contacts) and the gate contacts. Between the working points the embedding method adds additional operating points automaticly and more or less regularly.

Executing the `STep`-command the solution of the equations – the potential and the electron- and hole densities – are calculated and can be printed (to the terminal) and saved. Moreover, some more quantities (e.g., the contact and recombination currents and others) are calculated and can be printed/plotted.

If an ac-analysis frequency `OMega()` is given, a resistance and capacity matrix are calculated.

A typical `WIAS-TeSCA` output on terminal or in the log file looks like this

```
-----------------------------------------------------------------------
***** BIAS-point:    1 **  Step:      4 ** Time:      0.00000E+00
contact   voltage/ V  current/ A  **  contact   voltage/ V  current/ A
contact   0.0000E+00  -3.13474E-05  substrat  0.0000E+00   3.13475E-05
I-Shockley-Read-Hall*  I-Auger  *I-Avalanche*  I-surface*
           2.069E-25   0.000E+00   0.000E+00   0.000E+00
  I-Reabs   *   I-Korro   *   I-Tunnel  *   I-Photo
  0.00000E+00   0.00000E+00   0.00000E+00   0.00000E+00
-----------------------------------------------------------------------
N-charge  = 3.7207188030E-10 P-charge  = 2.8164628214E-30
-----------------------------------------------------------------------
dN-charge = 2.3785474651E-17 dP-charge = 5.0122137430E-38
-----------------------------------------------------------------------
relative distribution of N-charge:
    4.0503E-01  2.9225E-01  3.0272E-01
-----------------------------------------------------------------------
relative distribution of P-charge:
    6.7439E-01  1.4861E-05  3.2560E-01
-----------------------------------------------------------------------
   GU-st:   1, IT=  1, UDI= 1.691E-08, JDI= 0.000E+00, DEF= 9.059E-06
   GU-st:   1, IT=  1, UDI= 1.481E-07, JDI= 6.353E-07, DEF= 9.059E-06

  CPU-Time: 3.00000E-02,  Control: 3.79198E-05
  GUMMEL-steps:  1, NEWTON-steps:  0, Test: 1.481E-07, Defect: 9.059E-06
```

The meaning of the key-words is contained in the following table.

2.17.2 Parameters

name	unit *type* option	default	comment
CApacity()	*real*	0.d-12	capacity at the contact No. IPAC (≤mbias, see (2.89))
CHARGEN()	*real*	rundef	Length≤ mreg
CHARGEP()	*real*	rundef	Length≤ mreg
Continue	*integer*	0	continue the calculation on the current point, no recalculation of the thermodynamic equilibrium is required, after processing the Step-command the value is reset to 0
Execute	*l*	set by SET	toggle the command execution
IGUMAX	*integer*	100	max. gummel steps
Dibias	*array* *V*	0. ...	bias on Dirichlet-contacts. Length = IDIRI · NBias
Gabias	*array* *V*	0. ...	bias on gate-contacts Length = INatur · NBias
IPRINT	*integer*	1	The solution is printed at each IPRINT time point (only transient case).
IPROT	*integer*	10	output control
IPOT	*integer*	0	solve only Poisson eqn.
ITSTEP	*integer*	1	actual working step number
MAstep()	*integer*	iundef	max. step number (≤mbias). In the transient case DELT = TIMINT/MASTEP is the starting time step
MBIAS	*integer*	mbias	number of working points
MISTep()	*integer*	iundef	min step number Length≤mbias

continued on next page

name	unit *type* option	default	comment
MOckeu	*integer*	20	For TIMINT(I) > 0 MOCKEU determines the type of time discretization. MOCKEU = 1 means Mock-Method
NBias	*integer*	iundef	working point number
OMega()	*real*	0.d9	ac-analysis frequency Length≤mbias
NOMEGA	*integer*	1	ac-analysis
DELTAu	*real* [V]	0.00048d0	ac-analysis
SETTIM	*integer*	0	time will be initialed after each time interval TImint
Save	*integer*	0	toggle result saving
		0	no save
		1	save each time step
		2	save before and after call dynewt only
Load	*integer*	0	toggle result loading
		100	load the 100. dataset
		-100	load all datasets 1...100
STrom()	*real*	0.d-12	current at the contact No. IPAC (≤mbias, see (2.89))
TImint()	*real* [s]	-1.d0	time intervalls
VAOXDI()	*real* [cm]	0.d0	Length≤50
WORKstor	*integer*	2000000	max. integer workspace
RWORKstor	*integer*	2000000	max. real workspace
Functions	*integer*	55	max. number of functions

2.17.3 Some more comments on parameters

The capacity C (the corresponding parameter is `CApacity`) can be entered in the transient calculations on the `ICAP`-th edge according to the boundary conditions

$$J - J_i = CA_i \frac{dV}{dt}, \ i = 1, ..., \texttt{mbias} \tag{2.89}$$

The number of the contact `ICAP` has to be defined in the `STEP` command.

2.18 Recombination command

The `REcombination`-command defines the parameters of the generation-recombination model.

2.18.1 Models

For the generation-recombination we use the general additive ansatz of different generation-recombination processes.

$$G - R = G_{\text{ava}} - R_{\text{SRH}} - R_{\text{Aug}} - R_{\text{Surf}} - R_{\text{rad}} \pm \dots \tag{2.90}$$

where

G_{ava} Avalanche-generation,
R_{SRH} Shockley-Read-Hall-recombination,
R_{Aug} surface-recombination,
R_{Surf} Auger-recombination,
R_{rad} radiative recombination.

We describe these generation-recombination processes in detail. The temperature T is always normalized $T = \frac{\text{Temp in } K}{300K}$, the dopant concentration is D

Moreover, the dependence on the material M can be considered by a constant factor.

Shockley–Read–Hall recombination

The models are taken from Selberherr [Sel84], S.105, Eqns.(4.2-14), and H.C. de Graaf and F. M. Klaassen, Compact Transistor Modelling for Circuit Design, Springer, Wien (1990)

$$R_{srh}(n,p) = \frac{np - N_i^2}{\tau_p(n + r_n) + \tau_n(p + r_p)} \tag{2.91}$$

with

$$\frac{1}{\tau_n} = T^{\gamma_n}\left(\frac{1}{\tau_{n0}} + C_n D\right) + T^{\delta_n} A_n p^2 \tag{2.92}$$

$$\frac{1}{\tau_p} = T^{\gamma_p}\left(\frac{1}{\tau_{p0}} + C_p D\right) + T^{\delta_n} A_p n^2 \tag{2.93}$$

Auger recombination

(see Selberherr[Sel84] , S.109,(4.2-35))

$$R_{aug}(n,p) = (a_b + a_n n + a_p p)(np - N_i^2) \tag{2.94}$$

Surface recombination

(see Selberherr[Sel84] , S.110,(4.2-36))

$$R_{surf}(n,p) = \frac{np - N_i^2}{\frac{n+r_n}{v_n} + \frac{p+r_p}{v_p}} \tag{2.95}$$

Avalanche generation

(see Selberherr [Sel84], S.110ff)

$$G_{ava} = a_1 |\mathbf{J}_n| \exp(-a_2/E\mathbf{J}_n) + a_x |\mathbf{J}_p| * \exp(-a_y/E\mathbf{J}_p) \tag{2.96}$$

with

$$E\mathbf{J}_n = |E * \mathbf{J}_n|/|\mathbf{J}_n|$$
$$E\mathbf{J}_p = |E * \mathbf{J}_p|/|\mathbf{J}_p|$$

$$a_x = a_3, \quad \text{for } EJp \le a_7 \tag{2.97}$$

$$a_y = a_4, \quad \text{for } EJp \le a_7 \tag{2.98}$$

$$a_x = a_5, \quad \text{for } EJp \ge a_7 \tag{2.99}$$

$$a_y = a_6, \quad \text{for } EJp \ge a_7 \tag{2.100}$$

$$a_1 \ge 0 \tag{2.101}$$

Trapped charges

The Shockley-Read-Hall recombination was extended to incorporate deep traps in volume and on interfaces. These trap levels can take different states (neutral, negatively charged, positively charged) that are governed by additional equations, see 1.3.

According to the parameters in Section 1.3 on page 11, the following parameters can be set:

`TER` $= E_r$, the difference between trap energy level and intrinsic Fermi energy;

`TNR` $= N$, the total density of the impurity;

`TSN` $= s_n$ and `TSP` $= s_p$ are the capture coefficient (this is the product of capture cross section and thermal velocity).

Incomplete Ionization

Incomplete Ionization is handled in the same way as trapped charges.

According to the parameters in section 1.3 on page 11, the following parameters can be used:

`EDR`$= E_D - E_i$, the difference between trap energy level and intrinsic Fermi energy;

`EAR`$= E_A - E_i$, the ionized part of traps

`SND`$= s_n$ and `SPA`$= s_p$ are the capture coefficient (this is the product of capture cross section and thermal velocity).

2.18.2 Parameters

Recombination

name	unit *type* option	default	comment
AUGB	*real*	0.d0	a_b in (2.94)
AUGN	*real*	2.8d-31	a_n in (2.94)
AUGP	*real*	9.9d-32	a_p in (2.94)
AVA1	*real* [1/cm]	1.00d6	a_1 in (2.96); AVA1= 0 turns off the avalanche-generation.
AVA2	*real* [V/cm]	1.66d6	a_2 in (2.96)
AVA3	*real* [1/cm]	1.582d6	a_3 in (2.97)
AVA4	*real* [V/cm]	2.036d6	a_4 in (2.98)
AVA5	*real* [1/cm]	6.71d5	a_5 in (2.99)
AVA6	*real* [V/cm]	1.693d6	a_6 in (2.100)
AVA7	*real* [V/cm]	4.d5	a_7 in (2.97,...,2.100)
CAUGn	*real*	0.d0	A_n in (2.92)

continued on next page

name	unit *type* option	default	comment
CAUGp	*real*	0.d0	A_p in (2.93)
CSRHn	*real*	0.d0	C_n in (2.92)
CSRHp	*real*	0.d0	C_p in (2.93)
DELTAn	*real*	0.d0	δ_n in (2.92)
DELTAp	*real*	0.d0	δ_p in (2.93)
GAMMAn	*real*	0.d0	γ_n in (2.92)
GAMMAp	*real*	0.d0	γ_p in (2.93)
IBulk	*integer*	-1	number of the BULK-contact
IVREN()	*real*	1.d0	factor of surface recombination velocity for gate contacts, Length$\leq$ mreg
IVREP()	*real*	1.d0	factor of surface recombination velocity for gate contacts, Length$\leq$ mreg
RAB	*real* [Ω]	0d0	
RABT	*real* [Ω]	0.d0	BULK-resistance, time dependent
REN	*real* [cm^{-3}]	1.09d10	r_n in (2.91, 2.95)
REP	*real* [cm^{-3}]	1.09d10	r_p in (2.91, 2.95)
RENI	*real*	0.d0	intrinsic carrier density for SRH recombination
TAUN0	*real* [s]	2d-4	τ_{n0} in (2.92), life time (electrons)
TAUNFA()	*real*	1.d0	$\tau_n^{\mathrm{mat}}(M)$, $M = 1, ..., $mreg
TAUP0	*real* [s]	2d-6	τ_{p0} in (2.93), life time (holes)
TAUPFA()	*real*	1.d0	$\tau_p^{\mathrm{mat}}(M)$, $M = 1, ..., $mreg
VREN	*real* [cm/s]	5.d0	v_n in (2.95), recombination speed
VREP	*real* [cm/s]	5.d0	v_p in (2.95), recombination speed

Trap Model

name	unit *type* option	default	comment
ITRAP	*integer*	0	ITRAP number of traps, ITRAP>1 activate the trap model
IZTR()	*integer*	0	ITRAP numbers of zones, in which relevant traps are active, Length≤ mreg
TER()	*real* [V]	0.d0	Trap model, E_r in formula (1.33c) and (1.33d), Length≤ mreg
TNR()	*real* [cm^{-3}]	0.d0	Trap model, Length≤ mreg if TNR<0: N_k^{trap} = -TNR, acceptor-type trap in formula (1.32a); if TNR>0: N_k^{trap} = +TNR, donor-type trap in formula (1.32a);
TSN()	*real*	1.3d-6	Trap model, electron capture coefficient s_n in formula (1.33), Length≤ mreg
TSP()	*real*	1.3d-7	Trap model, hole capture coefficient s_p in formula (1.33), Length≤ mreg

Incomplete Ionization

name	unit *type* option	default	comment
INCOMplete	*integer*	0	INCOM=1 activates incomplete ionization

continued on next page

name	unit *type* option	default	comment
EAR	*real* [eV]	0.d0	Trap model, Length≤ mreg if EAR<0: N_a = -EAR, N_d = 0 in formula 1.32b, 1.32c; if EAR>0: $N_a = 0$, N_d =EAR,
EDR	*real* [eV]	0.d0	E_r , Length≤ mreg
END	*real*	1.d-0	
EPA	*real*	1.d-0	
SND	*real* [cm^3/s]	1.3d-6	s_n , Length≤ mreg
SPA	*real* [cm^3/s]	1.3d-7	s_p , Length≤ mreg

2.19 Load command

The LOAD-command is used, to read WIAS-TeSCA save files from previous simulations and to load analytical profiles or interpolate profiles from external meshes. Loading a WIAS-TeSCA save file is the default. When loading a WIAS-TeSCA save file, grid, layer structure and doping profiles are read from the file and the simulation can be continued. The save files may be compressed. Incompatibilities of the storage sizes between loaded file and current WIAS-TeSCA run are indicated and corrected internally.

Incompatibilities of old save files with newer program versions, reflect frequently only modifications in the parameter lists. By default the command parameters are not read from the save file. In this case all changes of default parameter values that had been made for the simulation are lost and have to be repeated after loading the file. LOAD(DEFAULT=on) can be used to force reading the command parameter values from the file.

Other incompatibilities in the save files are handled by different internal version numbers in the save file. A warning indicates, if the program has to modify the file content in order to be able to continue the simulation. If save files have to be exchanged between different machines, the file can be saved as compressed ASCII-file SAVE(File=...,FORMat=1). When loading the file LOAD(FORMat=on...) can be specified.

The LOAD command can be used also, to define analytical profiles or to interpolate values on external numerical results. The user grid and the layer structure have to be defined before loading the profiles. For all nodes in the existing grid the values in the loaded profiles are interpolated. By default, the interpolated values are added to the already existing nodal values. If ADD=off is specified, the old

doping values are erased on the entire grid and only the loaded new values are used. The final profile is used to refine the mesh automatically.

The analytical functions are defined in the entire x-y-plane and the profiles interpolated from an external file are extended in vertical and lateral directions, to cover the entire plane too. There is no extrapolation formula used, instead a "1D-continuation" is assumed in vertical or lateral direction.

Several species can be loaded at the same time. A list of species names can be supplied in the `LOAD` command. It is used to select some of the species from a file. If no species can be identified, the specified names are assigned to the profiles in the order as they appear in the file. The species names are used also for the analytical profiles. If no name can be identified, a net doping profile is assumed.

Doping profiles are interpreted as total concentrations. If the read file or the analytical function provide only a net profile, its absolute value is taken for the total doping profile. From the net and total doping profiles the donor and acceptor concentrations are computed and added to the specified acceptor and donator species:

$$\texttt{FNET} \cdot \mathit{net} = \mathit{donator} \text{ - } \mathit{acceptor}$$
$$\mathit{total} = \mathit{donator} + \mathit{acceptor}$$

The external prescription of active concentrations, net and total concentrations, electron and hole density and electrostatic potential is impossible, since these variables are defined internally from the total concentrations due to clustering and charge neutrality assumptions (or by solving device equations).

2.19.1 Parameters

name	unit *type* option	default	comment
FILE	*strin*80*	undefined	File name.
TYPe		dmp	Type of doping definition: dmp plt exp prf plx Constant Gauss Erf Prosim DIFfgaus Relief GAUSS3 Tesim WIAS-TeSCA XGraph Mdraw dmp.Z dmp.gz

continued on next page

name	unit *type* option	default	comment
IGNore	*boolean*	off	Applies to TYPe=dmp, dmp.Z, dmp.gz If a save file is created during the diffusion process (NSAVE Saveeach) the already passed process time is saved into the file. By default IGNore=off, and this time is read and the diffusion time, immediately after the LOAD command is reduced by this time. This enables continuation of the diffusion simulation. For IGNore=on, the loaded time is ignored.
SPecies()	*record*		Names of species, to be selected from the file are defined by analytical profiles.
XLeft	Length	um	Left window boundary. Must be specified for TYPe=DIFfgaus, GAUSS.
XRight	Length	um	Right window boundary. Must be specified for TYPe=DIFfgaus, GAUSS.
EPS	Length	0.1um	Length of linear decay at the sides of the window if ULeft, URight, UTop or UBottom are undefined.
XSYLeft	Length	um	Left symmetry line.
XSYRight	Length	um	Right symmetry line.
SHIFt	*real*	1.e10	Vertical shift transformation. If SHIFt > 1e9 the profile is shifted to the local substrate surface.

continued on next page

name	unit *type* option	default	comment
FACtor	*real*	-1000.	Vertical scaling factor for 1D profiles.
ADD	*boolean*	on	Summation of already existing and newly loaded profiles. off: erase the existing profiles in the entire grid, before loading the new profile.
ULeft	Length	um	Lateral "diffusion" length at the left side. Must be specified for TYPe=DIFfgaus, GAUSS3.
URight	Length	um	Lateral "diffusion" length at the right side. Must be specified for TYPe=DIFfgaus, GAUSS3.
Dot	Concentration	/cm3	Doping concentration (with sign). Must be specified for TYPe=Constant, Gauss, Erf, DIFfgaus, GAUSS3. Additional scaling factor of the doping for TYPe=Prosim, Relief: Net(WIAS-TeSCA) = FNET · Dot · *net(file)* Total (WIAS-TeSCA) = Dot · *total(file)*
Y	Length	um	Position of the maximum doping. Must be specified for TYPe=Gauss, Erf. **Note!**YY=-1 !

continued on next page

name	unit *type* option	default	comment
S	Length	um	Standard deviation of the doping. Must be specified for TYPe=Gauss, Erf, DIFfgaus, GAUSS3.
L	Length	um	Diffusion length. Must be specified for TYPe=DIFfgaus.
R	Length	um	Projected range. Must be specified for TYPe=DIFfgaus.
LAT	Length	um	Lateral diffusion length. Must be specified for TYPe=DIFfgaus.
SAT	Concentration	/cm3	Saturation value of the profile. Must be specified for TYPe=GAUSS3.
YTop	Length	um	Top window boundary. If YTop=undefined no upper limit is assumed. Must be specified for TYPe=GAUSS3.
YTop	Length	um	Top window boundary. If YTop=undefined no upper limit is assumed. Must be specified for TYPe=GAUSS3.
YBottom	Length	um	Bottom window boundary. If YBottom=undefined no bottom window boundary is assumed. Must be specified for TYPe=GAUSS3.

continued on next page

name	unit *type* option	default	comment
UTop	Length	um	"Diffusion length" at the top window boundary.
UBottom	Length	um	"Diffusion length" at the bottom window boundary.
FNET	*real*	1	Scaling factor of the net doping. In DIOS : Net = donator- acceptor
ACCeptor	Dopant	B	Acceptor element when loading Net and Total profiles.
DONator	Dopant	P	Acceptor element when loading Net and Total profiles.
DEFaults	*boolean*	off	
	off		Prevents WIAS-TeSCA from reading the default parameter values for the command interpreter from a save file. Only the WIAS-TeSCA layer system, grid, doping and work arrays are read. **Note!** Default values, modified in the previous simulation run, have to be changed again after loading the file.
	on		Default values are read from the save file. Should be used only, if SAVE and LOAD are done with exactly the same WIAS-TeSCA version.

2.20 Use command

This command should be used always when the transition from process simulation to a device simulation with WIAS-TeSCA is done. The command has to be used if regions that are defined in WIAS-TeSCA should be omitted in the WIAS-TeSCA simulation.

In particular the contacts for the device simulation can be defined in the Use command.

2.20.1 Parameters

parameter name	unit *type* options	default value	comment
Triangle	*task*		Selection of triangles for the device simulation.
Material()	*record*		All triangles of a WIAS-TeSCA material can be treated as one zone. They may not be connected Material(SI=1,OX=2)
Area(	*task*		Deliberate selection of areas. First an area number followed by a zone number have to be specified. This overrides a zone number specied from the material list Area(area=52,zone=1, area=53,zone=2).
Area	*integer*	undefined	Number of the area.
Zone	*integer*	undefined	WIAS-TeSCA zone number.
)			end of task Area.
)			end of task Triangle.

continued on next page

name	unit *type* option	default	comment
`Contacts(`	*task*		Denition of contacts for the device simulation. The contact number is composed of: `sign(el)*(opt*100+abs(el))`, where `el` denotes the number of the gate or metal contact and `opt` the number of the optical or thermal contact. If `Energy(IEnergy=-1)` no thermal contacts are assumed, and if `FERmi(ISPec=0)` no optical contacts are assumed. `el=-1...-10` for metal contacts `el=1...5` for gate contacts `opt=1...20` for optical or thermal contacts.
`Points(`	*record*		Specification of contacts by approximate location of starting and end point. The outer boundary of the selected triangulation is used, to define the closest points for the starting and end point of the specified contacts. The corresponding edges on the outer boundary of the triangulation is selected to define the contact. The outer connectivity component is surrounded counterclockwise, the inner components (e.g., omitted polysilicon inclusions) clockwise. Up to 15 contacts can be specified.
`TYPE`	*integer*		Contact type of the first contact.

continued on next page

name	unit *type* option	default	comment
XB1	*Length*	um	Approximate lateral position of the starting point of the first contact.
YB1	*Length*	um	Approximate vertical position of the starting point of the first contact.
XE1	*Length*	um	Approximate lateral position of the end point of the first contact.
YE1	*Length*	um	Approximate vertical position of the end point of the first contact.
TYPE0	*integer*	999	Default boundary condition type, symmetry condition.
			For completeness the parameters XRT YRT XLT YLT XLB YLB XRB YRB BCLeft BCRight BCBottom BCTop can be specified, too. (cf. GRID command)
)			End of record Points.
Dibez()	*record*		Names (*string* $*$ 8) of the Dirichlet (metal) contacts in the order of their type numbers -1, -2, -3, ..., -10.
Gabez()	*record*		Names (*string* $*$ 8) of the gate contacts in the order of their type numbers 1, 2, 3, 4, 5.
)			End of task Contacts.

3 Numerical methods

3.1 Discretization of space

In WIAS-TeSCA, the discretization of space is realized with the finite elements method. As finite elements, triangles are used. By means of the DOMAIN command, different triangulation possibilities can be selected. In particular, it is possible to include user-defined grids. The potential and the charge carrier densities are the unknowns in the discrete versions of the Poisson continuitity and total current equations. These were derived with the aid of similar thoughts as described by Buturla et al. [ECGS81] for the programming system FIELDAY. In particular, the discretization of the continuity equations is based on Scharfetter's and Gummel's assumption of constant current densities along the edges of the triangles. The command POTENTIAL provides for an automatic grid refinement.

3.2 Discretization of time

In WIAS-TeSCA, the necessary complexity for the realization of one time step is essentially determined by the setting of the iteration parameter MOCKEU in the STEP command. For MOCKEU=1, a method examined and described by Mock [Moc83] is used that is based upon the successive solution of the continuity and the total current equations at one single time step. The iteration of this procedure depends on the choice of MOCKEU. For sufficiently high values of MOCKEU, this results in the implicit Euler method.

The time-step is controlled by the use of an energy functional, which plays a key role in analytical investigations where it is known as Lyapunov function [Gaj85].

3.3 Linearization

For the calculation of characteristics, WIAS-TeSCA works with the natural embedding method which uses the terminal voltage as embedding parameter. Unless a new start with saved values is carried out, the first operation point on a characteristic is reached from the thermodynamic equilibrium. Each time, from two calculated operation points, a starting point for the iterative calculation of the new operation point is gained by extrapolation of the electrostatic and the quasi-Fermi potential. The iteration is always started with the successive Gummel method.

It is switched to the simultaneous Newton method automatically if the rate of convergence falls below the parameter value GUMNEW. In the Gummel method, the necessary solution of the non-linear Poisson equation is generally carried out with the Newton method.

3.4 Solution of linear systems of equations

As a result of discretization and linearization, sparse systems of linear equations appear that are solved in `WIAS-TeSCA` through a combination of sparse-matrix-techniques and iteration methods.

To solve the decoupled Poisson equation, a multi-grid method is used, and if necessary an automatic grid adaption to the given doping profile is employed.

Due to the Gummel method a linear system of equations arrises from the non-linear Poisson equation. It is solved with a conjugated gradient method with pre-conditioning and if necessary with a sparse-matrix-correction. The sparse-matrix-technique is also used for the solution of the discretized continuity and total current equations. However, the necessary relatively complex factorization is not carried out in each iteration cycle. On the contrary, as long as a sufficient speed of convergence is achieved the old factoring is used for the iteration. The coupled linear systems of equations that arise from the simultaneous Newton method are successively solved with a block iteration method that is based upon the Gummel method with the additional aid of the sparse-matrix-technique [GG92] .

3.5 Current calculation

In `WIAS-TeSCA`, consistent with the realized discretiziaton of the continuity equations, the current J_j through the j-th contact is calculated according to the following formula which is based on Gauss' theorem.

$$J_j = (J, H_j) := \int \mathbf{J} \cdot \nabla H_j \ . \tag{3.1}$$

Here, the expression on the right hand side denotes an area integral of the scalar product of the vectorial current density $\mathbf{J}$ and the gradient of a test-function H_j, which is equal to one in the neighborhood of the j-th contact and disappears in the proximity of the remaining contacts. To generate these test-functions H_j, solutions of the discrete Laplace equation with appropriate boundary conditions are utilized.

4 External tools

4.1 DEVICE – Grid and doping generator for TeSCA

The DEVICE tool is used to generate two-dimensional grids and doping profiles for the semiconductor simulation tool WIAS-TeSCA. DEVICE takes a '.dev' file as input and outputs a '.dom' and a '.dot' file, each with the same base name as the input file, e.g. example.dev yields example.dom and example.dot. The first file contains the grid description while the second describes the doping profile.

4.1.1 Usage of device

DEVICE is called on the command line with the '.dev' file (without suffix!) as only parameter:

```
> device example
```

4.1.2 Structure of the input files

The '.DEV' file consists of several sections – one grid section and several doping sections:

```
1  ! This is a comment
2  ! FILE: example.dev
3  &grid
4     nw        =  4                              ! number of columns
5     w         =  0.1   0.05   0.1    0.2       ! width of each column in [μm]
6                                                 ! (left to right)
7     ncol      =  10    5      10     20        ! subdivision of each column
8     xstretch  =  +0.4  -0.3   +0.4   -0.4      ! lateral stretch factor for each column
9     nd        =  3                              ! number of rows
10    d         =  0.1  0.2     0.3               ! thickness of each row in [μm]
11                                                ! (bottom to top)
12    nrow      =  6    8       16                ! subdivisions of each row
13    ystretch  = +0.4  0.0    -0.4               ! vertical stretch factor of each row
14    mat       = 1 1 2 2                         ! material zones bottom row
15                3 3 2 2                         ! material zones middle row
16                4 4 5 0                         ! material zones top row
17    diag      = +1                              ! direction of triangle diagonal +/-1
18    yorigin   =  1                              ! number of row that contains origin
19    xorigin   =  1                              ! number of column that contains origin
20    ypos      = 'b'                             ! position of origin in row
21                                                ! ('b'ottom, 'c'enter, 't'op)
22    xpos      = 'l'                             ! position of origin in column
23                                                ! ('l'eft, 'c'enter, 'r'ight)
24
```

```
25 /                                              ! end of grid description
26 &doping
27   net= 1.0E18  1.0E16 -1.0E19                  ! row-wise doping for all columns
28 /                                              ! end of first doping section
29 &doping
30   net= 1.0E18  0.0     0.0                     ! row-wise doping for first column
31                                                ! (added to existing doping)
32 /                                              ! end of second doping section
33 ...
```

The stretch factors

The lateral and vertical stretch factors `xstretch` and `ystretch` are used to refine the subdivisions of a column or row in a direction and can take values in the interval $(-\frac{1}{2}, +\frac{1}{2})$. A positive value means that the length of the subdivisions is increasing from left to right in the lateral direction and increasing from bottom to top in the vertical direction. For negative values it is the other way round.

The doping profile

The doping is specified in the `doping` section. The first section describes the doping for ALL columns in a row-wise fashion (bottom to top). The succeeding doping sections describe the doping in the columns: The first section describes the first column, the second section describes the second column, and so on. Doping levels for the same area are added.

4.1.3 Including the grid and doping profile in TeSCA simulations

The grid generated with `DEVICE` is used in `WIAS-TeSCA` via the `GRID` command:

```
! example.dio
...
! load grid
grid(type=dom, file='example.dom')
...
```

However, the `'DOM'` file does not contain information about the position and types of boundary conditions. Hence, these have to be also defined in the `grid` command using the `bc` subcommand:

```
1  ! example.dio
2  ...
3  ! load grid and define boundary conditions
4  grid(type=dom, file='example.dom'
5       bc(type0=999,
6          type1=ID1, XA1, YA1, XB1, YB1
7          type2=ID2, XA2, YA2, XB2, YB2
8          type3=ID3, XA3, YA3, XB3, YB3
9          type4=ID4, XA4, YA4, XB4, YB4
10         type5...
11      )
12 )
```

```
...
```

Here, `ID`i is a placeholder for the type of the boundary condition. The first type always has to be `type0 = 999`, which corresponds to symmetry conditions or homogeneous Neumann conditions. The succeeding boundary condition types are of the form `type`$i = \pm(i \cdot 100 + k)$, where i should be an (increasing by one) positive number, which can be used for the definition of temperature and optic boundary conditions. Moreover, k is a unique nonnegative number that describes the electrical contact, that is referenced e.g. in the `device` command. If $k = 0$, then no-flux boundary conditions are assumed. Finally, if `type`i is positive, e.g. `type1=101`, gate contacts are defined, otherwise, for `type1=-101` metal contacts are assumed.

The coordinates (`XA`i, `YA`i) and (`XB`i, `YB`i) denote the starting and end of a boundary segment. The starting and end points for the segments have to be given counter-clockwise.

The triangulation along with the boundary types can be plotted in `WIAS-TeSCA` with the `graphic` command, namely

```
graphic(isol=no, contact=on, lay=no, glay=no, tria=on, text='Triangulation', plot)
```

The information on the doping profile from the `'DOT'` file is included using the `@` operator which embeds the `'DOT'` file into the `'DIO'` file:

```
! example.dio
...
! include doping file
@example.dot
...
```

4.1.4 Full example

```
! example.dev
&grid
 nw       =  4
 w        =  0.1   0.05   0.1    0.2
 ncol     =  10    5      10     20
 xstretch = +0.4  -0.3   +0.4   -0.4
 nd       =  3
 d        =  0.1  0.2     0.3
 nrow     =  6    8       16
 ystretch = +0.4  0.0    -0.4
 mat      =  1 1 2 2
             3 3 2 2
             4 4 5 0
 diag     = +1
 yorigin  = 1
 xorigin  = 1
 ypos     = 'b'

/
&doping
net = 1.0E18 1.0E16 -1.0E19
/
```

```
&doping
net = 1.0E18 0   0
/
```

```
! example.dio
! ============
! set title
title('Grid test',
    iphy = 0,
    maxv = 20000,
    mxt  = 40000)

! load grid and define boundary conditions
grid( typ = dom,
      file = example.dom,
      bc( type0 =  999
          type1 = -101,  0.25, 0.6, 0.0,  0.6
          type2 =  200,  0.0,  0.6, 0.0,  0.0
          type3 = -301,  0.0,  0.0, 0.45, 0.0
          type4 =  400,  0.45, 0.0, 0.45, 0.6
        )
    )

! initialize layer system
subs()

! plot triangulation and boundary conditions
graphic(isol=no, contact=on, scale(equal=no), lay=no, glay=no, tria=on,
       text='Triangulation' plot)

! pause, continue with GO
break

! plot material regions
gra(isol=no, text='Material zones', contact=on notria lay=no glay=zone plot)

! pause, continue with GO
break

! include doping file
@example.dot

! plot net doping
graphic(junction, lay=no, isol=no, noabs, text='Doping',
        contacts=no, glay=no, spec(net), isol=fill,
        isolin=no, glay=no, isol=fill, plot
)

! pause, continue with GO
break
```

Bibliography

[ECGS81] Buturla E.M., P.E. Cotrell, B.M. Grossman, and K.A. Salsburg. Finite-element analysis of semiconductor devices: The fielday program. *IBM J.Res.Develop.*, 1981.

[Gaj85] H. Gajewski. On existence, uniqueness and asymptotic behavior of solutions of the basic equations for carrier transport in semiconductors. *Z. Angew. Math. Mech.*, 65:101–108, 1985.

[GG92] H. Gajewski and K. Gärtner. On the iterative solution of van Roosbroeck's equations. *Z. Angew. Math. Mech.*, 72:19–28, 1992.

[Lau85] S. Laux. Techniques for small-signal analysis of semi-conductor devices. *IEEE Trans. on Comp. Aid. Des.*, 1985.

[Moc83] M.S. Mock. *Analysis of Mathematical Models of Semiconductor Devices*. Boole Press Dublin, 1983.

[SCW+81] C.T. Sah, P.C.H. Chan, C.-K. Wang, R.L.Y. Sah, K.A. Yamakawa, and R. Lutwack. Effect of zinc impurity in solar-cell efficiency. *IEEE Trans.Electron Devices*, 1981.

[Sel84] S. Selberherr. *Analysis and Simulation of Semiconductor Devices*. Springer, Wien, 1984.

[SSP82] Alfred Schütz, Siegfried Selberherr, and Hans W. Pötzl. A two-dimensional model of the avalanche effects in {MOS} transistors. *Solid-State Electronics*, 25(3):177 – 183, 1982.

[vR50] W. van Roosbroeck. Theory of the Flow of Electrons and Holes in Germanium and Other Semiconductors. *Bell System Technical Journal*, 29:560, 1950.

[WBW93] H. J. Wünsche, U. Bandelow, and H. Wenzel. Calculation of Combined Lateral and Longitudinal Spatial Hole Burning in $\lambda/4$ shifted DFB Lasers. *IEEE Journ. of Quant. electron.*, 1993.

[Yam83] K. Yamaguchi. *IEEE Transactions*, 1983.

[YTK75] K. Yamaguchi, T. Toyabe, and H. Kodera. Effect of field dependent carrier diffusion on the two-dimensional analysis of a junction gate fet. *Japan J.Appl.Phys.*, 1975.