PADRE Simulator

Prepared by:

Dragica Vasileska
Associate Professor

Arizona State University

Some Hints on Using the PADRE Simulation Software

PADRE is a modular and extensible framework for one, two and three dimensional semiconductor device simulation [1]. It is implemented using modern engineering practices that promote reliability, maintanability and extensibility. Products that use the PADRE framework meet the device simulation needs of many semiconductor applications. In this section, we point out to the most critical issues one faces when creating the input file for particular device structures. A complete description of each PADRE statement and its parameters is given in the PADRE manual (http://www.nanohub.org/resource_files/tools/padre/doc/).

The PADRE syntax

A PADRE command file is a list of commands for PADRE to execute. This list is stored as an ASCII text file using any text editor. The input file contains a sequence of statements. Each statement consists of a keyword that identifies the statement and a set of parameters. The general format is:

```
<STATEMENT> <PARAMETER>=<VALUE>
```

Some hints on the proper structure of the statements are given below:

1. The statement keyword must come first, but after this the order of parameters within a statement is not important.
2. It is only necessary to use enough letters of any parameter to distinguish it from any other parameter on the same statement.
3. Logicals can be explicitly set to false by preceding them with the ^ symbol.
4. Any line beginning with # is ignored. These lines are used as comments. Note that the "#" can be put on any PADRE input line; all information to the left of the character is processed, and that to the right is ignored.
5. PADRE can read up to 256 characters on one line. However, it is best to spread long input statements over several lines to make the input file more readable. The character + at the beginning of a line indicates continuation.

A simple example of PADRE statements is given next. It is designed to model simple MOS capacitor and calculate the potential, the electric field densities, the total charge density and the electron concentration under DC bias conditions.

```
$ Mesh Specification
mesh     rect nx=3 ny=60
y.m      n=1  l=0 r=1
y.m      n=10 l=0.001 r=0.8
y.m      n=60 l=0.1 r=1.05
x.m      n=1 l=0 r=1
x.m      n=3 l=1 r=1

$ Regions specification
region   num=1 ix.l=1 ix.h=3 iy.l=1 iy.h=10 name=siO2 INS
region   num=2 ix.l=1 ix.h=3 iy.l=10 iy.h=60 name=silicon SEMI

$ Electrodes specification
elec     num=1 ix.l=1 ix.h=3 iy.l=1 iy.h=1
elec     num=2 ix.l=1 ix.h=3 iy.l=60 iy.h=60

$ Doping specification
dop reg=2 p.type conc=1e18 uniform
$ doping can be p.type or n.type (two options), conc is a parameter
too

$ Contact specification
contact all neutral
contact  num=1 aluminum
$ one can add as options: n.polysilicon, p.polysilicon, tungsten

$ Specify models
models srh conmob fldmob
system electrons holes newton

$ Solve for initial conditions
solve init
plot.1d pot a.y=0 b.y=0.1 a.x=0.5 b.x=0.5 ascii outf=pot.plot
plot.1d ele a.y=0 b.y=0.1 a.x=0.5 b.x=0.5 ascii outf=ele.plot
plot.1d net.charge a.y=0 b.y=0.1 a.x=0.5 b.x=0.5 ascii outf=ro.plot
plot.1d e.field a.y=0 b.y=0.1 a.x=0.5 b.x=0.5 ascii outf=efield.plot

$ Solve for applied bias
solve prev
solve proj vstep=0.2 nsteps=10 elect=1
plot.1d pot a.y=0 b.y=0.1 a.x=0.5 b.x=0.5 ascii outf=potne.plot
plot.1d ele a.y=0 b.y=0.1 a.x=0.5 b.x=0.5 ascii outf=elene.plot
plot.1d net.charge a.y=0 b.y=0.1 a.x=0.5 b.x=0.5 ascii outf=rone.plot
plot.1d e.field a.y=0 b.y=0.1 a.x=0.5 b.x=0.5 ascii outf=efieldne.plot

end
```

As we can see from the above example, the order in which statements occur in PADRE input file is very important. There are four groups of statements, and these must occur in the correct order. These groups are indicated in Figure 1. **Each input file must contain these four groups in order**. Failure to do this will usually cause an error message and termination of the

program, but it could lead to incorrect operation of the program. For example, material parameters or models set in the wrong order may not be used in the calculations. The order of statements within the mesh definition, structural definition, and solution groups is also important.

Group		Statements
1. Structure Specification	_____	MESH REGION ELECTRODE DOPING
2. Material Models Specification	_____	MATERIAL MODELS CONTACT INTERFACE
3. Numerical Method Selection	_____	METHOD
4. Solution Specification	_____	LOG SOLVE LOAD SAVE

Figure 1. PADRE command groups with the primary statements in each group.

Choice of the numerical method

Several different numerical methods can be used for calculating the solutions to semiconductor device problems. Different solution methods are optimum in different situations and some guidelines will be given here. Different combinations of models will require PADRE to solve up to six equations. For each of the model types there are basically three types of solution techniques: (a) de-coupled (`GUMMEL`), (b) fully coupled (`NEWTON`) and (c) `BLOCK` that are specified on the SYSTEM line. In simple terms, the de-coupled technique like the Gummel method will solve for each unknown in turn keeping the other variables constant, repeating the process until a stable solution is achieved. Fully coupled techniques such as the Newton method solve the total system of unknowns together. The coupled or block methods will solve some equations fully coupled, while others are decoupled.

In general, the Gummel method is useful where the system of equations is weakly coupled, but has only linear convergence. The Newton's method is useful when the system of equations is strongly coupled and has quadratic convergence. The Newton method may however spend extra time solving for quantities which are essentially constant or weakly coupled. Newton also requires a more accurate initial guess to the problem to obtain convergence. Thus, a coupled method can provide for faster simulation times in these cases over Newton. Gummel can often provide better initial guesses to problems. It can be useful to start a solution with a few Gummel iterations to generate a better guess and then switch to Newton to complete the solution. The following specifies a PDE system for a simulation with only holes and using the Gummel method:

```
SYSTEM  GUMMEL CARR=1 HOLES
```

Basic Drift Diffusion Calculations

The isothermal drift diffusion model requires the solution of three unknowns represented by the potential, electron and hole concentrations. Specifying `GUMMEL` or `NEWTON` alone will produce simple Gummel or Newton solutions as detailed above. For almost all cases the Newton method is preferred and it is the default.

The basic drift-diffusion equations in 3D are of the form:

$$\begin{aligned}
&\nabla\cdot\left(\varepsilon\nabla V\right) = q\left(n - p + N_B\right)\\
&\nabla\cdot J_n = qU(n,p) + q\frac{\partial n}{\partial t}\\
&\nabla\cdot J_p = -qU(n,p) + q\frac{\partial p}{\partial t}\\
&J_n = q\mu_n\left(-n\nabla V + \frac{k_B T}{q}\nabla n\right)\\
&J_p = q\mu_p\left(-p\nabla V - \frac{k_B T}{q}\nabla p\right)
\end{aligned} \tag{1}$$

with $N_B = N_A - N_D$. Note that the above equations are valid in the limit of small deviations from equilibrium, since the Einstein relations have been used for the diffusion coefficient, normally valid for low fields or large devices. The generation-recombination term *U* will be in general a function of the local electron and hole concentrations, according to possible different physical

mechanisms. In Eqs. (1) ε is the dielectric constant, V is the potential , n is the electron density, p is the hole density, q is the elementary charge, J_n and J_p are the electron and hole current densities, μ_n and μ_p are the electron and hole mobilities, and T is the temperature.

The Poisson equation is always solved, and optionally one can specify that continuity and/or energy balance partial differential equation (PDEs) be solved for the carriers. The continuity equations can be selected by setting `CAREERS` to the number of carriers (0, 1, 2) with the single carrier defined through the `ELECTRONS` and/or `HOLES` parameter; the default is `ELECTRONS` for `CARRIERS=1`. In general, one can specify under the `SYSTEM` line a general non-linear construct (with arbitrary groupings of PDEs) using the `COUPLING` parameter. In this case the PDEs are selected using an integer code as follows:

```
1    Poisson
2    Electron continuity equation
3    Hole continuity equation
4    Electron energy balance equation
5    Hole energy balance equation
```

Drift Diffusion Calculations with Lattice Heating

When the lattice heating model is added to drift diffusion an extra equation is added. The `BLOCK` algorithm solves the three drift diffusion equations as a Newton solution and follows this with a decoupled solution of the heat flow equation. The `NEWTON` algorithm solves all four equations in a coupled manner. `NEWTON` is preferred once the temperature is high, however, `BLOCK` is quicker for low temperature gradients.

The heat flow equation implemented in PADRE has the form:

$$C\frac{\partial T_L}{\partial t} = \nabla\left(\kappa \nabla T_L\right) + H \qquad (2)$$

where C is the heat capacity per unit volume, κ is the thermal conductivity, H is the heat generation and T_L is the local lattice temperature. The heat capacity can be expressed as $C = \rho C_p$, where C_p is the specific heat and ρ is the density of the material.

Energy Balance Calculations

The conventional drift-diffusion model of charge transport neglects non-local transport effects such as velocity overshoot, diffusion associated with the carrier temperature and the dependence

of impact ionization rates on carrier energy distributions. These phenomena can have a significant effect on the terminal properties of submicron devices. The energy balance model implemented in PADRE introduces two new independent variables T_n and T_p, the carrier temperature for electrons and holes. The energy balance equations consist of an energy balance equation with the associated equations for current density and an energy flux $\mathbf{S}_{n,p}$. For electrons, the energy balance equation consists of

$$\begin{aligned}
\nabla\cdot\mathbf{S}_n &= -\mathbf{J}_n\cdot\nabla\psi - W_n - \frac{3k}{2}\frac{\partial}{\partial t}\left(\lambda_n^* n T_n^*\right)\\
\mathbf{J}_n &= qD_n\nabla n - \mu_n n\nabla\psi + qnD_n^T\nabla T_n\\
\mathbf{S}_n &= -K_n\nabla T_n - \left(\frac{\kappa\delta_n}{q}\right)\mathbf{J}_n T_n
\end{aligned} \tag{3}$$

where $\mathbf{S}_n$ is the energy flux density associated with electrons, W_n is the energy density loss rate for electrons, K_n is the thermal conductivity for electrons, D_n is the thermal diffusivity and μ_n is the electron mobility. The remaining terms are defined by the following equations:

$$\begin{aligned}
&D_n = \frac{\mu_n kT_n}{q}\lambda_n;\quad \lambda_n = \frac{F_{1/2}(\eta_n)}{F_{-1/2}(\eta_n)};\quad \eta_n = \frac{E_{Fn}-E_C}{kT_n}\\
&D_n^T = \left(\mu_{2n} - \frac{3}{2}\lambda_n\mu_n\right)\frac{k}{q};\quad \mu_{2n} = \mu_n\left(\frac{5}{2}+\xi_n\right)\frac{F_{\xi_n+3/2}(\eta_n)}{F_{\xi_n+1/2}(\eta_n)}\\
&K_n = qn\mu_n\left(\frac{k}{q}\right)^2\Delta_n T_n;\quad \Delta_n = \delta_n\left[\left(\frac{7}{2}+\xi_n\right)\frac{F_{\xi_n+5/2}(\eta_n)}{F_{\xi_n+3/2}(\eta_n)} - \left(\frac{5}{2}+\xi_n\right)\frac{F_{\xi_n+3/2}(\eta_n)}{F_{\xi_n+1/2}(\eta_n)}\right];\quad \delta_n = \frac{\mu_{2n}}{\mu_n}
\end{aligned} \tag{4}$$

where

$$\begin{aligned}
&\lambda_n = 1\\
&\Delta_n = \delta_n = \left(\frac{5}{2}+\xi_n\right)\\
&\xi_n = \frac{T_n}{\mu_n}\left(\frac{\partial\mu_n}{\partial T_n}\right)
\end{aligned} \tag{5}$$

Similar equations hold for holes. Therefore, the energy balance model requires the solution of up to 5 coupled equations. `GUMMEL` and `NEWTON` have the same meanings as with the drift diffusion model (i.e. `GUMMEL` specifies a decoupled solution and `NEWTON` specifies a fully coupled solution). However, `BLOCK` performs a coupled solution of potential, carrier continuity equations followed by a coupled solution of carrier energy balance, and carrier continuity equations.

Energy Balance Calculations with Lattice Heating

When non-isothermal solutions are performed in conjunction with energy balance models, a system of up to six equations must be solved. `GUMMEL` or `NEWTON` solve the equations iteratively, or fully coupled, respectively. `BLOCK` initially performs the same function as with energy balance calculations; then solves the lattice heating equation in a de-coupled manner.

Solutions obtained

PADRE can calculate DC, AC small signal, and transient solutions. Obtaining solutions is rather analogous to setting up parametric test equipment for device tests. One usually defines the voltages on each of the electrodes in the device. PADRE then calculates the current through each electrode. PADRE also calculates internal quantities, such as carrier concentrations and electric fields throughout the device. This is information that is difficult or impossible to measure. In all simulations the device starts with zero bias on all electrodes. Solutions are obtained by stepping the biases on electrodes from this initial equilibrium condition. Results are saved using the `LOG` or `PLOT` statements.

The following example performs an initial bias point, saving the solution to the data file `OUT0`:

```
SOLVE  INIT OUTF=OUT0
```

In the next example, bias stepping is illustrated. The two solve lines produce the following bias conditions:

```
Bias point #     V1     V2     V3
1               0.0    0.5    -0.5
2               1.0    0.5    0.0
3               2.0    0.5    0.0
4               3.0    0.5    0.0
5               4.0    0.5    0.0
6               5.0    0.5    0.0
```

The solutions for these bias points will be saved to the files OUT1, OUTA, OUTB, OUTC, OUTD and OUTE. Note that the initial guess for the first bias point is obtained directly from the preceding solution because the `PREVIOUS` option was specified. The initial guesses for bias

points 2 and 3 will also be obtained as if PREVIOUS had been specified since two electrodes (numbers 1 and 3) had their biases changed on bias point 2. However, for bias points 4, 5 and 6, *PADRE* will use a projection to obtain an initial guess since starting with bias point 4, both of its preceding solutions (bias points 2 and 3) only had the same electrode bias (number 1) altered.

```
SOLVE  PREV V1=0 V2=.5 V3=-.5 OUTF=OUT1
SOLVE  PROJ V1=1 V2=.5 V3=0 VSTEP=1 NSTEPS=4
+      ELECT=1 OUTF=OUTA
```

Here is a case where two electrodes are stepped (2 and 3). The bias points solved for will be (0,0,1), (0,.5,1.5), (0,1,2) and (0,2,3). *PADRE* will use the PROJECT option to predict an initial guess for the third and fourth bias points since the bias voltages on both electrodes 2 and 3 have been altered by the same amount between each point.

```
SOLVE  PROJ V1=0 V2=0 V3=1 VSTEP=.5 NSTEPS=2 ELECT=23
SOLVE  PROJ V2=2 V3=3
```

If no new voltages are specified and a VSTEP is included, the first bias point solved for is the preceding one incremented appropriately by VSTEP. This is illustrated by repeating the above example as a three line sequence:

```
SOLVE  V1=0 V2=0 V3=1
SOLVE  PROJ VSTEP=.5 NSTEPS=2 ELECT=23
SOLVE  PROJ VSTEP=1  NSTEPS=1 ELECT=23
```

The following sequence is an example of a time-dependent solution. The METHOD line specifies the second-order discretization and automatic time-step selection option, along with Newton-Richardson. The first SOLVE line then computes the solution for a device with 1 volt on V1 and 0 on V2 in steady-state. The second SOLVE line specifies that V1 is to be ramped to 2 volts over a period of 10ns and is left on until 25 ns. Each solution is written to a file; the name of the file is incremented in a manner similar to that described above for a dc simulation (UP1, UP2, etc.). Note that an initial time step had to be specified on this line. The third SOLVE line ramps V1 down from 2 volts to -1 volts in 20 ns (end of ramp is at t = 45ns). The device is then solved at this bias for another 55 ns (out to 100 ns). Note that again each solution is saved in a separate file (DOWN1, DOWN2, etc.) and that no initial time-step was required since one had been estimated

from the last transient solution for the previous SOLVE line. Finally, the fourth SOLVE line performs the steady-state solution at V1=-1 and V2=0.

```
METHOD 2ND TAUTO AUTONR
SOLVE  V1=1 V2=0
SOLVE  V1=2 TSTART=1E-12 TSTOP=25E-9 RAMPTIME=10E-9
+      OUTF=UP1
SOLVE  V1=-1 TSTOP=100E-9 RAMPTIME=20E-9 OUTF=DOWN1
SOLVE  V1=-1 V2=0
```

In all of the above examples, to obtain convergence for the equations used it is necessary to supply a good initial guess for the variables to be evaluated at each bias point. The PADRE solver uses this initial guess and iterates to a converged solution. Provided a reasonable grid is used, almost all convergence problems in PADRE are caused by a poor initial guess to the solution. During a bias ramp, the initial guess for any bias point is provided by a projection of the two previous results. Problems tend to appear near the beginning of the ramp when two previous results are not available. If one previous bias is available, it is used alone.

The LINALG line sets parameters associated with the numerical methods used for the solution of linear algebraic systems. There can be more than one LINALG line in a single simulation, so that parameters can be altered. Subsequent LINALG lines only alter those coefficients specified. The METHOD, PRECONDITION and ACCELERATION vectors determine the basic method, the preconditioner and the iterative acceleration technique used to perform the linear algebraic solutions. Each of these consists of a set of single digits corresponding exactly to the PDE groupings on the SYSTEM line, defined as follows:

```
Method        =  1     Direct sparse LU decomposition
                 2     Point iterative
                 3     ABF

Precondition  =  1     LDU
                 2     ILU
                 3     Block-ILU

Acceleration  =  1     Conjugate gradients
                 2     Orthomin
                 3     Biconjugate gradients
                 4     CGS
                 5     Generalized conjugate residuals
                 6     GMRES
                 7     CGNE
                 8     LSQR
```

```
9     USQR
10    BICR
11    BICG/CGS
12    BICG1
13    BICG2
```

Similarly, the S.METHOD, S.PRECONDITIONER and S.ACCELERATION vectors optionally specify the method, preconditioner and acceleration for any plug-in smoothing loops. The following is an example for a 2 carrier system, where we set the linear algebra default to iterative methods, but redefine all the accelerators to use bicg2.

```
SYSTEM   NEWTON CARR=2
LINALG   ITER.DEF ACCEL=13 S.ACCEL=13,13,13.
```

Next, an AC example is presented. It is assumed that the device to be simulated has 3 electrodes. Starting from solved DC conditions at V1 = 0, 0.5, 1.0, 1.5 and 2.0 volts, 10 mV AC signals of frequency 1 MHz, 10 MHz, 100 MHz, 1 GHz, 10 GHz and 100GHz are applied to each electrode in the device. And for each of these frequencies, also compute the current noise parameters, and write the output in the "noisefile". Note that the number of AC solutions to be performed is 5*6*3=90.

```
SOLVE  PROJ V1=0 V2=0 V3=0 VSTEP=0.5 NSTEPS=4 ELECT=1
+      AC FREQ=1E6 FSTEP=10 MULT.F NFSTEP=5 VSS=0.01
+      NOISE  I.NOISE=t  NFILE=noisefile
```

Advanced solution techniques

Obtaining Solutions Around Breakdown Voltage

Obtaining solutions around the breakdown voltage can be difficult using the standard PADRE approach. It requires special care in the choice of voltage steps and also in interpreting the results. The curve tracer is the most effective method in many cases. The following statement traces an IV curve using terminal 4. Extrapolation is used to obtain initial guesses, and the simulation will terminate when V4 is 5 volts or I4 is 1E-4 Amps/mum. The initial step will be <= .05*5 = .25v on V4 and <= 5E-6A in I4.

```
CONTIN  ELECT=4 SIGMA=.05 VMAX=5 IMAX=1E-4 TOL.SIG=5E-3 EXTRA
```

Using Current Boundary Conditions

In all of the examples considered in the basic description of the SOLVE statement it was assumed that voltages were being forced and currents were being measured. PADRE also supports the reverse case through current boundary conditions. The current through the electrode is specified in the SOLVE statement and the voltage at the contact is calculated. Current boundary conditions are set using the CONTACT statement. The syntax of the SOLVE statement is altered once current boundary conditions are specified. Often it is best to ramp the voltage until the current is above 1pA/mm and then switch to current forcing. When interpreting the results, it is important to remember the calculated voltage on the electrode with current boundary conditions is stored as the 'internal bias'.

Pseudo continue

```
TRAp       =  logical   (default is true)
DGmin      =  real      (default is 3.0e-9)
A.Trap     =  real      (default is 0.5)
N.Trap     =  real      (default is 2)
M.Trap     =  real      (default is 1)
I.Trap     =  real      (default is 10)
MAxneg     =  integer   (default is 10)
DI.Trap    =  real      (default is 1.0e-5)
DV.Trap    =  real      (default is 1.0e-5)
OUT.Trap   =  logical   (default is true)
IGN.Inner  =  logical   (default is false)
STop       =  logical   (default is true)
```

TRAP specifies the initiation of a "pseudo-continuation" method which attempts to detect if a solution process starts to diverge. If it does, the electrode bias steps taken from the initial guess are reduced by the multiplicative factor A.TRAP (up to a maximum of I.TRAP cutbacks or until DI.TRAP or DV.TRAP are violated - see below). One criterion for determining divergence is that the maximum number of Newton iterations is exceeded with any group of PDEs. Also, PADRE can monitor internal quantities - e.g. the norm of the residuals (RHSNORM), the norm of the updates (XNORM) and the size of any damping coefficient - during the Newton iterations if it is clear that convergence is slow (this feature can be disabled by specifying IGN.INNER). Divergence tendencies are ignored if the residual norms are below a small threshold (to account for round-off error), DGMIN. N.TRAP and M.TRAP specify the number of Newton iterations to be ignored before checking and the consecutive number of occurrences on which the norm must go up before the TRAP feature is enabled. A trap can also be initiated by reaching a maximum

number of consecutive iterations which produce negative concentrations (MAXNEG). DI.TRAP and DV.TRAP specify the minimum size of the allowed bias steps after application of A.TRAP; DV.TRAP is given in volts, while DI.TRAP is defined in a relative sense. OUT.TRAP controls whether solution output files are written for any points computed due to a trap; the file names used are derived from the name given to the originally attempted point, with the string "Xn" (n is a digit, starting from 0) appended. If the TRAP feature is not used, setting the STOP parameter will force the simulator to suspend execution if convergence is not reached.

Damping

```
DAMPEd    =  character  (default is "single")  [Expert]
ITDamp    =  integer    (default is 1)         [Expert]
DElta     =  real       (default is 1.0e-6)    [Expert]
DAMPLoop  =  integer    (default is 10)        [Expert]
DFactor   =  real       (default is 10.0)      [Expert]
DPower    =  real       (default is 2)         [Expert]
DVLimit   =  real       (default is 10.0)      [Expert]
TRUncate  =  logical    (default is false)     [Expert]
VMargin   =  real       (default is 0.01*kT/q) [Expert]
```

The DAMPED parameter indicates the use of a more sophisticated damping scheme proposed by Bank and Rose (this is the recommended option, particularly for large bias steps). The method is recommended for single PDE loops (DAMPED="single"), but it can also be performed for coupled groups of PDEs (DAMPED="all"). To turn this damping method off, use DAMPED="none". The Bank/Rose damping scheme is controlled by the parameters ITDAMP, DELTA, DAMPLOOP, DFACTOR and DPOWER which specify the iteration on which to start to damp, the reduction threshold, maximum number of damping loops, update reduction factor and update reduction power respectively. Other damping schemes that can be performed either separately or in conjunction with Bank/Rose include a simple limit on any potential updates (DVLIMIT) and truncation of all potentials so that they lie between the minimum and maximum possible voltages (TRUNCATE) with a margin on either end of VMARGIN.

Plot and log statement

The PLOT.1D line plots a specific quantity along a line segment through the device (mode A), or plots an I-V curve of data (mode B). These data are then used as inputs in Rappture for efficient graphical presentation.

```
X.Start or A.X  =  real
```

```
Y.Start or A.Y  =  real
X.End or B.X    =  real
Y.End or B.Y    =  real
Z.pos           =  real      (default is 0)
COordinate      =  character
```

The above parameters define the Cartesian coordinates of the start (A.X,A.Y) and the end (B.X,B.Y) of the line segment along which the specified quantity is to be plotted. By default, the data is plotted as a function of distance from the start (A) on the z-plane closest to the z-coordinate given by Z.POS; the data can alternatively be plotted against a particular coordinate ("x", "y" or "z") along the same segment. The line segment may not be defaulted, and it is required in mode A.

function is one of:

```
POTential    =  logical   Mid-gap potential
QFN          =  logical   Electron quasi-fermi level
QFP          =  logical   Hole quasi-fermi level
N.temp       =  logical   Electron temperature
P.temp       =  logical   Hole temperature
DOping       =  logical   Total net impurity concentration
IOn.imp      =  logical   Net ionized impurity concentration
ELectrons    =  logical   Electron concentration
Holes        =  logical   Hole concentration
NET.CHarge   =  logical   Net charge concentration
NET.CArrier  =  logical   Net carrier concentration
J.Conduc     =  logical   Conduction current
J.Electr     =  logical   Electron current
V.Electr     =  logical   Electron velocity
J.Hole       =  logical   Hole current
V.Hole       =  logical   Hole velocity
J.Displa     =  logical   Displacement current
J.Total      =  logical   Total current
E.Field      =  logical   Electric field
REcomb       =  logical   Net recombination
BAND.Val     =  logical   Valence band potential
BAND.Con     =  logical   Conduction band potential
               or
X.Axis       =  character
Y.Axis       =  character
Freq         =  real
INFile       =  character
```

The above parameters specify the quantity to be plotted. There is no default. In mode A, one of the solution variables is plotted versus distance into the device. For vector quantities, the magnitude is plotted. In mode B, terminal characteristics can be plotted against each other by choosing the value to be plotted on each axis (XAXIS=,YAXIS=). Quantities available for

plotting include applied device biases (XAXIS/YAXIS=VA1, VA2, ..., VA9, VA0), actual contact bias which may differ from applied bias in the case of lumped element boundary conditions (V1,V2, etc.), terminal current (I1,I2, etc.), AC capacitances (C11,C12,C21, etc.), AC conductance (G11,G12,G21, etc.) and AC admittance (Y11,Y12,Y21, etc.). Also, voltages at circuit nodes can be plotted as "Vnode_name" where "node_name" is the name of the node, and currents for labeled lumped elements can be plotted by name as "Iname" where "name" is the lumped element name (see LUMP.ELEMENT line). Additionally, any of the voltages or currents can be plotted versus time for transient simulations, and any AC quantity can be plotted versus frequency. For plotting AC parameters versus bias at a particular frequency, use the FREQUENCY parameter. The values plotted are the I-V or AC data of the present run, provided a log is being kept (see the LOG line). Alternatively, a different log file can be loaded with INFILE.

data-control

```
ABsolute     =  logical  (default is false)
LOgarithm    =  logical  (default is false)
X.Log        =  logical  (default is false)
DEcibels     =  logical  (default is false)
INTegral     =  logical  (default is false)
NEGative     =  logical  (default is false)
INVerse      =  logical  (default is false)
D.order      =  real     (default is 0)
X.Component  =  logical  (default is false)
Y.Component  =  logical  (default is false)
MIx.mater    =  logical  (default is false)  [Expert]
SPline       =  logical  (default is false)
NSpline      =  logical  (default is 100)
```

ABSOLUTE specifies that the absolute value of the variable be taken. For rapidly varying quantities, the LOGARITHM (X.LOG) is often more revealing. To get the true logarithm of a quantity, specify ABSOLUTE and LOGARITHM - the absolute is taken first and there is no danger of negative arguments. DECIBELS converts the function to a measure of gain in decibels defined by INTEGRAL plots the integral of the specified ordinate, and NEGATIVE negates the ordinate values; INVERSE plots the value of the function raised to the (-1) power. D.ORDER0 specifies that the derivative of the function (order=D.ORDER) with respect to the ordinate be plotted. X.COMPONENT and Y.COMPONENT force the x and y components respectively of any vector quantities to be plotted as opposed to the default total magnitude. MIX.MATER specifies that local vector averaging should be done over all materials to which the node belongs as opposed to just

the hierarchical choice (see the PRINT line). The SPLINE option indicates that spline-smoothing should be performed on the data using NSPLINE interpolated points (maximum is 500).

Examples

The following plots a graph of potential along a straight line from (0.0,0.0) to (5.0,0.0):

```
PLOT.1D  POTEN A.X=0 A.Y=0 B.X=5 B.Y=0
```

In the next example, the log of the electron concentration is plotted from (1.0,-0.5) to (1.0,8.0) with bounds on the plotted electron concentration of 1.0e10 and 1.0e20. A spline interpolation is performed with 300 interpolated points. The non-spline-interpolated points are marked.

```
PLOT.1D  ELECT LOG A.X=1 A.Y=-.5 B.X=1 B.Y=8
+        MIN=10 MAX=20 SPLINE NSPL=300 POINTS
```

In the following example, the current in contact 1 is plotted as a function of contact 2 voltage, then the curve is compared with a previous run.

```
PLOT.1D  X.AXIS=V2 Y.AXIS=I1
PLOT.1D  X.AXIS=V2 Y.AXIS=I1 INF=logf0 UNCH
```

The following plots the actual contact voltage on a contact versus the applied voltage.

```
PLOT.1D  X.AXIS=V3 Y.AXIS=VA3
```

Finally, the following shows a plot of two capacitance components versus the log of frequency. A different line type is chosen for the second component.

```
PLOT.1D  X.AXIS=FREQ Y.AXIS=C21 X.LOG
PLOT.1D  X.AXIS=FREQ Y.AXIS=C31 X.LOG UNCH LINE=4
```

Log Statement

The LOG line allows the I-V and/or AC characteristics of a run to be logged to disk. Then, they are ready for plotting using Rappture. Any I-V or AC data subsequent to the line is saved. If a log file is already open, it is closed and a new file opened. The following example specifies the command: Save the I-V data in a file called IV1 and AC data in AC1.

```
LOG OUTF=IV1 ACFILE=AC1
```

PADRE Simulation Examples

In this section we will describe two simulation examples: (1) simulation of a pn-diode and (2) MOSFET simulation.

a) pn-diode simulation

In this example, we demonstrate to the user of PADRE simulation software how one can specify the mesh in the device, regions, electrodes and doping profiles. The transport model includes Shockley-Read-Hall generation/recombination process, concentration dependent mobility model, field dependent mobility and includes the impact ionization process. Both electrons and holes are included in the simulation. In the first part, the user solves for the equilibrium quantities and plot of different variables is demonstrated. Then, the applied bias on the anode is incremented by 0.1 V up to 0.6 V. Again, an example of plotting the internal variables under non-equilibrium conditions is presented. The listing of the pn-junction simulation example using PADRE syntax is given below:

```
title    pn diode (setup)
options  PO

$ Mesh Specification
mesh     rect nx=60 ny=3 outf=pn.mesh
x.m      n=1  l=0 r=1
x.m      n=30 l=0.5 r=0.8
x.m      n=60 l=1 r=1.05
y.m      n=1  l=0 r=1
y.m      n=3   l=1 r=1

$ Regions specification
region   num=1 ix.l=1 ix.h=60 iy.l=1 iy.h=3 silicon
elec     num=1 ix.l=1 ix.h=1 iy.l=1 iy.h=3
elec     num=2 ix.l=60 ix.h=60 iy.l=1 iy.h=3

$ Doping specification
dop      reg=1 n.type conc=1e17 uniform
dop      reg=1 p.type conc=2e17 x.l=0 x.r=0.5 y.top=0 y.bot=1 uniform
plot.1d  log dop abs a.x=0 b.x=1 b.y=0.5 a.y=0.5 points ascii outf=dop.plot

$ Specify models
models srh conmob fldmob impact
system electrons holes newton

$ Solve for initial conditions
```

```
solve init
plot.1d pot a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=pot.plot
plot.1d qfn a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=qfn.plot
plot.1d qfp a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=qfp.plot
plot.1d ele a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=ele.plot
plot.1d hole a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=hole.plot
plot.1d net.charge a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=ro.plot
plot.1d e.field a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=efield.plot

$ Solve for applied bias
solve prev
solve proj vstep=0.1 nsteps=6 elect=1 ascii outf=iv.plot
plot.1d pot a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=potiv.plot
plot.1d qfn a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=qfniv.plot
plot.1d qfp a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=qfpiv.plot
plot.1d ele a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=eleiv.plot
plot.1d hole a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=holeiv.plot
plot.1d net.charge a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=roiv.plot
plot.1d e.field a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=efieldiv.plot
plot.1d j.electr a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=jelectr.plot
plot.1d j.hole a.x=0 b.x=1 a.y=0.5 b.y=0.5 ascii outf=jhole.plot

end
```

Note that the above program is part of the PN-junction learning lab and all these output data files generated with the `plot.1d` statement are plotted using Rappture.

b) MOSFET simulation example

Like the pn-diode, this example as well demonstrates the users of PADRE how to specify the mesh, regions, materials, electrodes/contacts and doping. The model statement specifies that the simulation is at 300K, band-gap narrowing process is included (important for the source and drain regions), concentration and field dependent mobility. Next, an example is given how to calculate device transfer characteristics and the device output characteristics. Finally, examples are given on loading and saving device simulation parameters, such as IV characteristics, and internal parameters like potential, charge, etc.

```
#Example for Simulation of a 25nm gate length MOSFET device
#
#(1) Transfer Characteristics
#    Students need to extract:
#       - threshold voltage
#       - Subthreshold slope
#       - maximum transconductance

#(2) Device output characteristics with and without impact ionization
#    included in the model
```

```
#    Students need to extract:
#       - Is there a punchthrough?
#       - Is this a long-channel or a short-channel device?
#       - Is there DIBL in the output characteristics
#       - Breakdown voltage
#       - Examine series resistance effects when source drain doping is
lowered
#         to 1E19

# Set up the mesh
MESH  RECT NX=50 NY=75
X.M N=1 LOC=0
X.M N=15 LOC=0.05  RATIO=1.05
X.M N=25 LOC=0.0625 RATIO=0.8
X.M N=35 LOC=0.075 RATIO=1.05
X.M N=50 LOC=0.125

Y.M N=1 LOC=0
Y.M N=5 LOC=0.0012 RATIO=1.05
Y.M N=25 LOC=0.022 RATIO=0.8
Y.M N=50 LOC=0.032 RATIO=1.05
Y.M N=75 LOC=0.1

# Set regions and materials
REGION NUM=1 ix.l=1 ix.h=50 iy.l=1 iy.h=5 oxide
REGION NUM=2 ix.l=1 ix.h=15 iy.l=5 iy.h=50 silicon
REGION NUM=3 ix.l=35 ix.h=50 iy.l=5 iy.h=50 silicon
REGION NUM=4 ix.l=15 ix.h=35 iy.l=5 iy.h=50 silicon
REGION NUM=5 ix.l=1 ix.h=50 iy.l=50 iy.h=75 silicon

# Set up the electrodes/contacts
ELECT NUM=1 ix.l=1 ix.h=15 iy.l=5 iy.h=5
ELECT NUM=2 ix.l=35 ix.h=50 iy.l=5 iy.h=5
ELECT NUM=3 ix.l=15 ix.h=35 iy.l=1 iy.h=1
ELECT NUM=4 ix.l=1 ix.h=50 iy.l=75 iy.h=75

# Set up doping density
dop region=2,3 unif conc=1E20 n.type
dop region=4,5 unif conc=5E18 p.type
contact num=3 n.polysilicon

# Definition of the models used
MODELS TEMP=300 BGN CONMOB FLDMOB
SYSTEM NEWTON CARR=1 ELECTRONS

# DEVICE SIMULATION PART I

# Calculate the device transfer characteristics
log outf=idvg
SOLVE INIT
PLOT.3D POTEN ELECT OUTFILE=plt.wmc
SOLVE PREV V1=0 V2=0.10 V3=0 VSTEP=0.1 NSTEPS=20 ELECT=3

# DEVICE SIMULATION PART II

# Calculate the device output characteristics without impact ionization
SOLVE INIT
```

```
SOLVE V3=0.5 OUTF=VG1.DAT
SOLVE V3=1.0 OUTF=VG2.DAT
SOLVE V3=1.5 OUTF=VG3.DAT
SOLVE V3=2.0 OUTF=VG4.DAT

LOAD INFILE=VG1.DAT
log outf=idvd1
SOLVE V1=0 V2=0 VSTEP=0.1 NSTEPS=20 ELECT=2
LOAD INFILE=VG2.DAT
log outf=idvd2
SOLVE V1=0 V2=0 VSTEP=0.1 NSTEPS=20 ELECT=2
LOAD INFILE=VG3.DAT
log outf=idvd3
SOLVE V1=0 V2=0 VSTEP=0.1 NSTEPS=20 ELECT=2
LOAD INFILE=VG4.DAT
log outf=idvd4
SOLVE V1=0 V2=0 VSTEP=0.1 NSTEPS=20 ELECT=2

END
```

References

1 http://www.nanohub.org/resource_files/tools/padre/doc/ .