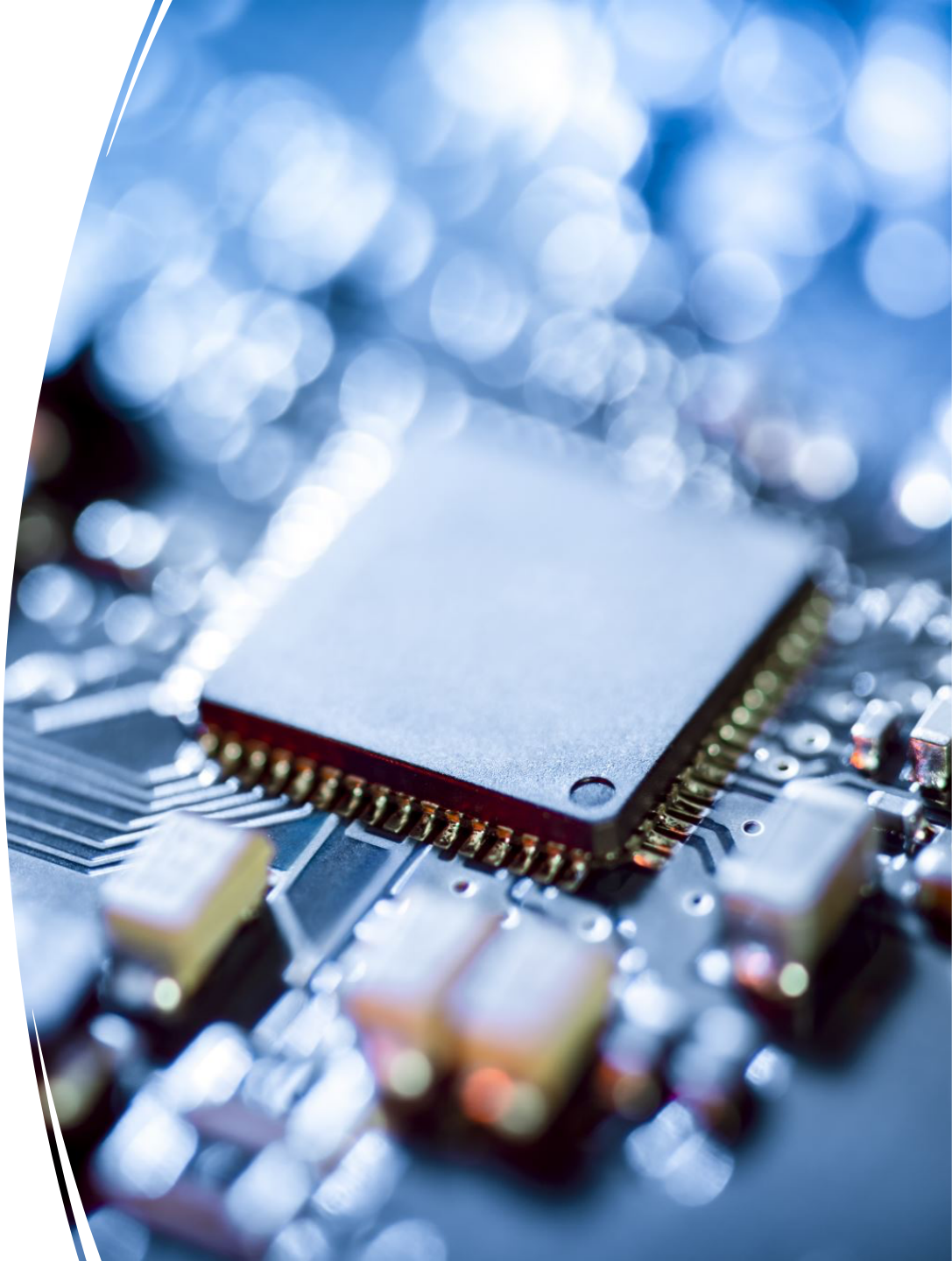
# Semiconductor Device Modeling and Simulation

Dragica Vasileska

*Arizona State University*

National Nanotechnology Coordinated Infrastructure



# Introduction to Silvaco VictoryDevice

- Some general comments
- VictoryDevice Syntax
  - Structure specification
  - Materials and models specification
  - Numerical method selection
  - Solution specification
  - Results analysis
- VictoryDevice Extract statements description
- Physical Models:
  - Mobility Modeling
  - Generation-Recombination Processes Modeling
- Invoking Energy Balance Model

# VictoryDevice Syntax

- The form of the input file statements is:

  **<STATEMENT>** **<PARAMETER> = <VALUE>**

  The parameter can be real, integer, character and logical

- The order in which the ATLAS commands occur is the following:

  a) **Structure specification:** MESH, REGION, ELECTRODE, DOPING

  b) **Material models specification:** MATERIAL, MODELS, CONTACT, INTERFACE

  c) **Numerical method selection:** METHOD

  d) **Solution specification:** LOG, SOLVE, LOAD, SAVE

  e) **Results analysis:** EXTRACT, TONYPLOT

# Numerical Method Selection

- **METHOD** statement – allows for several different choices of numerical method selection. The numerical methods that can be specified within the METHOD statement include

  **GUMMEL** ➔ Decoupled Gummel scheme which solves the necessary equations sequentially, providing linear convergence. Useful when there is weak coupling between the resultant equations.

  **NEWTON** ➔ Provides quadratic convergence and needs to be used for the case of strong coupling between the resultant equations.

  **BLOCK NEWTON** ➔ more efficient than NEWTON method

  ```
  method  gummel  block  newton
  method  carriers=0
  ```

  One can also alter the parameters relevant for the numerical solution procedure:

  **CLIMIT.DD** ➔ Specifies minimum value of the concentration to be resolved by the solver.

  **DVMAX** ➔ Maximum potential update per iteration. Default value is 1V.

# Solution Specification

VictoryDevice allows for four different types of solutions to be calculated: ***DC, AC, small signal*** and ***transient solutions***. The previously set bias at a given electrode is remembered and does not need to be set again.

**❶ DC solution procedures and statements:**

- A stable DC solution is obtained with the following two-step procedure:
  - Find good initial guess by solving equilibrium case (initial guess is found based on the local doping density)

    **solve init**
  - Step the voltage on a given electrode for a convergent solution:

    **solve vcollector=2.0**

    **solve vbase=0.0 vstep=0.05 vfinal=1.0 name=base**
- To overcome the problems with poor initial guess, one can use the **TRAP** statement, where **MAXTRAPS** is the maximum allowed number of trials (default value is 4)

  **method trap**

  **solve init**

  **solve vdrain=2.0**

# Solution Specification

To generate a family of curves, use the following set of commands:

```
solve  vgate=1.0  outf=solve_vgate1.str
solve  vgate=2.0  outf=solve_vgate2.str

load infile=solve_vgate1.str
log outfile=mos_drain_sweep1.log
solve  name=drain  vdrain=0  vfinal=3.3  vstep=0.3

load infile=solve_vgate2.str
log  outfile=mos_drain_sweep2.log
solve  name=drain  vdrain=0  vfinal=3.3  vstep=0.3
```

The **log** statement is used to save the Id/Vds curve from each gate voltage to separate file.

# Solution Specification

## ❷ AC solution procedures and statements:

The AC simulation is simply an extension to the DC simulation procedure. The **final result** of this analysis is the **conductance** and **capacitance** between each pair of electrodes. The two types of simulations are:

- Single frequency solution during a DC Ramp:

  **solve vbase=0. vstep=0.05 vfinal=1 name=base AC freq=1e6**

- Ramped frequency at a single bias:

  **solve vbase=0.7 ac freq=1e9 fstep=1e9 nfsteps=10**

  **solve vbase=0.7 ac freq=1e6 fstep=2 mult.f nfsteps=10**

# Solution Specification

## ❸ Transient solution procedures and statements:

For transient solutions, one needs to use piecewise-linear, exponential and sinusoidal bias functions. For a linear ramp, one needs to specify the following parameters: **TSTART**, **TSTOP**, **TSTEP** and **RAMPTIME**.

```
solve  vgate=1.0  ramptime=1e-9  tstop=10e-9  tstep=1e-11
```

# Solution Specification

❹ **Advanced solution procedures:**

- **Obtaining solutions around a breakdown point** – uses MAXTRAPS
- **Using current boundary conditions**

  Instead of voltage, one can also specify current boundary conditions.

  This is important, for example, when simulating BJTs:

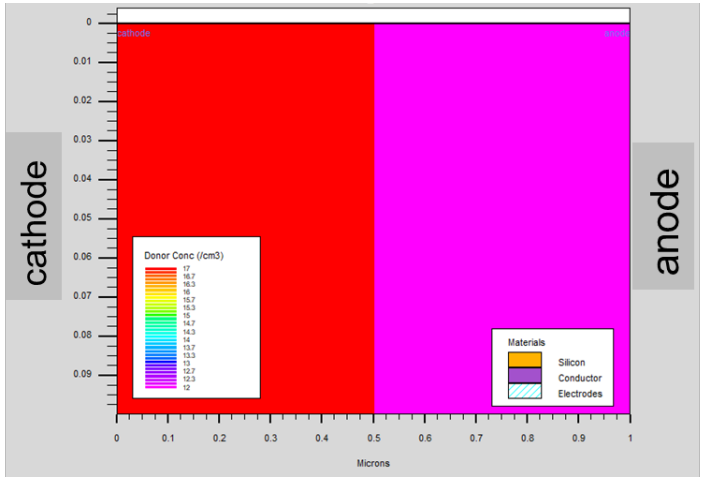
  ```
  solve  ibase=1e-6
  solve  ibase=1e-6  istep=1e-6  ifinal=5e-6  name=base
  ```

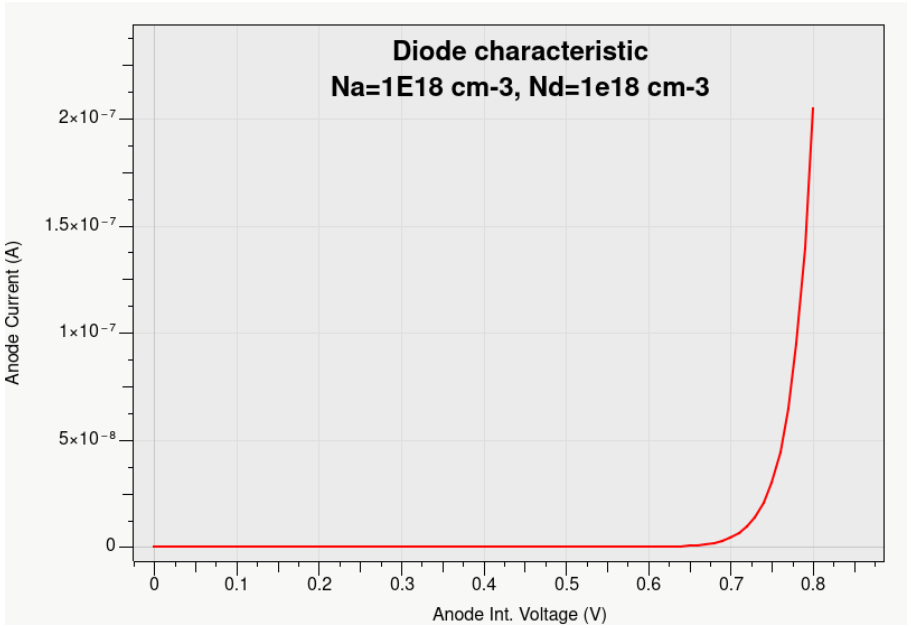
- **The compliance parameter**
  This parameter is used to stop simulation when appropriate current level is reached.

  ```
  solve  vgate=1.0
  solve  name=drain  vdrain=0  vfinal=2  vstep=0.2  \
         compl=1e-6  cname=drain
  ```

- **The curve trace capability** – enables tracing out of complex IV curves

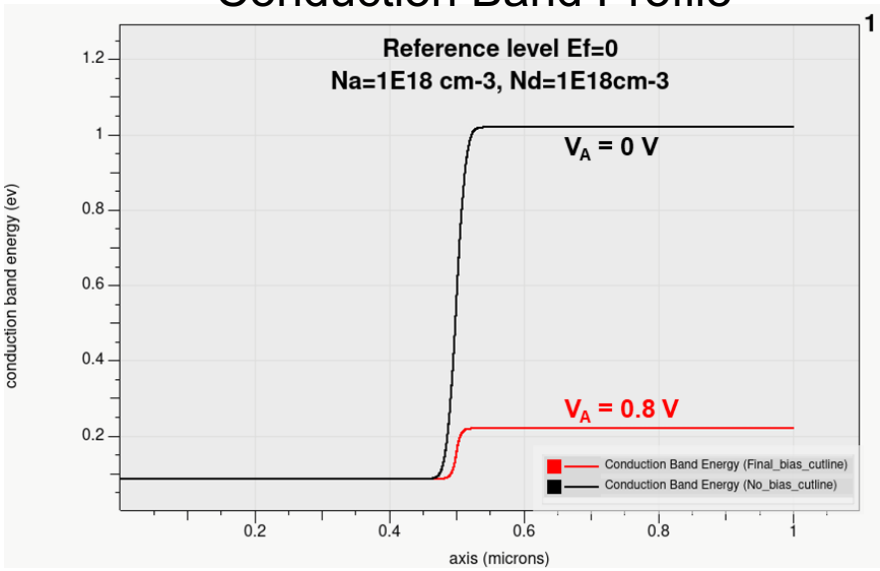
```
##################################################
# simulation of a silicon pn-diode
##################################################
go victoryd

mesh  space.mult=1.0

set x_n = 0.5
set x_p = 0.5
set x_tot = $x_n + $x_p
#
x.mesh loc=0.00 spac=0.01
x.mesh loc=$x_n spac=0.0001
x.mesh loc=$x_tot spac=0.01
#
y.mesh loc=0 spac=0.05
y.mesh loc=0.10 spac=0.05

# REGIONS AND ELECTRODES
region num=1 y.min=0 silicon
elect num=1 name=cathode y.min=0.0 y.max=0.1 x.min = 0 x.max = 0
elect num=2 name=anode y.min=0.0 y.max=0.1 x.min = $x_tot x.max = $x_tot

# DEVICE DOPING
doping uniform n.type conc=1.e18 x.max = $x_n
doping uniform p.type x.min = $x_n conc=1.e18

# MATERIAL CONTACT INTERFACE MODELS METHOD
models temp=300 conmob fldmob
method newton trap
output con.band
output val.band

# INITIAL SOLUTION
solve init

save outfile = no_bias.str
##################################################
# RAMP THE ANODE VOLTAGE
##################################################
log outf = IV.log
solve vanode = 0 vstep = 0.01 vfinal = 0.8 name = anode
save outfile = final_bias.str

quit
```



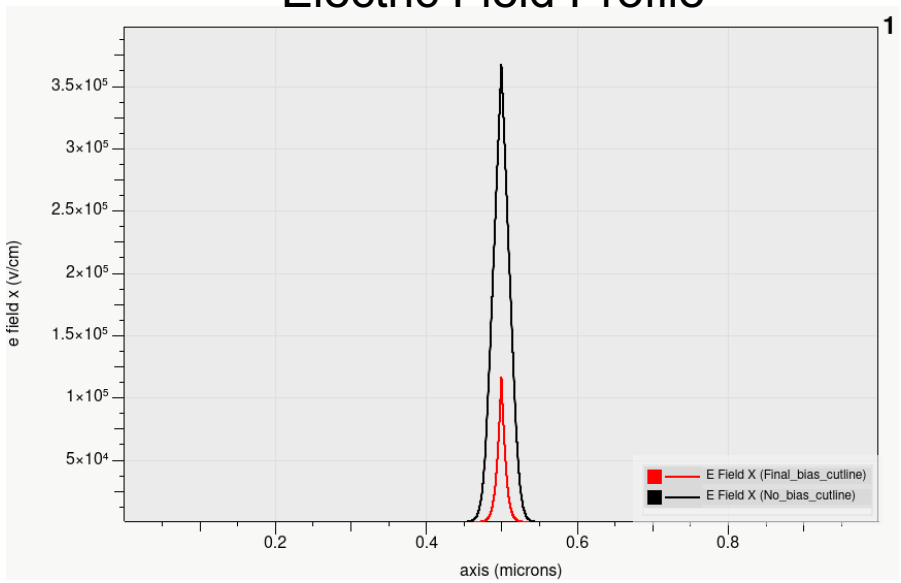



# Conduction Band and Electric Field Profile

Conduction Band Profile




Electric Field Profile



# Results Analysis

Three types of outputs are produced by the ATLAS tool: *run-time outputs, log files* and *solution files*.

**❶ Run-time outputs:**

The various parameters displayed during the SOLVE statement are listed below:

**proj** ➔ initial guess methodology used (previous, local or init)

**i, j, m** ➔ iteration numbers of the solution and the solution method

**i** = outer loop iteration number

**j** = inner loop number for decoupled solutions

**m** = solution method used: G=Gummel, B=Block, N=Newton

**x, rhs** ➔ norms of the equations being solved

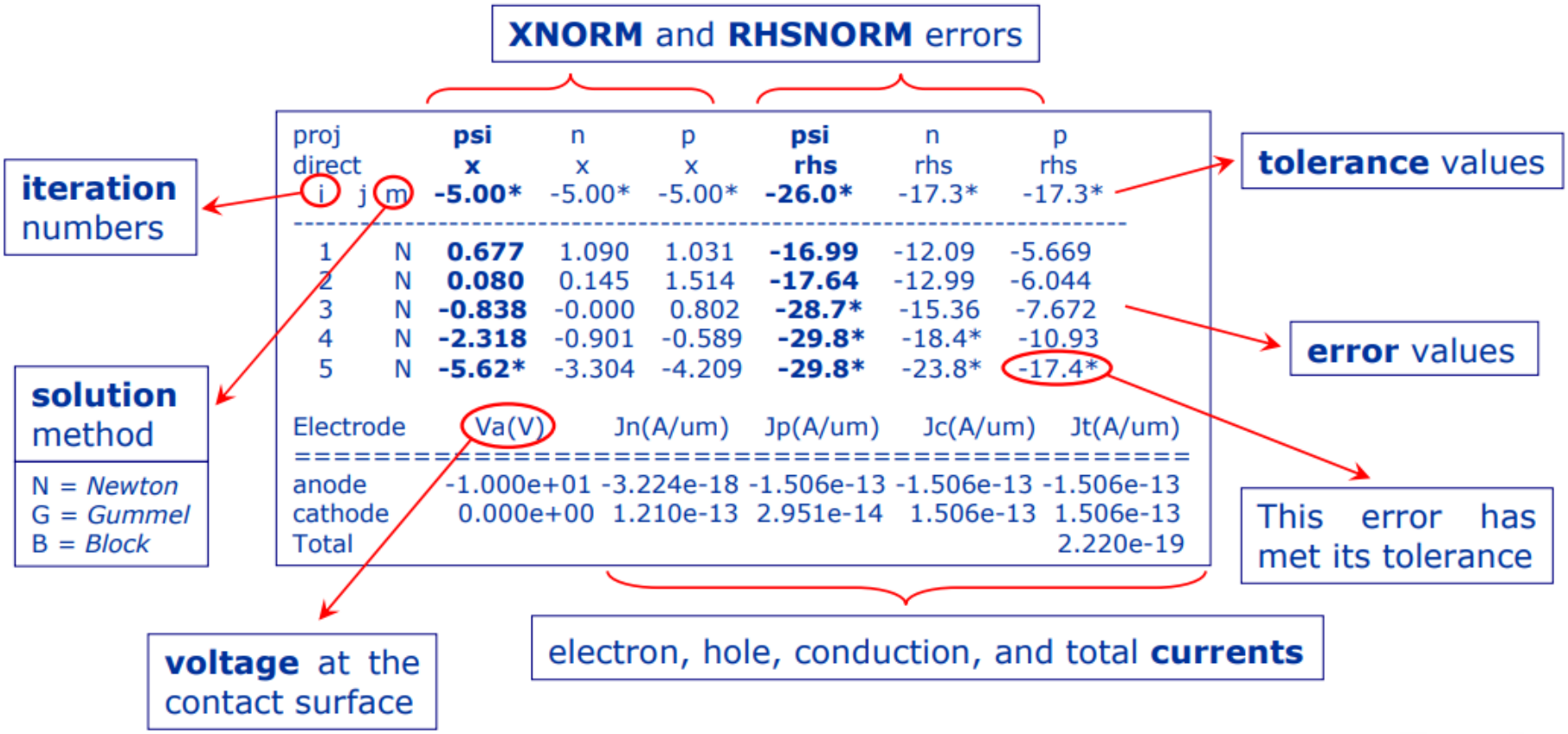
(*) ➔ the error measure has met its tolerance

**❷ Log files:**

The **LOG** parameter is used to store the device characteristics calculated using ATLAS:

**log outfile=<file_name>**

# Run-Time Output



# Results Analysis (cont'd)

### ❸ Solution files:

The syntax to produce the solution files that can be used in conjunction with VictoryVisual is:

**load infile=<file_name_in>.str**

**solve . . . .**

**save outfile=<file_name_out>.str**

### ❹ Invoking VictoryVisual

- To create overlayed plots with VictoryVisual, one needs to use the following command:

  **victoryvisual -overlay file1.log file2.log**

- To load structure files, containing mesh, doping profile information, etc., one can use the following statement:

  **victoryvisual file.str -set mx.set iv.data**

This command allows loading of the file called "*file.str*" and sets its display to a previous setup stored in the "*mx.set*" file, and then loads the file containing the *IV*-data.

# Parameter Extraction

1) Using the **EXTRACT** command that operates on previously solved **curve** or **structure** file:

- To override the default of using open log file, the name of the file that needs to be used is specified in the following manner:

```
extract  init  infile="<file_name>"
```

- Parameters that can be extracted using this EXTRACT statement include: threshold voltage, cutoff frequency, etc. The extraction of the threshold voltage is accomplished with the following statement:

```
extract  name="nvt"  xintercept(maxslope(curve (v."gate",  \
          (i."drain"))) - (ave(v."drain"))/2.0)
```

- Default file for saving results is **results.final**. The results can be stored in other file using the following options:

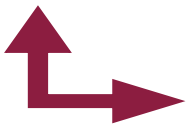
```
extract … . Datafile="<file_name>"
```

2) Using the Functions Menu in VictoryVisual allows one to use saved data for post-computation

3) Using the LOG statement for AC parameter extraction

# Parameter Extraction (cont'd)

1) The extract statement can be used in conjunction with:
   - **Process extraction**, after running VictoryProcess simulator
   - **Device extraction**, after obtaining the electrical characteristics of the device structure being simulated



   - **Log-files: contain the electrical information, more precisely, the *IV*-data obtained via the VictoryDevice simulation process**
   - **Structure files: contain the additional electrical information, such as electric field, electrostatic potential, etc.**

2) One can construct a *curve* using separate X and Y-axes. For each of the electrodes, one can choose one of the following: Voltage (*v*), Current (*i*), Capacitance (*c*), Conductance (*g*), Transient time for AC simulations (*time*), Frequency for AC simulations (*frequency*), Temperature (*temperature*), etc.

# Parameter Extraction (cont'd)

3) More in-depth description of the use of the EXTRACT statement:

- **Curve**, basic element in the extract statement. The syntax is as follows:

  **extract name="curve_name" curve(v."name", i."name")**

  "curve_name" = name of the curve to which one can refer to in later post-processing steps

- **Axes manipulation:**
  - algebra with a constant (multiplication, division)
  - operators application (abs, log, log10, sqrt)
- **Curve manipulation primitives:**

  *min*, *max*, *ave*, *minslope*, *maxslope*, *slope*, *xintercept*, *yintercept*,

  *x.val* from curve where *y.val=Y* (*val.occno=1*, would mean first occurrence of the preset condition)

- **Example:** Find max $\beta = I_C/I_B$ vs. $I_C$

  **extract "maxbeta" max(curve(i."colector", i."colector"/i."base")**

# Summary

- In this lecture we completed the discussion on basic Silvaco syntax:
  - Numerical Methods selection
  - Solution specification
  - Types of analyses
  - Parameter extraction

# Semiconductor Device Modeling and Simulation

Dragica Vasileska

*Arizona State University*

National Nanotechnology Coordinated Infrastructure