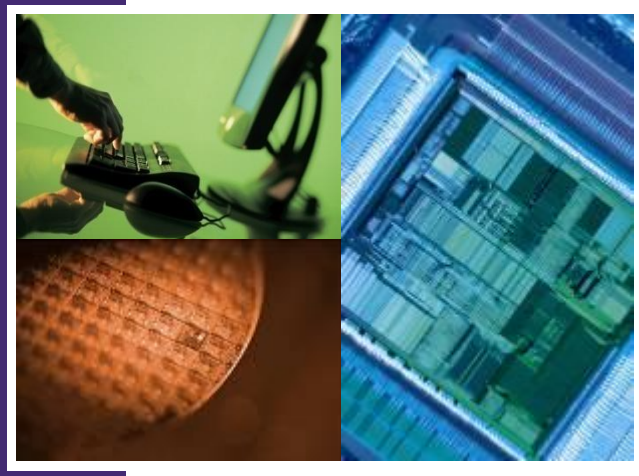
# Sentaurus TCAD Training for CMOS Application



SYNOPSYS®

Predictable Success

# Course Outline

- Sentaurus TCAD Overview
- Sentaurus Workbench
- Sentaurus Process 1D & 2D
- Tecplot SV
- Sentaurus Structure Editor – Building Meshes
- Sentaurus Device I-V simulation
- Inspect

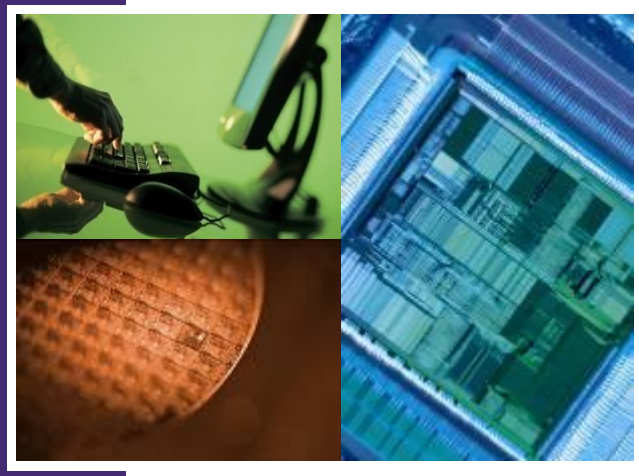
© 2007 Synopsys, Inc. (2)

# Course Outline

- 90nm nMOSFET Exercise
- C-V Device Simulation
- Breakdown Device Simulation
- pMOSFET Device Simulation
- Ligament introduction
- SolvNet Resources
  - SolvNet Introduction
  - 2D Strained Silicon 45nm CMOS Reference Flow Demo
  - 3D nMOSFET Demo

© 2007 Synopsys, Inc. (3)

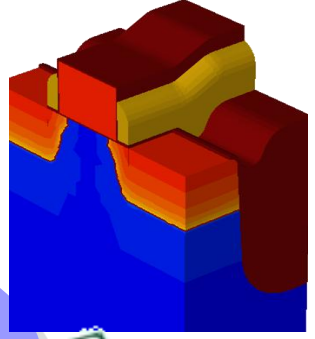
# Synopsys TCAD Overview



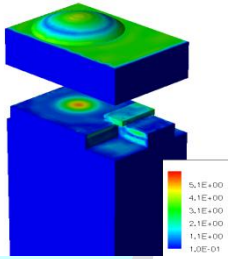
SYNOPSYS®

Predictable Success

# TCAD Application Areas



## Opto



- LED, LASER
- Image sensor
- Photodetector
- Solar cell

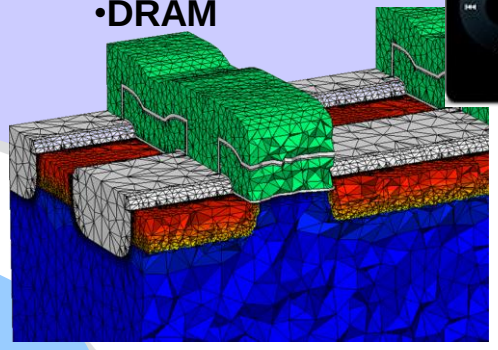
## CMOS

- Deep submicron transistor
- Sophisticated models
- Atomistic modeling


**TCAD Application Markets**

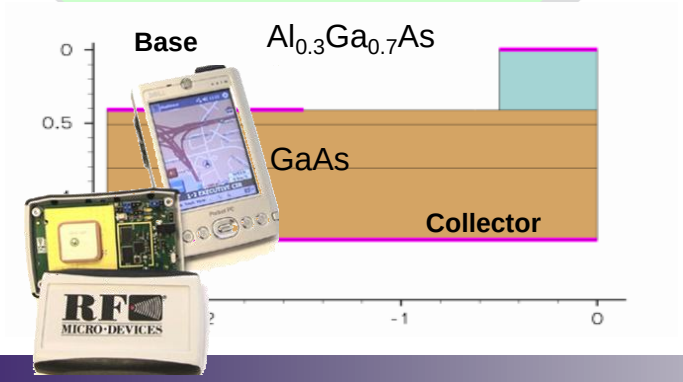
## Memory

- Flash
- DRAM



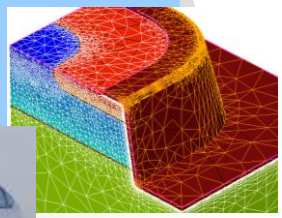
## RF

- High-speed device
- Compound semiconductor




## Power

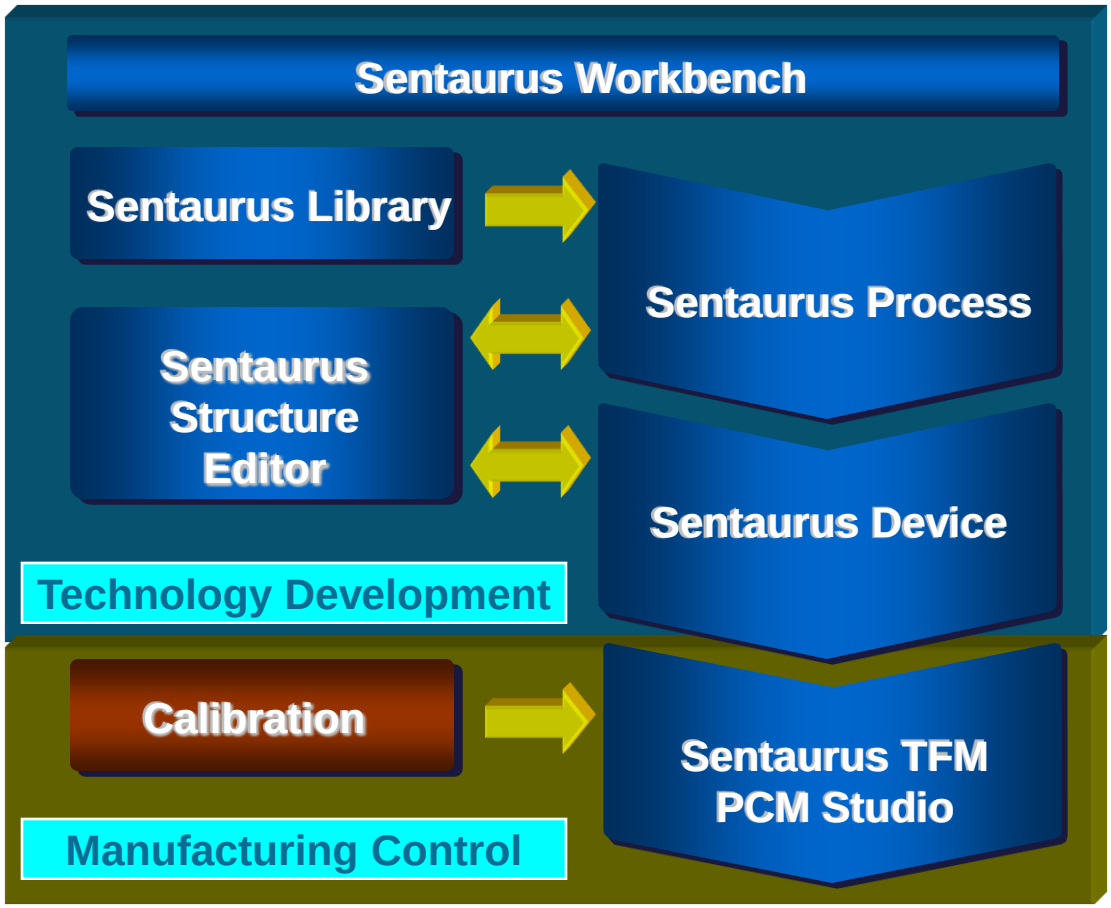
- Complex structure & process
- Mixed mode simulation



© 2007 Synopsys, Inc. (5)

SYNOPSYS®
Predictable Success

# Sentaurus Product Family



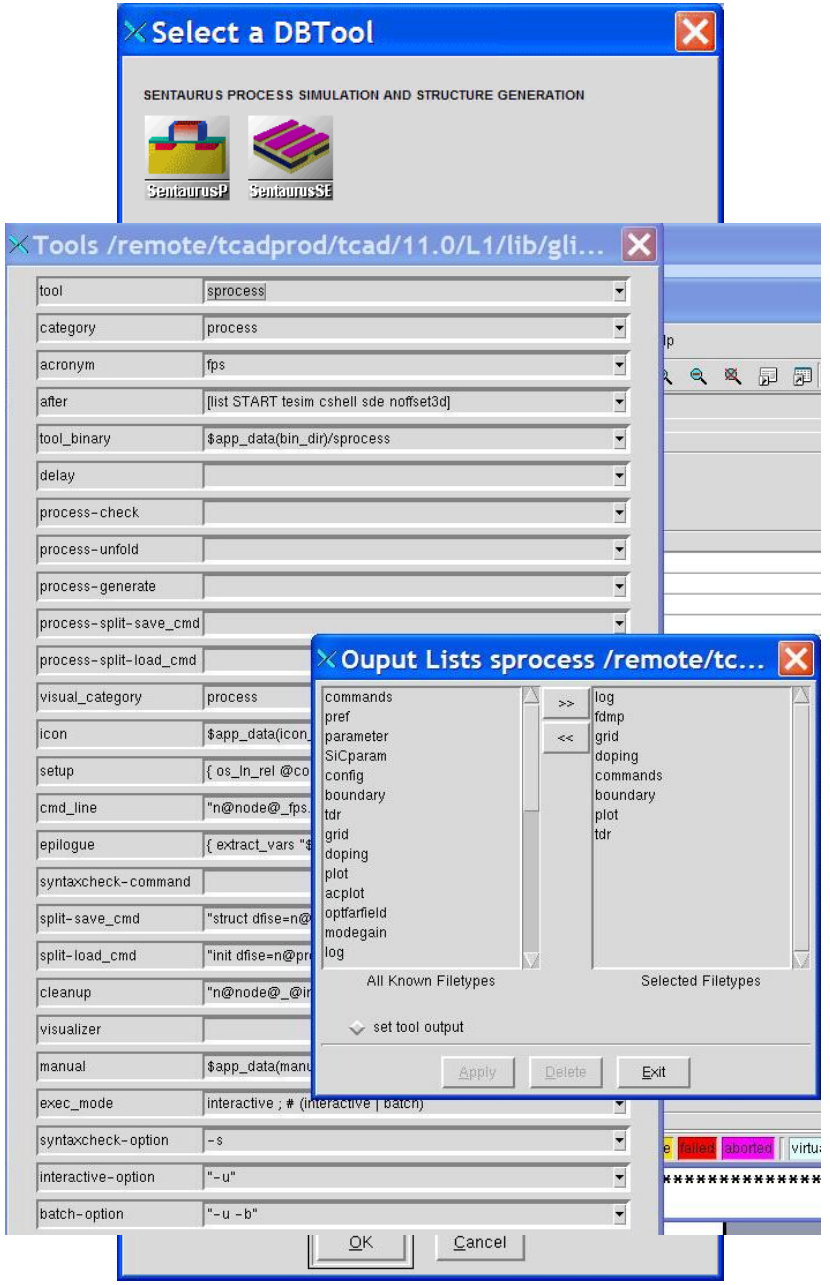

- Flexible Framework Environment
  - Advanced Visualization
- 2D/3D Process Simulation
  - Calibration Library
  - Structure Editor Interface
- 2D/3D Device Simulation
  - Structure Editor and Mesher
  - Application Specific Options
- Process Compact Models
  - Based on Calibrated Flow
  - Links Process Variation and Device Performance

**Integrated TCAD Flow from Development to Manufacturing**

© 2007 Synopsys, Inc. (6)

# Framework

- Provides a GUI-based simulation environment
- Organizes simulation projects, runs and results
- Allows large Design of Experiment (DOE) and statistical analysis
- Manages job scheduling & network computing
- Enables interactive visualization and analysis of simulation results

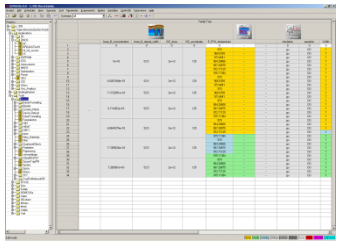



© 2007 Synopsys, Inc. (7)

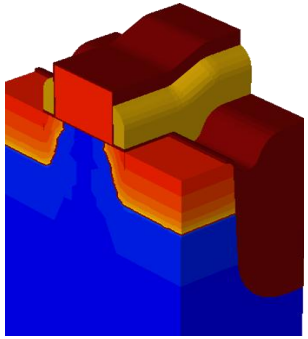
SYNOPSYS®
Predictable Success

# TCAD Product Architecture

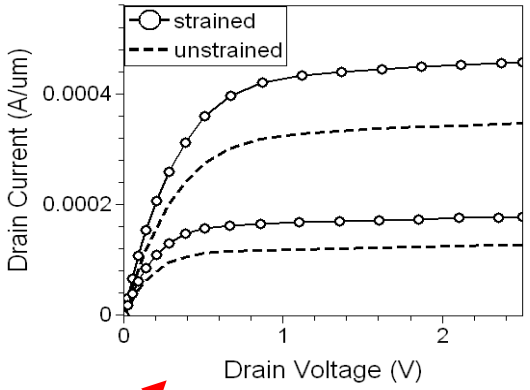
**Sentarus Workbench**



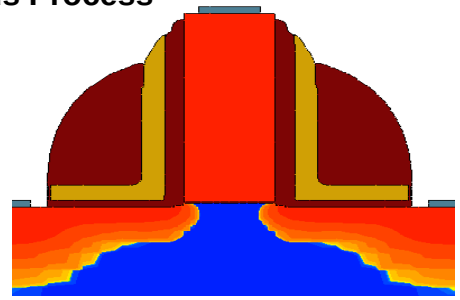
**Sentaurus Device Editor**



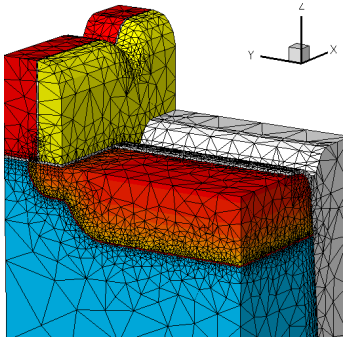
**Sentaurus Device**



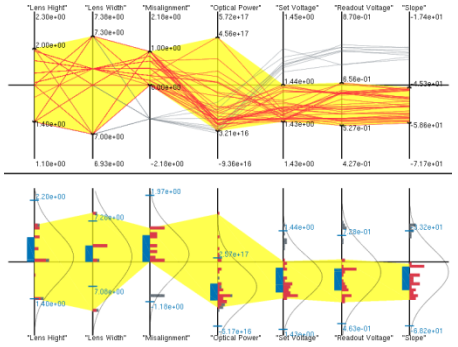

**Sentaurus Process**



**Mesh**



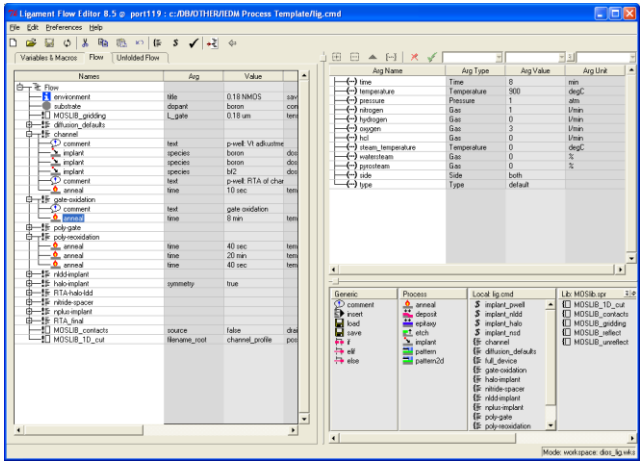
**PCM Studio**



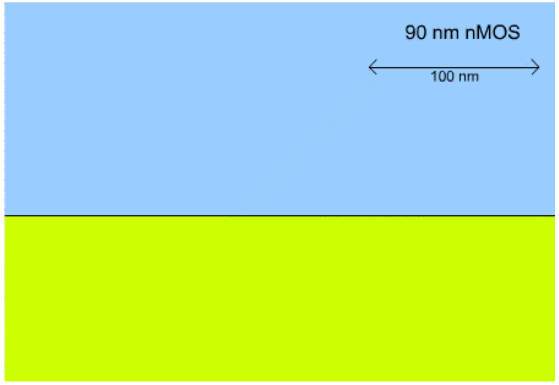
© 2007 Synopsys, Inc. (8)

# Process Simulation

- **Models wafer fabrication steps**
  - **Implantation, diffusion, oxidation & lithography models are calibrated and highly predictive**
  - **Etching and deposition are typically modeled geometrically**
- **Starts from flow description and layout**



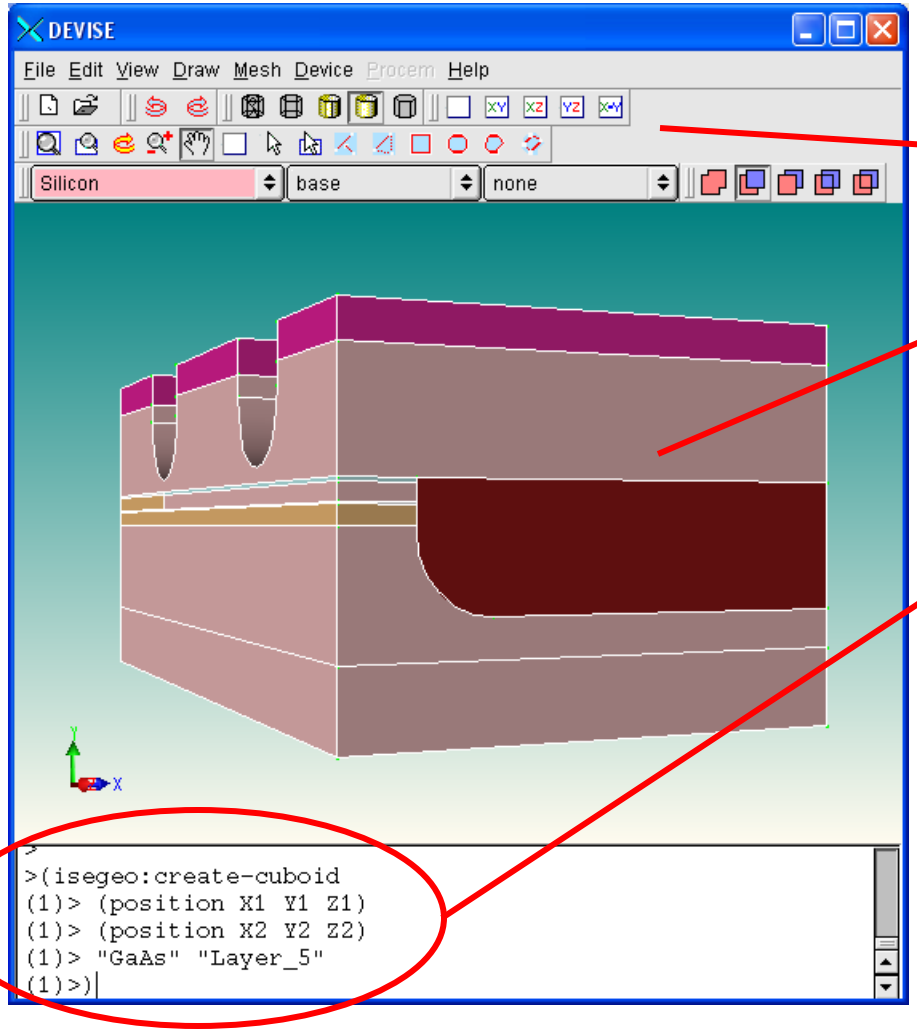
**TCAD process flow editor**




**90nm nFET**

© 2007 Synopsys, Inc. (9)

# Structure Editor

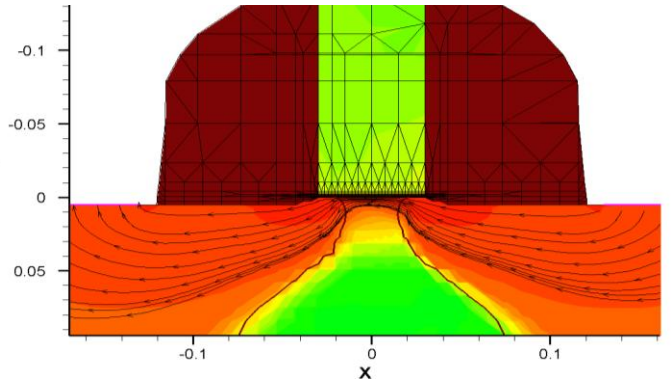



- 3D Emulation
  - From layout and geometrical photo, etch, & deposition to generate 3D structure.
- 2D and 3D iterative editor
  - Intuitive user interface
  - Interactive scripting
    - record GUI actions
    - type/paste script command
    - easy to debug

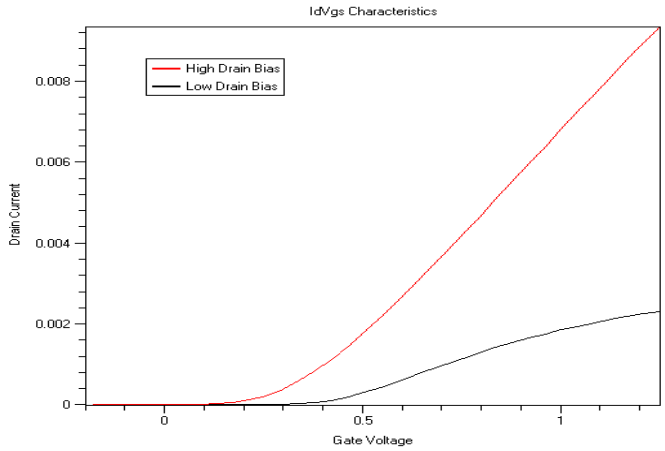
© 2007 Synopsys, Inc. (10)

# Device Simulation

- **Models the electrical, optical, mechanical & magnetic behavior of semiconductor devices**
  - **Simulation is typically performed on structures created by process simulation**
  - **Modes of simulation**
    - **Static, time-dependent, large and small signal frequency dependent and noise modeling**
- **Highly accurate CAD models can be extracted from device simulation results**




**Simulated current density and flow lines in 100nm device**



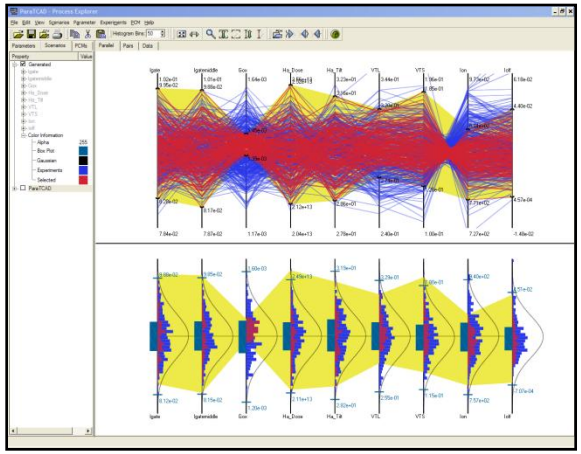

**Simulated electrical characteristics**

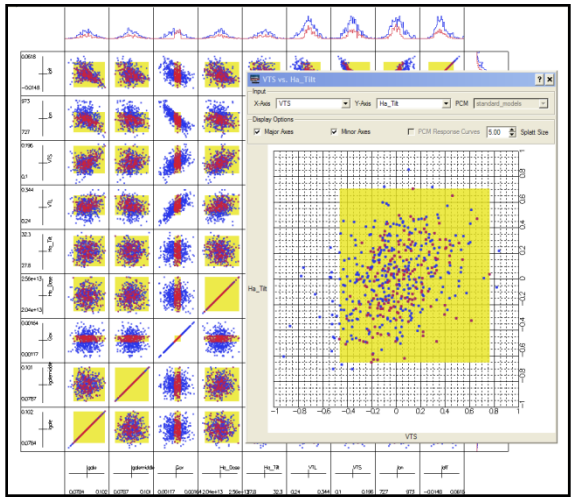
© 2007 Synopsys, Inc. (11)

SYNOPSYS®
Predictable Success

# PCM Studio

- PCM acts as a link between the Yield Management System (YMS) and TCAD
  - PCM from TCAD
  - In-process Metrology and Device Characteristics from Manufacturing
- Graphical & Statistical analysis
- Visualization of experimental data
- Visualization of process-device relations
- Algorithmic analysis
  - Reverse analysis
  - Feed-forward analysis
  - Process Window analysis

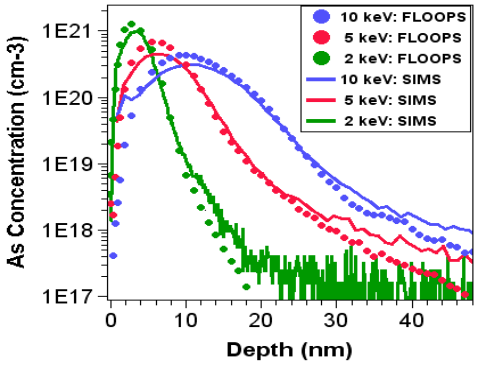


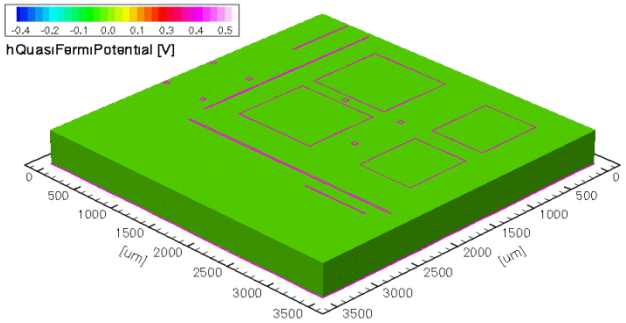


© 2007 Synopsys, Inc. (12)

# TCAD Consulting and Engineering

- *Dedicated* team of highly proficient engineers with long professional experience
- Close collaboration with the Synopsys *software engineers*
- Close collaboration with *customers* in consulting and engineering projects
- Service project examples
  - Calibration
  - Process analysis and optimization
  - Difficult simulation types, such as 3D, full-chip, SEU/SER, ESD
  - Customer specific technology templates
  - Customer specific training and know-how transfer
  - Model development and integration
  - Software integration: TCAD FabLink
  - Dedicated engineering projects




© 2007 Synopsys, Inc. (13)

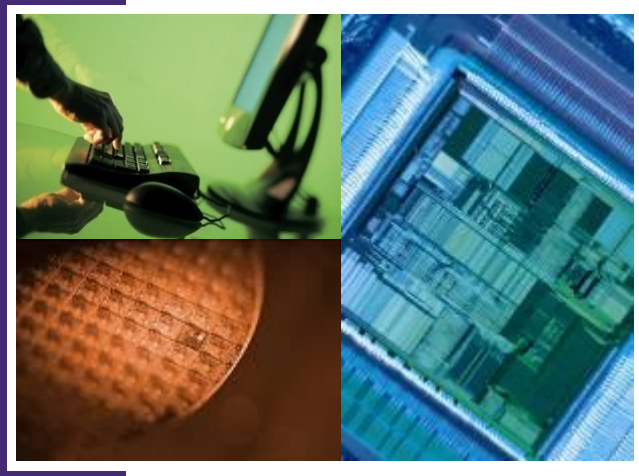
SYNOPSYS®
Predictable Success

# Synopsys TCAD


- Used by 19 out of 20 top semiconductor companies worldwide
- Technical and market leadership across all technologies: DSM, Power, Memory, Compound and Optoelectronics
- Strong R&D program with research & academia
- Dedicated support organization focusing on customer success
- Complementary Consulting and Engineering Service Offerings

© 2007 Synopsys, Inc. (14)

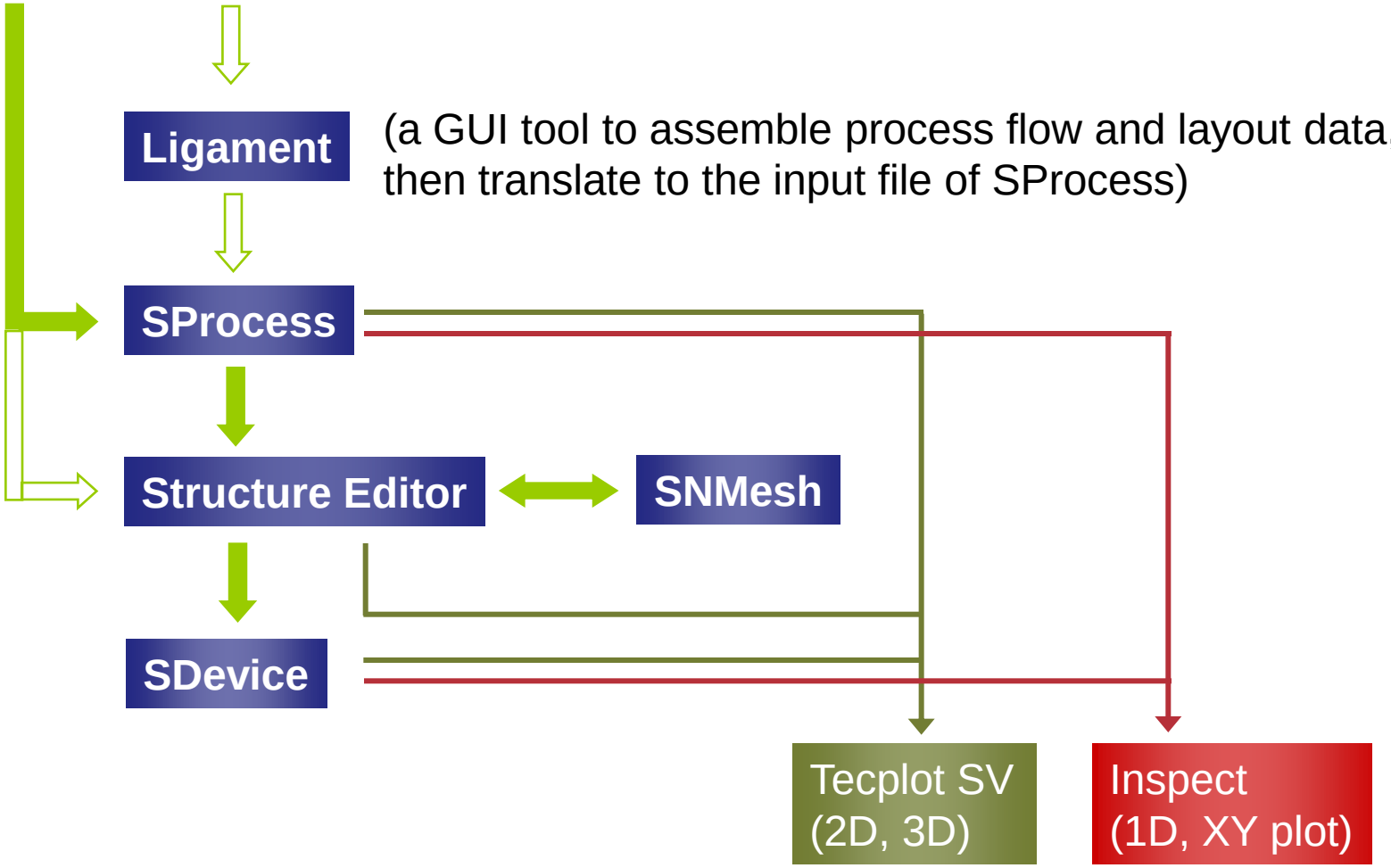
# Sentaurus Work Bench (SWB)



SYNOPSYS®

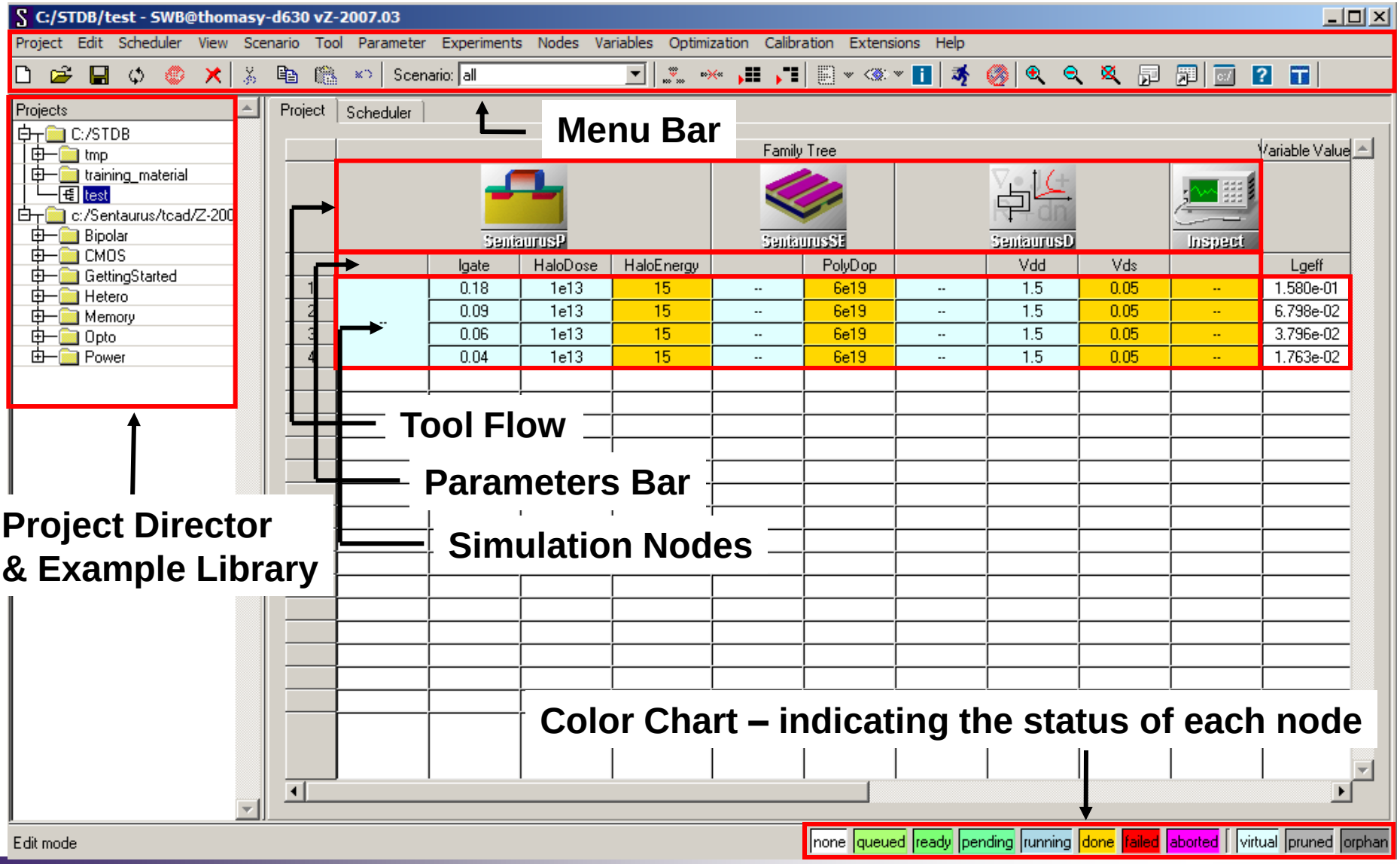
Predictable Success

# Sentaurus Workbench




© 2007 Synopsys, Inc. (16)

# Sentaurus Workbench




© 2007 Synopsys, Inc. (17)

SYNOPSYS®
Predictable Success

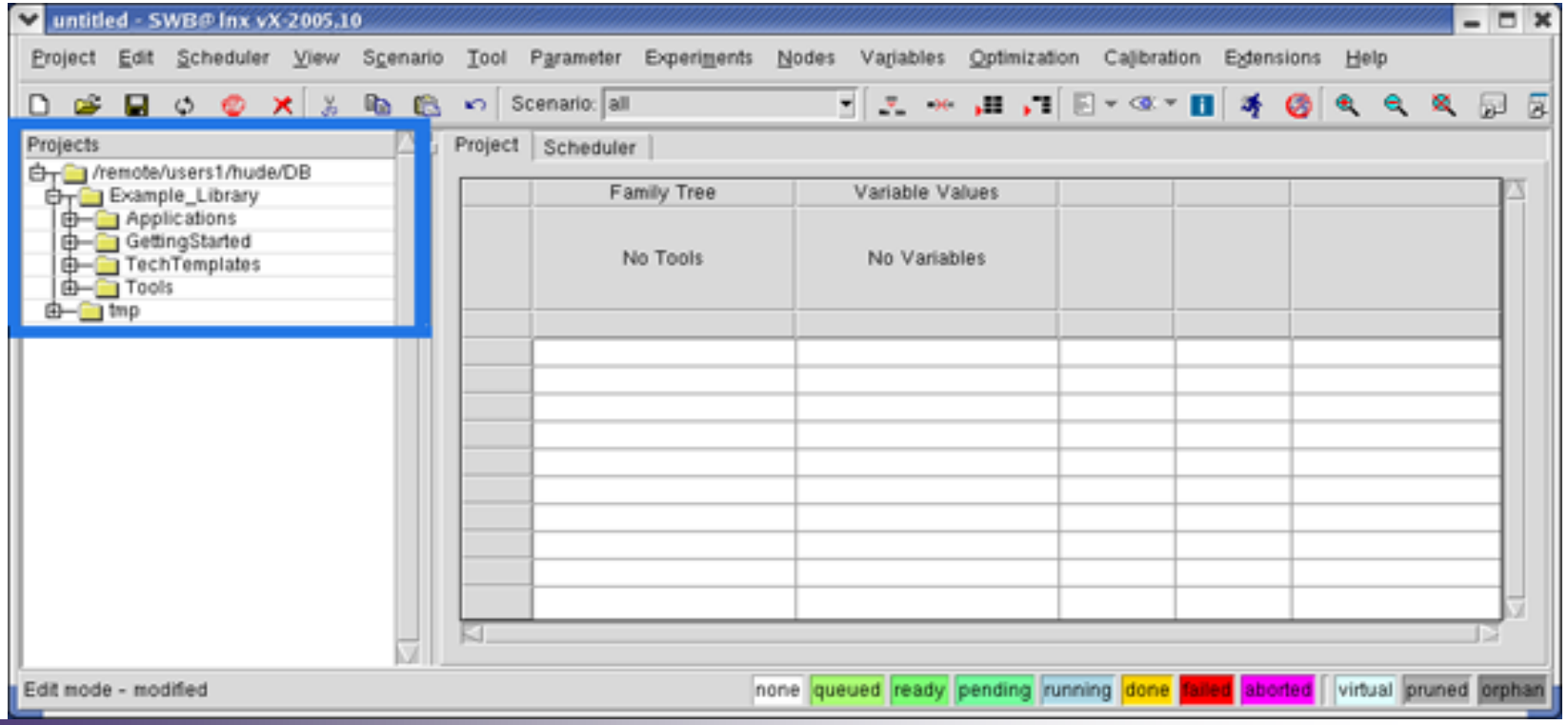
# Overview

- Sentaurus Workbench is the primary graphical front end that integrates Synopsys simulation programs into one environment.
- Simulations are comprehensively organized into projects.
- Sentaurus Workbench automatically manages the information flow, which includes preprocessing of user input files, parameterizing projects, setting up and executing tool instances, and visualizing results.
- Sentaurus Workbench allows users to define parameters and variables in order to run comprehensive parametric analyses. The resulting data can be used with statistical and spreadsheet software.

© 2007 Synopsys, Inc. (18)

# Starting SWB

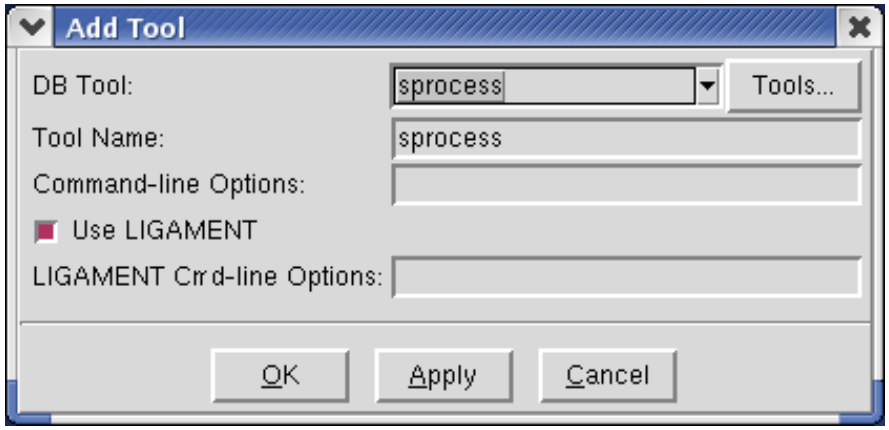
- Before starting Sentaurus Workbench, a database directory to store simulations must be created.
- Users must set the STDB environment variable to point to that directory.




© 2007 Synopsys, Inc. (19)

# Creating a Project

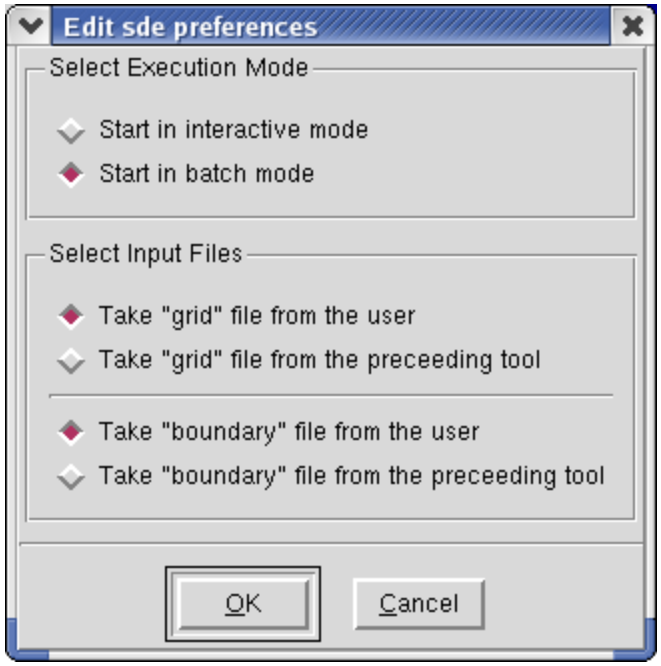
- To create a project:
  - Project > New
  - This creates a temporary project with a name similar to g_lnx_2879_0.tmp in the tmp folder that can be viewed in the Projects window
- To set up a tool flow:
  - Right-click the No Tools node under the Family Tree
  - The Add Tool dialog box is displayed




© 2007 Synopsys, Inc. (20)

# Editing Tool Command File and Preferences

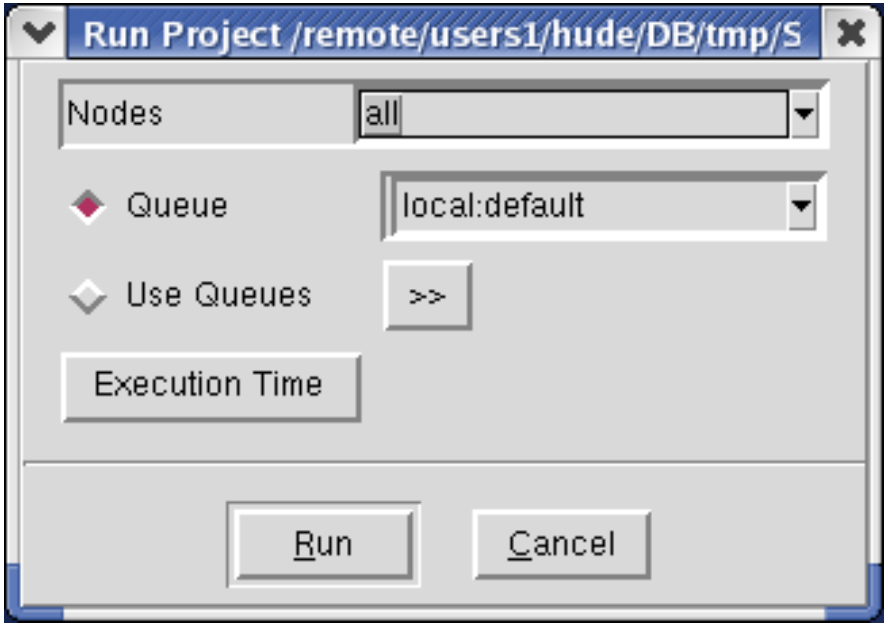
- Click on the tool icon
- Right click -> Edit Input -> Commands
- To change preferences, Edit Input -> Preferences




© 2007 Synopsys, Inc. (21)

# Running Projects

- To run a project:
  - Project > Run or Ctrl+R (alternatively, click the Run icon in the toolbar).
  - In the Run Project dialog box, click Run to execute the entire project




© 2007 Synopsys, Inc. (22)

# Viewing Output Results

- A given node has a number of input and output files associated with it. These can be viewed by right-clicking a node and selecting Visualize. All text and log files can be viewed using SEdit by selecting them.
- All output data files in the case of Dios, Sentaurus Process, Mdraw, Sentaurus Device, and Sentaurus Device EMW can be viewed using Tecplot SV, or a plot of .plx and .plt files in Dios and Sentaurus Device can be viewed using Inspect.
- In addition, the information written to standard output, while a simulation is running, can be viewed by selecting the View Output option.

© 2007 Synopsys, Inc. (23)

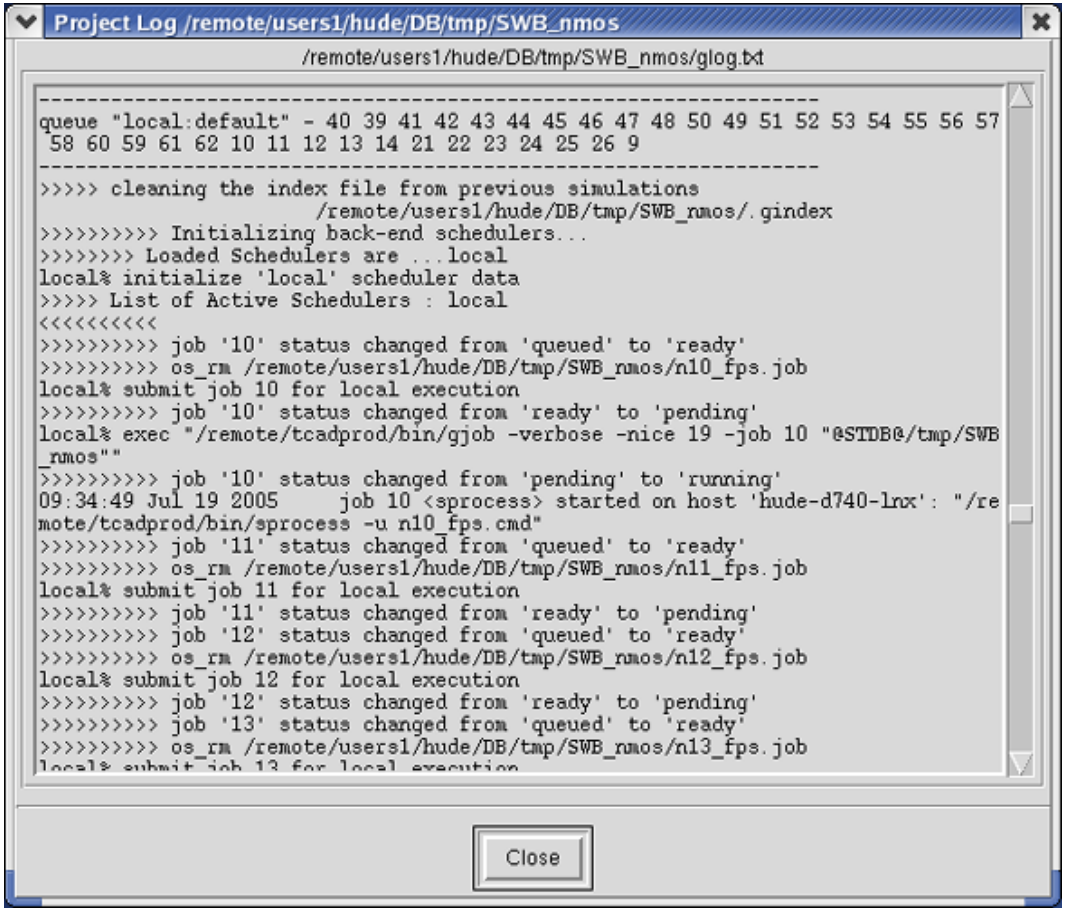
# Changing Tree Display Attributes


- Every simulation node in a project has a color associated with it that indicates its status. The color chart in the lower-right corner of the main window of Sentaurus Workbench shows what each color indicates.
- The format in which the Sentaurus Workbench project tree is displayed is very flexible and user controllable. Users can display solely the tool flow, or can number the various simulation nodes, or can display parameters (splits), variables, extracted values, and other details. To use this feature:
- From the View menu, select or clear the various options, or View > Tree Options for more features.

© 2007 Synopsys, Inc. (24)

# SWB Preprocessing

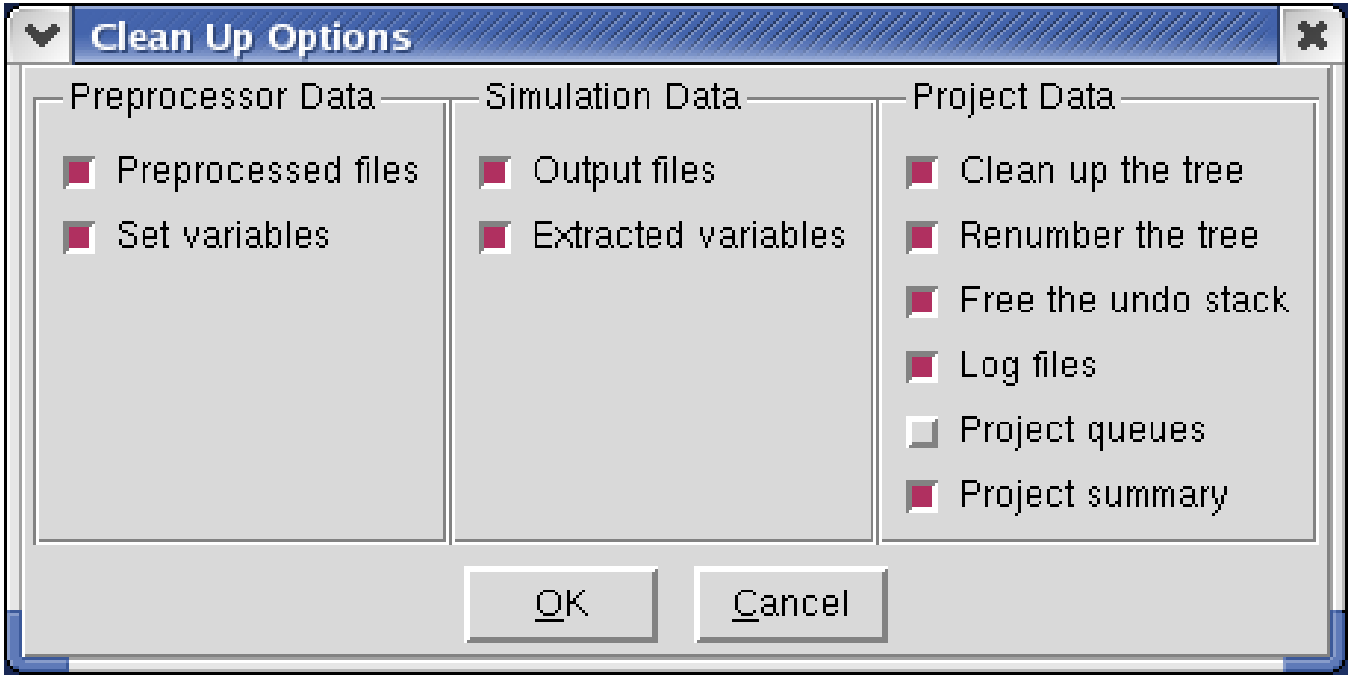
- As each node is run, SWB prints a message
- Before executing a simulation node, SWB preprocesses the input files.
- Upon preprocessing the files, SWB creates input files of the form pp<node_number>_<tool_name>.cmd.




© 2007 Synopsys, Inc. (25)

# Cleaning up Projects

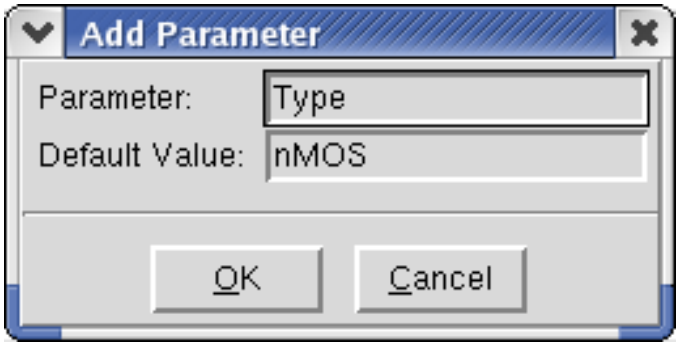
- To clean up a project:
  - Project > Clean Up.
  - In the Clean Up Options dialog box, select the items to be removed.
  - Click OK.




© 2007 Synopsys, Inc. (26)

# Adding Parameters

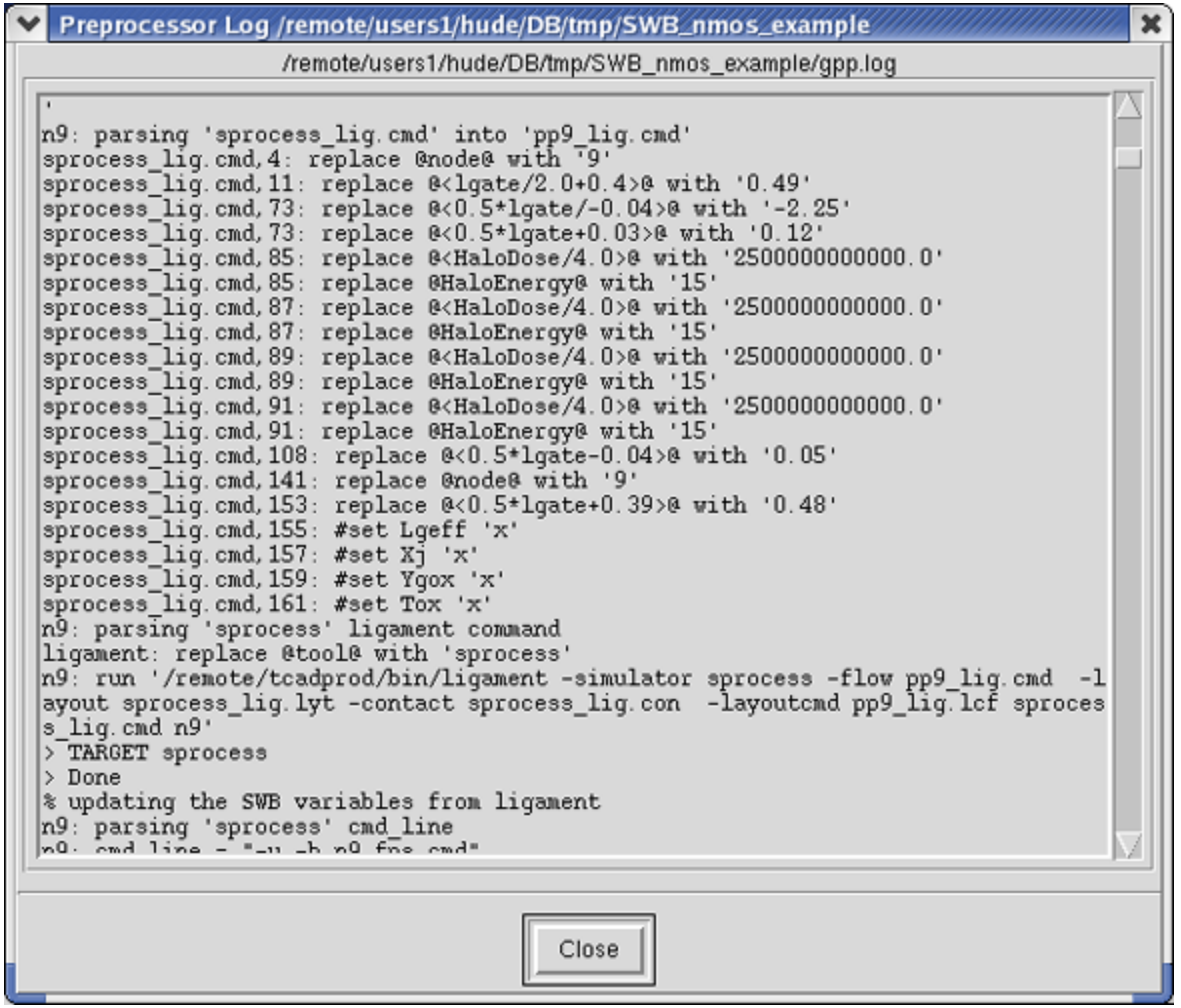
- In SWB, parameters can be defined and multiple values can be assigned to them to create splits in experiments.
- To create each parameter:
  - Right-click the gray box immediately below the Sentaurus Process icon in the main window of Sentaurus Workbench and select Add.
  - In the Add Parameter dialog box, enter the parameter name and default value.




© 2007 Synopsys, Inc. (27)

# Adding Parameters

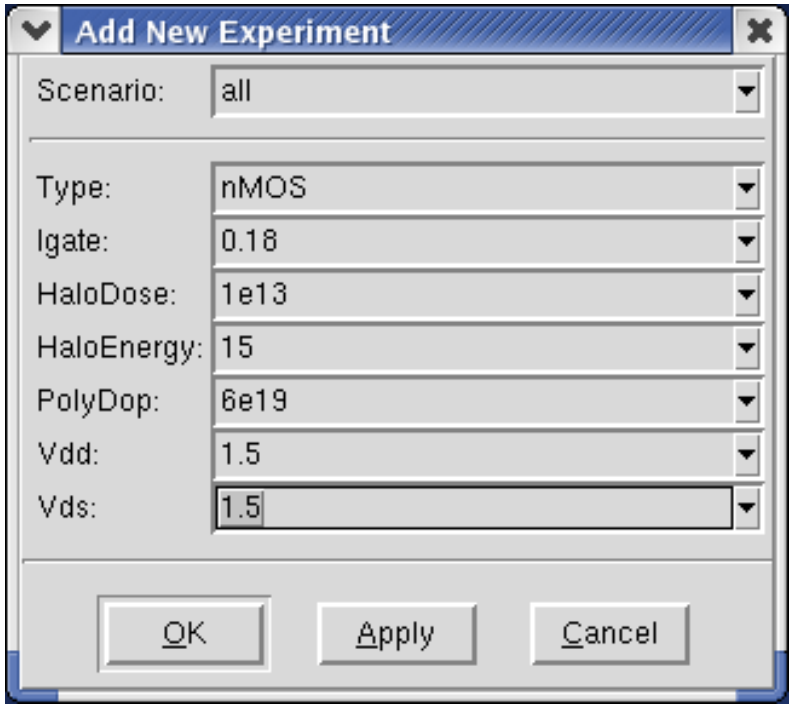
- Next, insert the parameters into the input file of the tool:
  - Right-click the tool icon and select Edit Input > Commands.
  - Insert the parameter as @Type@ where appropriate.
- To preprocess the project:
  - Project > Preprocess or Ctrl+P, and view the values.

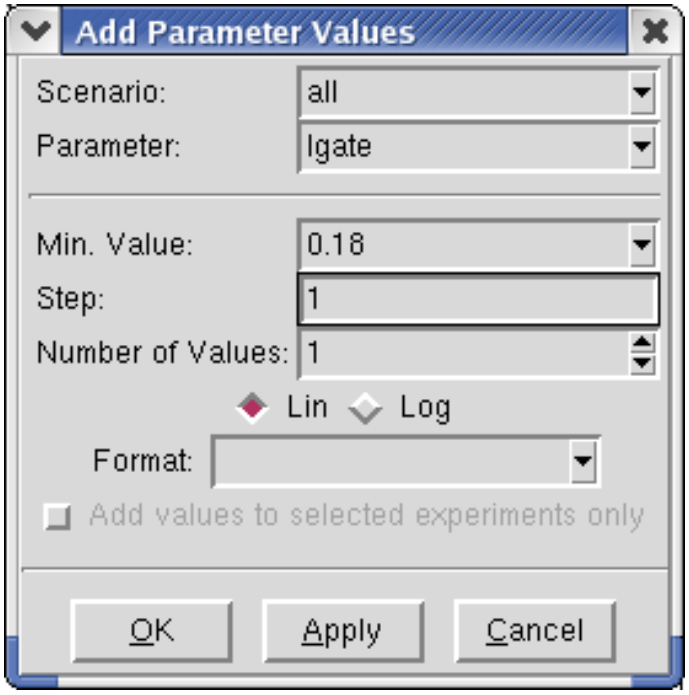



© 2007 Synopsys, Inc. (28)

# Adding New Experiments (Rows)

- To create multiple experiments:
  - Experiments > Add New Experiment.
  - Enter the values for parameters
- Or right-click the parameter name and select Add Values

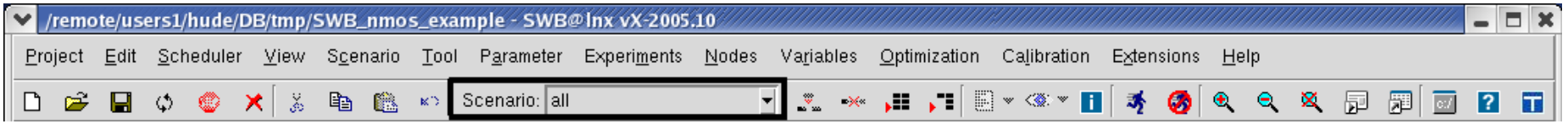



© 2007 Synopsys, Inc. (29)

# Using Scenarios

- Scenarios are sets of experiments (rows)
- Creation of different scenarios is particularly helpful when many parameters are used
- Experiments can be classified into separate scenarios to represent different physical situations.
- You can display different scenarios in the main window of Sentaurus Workbench by scrolling through the Scenario menu (black border) in the toolbar




© 2007 Synopsys, Inc. (30)

# Pruning the Tree

- Often, when many parameters are used in a project, certain combinations of parameters are not required to be simulated.
- In such cases, the project tree can be pruned by terminating such experiments.
- For example, assume that in the scenario New1 the experiments for lgate=0.18, HaloDose=1e13, HaloEnergy=25, and Vds=0.05 are not required to be simulated.
- Therefore, the project can be pruned at node 41 as follows:
  - Select node 41.
  - Node > Prune.
  - Nodes 41 and 53 are pruned and appear gray.

© 2007 Synopsys, Inc. (31)

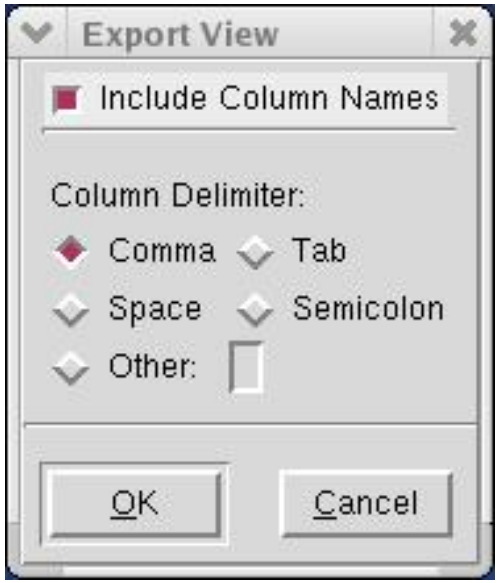
# Variables vs. Parameters

- If a certain parameter is going to change only for certain experiments, a variable could be used instead.
- To add a variable:
  - Variables > Add.
  - Enter Vsub in the Variable field with a default value of 0.0.
- This sets the substrate bias to 0.0 as before. The same could have been performed by defining a parameter called Vsub.
- Now assume you want to set the substrate bias to -1.0 V only for experiment 3, (node 41), retaining 0.0 V for all other experiments.
- Edit tool command file for node 41:
  - { Name="substrate" Voltage=@Vsub@ }
- Right-click node 41 ([n41]) and select Set Variable Value. In the dialog box, select Vsub and set the value to -1.0 V.
- Now, preprocess the project. After it is completed, view pp41_des.cmd. @Vsub@ is replaced by -1.0 V.

© 2007 Synopsys, Inc. (32)

# Exporting Scenarios and Experiments

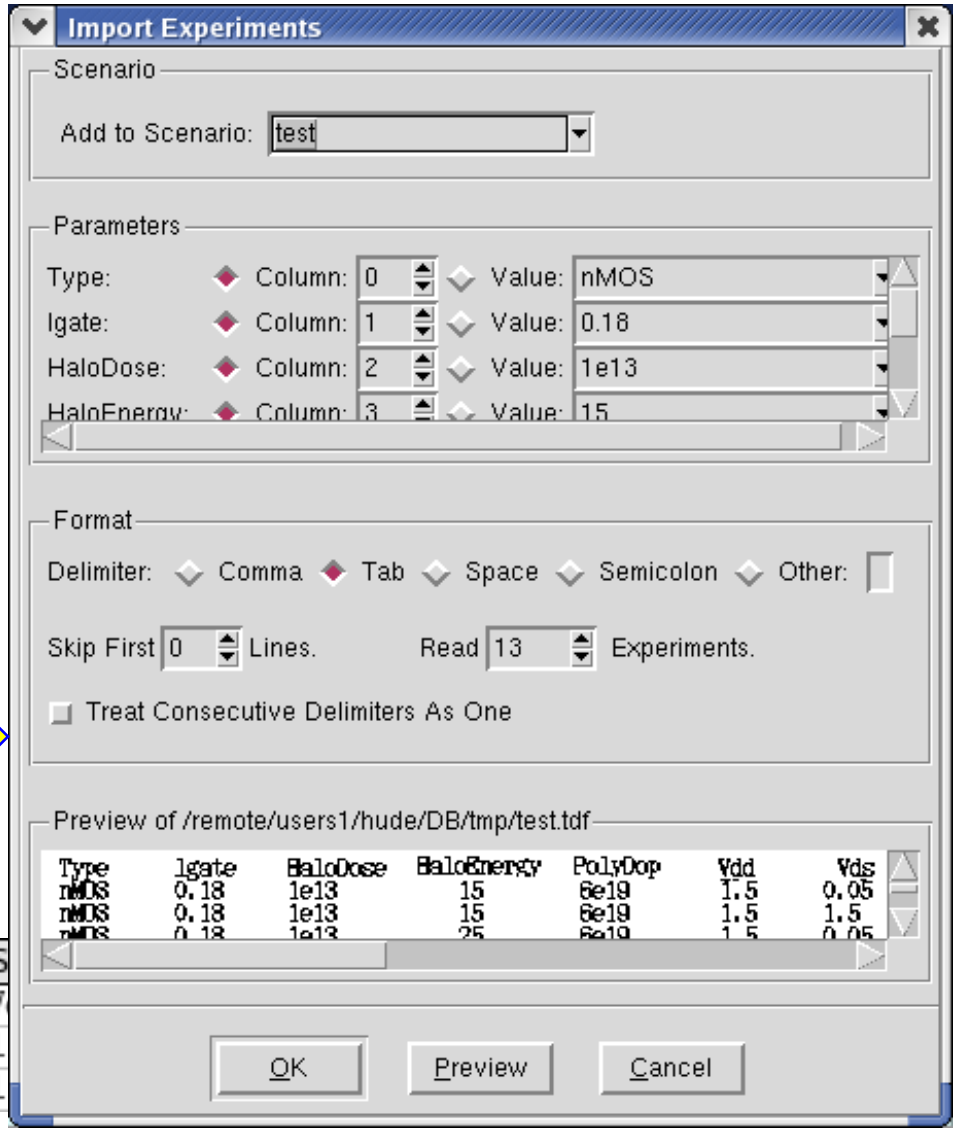
- Scenarios, that is, sets of parameters and variables visible in the SWB main window, can be exported to a file either for use with a spreadsheet application or an external editor.
  - View > Export.
  - In the Export View dialog box, select Tab for the Column Delimiter.
  - Click OK.
  - In the Save As dialog box, select Tab delimited (*.tdf) for Files of type and provide a file name (for example) test.tdf. This exports the scenario to a file.




© 2007 Synopsys, Inc. (33)

# Importing Experiments

- Experiments > Import from a File and select the file test.tdf.
- To create a scenario in an external text editor and import it into SWB, create a table, and save it in comma-delimited or tab-delimited format.




| | Standard | Standard | Standard | Standard | Standard | S |
|---|---|---|---|---|---|---|
| 1 | Type | lgate | HaloDose | HaloEnergy | PolyDop | V |
| 2 | nMOS | 0.14 | 5.00E+12 | 13 | 5.00E+19 | 1 |
| 3 | nMOS | 0.12 | 6.00E+12 | 14 | 6.00E+19 | 1 |

© 2007 Synopsys, Inc. (34)

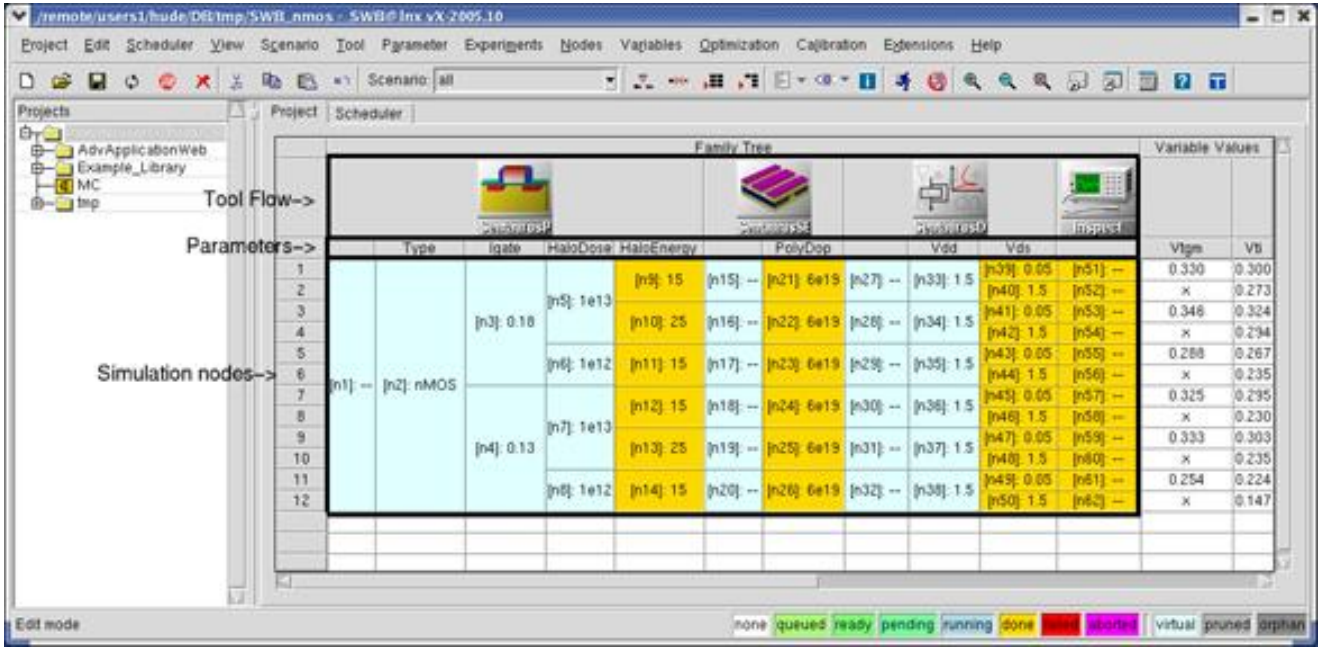
# Attaching Project Directories

- In addition to displaying the projects within the $STDB directory, projects located elsewhere can be attached to the displayed list:
  - From the Edit menu, select Attach Root
  - Select the required project directory
- This procedure can be repeated to attach up to five project directories.
- To remove a project directory:
  - Select the required project directory.
  - Edit > Detach Root.
- Only attached roots can be detached. The projects in the $STDB environment cannot be detached, but they can be deleted permanently.

© 2007 Synopsys, Inc. (35)

# Copying Sentaurus Workbench Projects

- To run a project, copy the project from the Examples Library:
  - Open the Examples Library > Getting Started, and select the project SWB_nmos.
  - Right-click and select Copy.
  - Select the tmp folder, right-click, and select Paste to place the project in tmp.
  - Open the tmp folder and double-click the project SWB_nmos.



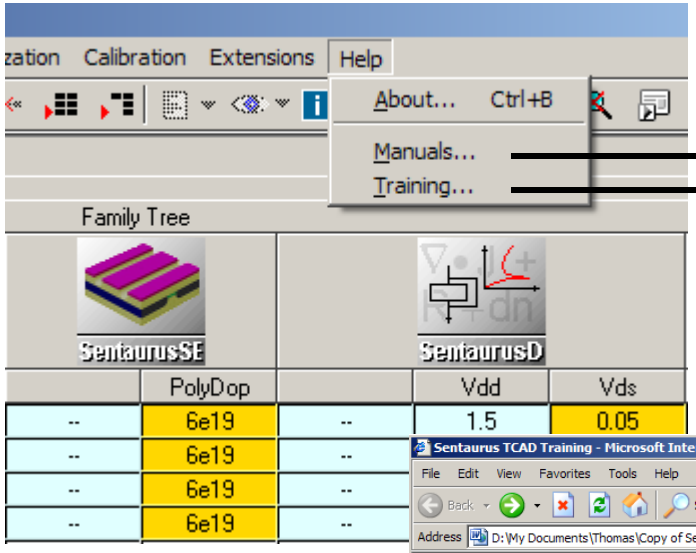

© 2007 Synopsys, Inc. (36)

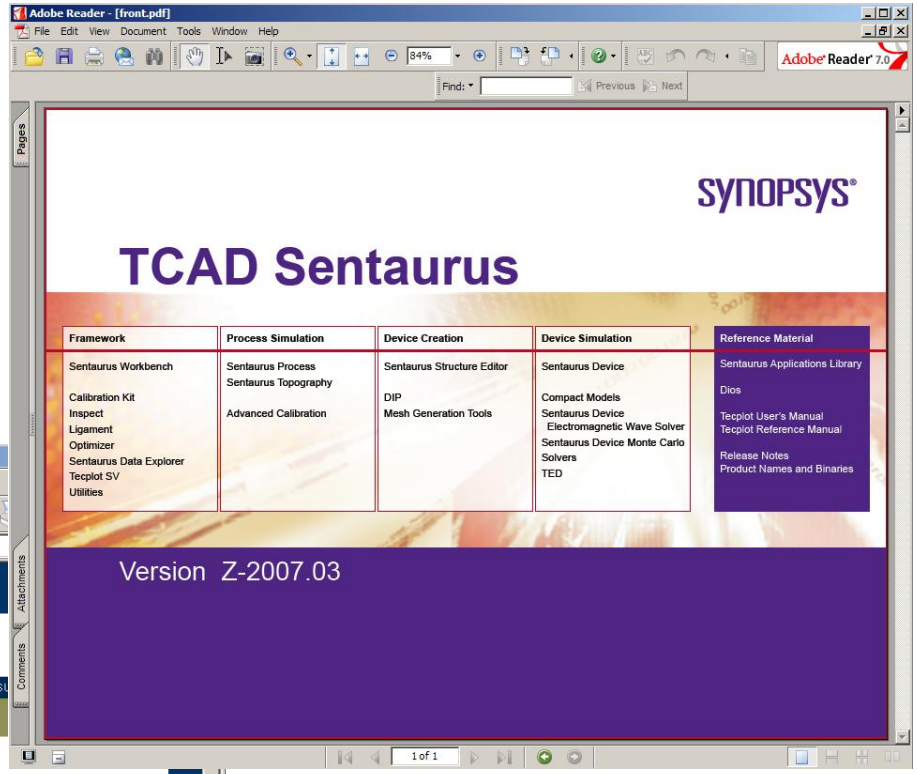
# Archiving Projects

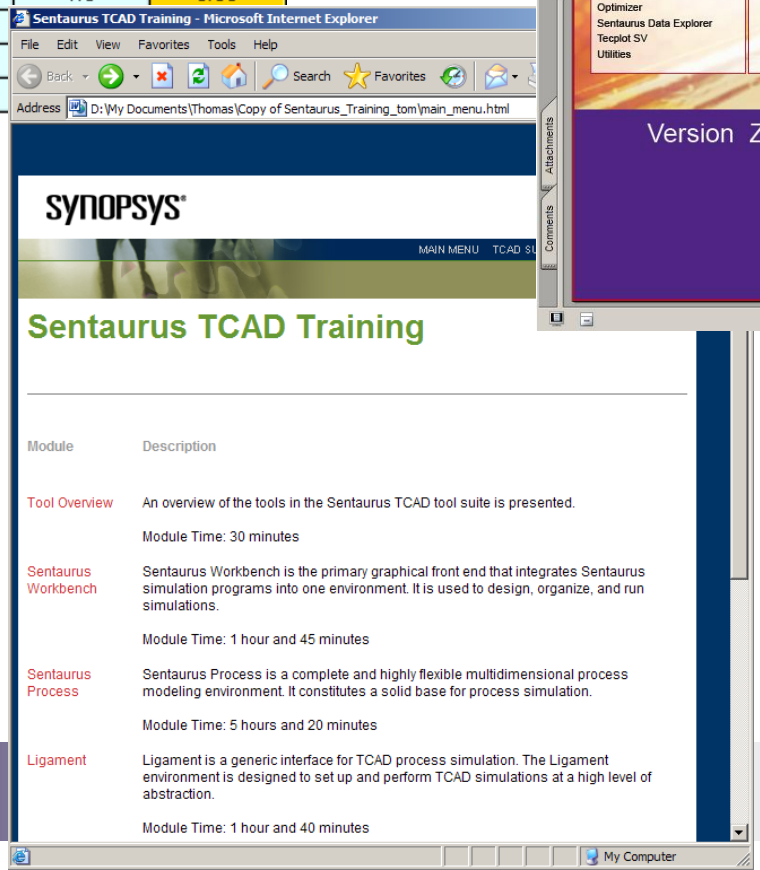
- Sentaurus Workbench projects can be archived, that is, they can be copied and compressed for later use.
- To archive a project: Project > Export (tar).
- All the files in the project directory are copied and compressed.
- To import archived projects: Project > Import (tar).
- In addition, projects can also be saved using Project > Save Clean As, which cleans up the project before it is saved in an uncompressed format.

© 2007 Synopsys, Inc. (37)

# Online Manuals and Training Material



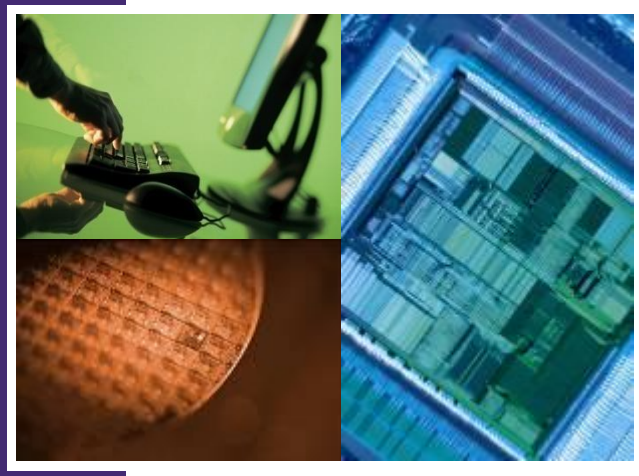




© 2007 Synopsys, Inc. (38)

SYNOPSYS®
Predictable Success

# Sentaurus Process (SProcess)



SYNOPSYS®

Predictable Success

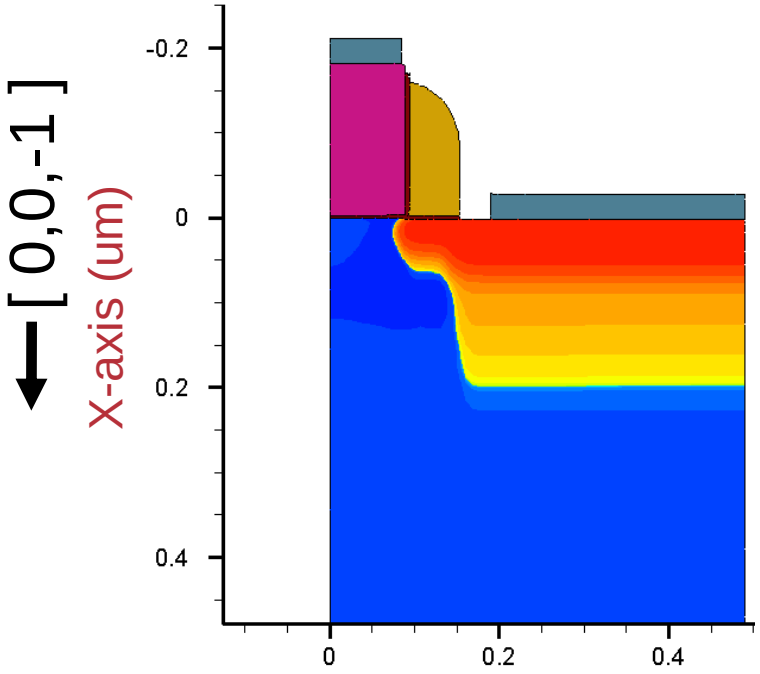
# Outline

- Sentaurus TCAD coordinate systems
- Script file setup sequence
- 1D example
- 2D example
- Implement external variable on Sentaurus Workbench

© 2007 Synopsys, Inc. (40)

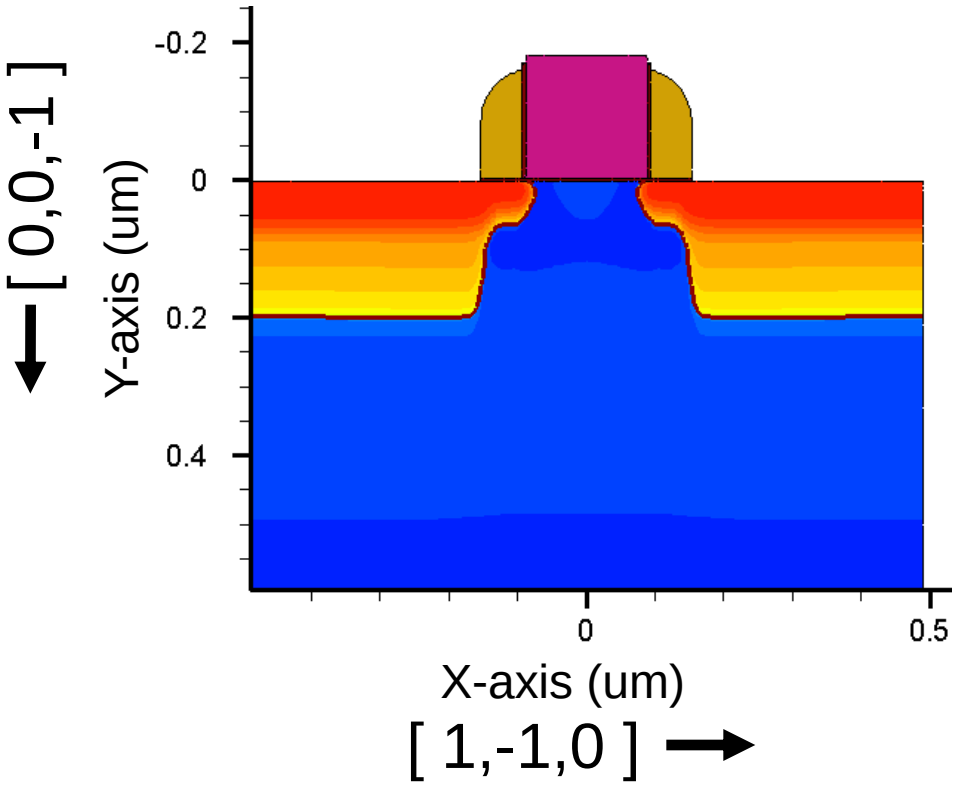
# Sentaurus TCAD Coordinate Systems (default setting)

SProcess




Silicon miller indices

Tecplot, SDE, SDevice



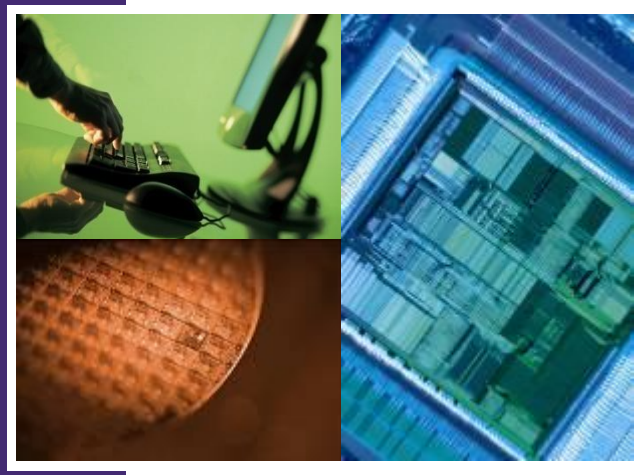

© 2007 Synopsys, Inc. (41)

# Script File Setup Sequence

- Setup initial mesh
- Define initial simulation domain
- Initialize simulation & define substrate condition
- Setup process flow
  - Oxidation / Deposition / Etching / Implantation / Annealing
  - Rebuild mesh at appropriate steps
- Define electrodes
- Save full structure
- Extract parameters

© 2007 Synopsys, Inc. (42)

# Ligament commands



SYNOPSYS®

Predictable Success

# Limitation of basic Ligament Flow Editor command

The below table is comparison of supported features.

| Support feature by basic command | Non-support feature by basic command |
|---|---|
| •Anneal (Oxidation, Diffusion)<br>•Anisotropic and Isotropic Etch<br>•Anisotropic and Isotropic Depo<br>•Epi<br>•Pattern(mask)<br>•Implant | •Trapezoidal and Polygon etch<br>•Fill<br>•Mesh strategy<br>•Extraction of parameter<br>•Changing models<br>… |

Complicated feature is not supported by basic command.

© 2007 Synopsys, Inc. (44)

# Insert command

- These special commands can be specified by insert command.
- These operation is used to insert original commands of Sentaurus Process directly.

***Basic rule***

- There are characters, $, ", [, ] and \.
  →
  \ is necessary to be placed just before these characters like below:
  **Example:**

Original command

```
refinebox name=emitter \
 min= {0 0} max= {1 0} \
 xrefine= {0.1} yrefine= {0.1}
```

→**Error!**

Modified command

```
refinebox name=emitter \\
 min= {0 0} max= {1 0} \\
 xrefine= {0.1} yrefine= {0.1}
```

→**OK!**

© 2007 Synopsys, Inc. (45)

# Basic command rule of Sentaurus Process (1/2)

- Long commands can be separated into several lines by ending each line with a backslash \.

**Example:**

```
etch Silicon type=polygon \
 polygon= { 0.0 0.5 1.0 0.6 \
            1.0 2.4 0.0 2.5 \
          }
```

- If there is combination, = {*value*}. The command should be:

=[space]{*value*}

**Example:**

```
refinebox name=emitter min={0 0} max={1 0} xrefine={0.1} yrefine={0.1}
```

→**Error!**

```
refinebox name=emitter min= {0 0} max= {1 0} xrefine= {0.1} yrefine= {0.1}
```

→**OK!**

space space space space

© 2007 Synopsys, Inc. (46)

# Basic command rule of Sentaurus Process (2/2)

- If there is combination, *value*<unit>. Ex) 50<nm>
  User should not insert space between value and <unit>

**Example**

```
Diffuse temperature=1000 <C> time=30 <min>
```

→**Error!**

```
Diffuse temperature=1000<C> time=30<min>
```

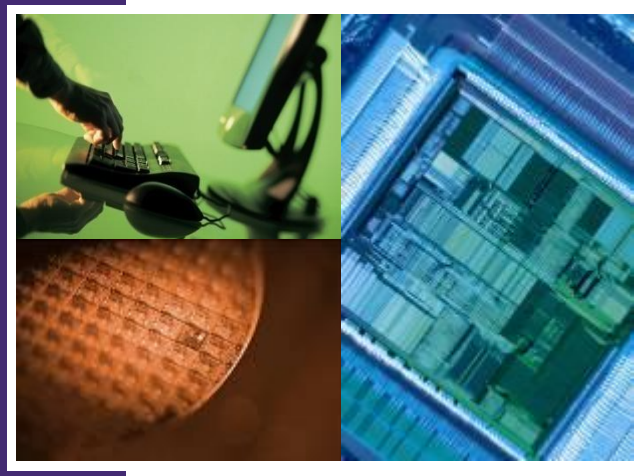
No space No space

→**OK!**

- Sentaurus Process supports “TCL script” command. Please refer to a reference book or web if you want to know the detail.

© 2007 Synopsys, Inc. (47)

# Taper etching

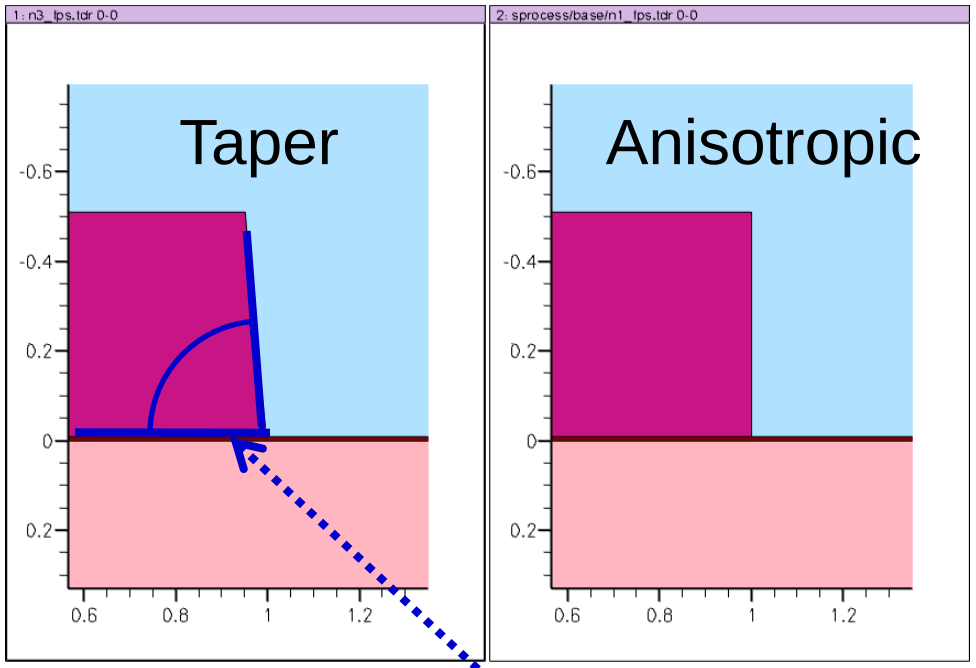


SYNOPSYS®

Predictable Success

# Taper etching

- Etch command supports basic feature
  - Isotropic
  - Anisotropic.
- Taper etching.
  → Simulator-specific command is necessary.
- Performs a trapezoidal etching.
  → Define the thickness and angle parameters.



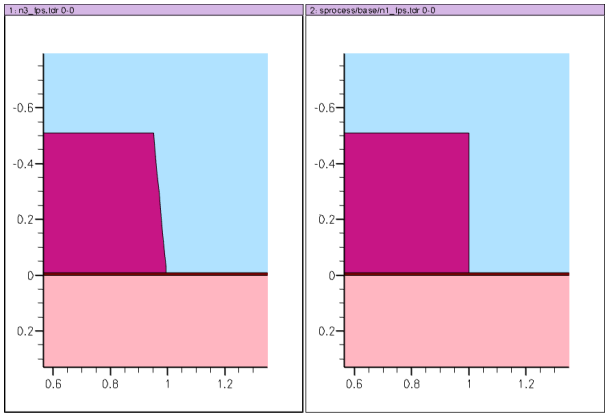

- An examples is:

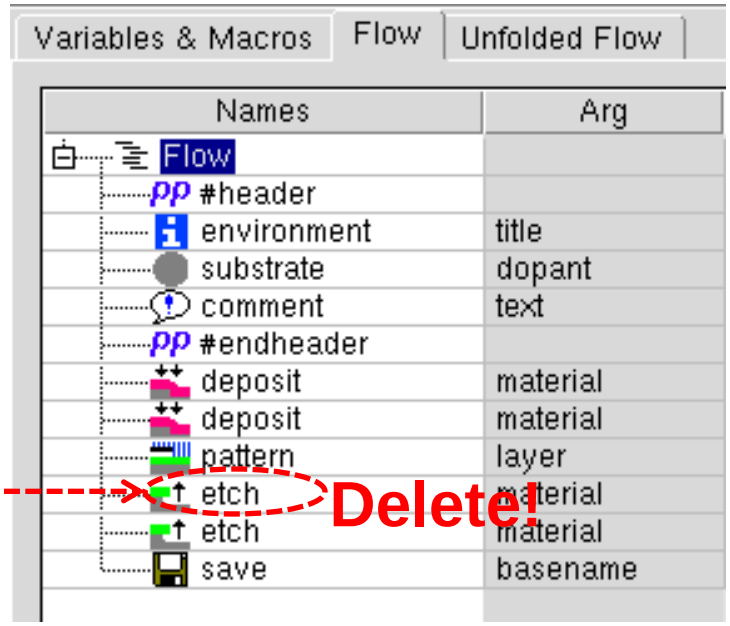
```
etch material=PolySilicon type=trapezoidal thickness=0.55 angle=85
```

© 2007 Synopsys, Inc. (49)

# Ex: taper etching

1. Copy the project “/PostBasic/question/sprocess/base” under STDB folder.
2. Change copied project’s name to “taper_etch”.
3. Open the project and start Ligament Layout Editor.
4. Delete the first **etch** command for Poly.



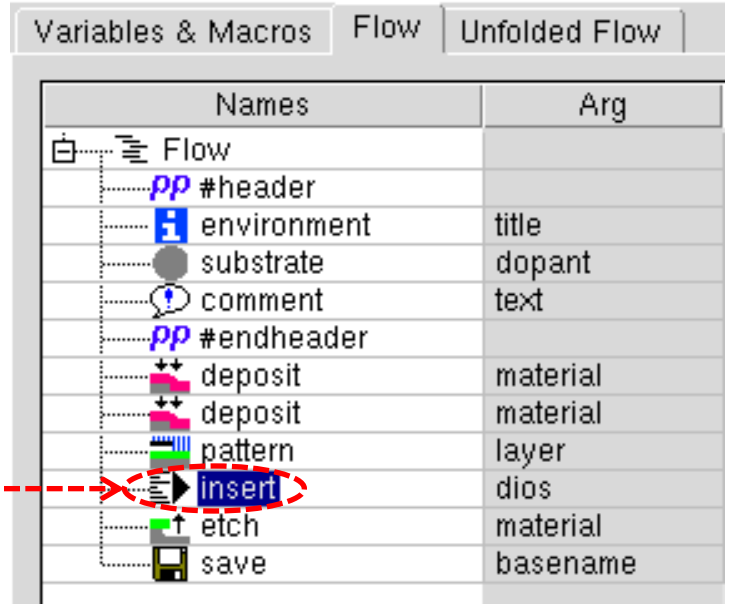



© 2007 Synopsys, Inc. (50)

SYNOPSYS®
Predictable Success

# Ex: taper etching (cont.)

5. Insert "**insert**" command
6. Specify etch of simulator-specific command like below.
7. Finish Ligament Flow Editor and run the project.

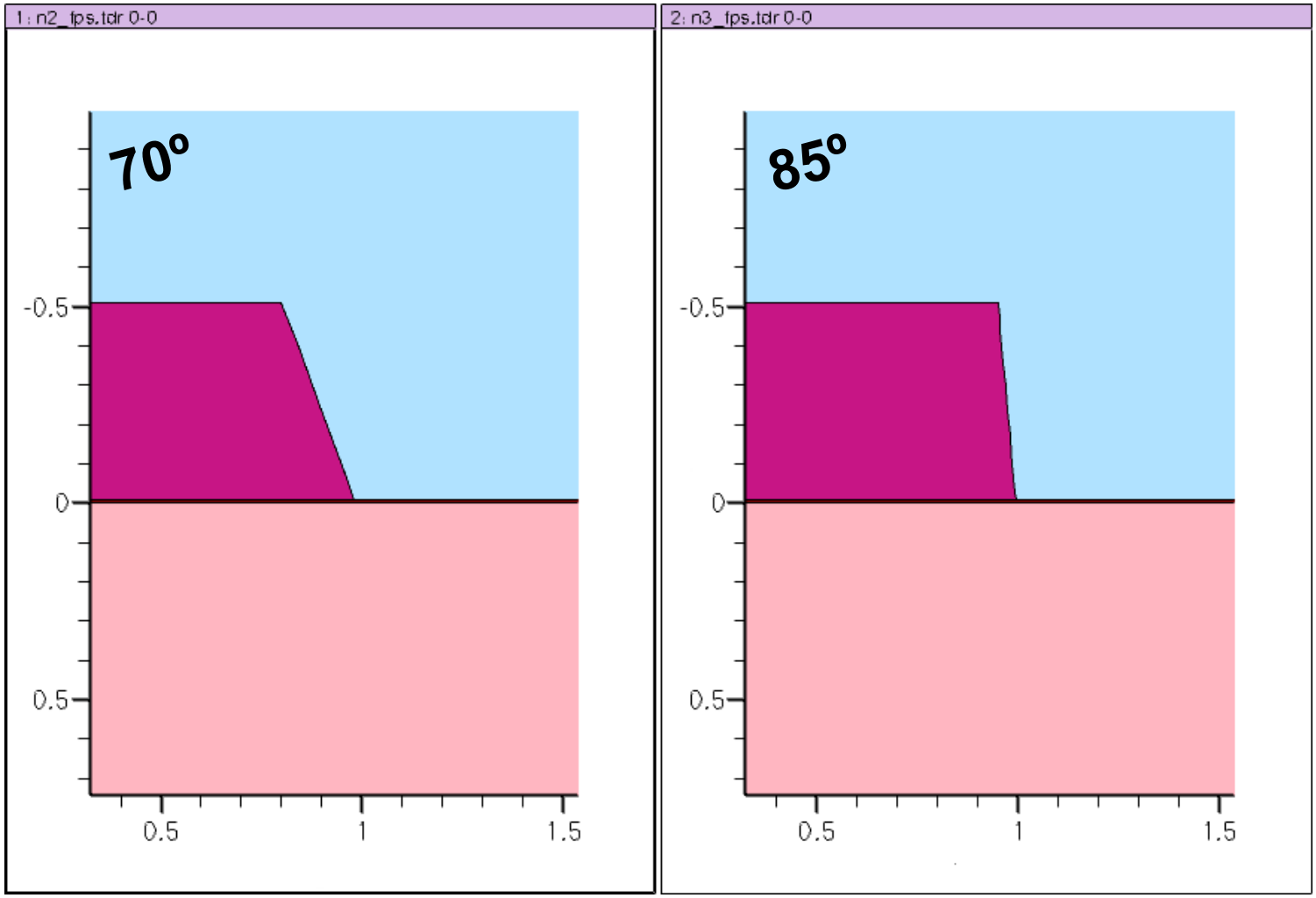



```
etch material=PolySilicon type=trapezoidal thickness=0.55 angle=85
```

8. If you have time, please try other angles.

© 2007 Synopsys, Inc. (51)

# Ex: taper etching ~the result~



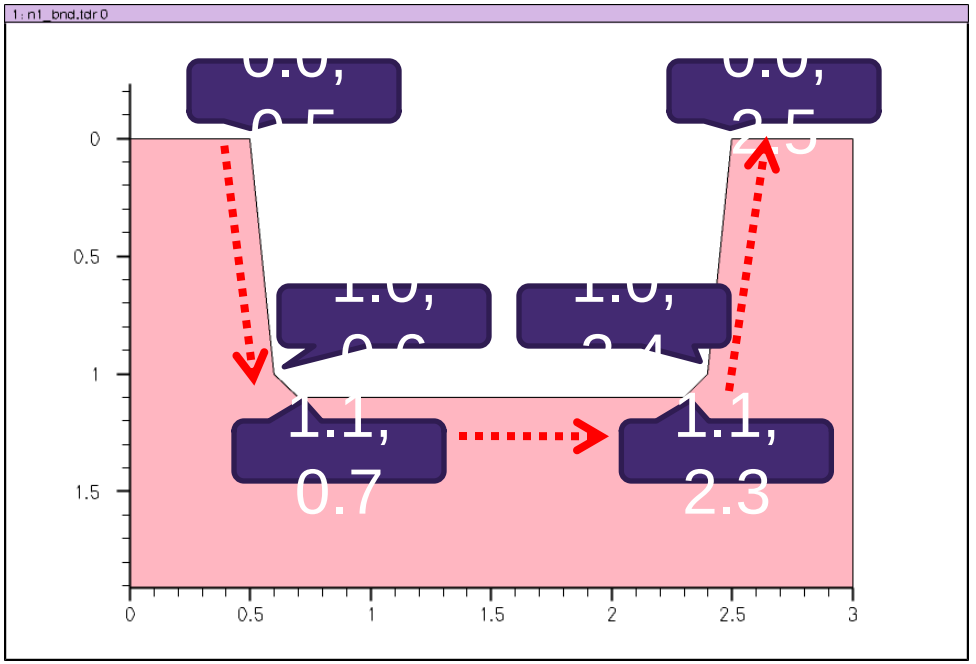

© 2007 Synopsys, Inc. (52)

# Appendix ~polygon etch~

- Complex figure
  → Use polygon etch to specify coordinates.

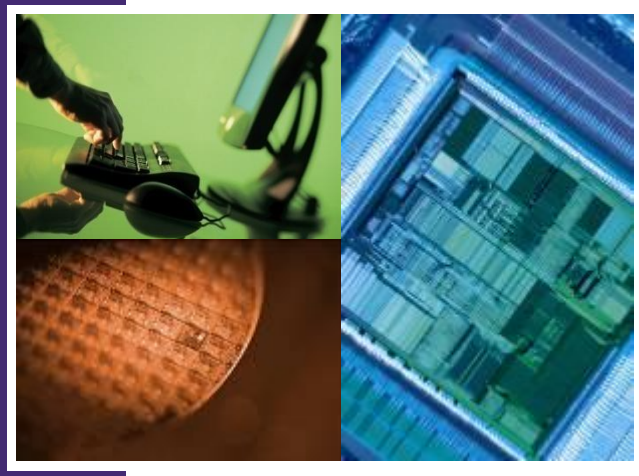
The example is:

```
etch Silicon type=polygon \
 polygon= { 0.0 0.5 \
            1.0 0.6 \
            1.1 0.7 \
            1.1 2.3 \
            1.0 2.4 \
            0.0 2.5 \
          }
```




© 2007 Synopsys, Inc. (53)

# Definition of mesh



SYNOPSYS®

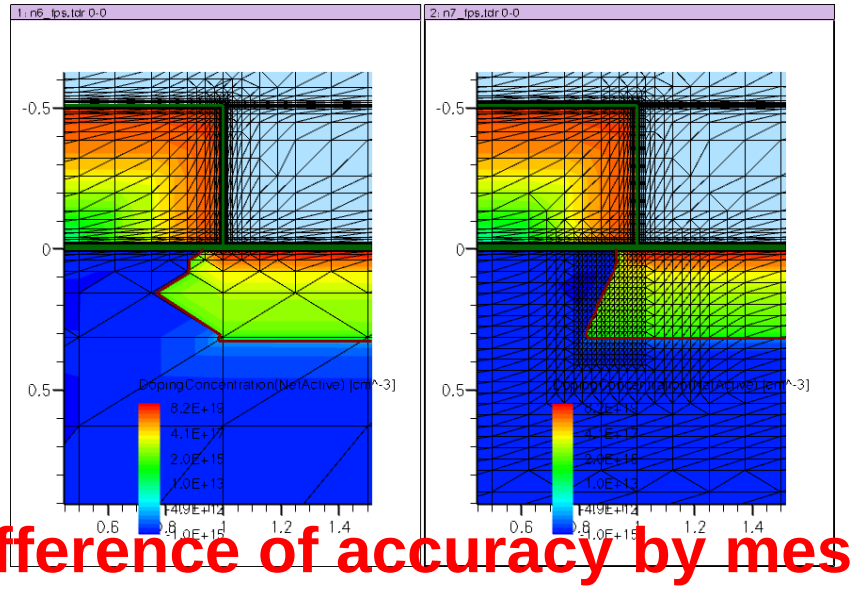
Predictable Success

# Mesh Overview (1/2)

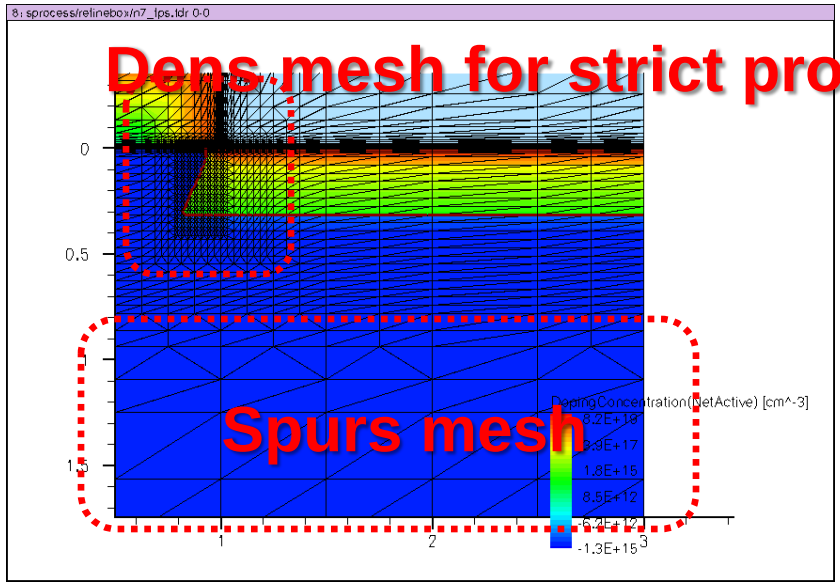
- Mesh is important :
  Accuracy are dependent on the mesh

- Surplus dens-mesh.
  → Long calculation time.

- To improve calculation time, dense mesh is placed at :
  - strict profile
  - strict geometry




Difference of accuracy by mesh




Dens mesh for strict profile

Spurs mesh

Combination improves the perfo

© 2007 Synopsys, Inc. (55)

SYNOPSYS®
Predictable Success

# Mesh Overview (2/2)

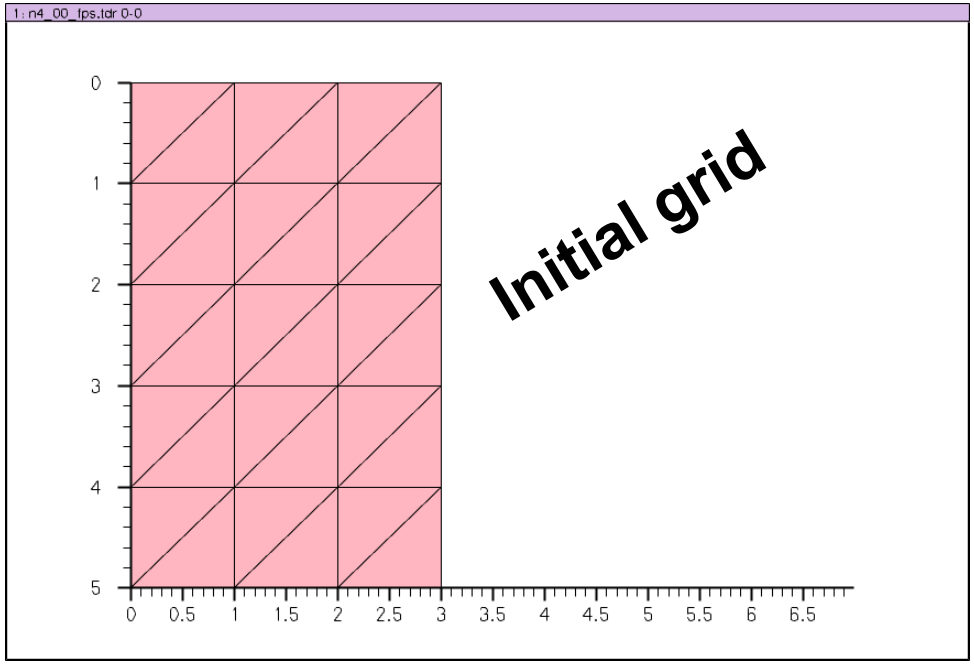
- The mesh command is inserted before the implantation or oxidation… process step
  → This is effective to reduce calculation time.

- User have to decide mesh strategy.
- There are three types of mesh:
  - line :
    Initial mesh defined at first.
  - mgoals :
    Additional mesh for the interface at etching and deposition
  - refinebox :
    Additional mesh for change of strict profile and geometry by oxidation.

© 2007 Synopsys, Inc. (56)

# Mesh Type ~line~

- This is used only for initial mesh (grid) size.
- Mesh size is changed after “mgoals” and “refinebox”.



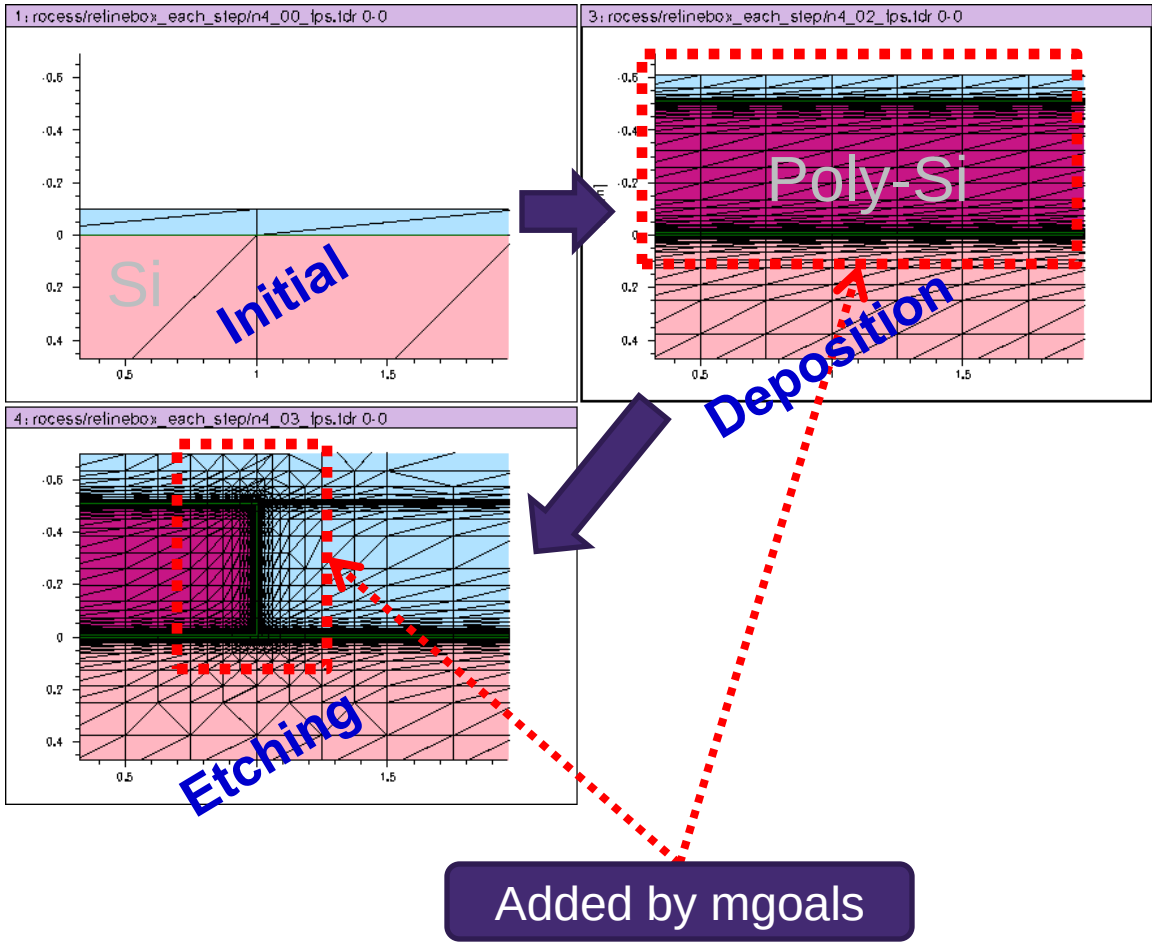

© 2007 Synopsys, Inc. (57)

# Mesh type ~mgoals~ (1/2)

- mgoals means “**M**esh and **G**eometry **O**per**A**tions using the **L**evel **S**et method”.

- **deposition and etching**
  → the geometry is changed
  →**Automatically** works along the interface

- The right example shows after deposition and etching.




© 2007 Synopsys, Inc. (58)

# Mesh type ~mgoals~ (2/2)

- User need to specify the parameter.
  - The parameters (mesh size) of “mgoals” are important for calculation of diffusion profile near the interface.

- It works on the interface of Si, Poly-Si and $SiO_2$ by default.

- The below statement “grid remesh” forces an “mgoals” remeshing step.

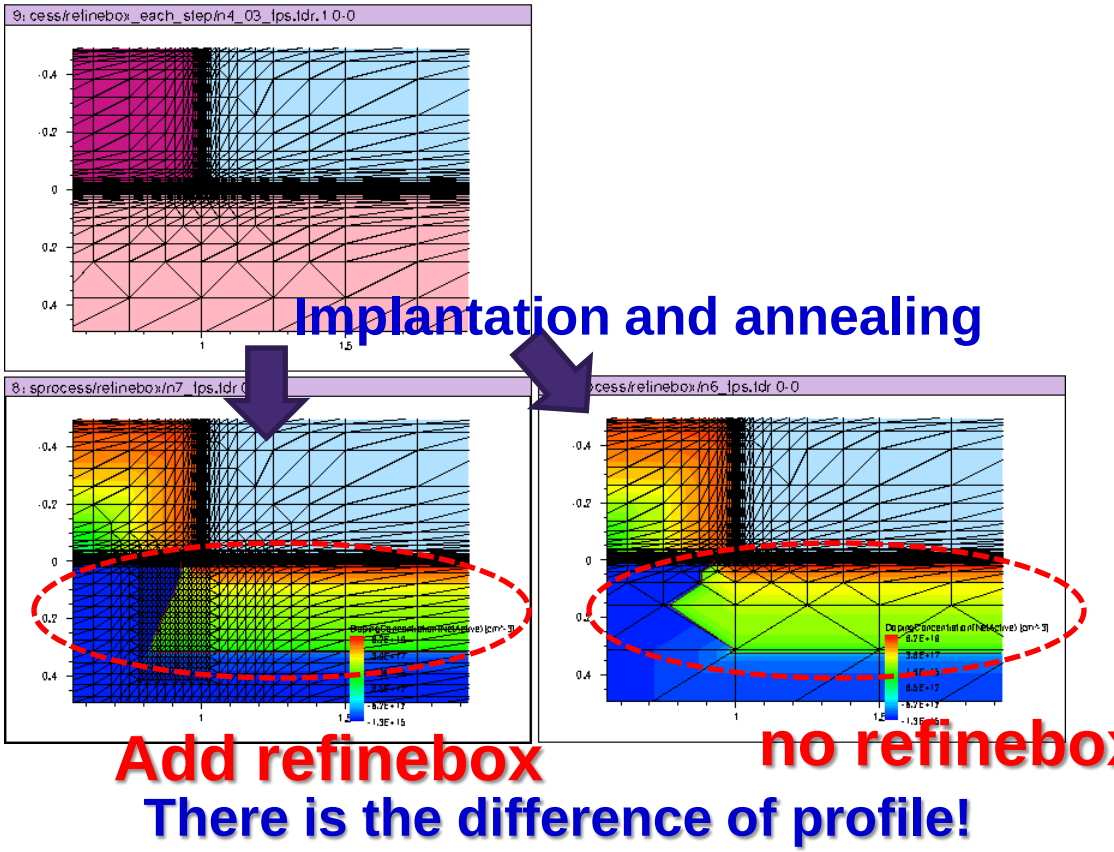
```
grid remesh
```

- “grid remesh” is recommended after an oxidation step because “mgoals” does not work oxidation step automatically.

© 2007 Synopsys, Inc. (59)

# Mesh type ~refinebox~ (1/2)

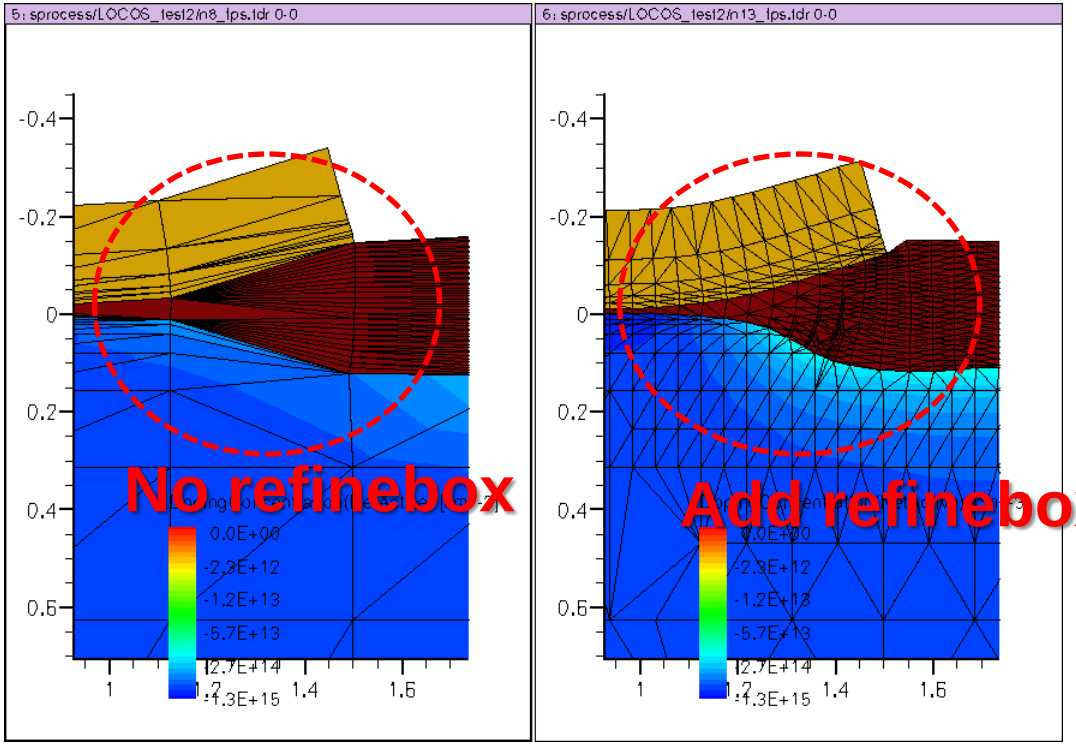
- refinebox works at the specified local area (“mgoals” works only the interface. )
- Explicitly specify the coordinate of area.
- Generally strict profile is appropriate area for refinebox, see the right figure.




© 2007 Synopsys, Inc. (60)

# Mesh type ~refinebox~ (2/2)

- Strict boundary by oxidation is also appropriate area for refinebox.




© 2007 Synopsys, Inc. (61)

# Mesh command : mgoals ~usage~ (1/2)

```
mgoals min.normal.size=5<nm> max.lateral.size=2.0<um> \
  normal.growth.ratio=2.0

pdbSet Grid Mgoals UseLines 1
```

**Command option**

- **min.normal.size** : The smallest normal (to the interface) mesh element size on either side of an interface. The default value is 1nm.
- **normal.growth.ratio** : Used to increase the normal size of the element, moving away from the interface. The default is 2.0.
- **max.lateral.size** : The maximum lateral (parallel to the interface) spacing between elements. This is, however, also used to define an upper bound on the normal size. The default value is 1um.

*Large Device :*

*→ Larger value of "min.normal.size" and "max.lateral.size" than default might be good. The calculation time might be improved.*

© 2007 Synopsys, Inc. (62)

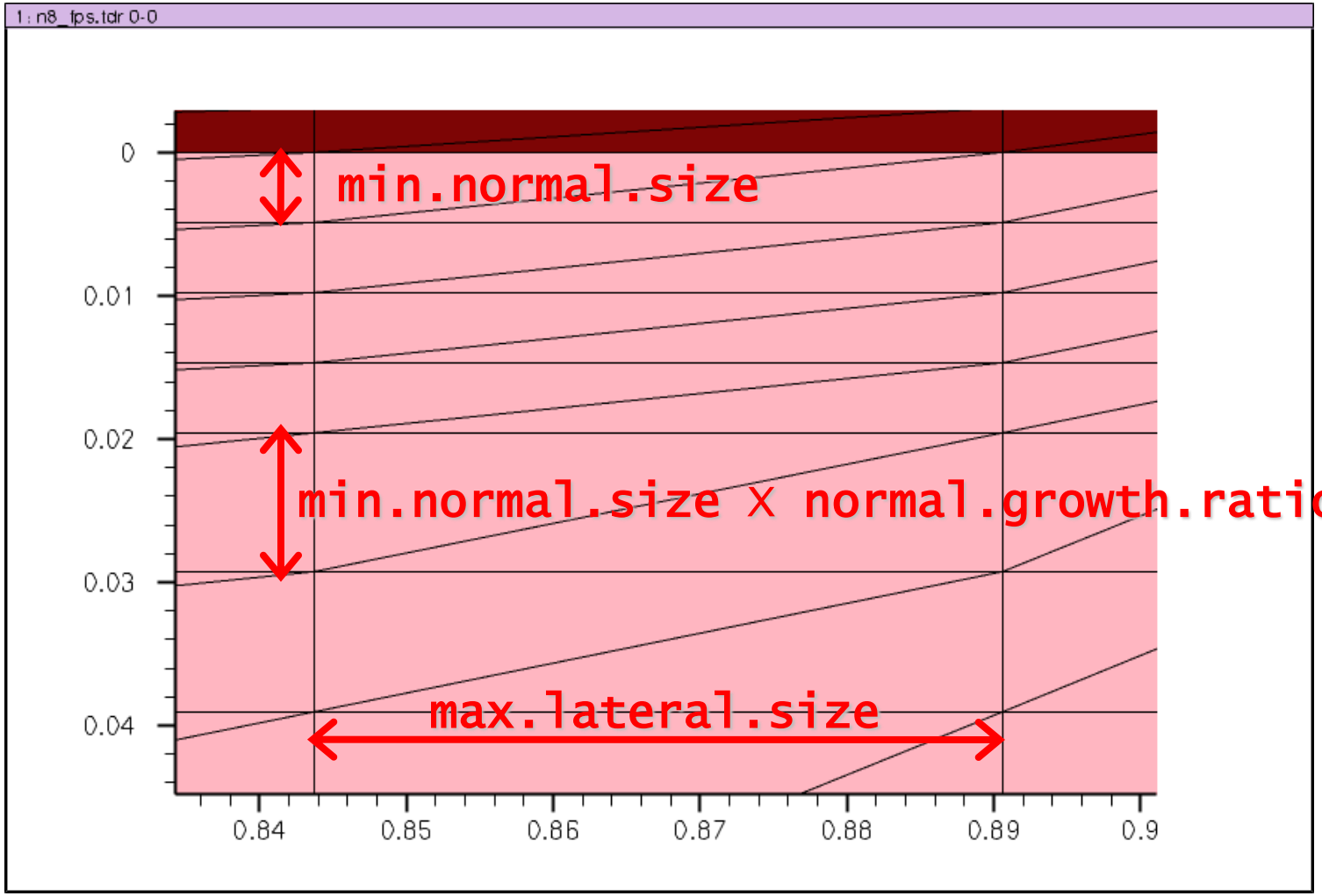
# Mesh command : mgoals ~usage~ (1/2)

```
mgoals min.normal.size=5<nm> max.lateral.size=2.0<um> \
  normal.growth.ratio=2.0

pdbSet Grid Mgoals UseLines 1
```

- **pdbSet Grid Mgoals UseLines 1** :
  It is recommended to use the UseLines mesher because this produces better element quality and better mesh reproducibility that reduces numeric error due to interpolation.

© 2007 Synopsys, Inc. (63)

# Mesh command: mgoals ~parameter~

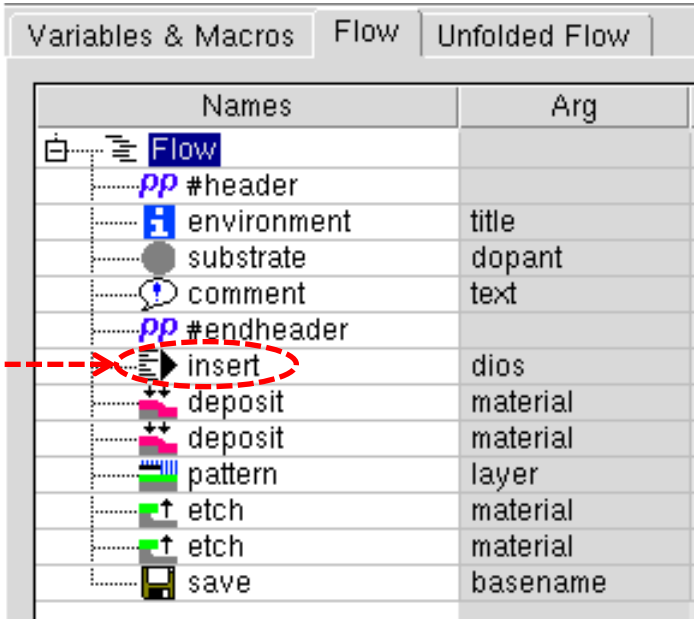



© 2007 Synopsys, Inc. (64)

# Ex: mgoals parameters (1/2)

- **Purpose :** confirm the parameters for “mgoals”.

1. Copy the project “/PostBasic/question/sprocess/base” under STDB folder.
2. Change copied project’s name to “mgoals”.
3. Open the project and start Ligament Flow Editor.
4. Insert the “**insert**” command after #**endheader** command.
5. Specify `mgoals` command like below:




```
mgoals min.normal.size=@MNS@ normal.growth.ratio=@NGR@
pdbSet Grid Mgoals UseLines 1
```

© 2007 Synopsys, Inc. (65)

# Ex: mgoals parameters (2/2)

6. Save and close Ligament Flow Editor.

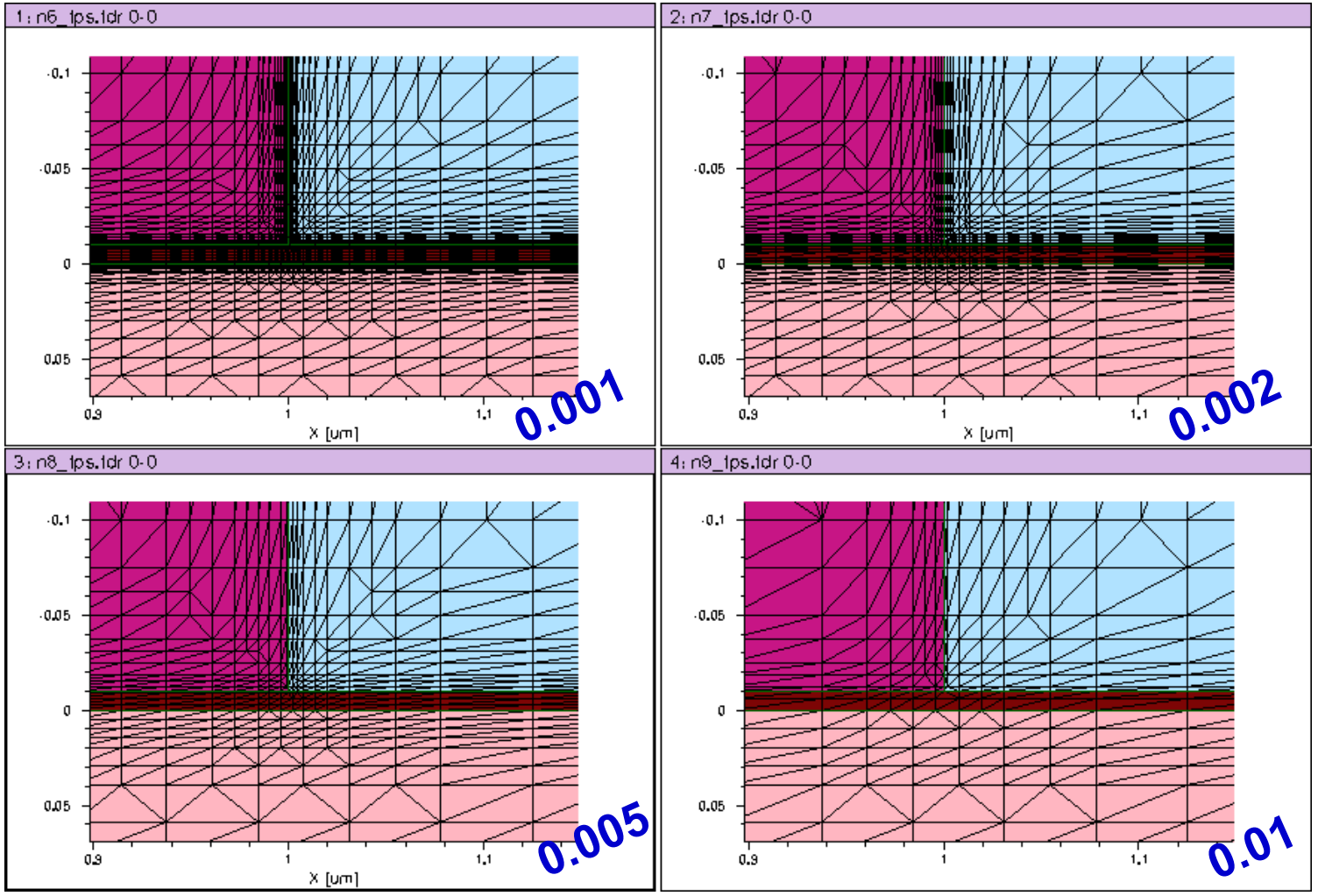
Then, try two items.

- Vary min.normal.size’s value : 0.001/0.002/0.005/0.01
  with normal.growth.ratio=1.41
  Use parameter “MNS” in swb environment .

- Vary normal.growth.ratio’s value : 1.1/1.41/1.73/2.0
  with min.normal.size=0.001
  Use parameter “NGR” in swb environment .

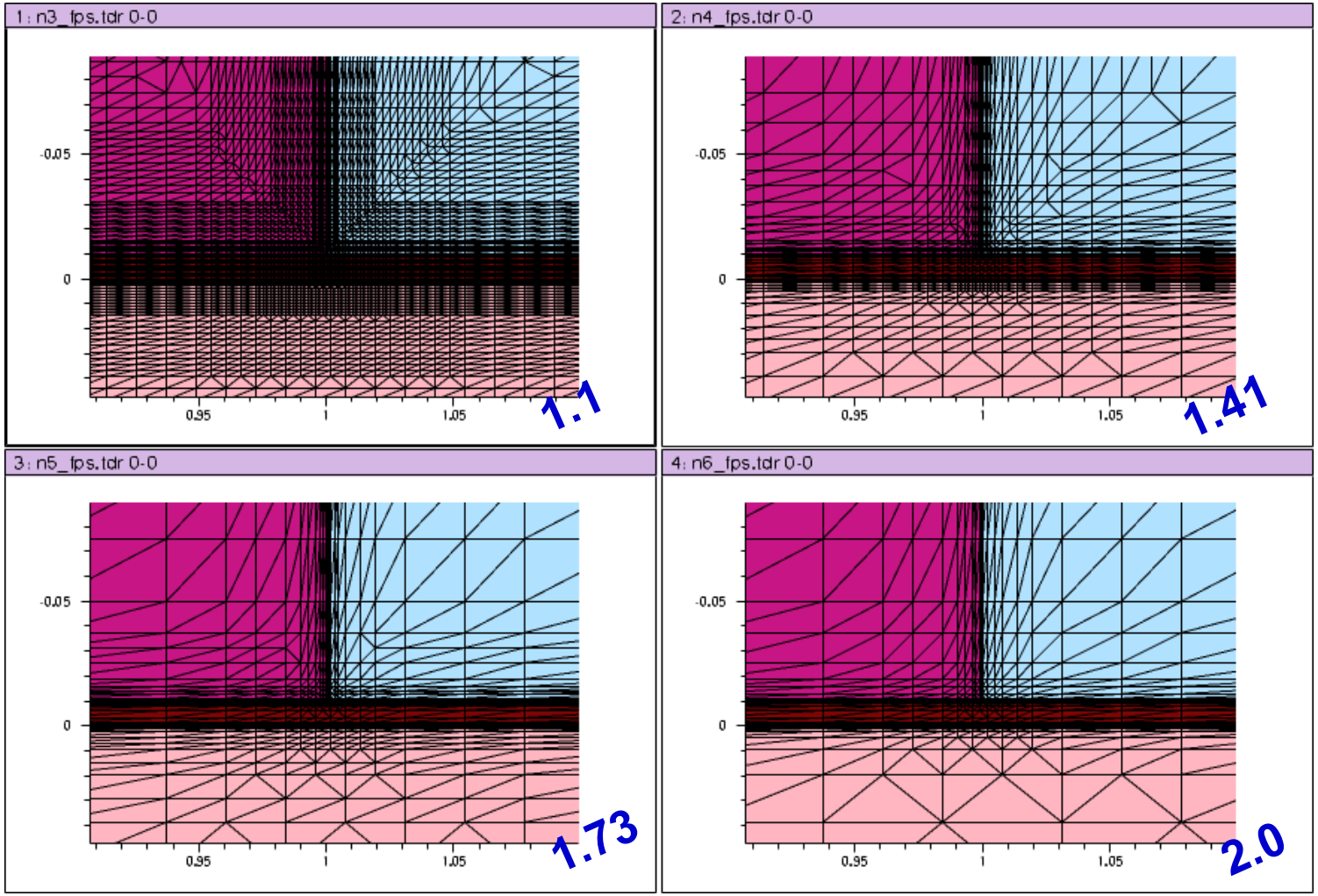
© 2007 Synopsys, Inc. (66)

# Ex: mgoals ~the results~ (min.normal.size)




© 2007 Synopsys, Inc. (67)

# Ex: mgoals ~the results~ (normal.growth.ratio)




© 2007 Synopsys, Inc. (68)

# Mesh command : refinebox ~usage~

Box name

Minimum x/y coordinate

Maximum x/y coordinate

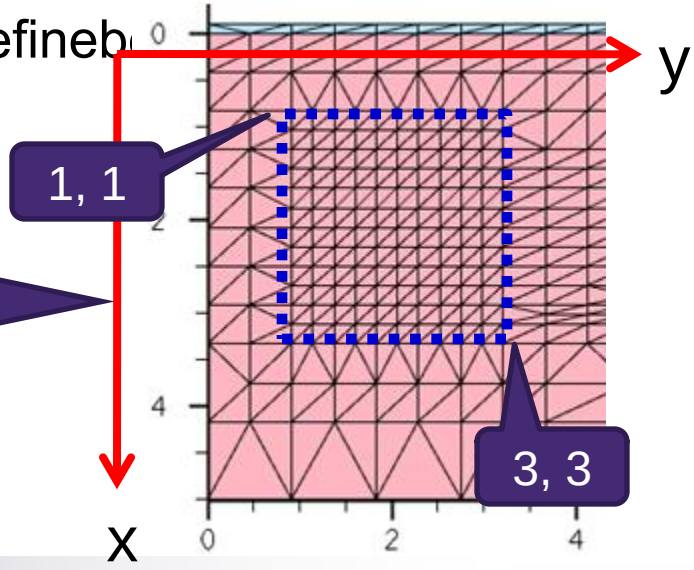
```
refinebox name=box min= { 1 1 } max= { 3 3 } \
   xrefine= { 0.25 } yrefine= { 0.25 }

grid remesh
```

Mesh size x direction

Mesh size y direction

- \ means that the command continues in next line.
- "grid remesh" is necessary just after “refineb

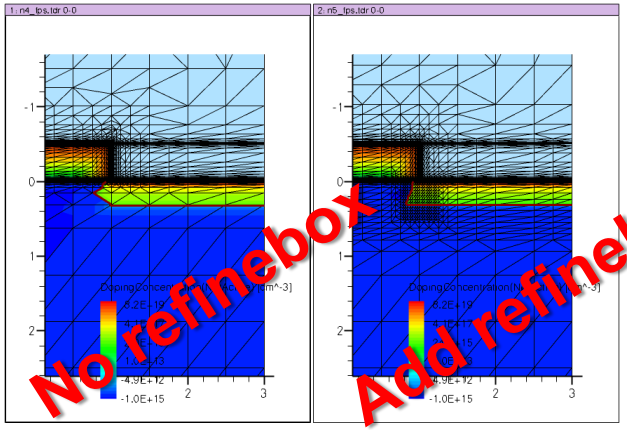
Caution!
Take care of coordinate system. Y is horizontal direction in Sprocess-specific commands.




© 2007 Synopsys, Inc. (69)

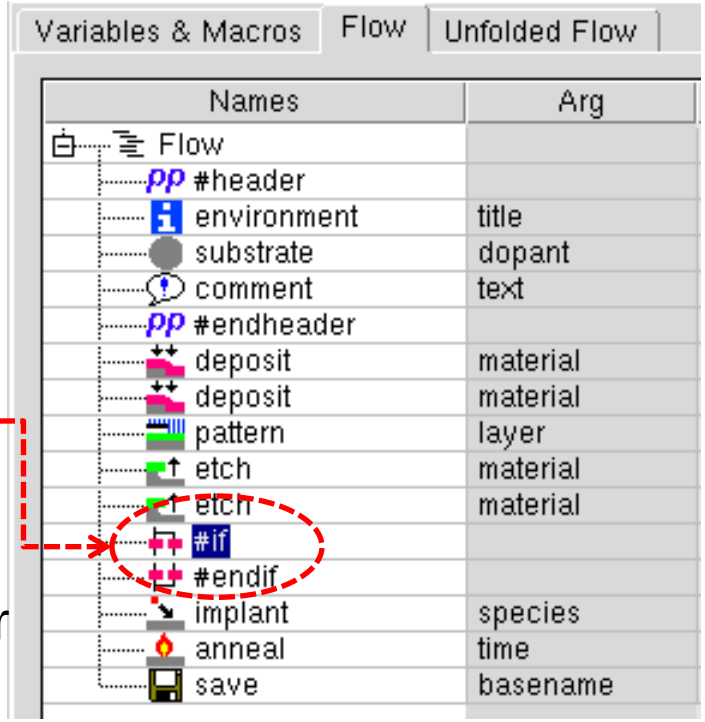
SYNOPSYS®
Predictable Success

# Ex: refinebox (1/3)




- **Purpose :** confirm the effects of `refinebox` command.

1. Copy the project "/PostBasic/question/sprocess/refinebox" under STDB folder.
2. Change copied project's name to "refinebox"
3. Open this project and start Ligament Flow Editor.
4. Insert #**if** command before the implant command. After that, if and end commands are added.
   Then, specify `@refine@` in `#if` parameter of if command's argument.




© 2007 Synopsys, Inc. (70)

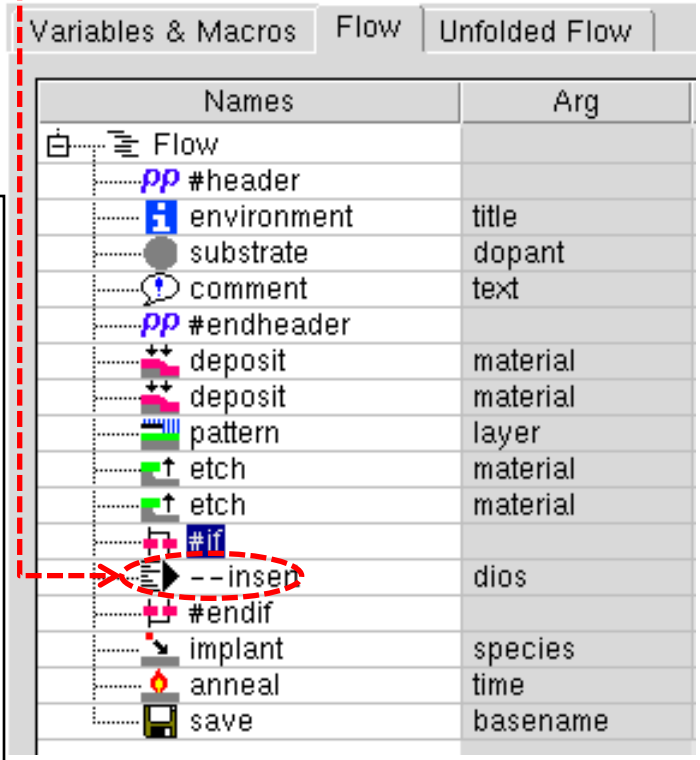
SYNOPSYS®
Predictable Success

# Ex: refinebox (2/3)

7. Insert the **insert** command between #**if** command and #**endif** command.

8. Specify `refinebox` command like below:

```
refinebox name=RB.Top \\
    min= {0 0} max= {0.75 3} \\
    xrefine= {0.07} yrefine= {0.5}
refinebox name=RB.Con \\
    min= {0 0.7} max= {0.5 1.2} \\
    xrefine= {0.05} yrefine= {0.05} \\
refinebox name=RB.Con \\
    min= {0 0.8} max= {0.4 1.0} \\
    xrefine= {0.02} yrefine= {0.02} \\

grid remesh
```

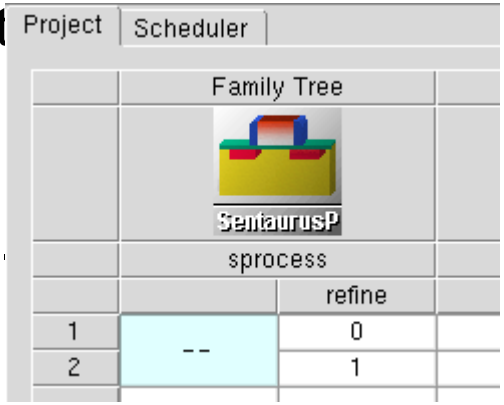



9. Save and close Ligament Flow Editor.

© 2007 Synopsys, Inc. (71)

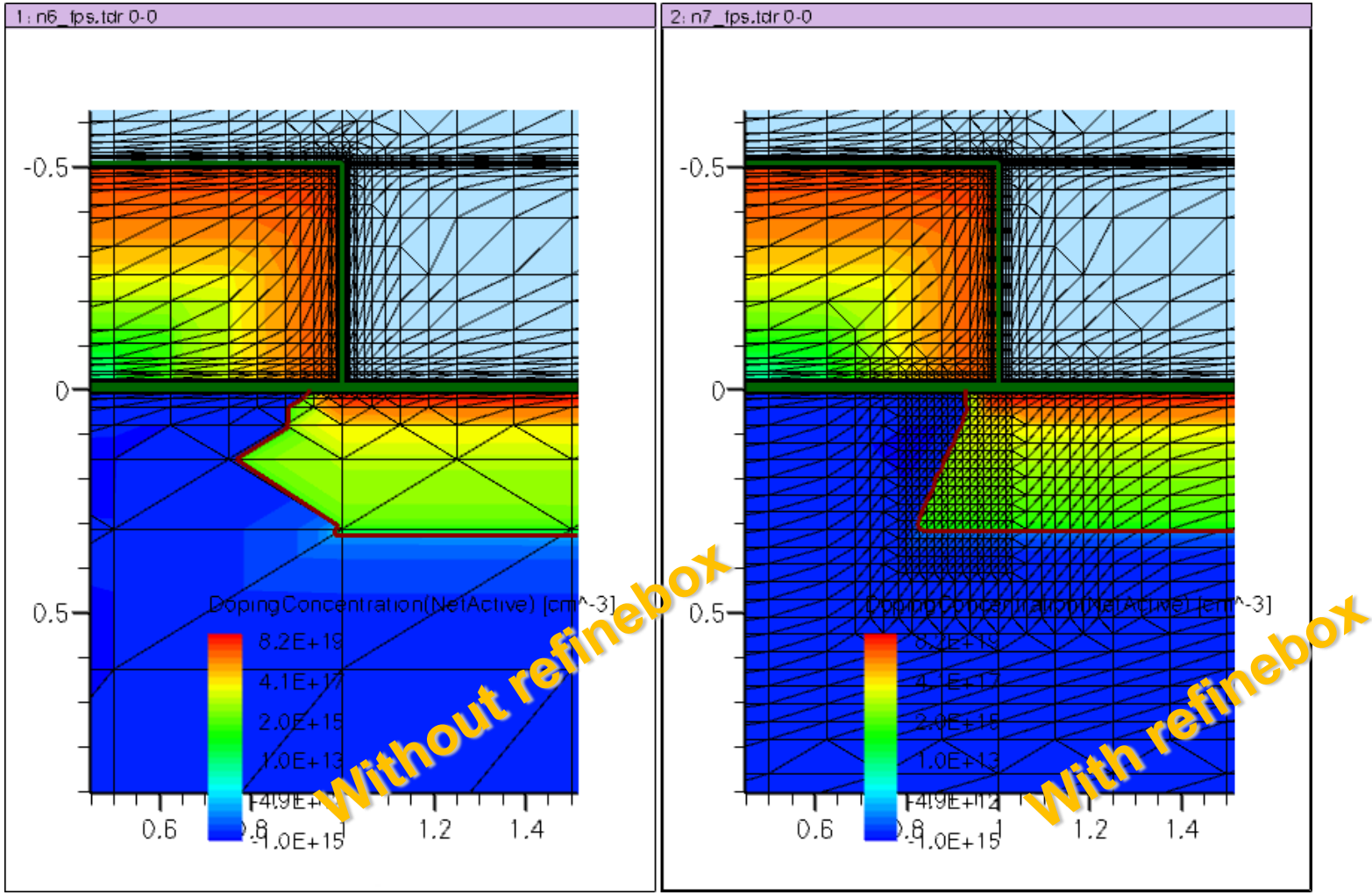
# Ex: refinebox (3/3)

10. “Parameter → Add” in the Sentaurus Workbench main window.
    The Add Parameter dialog box is displayed.
11. Enter `refine` in the Parameter field and “0” as the Default Value.
12. Click OK.
    The “refine” parameter is added to the Sent
    project.
13. Add value “1” to the `refine` parameter.
    An Experiment with “0” is without refinebox.
    An Experimant with “1” is with refinbox.



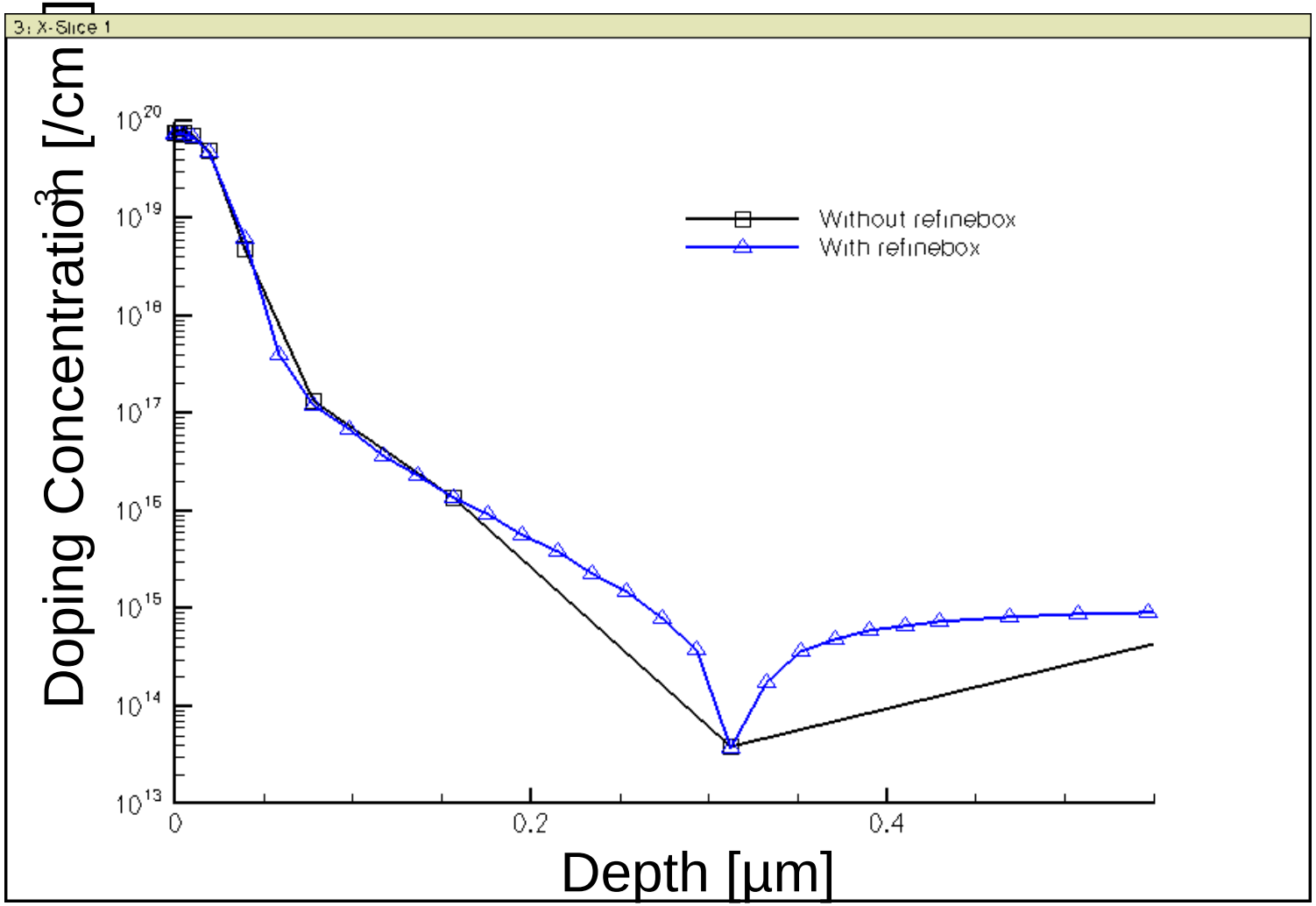

© 2007 Synopsys, Inc. (72)

# Ex: refinebox ~The results~




© 2007 Synopsys, Inc. (73)

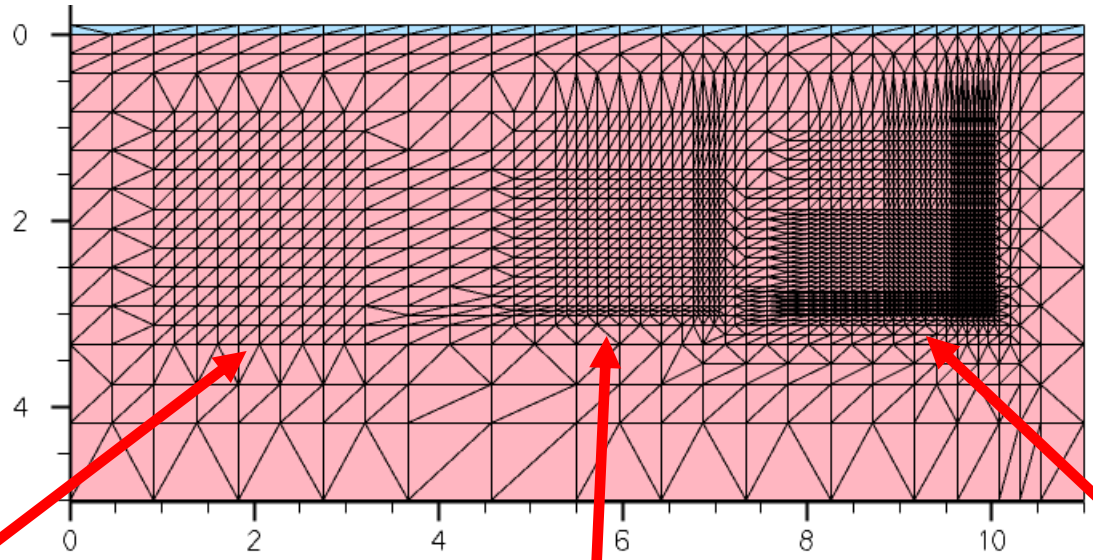
# Ex: refinebox
## ~ comparison of 1D profile at y=1.0 ~




© 2007 Synopsys, Inc. (74)

# Mesh command: refinebox ~Advance~

- Mesh size can be gradient by x/yrefine option



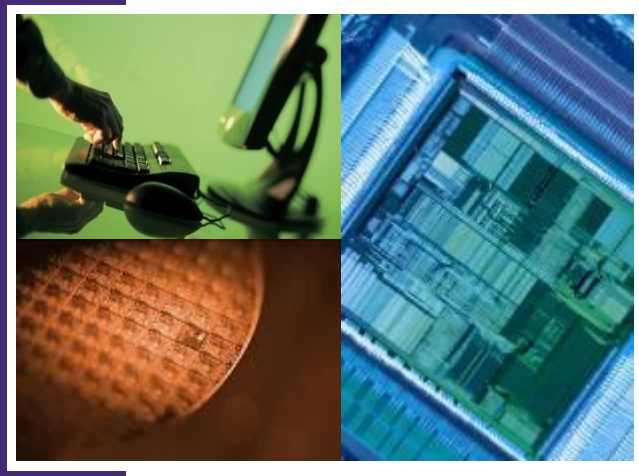

```
refinebox name=left
min= {1 1} max= {3 3}
xrefine= {0.25}
yrefine= {0.25}
```

```
refinebox name=mid
min= {1 5} max= {3 7}
xrefine= {0.25 0.1}
yrefine= {0.25 0.1}
```

```
refinebox name=right
min= {1 7} max= {3 10}
xrefine= {0.25 0.1 0.05}
yrefine= {0.25 0.1 0.05}
```

© 2007 Synopsys, Inc. (75)

# Extraction



SYNOPSYS®

Predictable Success

# Extraction

- Sentaurus Process supports the extraction, layer thickness, Xj and so on.
- This feature is useful to check these values.
- Some useful command is introduced from next slide.

© 2007 Synopsys, Inc. (77)

# Extracting oxide thickness

- Sprocess can extract the interface coordinates.
- User can extract material (layer) thickness.
- An example is :

```
set x1 [interface Oxide /Silicon y=2]
set x2 [interface Gas /Oxide y=2]

puts "DOE: Tox_A [format %.1f [expr ($x1 -$x2)*1000]]"
```

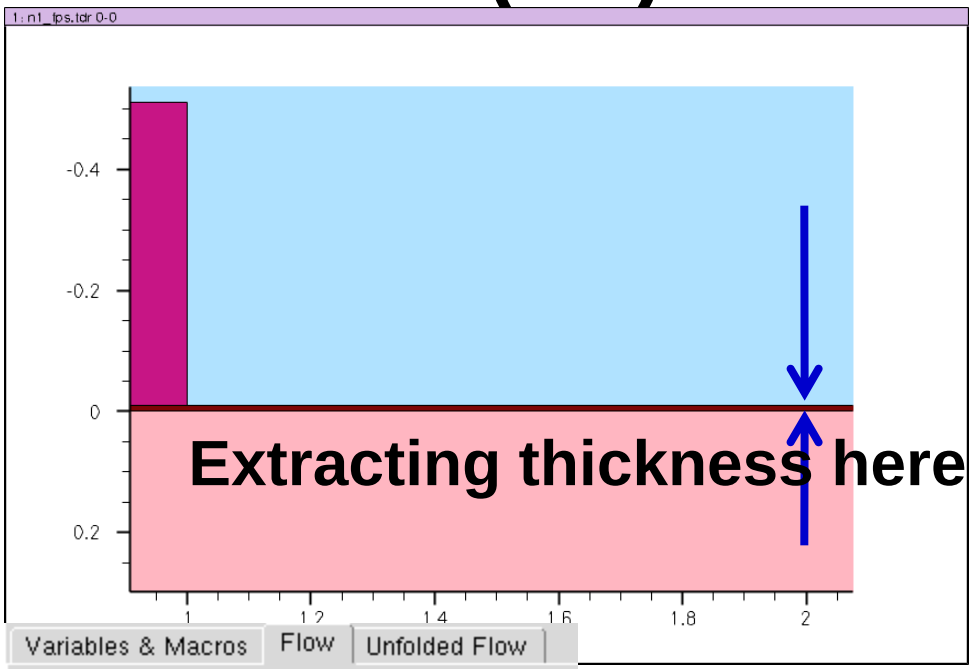
  - "`interface`" is the command for extraction of interface coordinate.
  - "`puts "DOE…`" is for displaying the result on swb.
  - "`Tox_A`" is variable name for output.
  - "`format %.1f`" is output format of the fractional part of decimal fractions.
  - "`expr ($x1 -$x2)*1000`" is for calculation the difference of each coordinate and convert unit from μm to angstrom.

© 2007 Synopsys, Inc. (78)

# Ex: extracting oxide thickness (1/2)

- **Purpose :** extract oxide thickness

1. Copy the project "/PostBasic/question/sprocess/base" under STDB folder.
2. Change copied project's name to "extract_tox"
3. Open this project and start Ligament Flow Editor.
4. Insert **insert** command the last etch command and define parameters :




Variables & Macros | Flow | Unfolded Flow

| Names | Arg |
|---|---|
| Flow | |
| #header | |
| environment | title |
| substrate | dopant |
| comment | text |
| #endheader | |
| deposit | material |
| deposit | material |
| pattern | layer |
| etch | material |
| etch | material |
| insert | dios |
| save | basename |

© 2007 Synopsys, Inc. (79)

# Ex: Extracting oxide thickness (2/2)

5. Specify sprocess-specific command in **insert** command like below:

```
set x1 \[interface Oxide /Silicon y=2\]
set x2 \[interface Gas /Oxide y=2\]

puts \"DOE: Tox_angstr \[format %.1f \[expr (\$x1 - \$x2)*10000\]\]\"
```

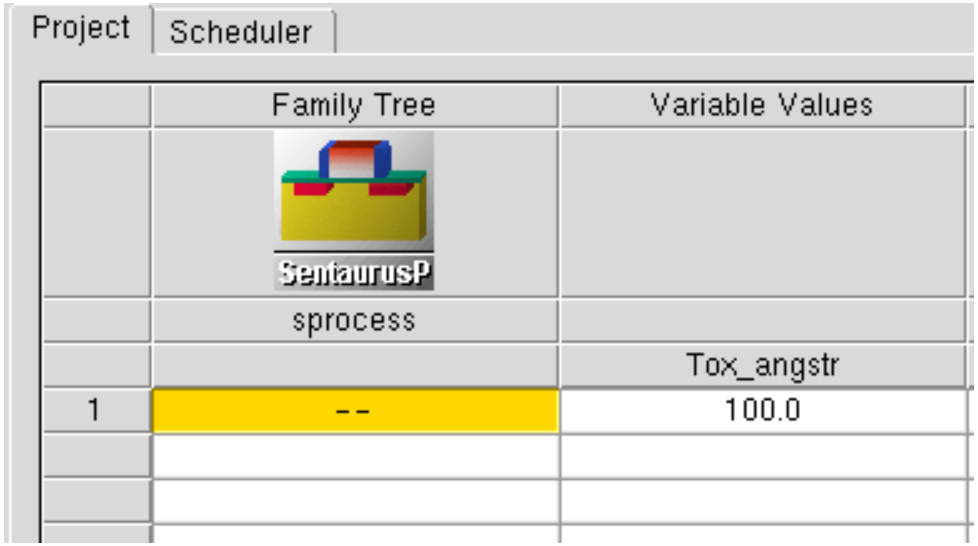
- ***Caution***
  *\ are needed before some special symbolic character, [, ], " and $ because Ligament cannot recognize special character without \.*

6. Finish Ligament Flow Editor and run the project.

© 2007 Synopsys, Inc. (80)

# Ex: Extracting oxide thickness ~The result~

- Extracted value is displayed on swb.




© 2007 Synopsys, Inc. (81)

# Extracting Xj

- Sprocess can extract xj at the specified location with “interpolate” command.
  An example for extraction of Xj is :

```
select z="NetActive"
set x0 [interpolate Silicon y=2 value=0]
set x1 [lindex $x0 0]

set x2 [interface Oxide /Silicon y=2]

puts "DOE: Xj_um [format %.1f [expr ($x1 -$x2)]]"
```

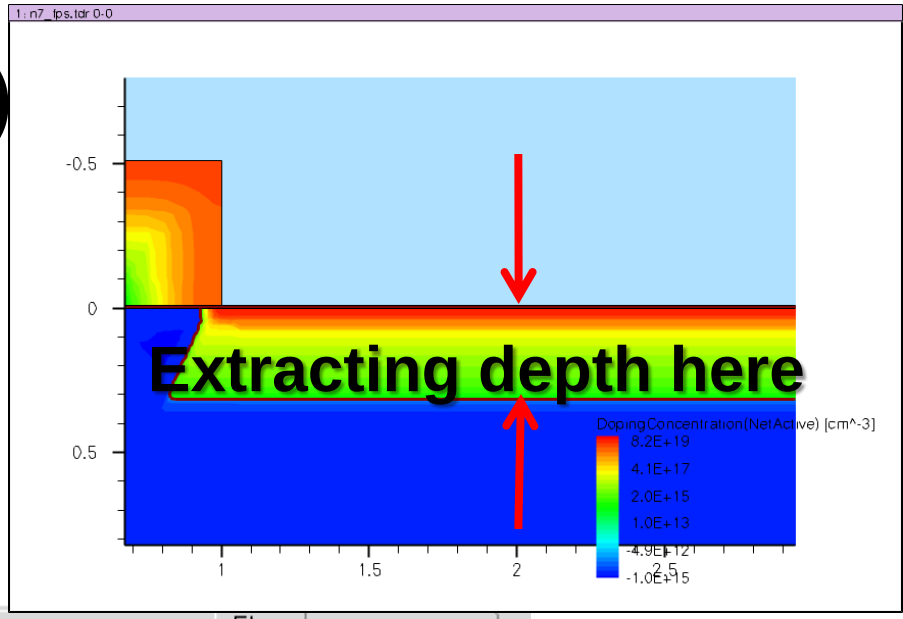
  - `select` command define extraction parameter for `value` keyword. `NetActive` means “*n*-concentration – *p*-concentration”
  - `y, value` : The combination of these parameters determines how the command operates. y and value are given, the locations along y where `value` is crossed are returned.
  - `lindex` : Return the first value if returned values are multiple.

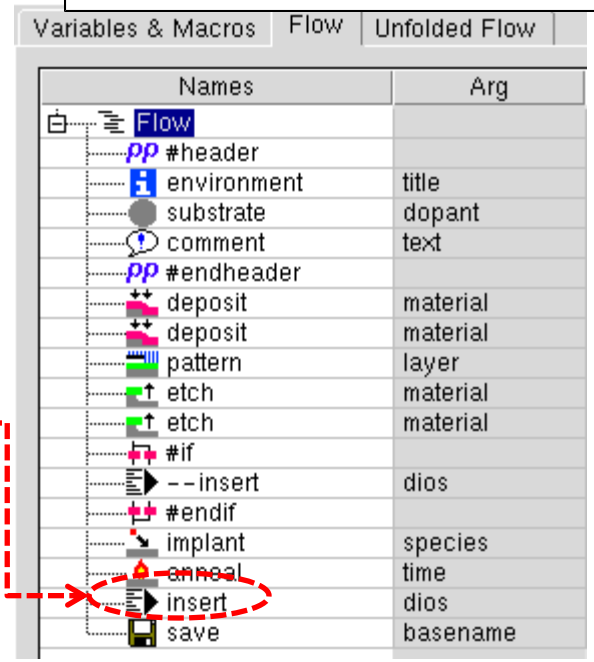
© 2007 Synopsys, Inc. (82)

# Ex: extracting Xj (1/2)

- Examine the following parameters for refinebox

1. Copy the project "/PostBasic/answer/sprocess/refinebox" under STDB folder.
2. Change copied project's name to "extract_xj"
3. Open this project and start Ligament Layout Editor.
4. Insert **insert** command the last etch command and define parameters :




© 2007 Synopsys, Inc. (83)

SYNOPSYS®
Predictable Success

# Ex: extracting Xj (2/2)

5. Specify sprocess command in **insert** command like below:

```
select z=\"NetActive\"
set x0 \[interpolate Silicon y=2 value=0\]
set x1 \[lindex \$x0 0\]

set x2 \[interface Oxide /Silicon y=2\]

puts \"DOE: Xj_um \[format %.3f \[expr (\$x1 -\$x2)\]\]\"
```

- ***Caution***
  *\ are needed before some special symbolic character, [, ], " and $ because Ligament cannot recognize special character without \.*

6. Save and close Ligament Flow Editor and run the project.

© 2007 Synopsys, Inc. (84)

# Ex: extracting Xj ~the result~

- Extracted value is displayed on swb.

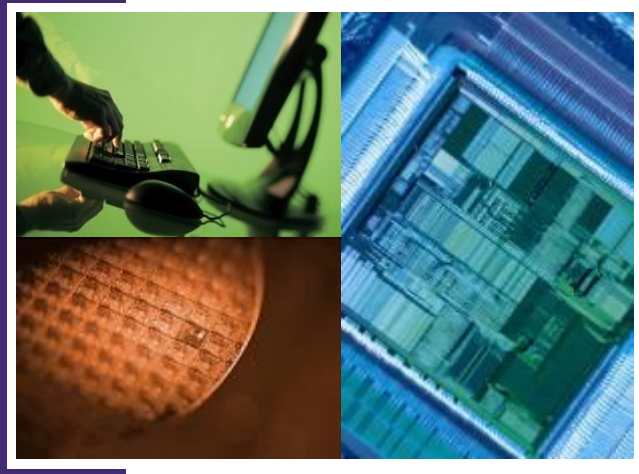
Project | Scheduler

| | Family Tree | Variable Values |
|---|---|---|
| |  | |
| | sprocess | |
| | refine | Xj_um |
| 1 | 0 | 0.325 |
| 2 | 1 | 0.316 |



© 2007 Synopsys, Inc. (85)

# Changing diffusion models



SYNOPSYS®

Predictable Success

# Diffusion Models

- **The ChargedPair diffusion model**: **(default)**
  - Used for high dose amount of implantation. (damage is high.)
  - Used for short-term anneals.
  - Used for advanced CMOS processes as they represent a balance between accuracy and computational expense.

- **The ChargedFermi diffusion model**:
  - Used for low dose amount of implantation. (damage is low.)
  - Used for long-term high-temperature anneals.
  - The calculation time is faster than ChargedPair diffusion model.

**The detail of model is introduced in 2D Advanced Sentaurus Process course.**

© 2007 Synopsys, Inc. (87)

# Change Models

The command to change the diffusion model is :

```
pdbSet Silicon Dopant DiffModel ChargedFermi
```

Simulators and available diffusion models

| | SProcess | Dios | TSuprem4 |
|---|---|---|---|
| 1-Stream | ChargedFermi<br>SetDiosEquilibriumModelMode | Equilibrium<Default> | NSTREAMS=1<br><Default> |
| 3-Streams | ChargedPair<Default><br>SetDiosPairModelMode | PairDiffusion | NSTREAMS=3 |

***Caution!***

*The default model is different between each simulator.*

*ChargedFermi model is generally appropriate for power device.*

© 2007 Synopsys, Inc. (88)

# 1D Example

- Copy project “1D_SProcess” from training_library
- Refer to the command file of example “1D_SProcess”
- Check 1D doping profile by INSPECT
- Output file introduction

© 2007 Synopsys, Inc. (89)

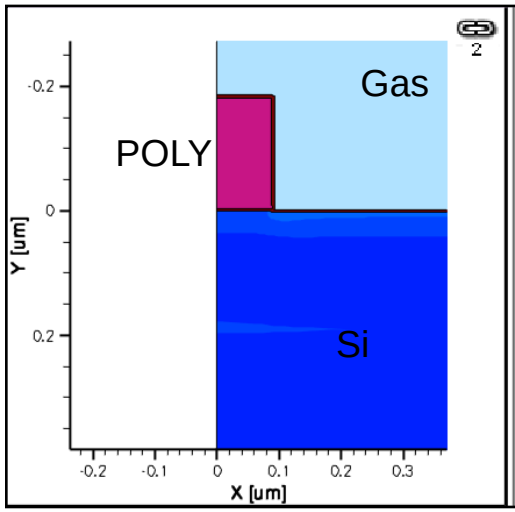
# 2D Example

- Copy project “2D_SProcess” from training_library
- Refer to the command file of “2D_SProcess”

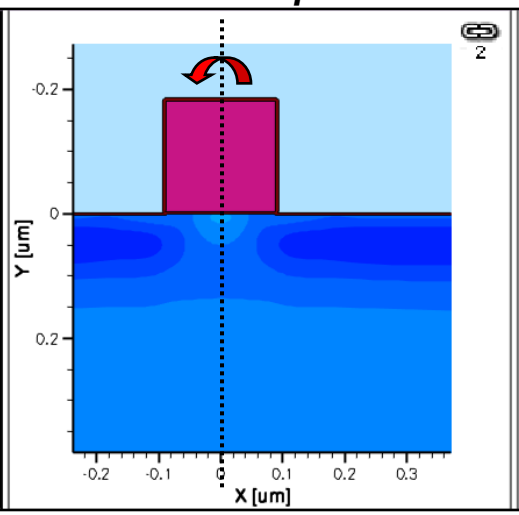
© 2007 Synopsys, Inc. (90)

# 2D Example Process Flow

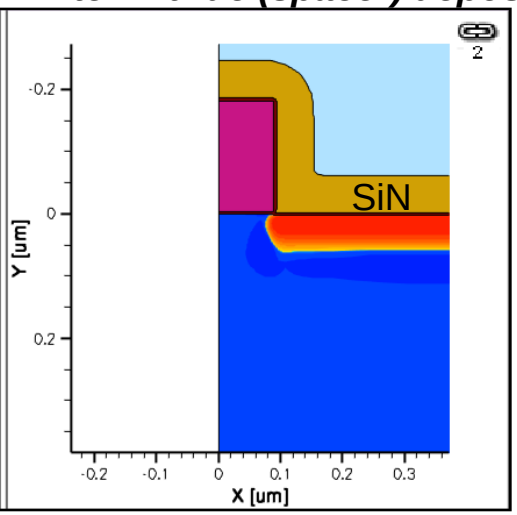
*After POLY re-oxidation*



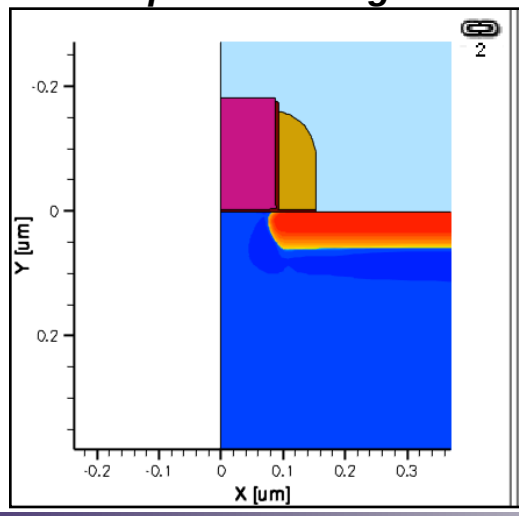

*After HALO implantation*



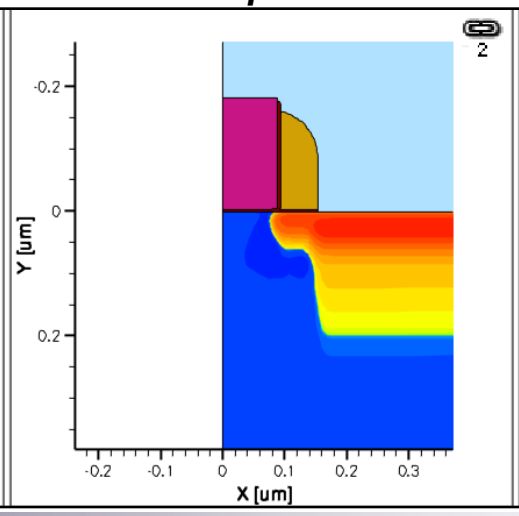

*After Nitride (spacer) deposition*



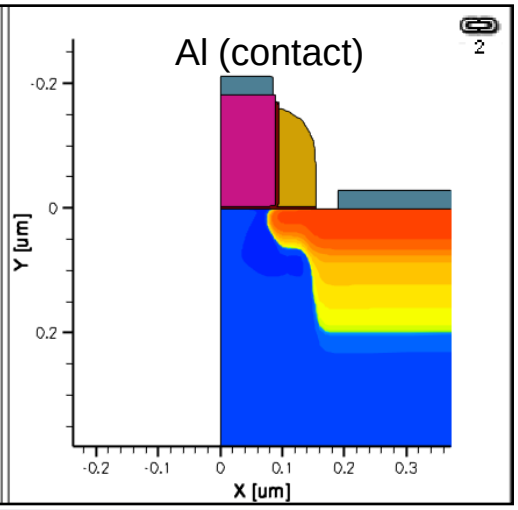

*After spacer etching*




*After S/D implantation*



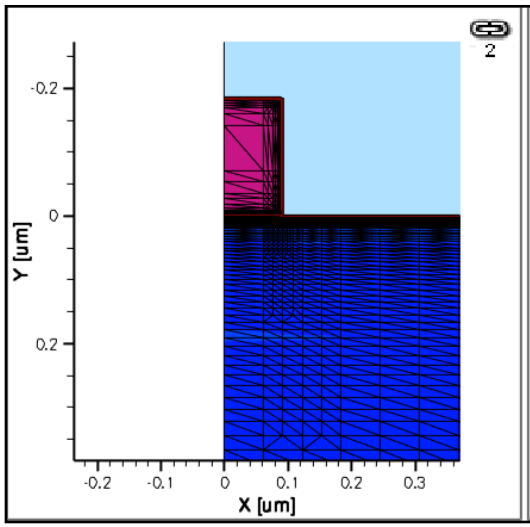

*Final structure*



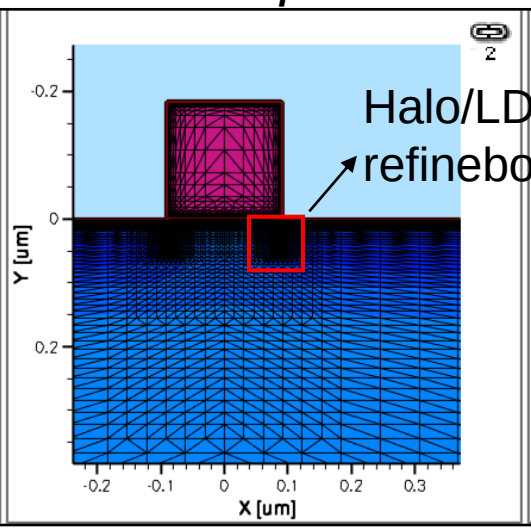

© 2007 Synopsys, Inc. (91)

# 2D Example Mesh Setting

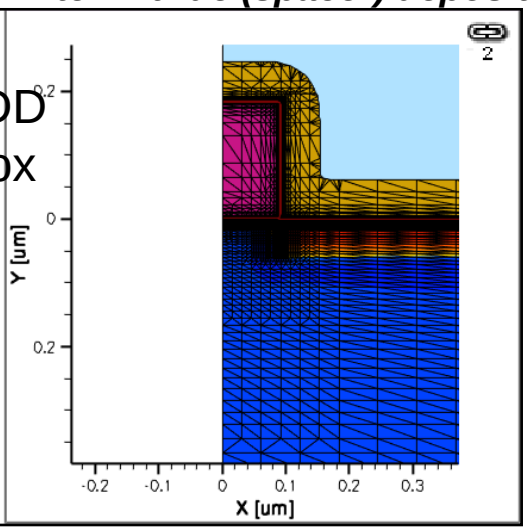
*After POLY re-oxidation*



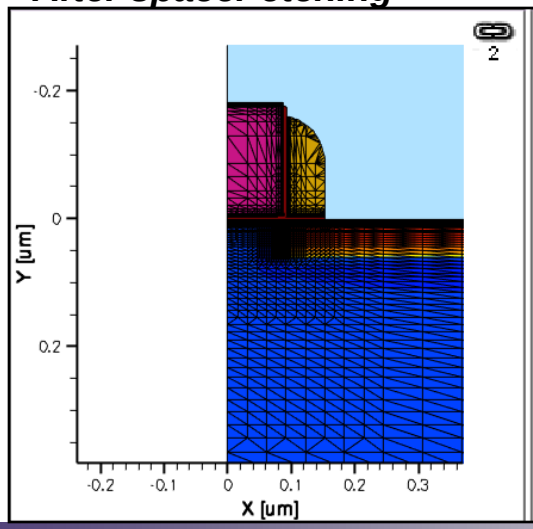

*After HALO implantation*



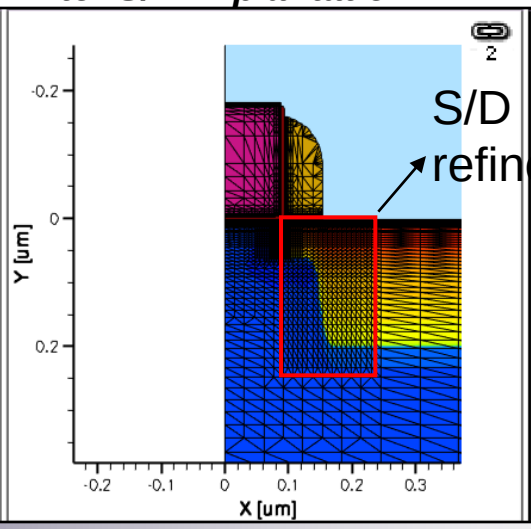

*After Nitride (spacer) deposition*



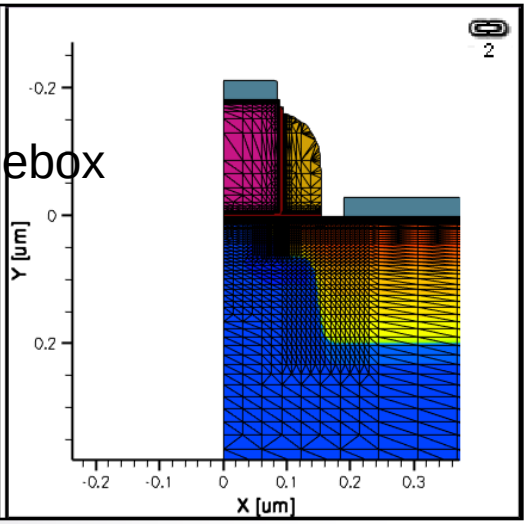

*After spacer etching*




*After S/D implantation*



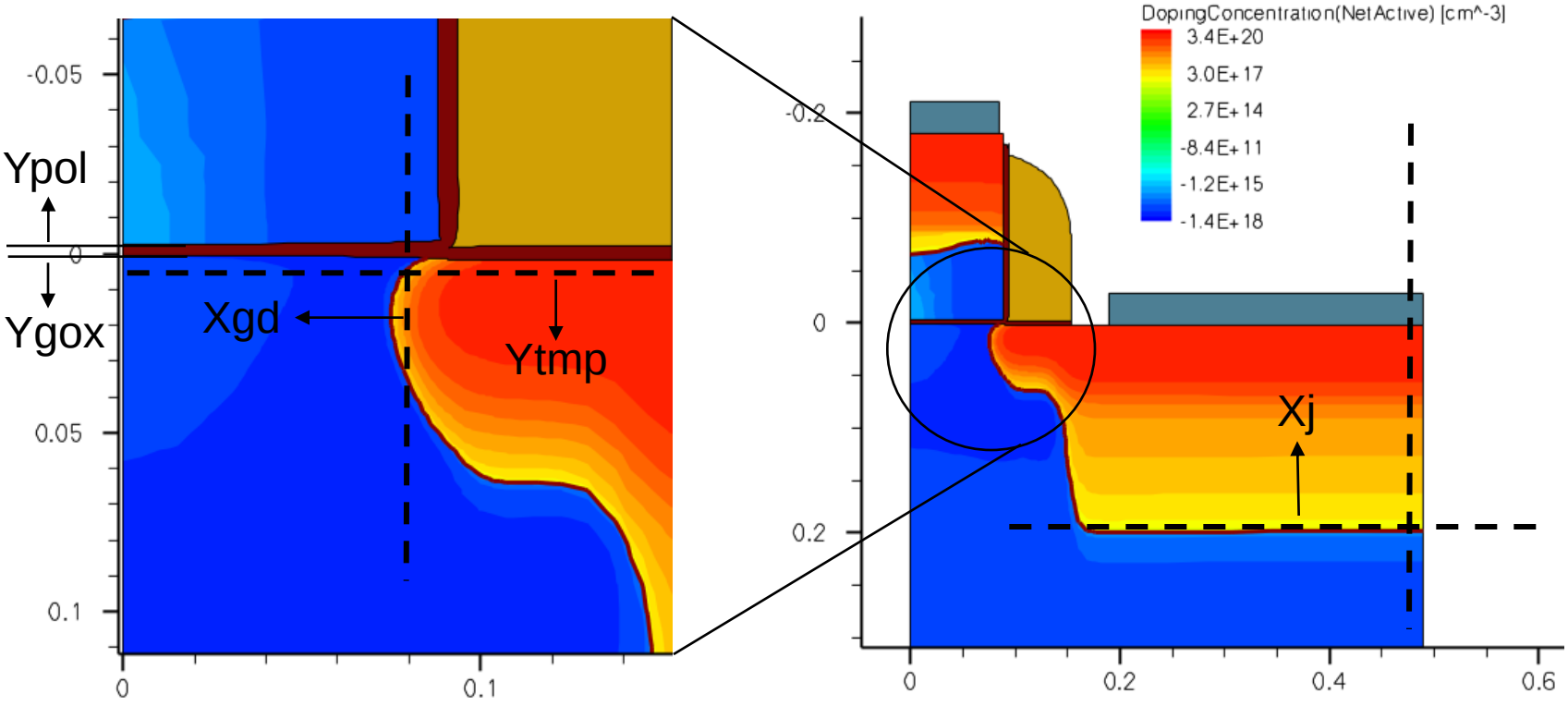

*Final structure*




© 2007 Synopsys, Inc. (92)

# Extract Parameters

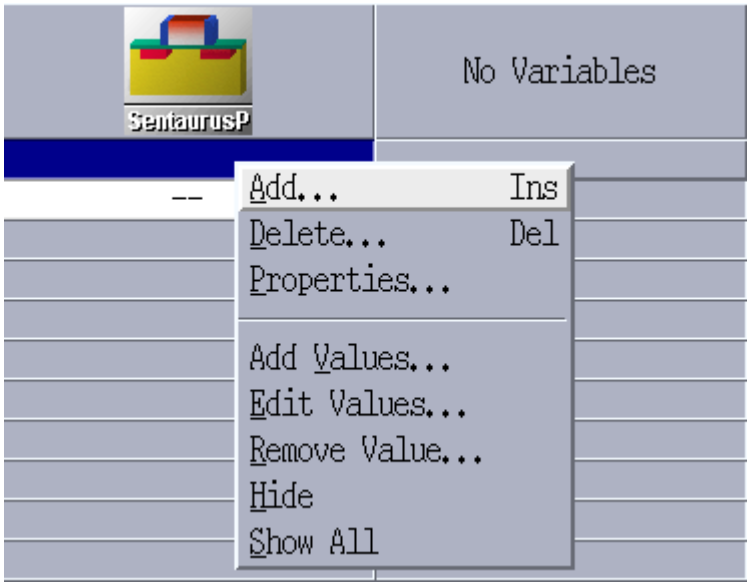
Ygox (X coordinate of oxide/silicon interface)
Ypol (X coordinate of POLY/oxide interface)
Tox (gate oxide thickness) = Ygox-Ypol
Lgeff (effective channel length) = 2*Xgd
Xj (Source/Drain junction depth)



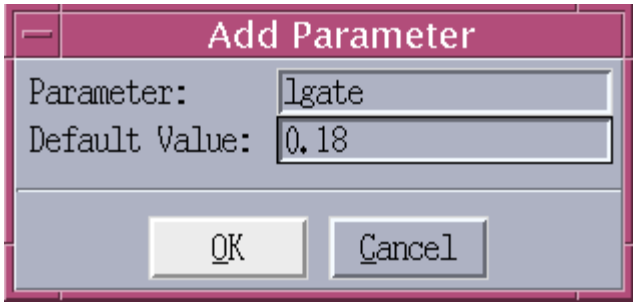

© 2007 Synopsys, Inc. (93)

# Implement External Variables on SWB

- copy 2D_SProcess > clean up > add parameters > modify command file > clear up & renumber the experiment tree




Add external parameters on workbench




| | Family Tree | | | | Variable Values | | | |
|---|---|---|---|---|---|---|---|---|
| | SentaurusP | | | | | | | |
| | | lgate | HaloDose | HaloEnergy | Lgeff | Xj | Ygox | Tox |
| 1 | -- | 0.18 | 1e13 | 15 | 1.571e-01 | 1.984e-01 | 6.497e-04 | 3.331e-03 |

© 2007 Synopsys, Inc. (94)

# Implement External Variables on SWB

- copy 2D_SProcess > clean up > add parameters > modify command file > clear up & renumber the experiment tree

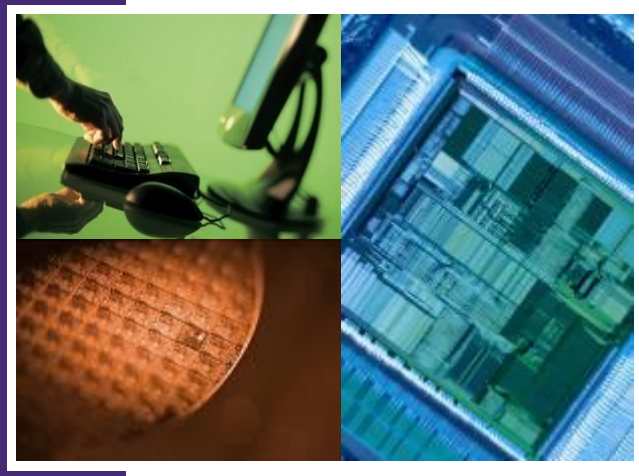
Link external variables to internal variables

```
#--- set variable ---------------------
set lgate          0.18
set ymax           [expr $lgate/2+0.4]
set HaloDose       1e13
set HaloEnergy     15
```

```
#--- set variable ---------------------
set lgate          @lgate@
set ymax           @<lgate/2+0.4>@
set HaloDose       @HaloDose@
set HaloEnergy     @HaloEnergy@
```

© 2007 Synopsys, Inc. (95)

# Tecplot SV



SYNOPSYS®

Predictable Success

# Overview

- Tecplot  is software for scientific visualization that has been extended by Synopsys to accommodate the special requirements of TCAD simulations.
- Synopsys provides an original equipment manufacturer (OEM) distribution of Tecplot that includes the original Tecplot distribution by Tecplot, Inc., the Synopsys TCAD add-on, and the Synopsys tecplot_sv launcher, which starts Tecplot in a special Synopsys configuration mode.
- The complete package consisting of these three components is called Tecplot_SV.

© 2007 Synopsys, Inc. (97)

# Starting Tecplot_SV

- to start Tecplot_SV:

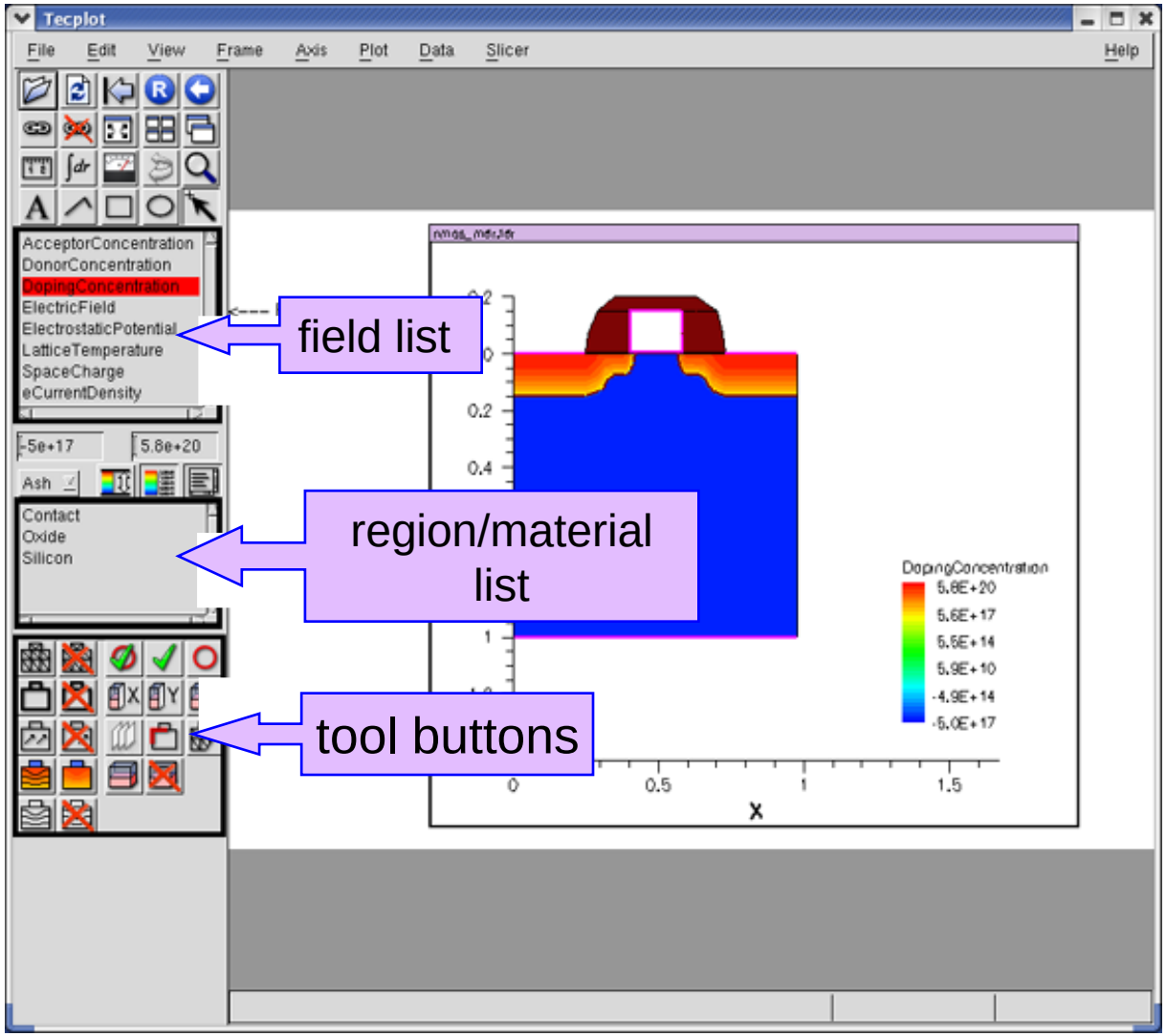
```
> tecplot_sv &
```

```
> tecplot_sv <filename> &
```

- in SWB : select Visualize → .tdr Files → Tecplot SV

© 2007 Synopsys, Inc. (98)

# Screen Elements




© 2007 Synopsys, Inc. (99)

SYNOPSYS®
Predictable Success

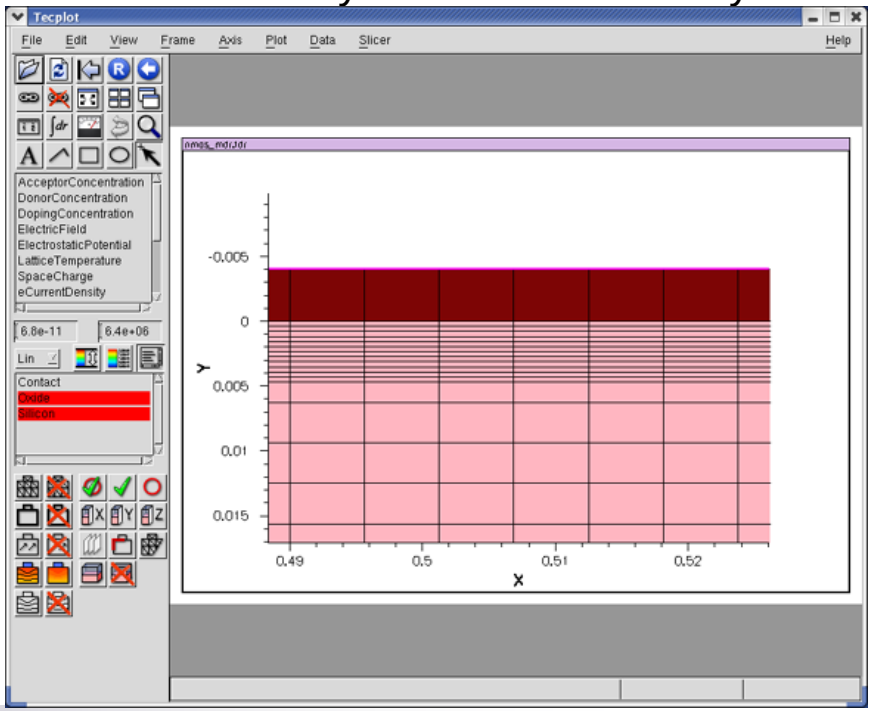
# View Control

- zooming: click on the zoom tool  and draw a rubber band, or use the middle mouse button and drag the pointer downwards/upwards for zoom in/out
- click on “last view” to return to the previous view, control-f fits the entire structure into the view window
- move the structure by holding the right mouse button and dragging the pointer

© 2007 Synopsys, Inc. (100)

# Displayed Objects

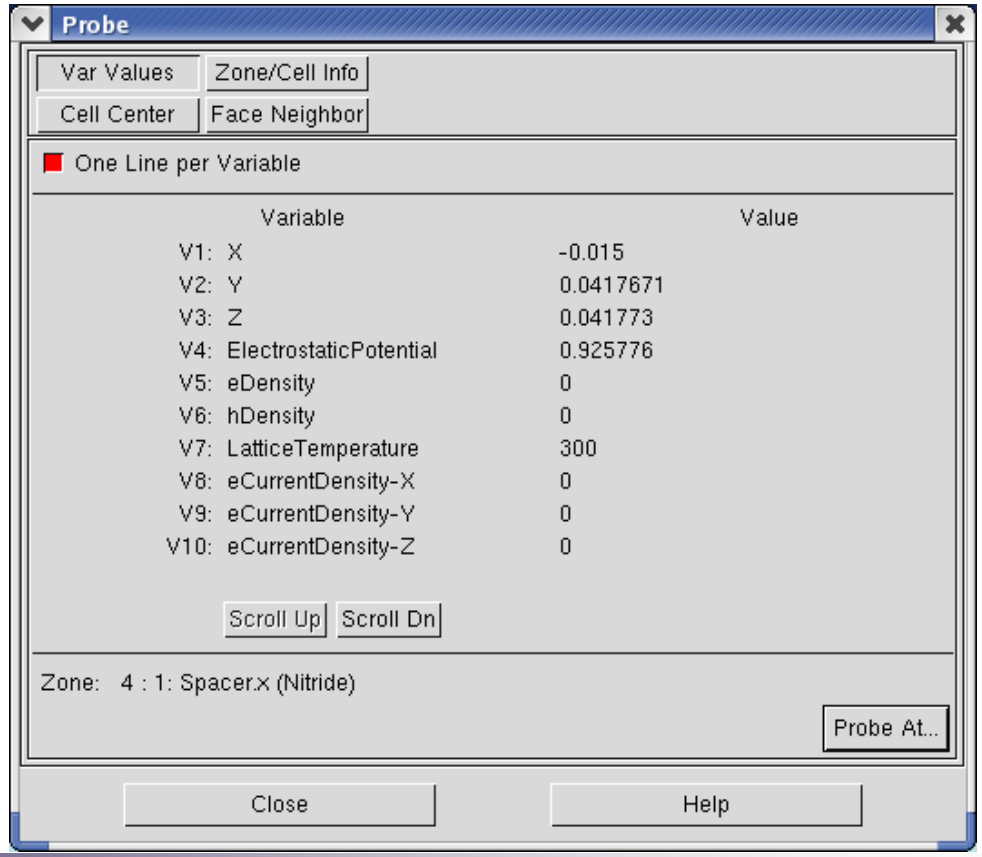
- Select both Silicon and Oxide in the Material list. Hold the Ctrl key to highlight both items.
- Click the Display Exclusive button to display these two material regions only and to switch off the display of any other regions.
- To limit the operations to only a specific region, click the Region List/Material List button to switch to the Region list, which selection by region.
- Click the Mesh On button.
- Click the Contours Off button on the sidebar to switch off any field that is currently displayed on the structure.




© 2007 Synopsys, Inc. (101)

# Data Sampling

- After a device structure has been loaded in Tecplot, the values of the physical fields existing on the structure can be seen using the data probing tool on the sidebar.




© 2007 Synopsys, Inc. (102)

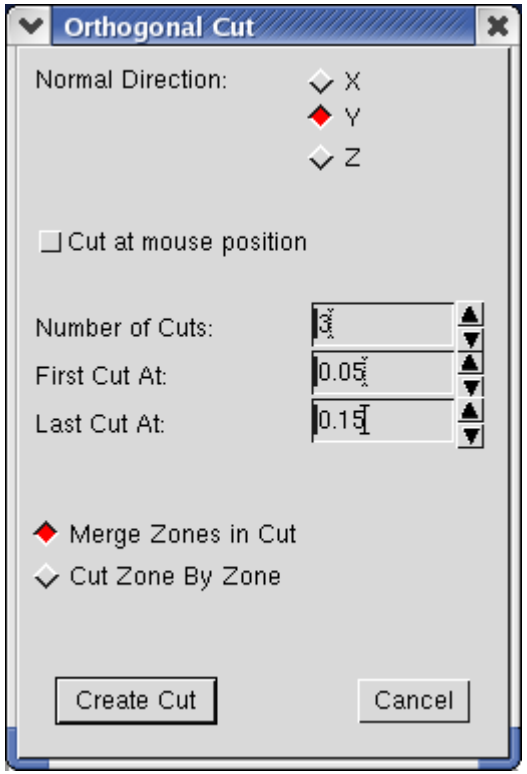
# Generating 1D Cuts

- Before a cut is made, it is necessary to decide whether the data obtained from the cut should go to an existing frame or a new frame.
  - Frame > Frame Linking. The Set Links for Current Frame dialog box is displayed
  - Select Slice Positions and click Apply Settings to All Frame, of this Group.
- One-dimensional cuts can be made along either the x or y coordinate axes
  - For example, to sample data along the x-axis, select the Orthogonal Cut  (Y for normal direction) function

© 2007 Synopsys, Inc. (103)

# Generating Multiple Cuts in One Step

- Multiple cuts in parallel to each other can be generated in one step.
- Slicer > Y Normal Cuts




© 2007 Synopsys, Inc. (104)

# Working with Frames

- For each successfully loaded data file, Tecplot starts a new frame and associates the frame with the loaded data. In addition, Tecplot SV automatically opens up new frames for newly generated cuts, either 1D or 2D.
- Tecplot permits multiple frames in the same workspace to be managed in a coordinated manner. These functions and commands are available from the Frame menu and are convenient to use. For example:
  - To send a frame to the back and bring another to the front: Frame > Push Current Frame Back.
  - To simultaneously show all the frames: Frame > Arrange Frames and select one of the options.

© 2007 Synopsys, Inc. (105)

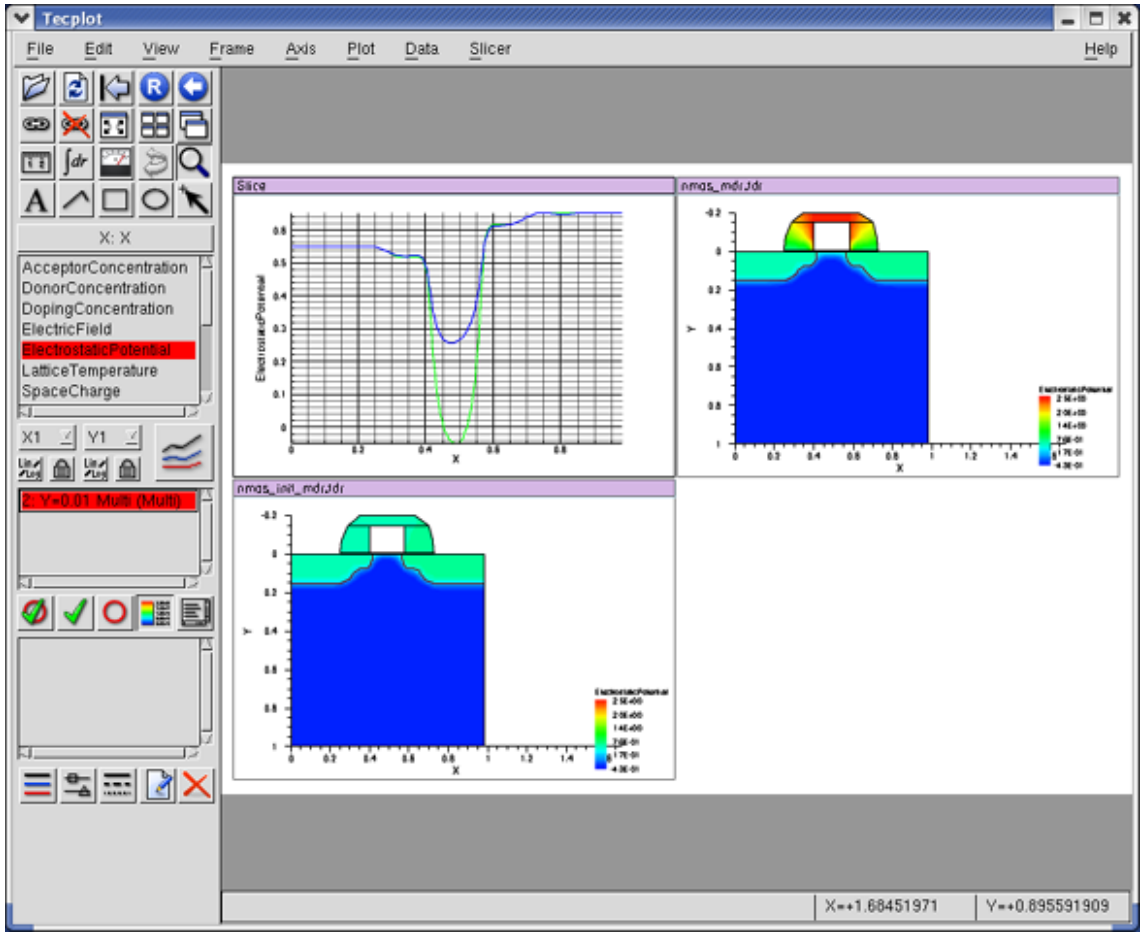
# Frame Linking

- The frame linking feature of Tecplot SV allows multiple frames to be linked by one or more of their common frame attributes.
- Changing an attribute in one frame results in the same change to all other frames linked with respect to that attribute.
- To link frames:
  - Click the All Frames button on the sidebar. 
  - Click the Link button. 
  - To break the link, click Unlink button. 

© 2007 Synopsys, Inc. (106)

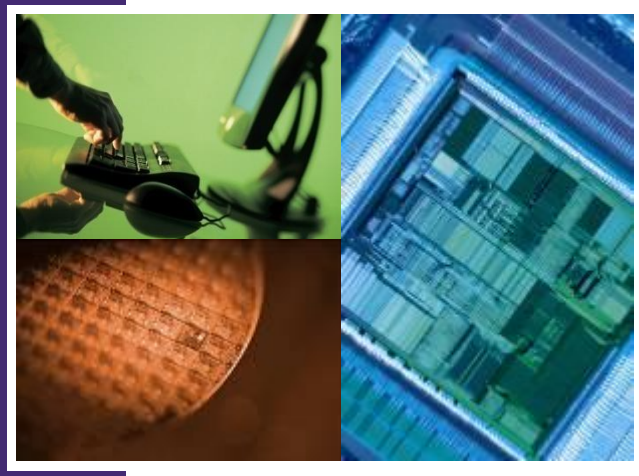
# Synchronized Cuts

- In addition to appearance coordination, linking allows synchronized cuts to be made simultaneously on all linked structures.




© 2007 Synopsys, Inc. (107)

# Sentaurus Structure Editor – Building Meshes (SDE)



SYNOPSYS®

Predictable Success

# Overview

- SDE is a structure editor for 2D and 3D device structures.
- From the graphical user interface, 2D and 3D device models are created geometrically, using 2D or 3D primitives, such as rectangles, polygons, cuboids, cylinders, and spheres.
- Rounded edges are generated by filleting, 3D edge blending, and chamfering. Complex shapes are generated by simply intersecting primitive elements.
- The graphical user interface of SDE features a command-line window, in which it prints script commands corresponding to the GUI operations.
- In process emulation mode (PROCEM), SDE translates processing steps, such as etching and deposition, patterning, fill and polish, into geometric operations.
- PROCEM supports various options, such as isotropic or anisotropic etching and deposition, and rounding and blending.

© 2007 Synopsys, Inc. (109)

# Modules of SDE

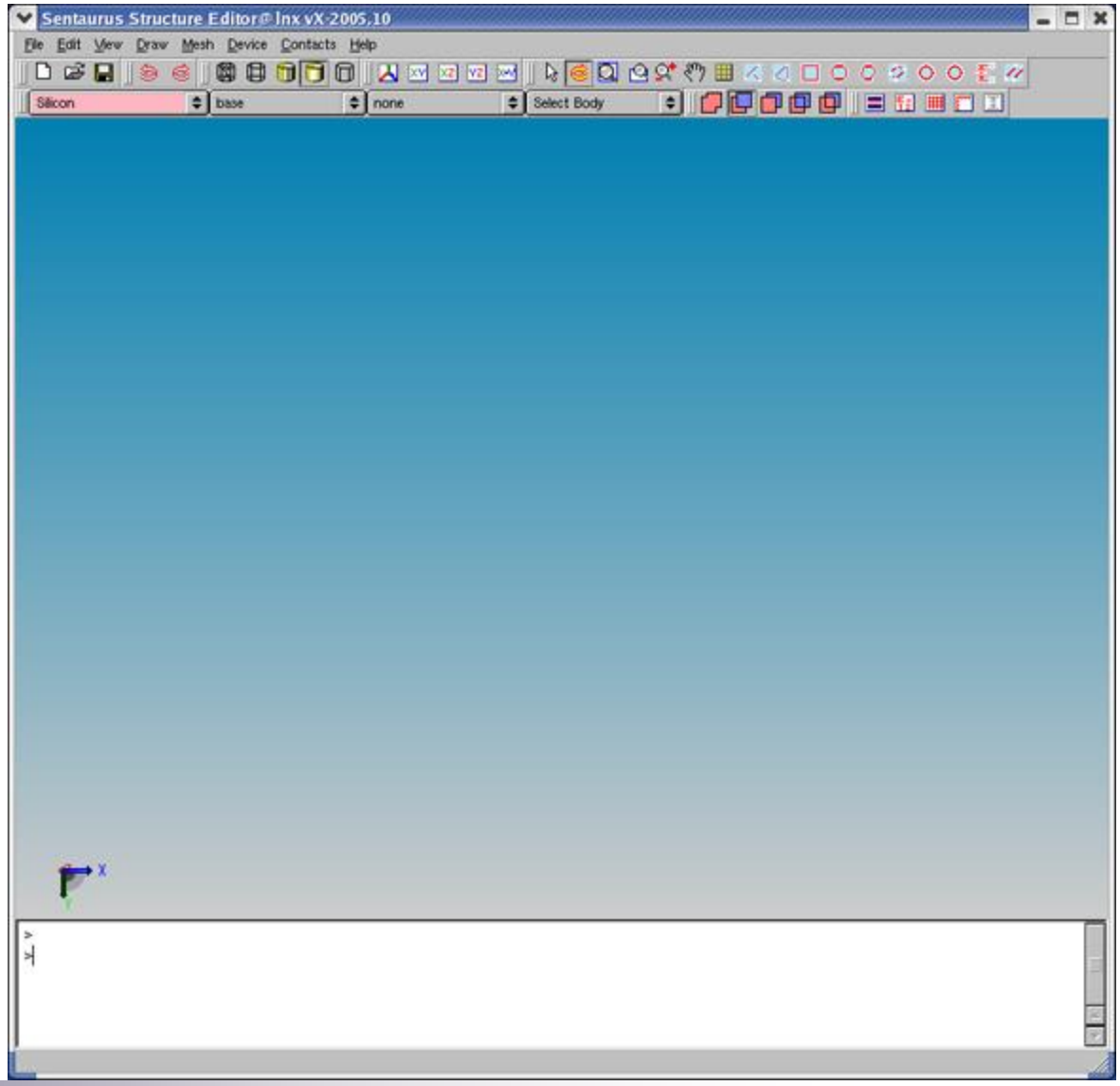
- Structure creation via primitives
- *Process emulation with PROCEM (3D only)*
- Contact definition
- Doping definition
- Mesh definition

© 2007 Synopsys, Inc. (110)

# Starting SDE in GUI Mode

- To start SDE, on the command line, enter:

```
sde
sde -2D
```




© 2007 Synopsys, Inc. (111)

# Creating Regions

- To create a rectangular region:
  - Draw > Create 2D Region > Rectangle, or click the corresponding toolbar button
  - Click and drag a rectangle
- Command line:

```
(sdegeo:create-rectangle
  (position 0 0 0)
  (position 1 1 0)
  "Silicon" "region_1"
)
```

© 2007 Synopsys, Inc. (112)

# Creating Polygons

- To create a polygonal region:
  - Draw > Create 2D Regions > Polygon, or click the corresponding toolbar button.
  - Click a location in the view window where the first vertex of the polygon is to be placed.
  - Repeat the previous steps for all the other vertices except the last vertex of the polygon.
  - For the final vertex, click the middle mouse button to place it.

```
(sdegeo:create-polygon
  (list  (position 0 0 0)
   (position 0 1 0)
   (position 1 2 0)
   (position 1 0 0) )
  "Silicon" "region_2"
)
```

© 2007 Synopsys, Inc. (113)

# Selecting Materials

- All materials used by Sentaurus Structure Editor are accessible from the Material list.
- To set the material to be used for new objects:
  - Click the Material list and select the required material, for example, Silicon.


© 2007 Synopsys, Inc. (114)

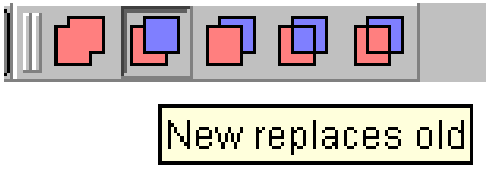
# Exact Coordinates Mode

- In Sentaurus Structure Editor, geometric objects can be drawn manually. However, for most applications, it is necessary to specify explicitly the coordinates of the object
- To do this, it is necessary to activate the Exact Coordinates mode.
  - Draw > Exact Coordinates.
- When the mode is active, all subsequent command operations that involve the placement of any object will display a dialog box in which exact coordinate values can be entered for the object being edited.

© 2007 Synopsys, Inc. (115)

# Selecting the Default Boolean Expression

- When the Sentaurus Structure Editor GUI is used to build a device with multiple regions, the later-added regions may intersect existing regions.
- A predefined scheme is required to resolve the overlapping region.
- Draw > Overlap Behavior > New Replaces Old, or click the corresponding toolbar button.




© 2007 Synopsys, Inc. (116)

# Types of Boolean Expressions

- Merge

```
(sdegeo:set-default-boolean "AB")
```

- New Replaces Old

```
(sdegeo:set-default-boolean "ABA")
```

- Old Replaces New

```
(sdegeo:set-default-boolean "BAB")
```

- New Overlaps Old

```
(sdegeo:set-default-boolean "ABiA")
```

- Old Overlaps New

```
(sdegeo:set-default-boolean "ABiB")
```

© 2007 Synopsys, Inc. (117)

# Rounding Edges

- To round corners of regions:
  - Click the Selection Level list and select Select Vertex.
  - Click the Aperture Select button in the toolbar.
  - Click the upper-left corner of the spacer to highlight the vertex.
  - Edit > Edit 2D > Fillet. Enter fillet radius.
  - The selected corner is now rounded.

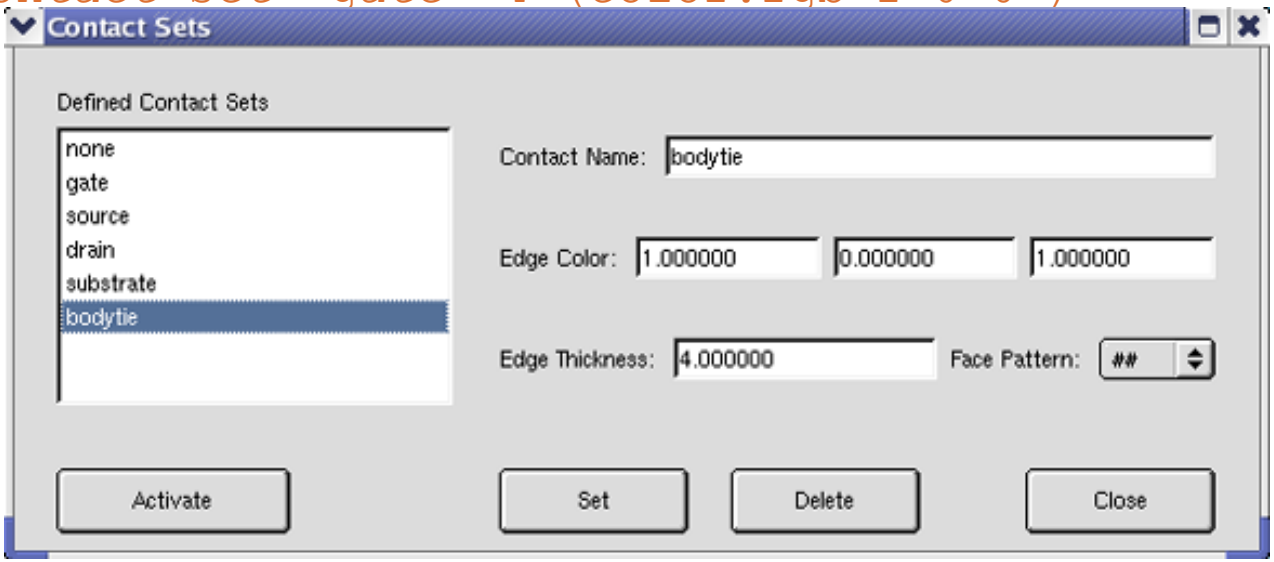
```
(sdegeo:fillet-2d
 (list (find-vertex-id (position 0 1 0))
 )
 0.2
)
```

© 2007 Synopsys, Inc. (118)

# Defining Contacts

- To define a contact:
  - Contacts > Contact Sets.
  - Define the properties of the contact. Enter the name of the contact in the Contact Name field.
  - Click Set to add the contact to the Defined Contact Sets list. Multiple contacts can be defined in one session.
  - Click Close.

```
(sdegeo:define-contact-set "gate" 4 (color:rgb 1 0 0 )
  "##")
```

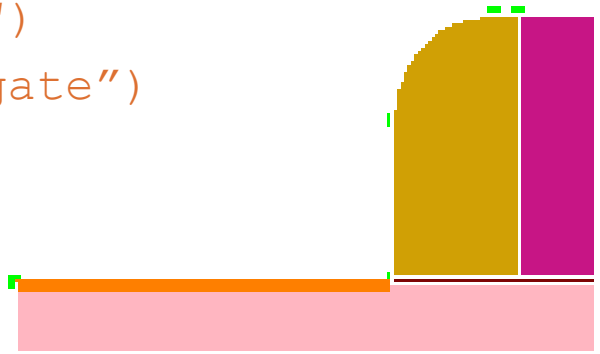



© 2007 Synopsys, Inc. (119)

# Associating Contacts with Edges

- Contacts become effective only after they are associated with an edge of the device structure:
  - Activate a contact by selecting it from the Contact list.
  - Set the selection level to Select Edge in the Selection Level list.
  - Click the Aperture Select button.
  - Click the edge of the structure where the contact, for example, source, will be defined. The selected edge is now highlighted.
  - Contacts > Set Edge(s). This defines the selected contact at the highlighted edge. The edge is now characterized by the color and line styles previously set for the contact.

```
(sdegeo:set-current-contact-set "gate")
(sdegeo:set-contact-edges edge-list “gate”)
```

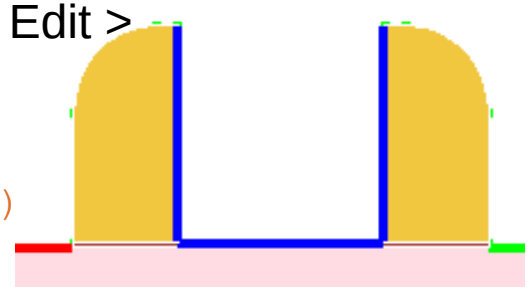


© 2007 Synopsys, Inc. (120)

# Defining a Region as a Contact

- To turn an entire region into a contact:
  - Select the contact from the Contact list.
  - Set the selection level to Select Body in the Selection Level list.
  - Click the Aperture Select icon on the toolbar.
  - Click the region where the contact is to be defined, for example, the poly gate region.
  - Contacts > Set Region Boundary Edges. This converts the selected region into a contact. The edges of the region have changed to the color and style of the selected contact.
  - To remove the gate region, select the region and Edit > Remove > Region

```
(sdegeo:set-current-contact-set "gate")
(sdegeo:set-contact-boundary-edges entity-list)
```



© 2007 Synopsys, Inc. (121)

# Defining Constant Doping

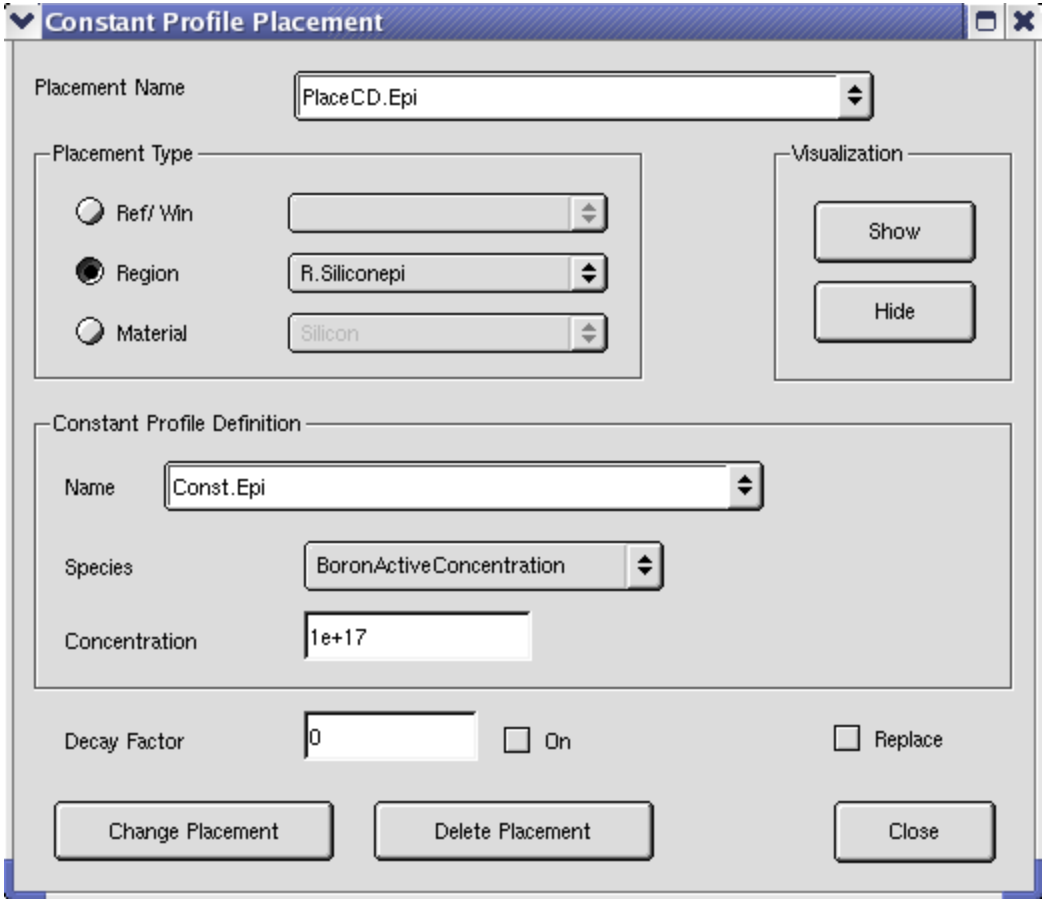
- To define a uniform doping profile in a region:
  - Device > Constant Profile Placement.
  - Type a name in the Placement Name field.
  - In the Placement Type group box, select Region, and select the region.

```
(sdedr:define-constant-profile "CProfileDef_1"
  "BoronActiveConcentration" 1e15)
(sdedr:define-constant-profile-region "CProfile_1"
  "CProfileDef_1" "region_1")
```

© 2007 Synopsys, Inc. (122)

# Constant Profile Placement

- In the Constant Profile Definition group box, type a name in the Name field.
- Select species from the Species list.
- Enter value in the Concentration field.
- Click Add Placement.
- Click Close.




© 2007 Synopsys, Inc. (123)

# Defining Analytical Profiles

- The placement of an analytic profile is performed in two steps:
  - Define the baseline. The baseline is used to determine the lateral extent of the profile and can also serve as the reference point for the depth of the peak position.
    - Mesh > Define Ref/Eval Window > Line.
    - In the view window, click the first point of the baseline.
    - Click again to define the end point of the baseline.
    - In the displayed dialog box, enter the name for the baseline and click OK

```
(sdedr:define-refinement-window "Myline" "Line"
  (position 0 0 0) (position 1 0 0)
)
```

© 2007 Synopsys, Inc. (124)

# Analytical Profile Placement

- Define the shape of the profile itself.
  - Device > Analytic Profile Placement.
  - Type the Placement Name field.
  - Select the baseline from the Ref/Win list.
  - In the Profile Definition group box, enter the Name field.
  - Select the Gaussian option from the Profile Type list and species from the Species list.
  - In the Concentration group box, fill in Peak Concentration, Peak Position, Junction and Depth.
  - In the Lateral Diffusion group box, enter the Factor field.
  - Click Add Placement.

© 2007 Synopsys, Inc. (125)

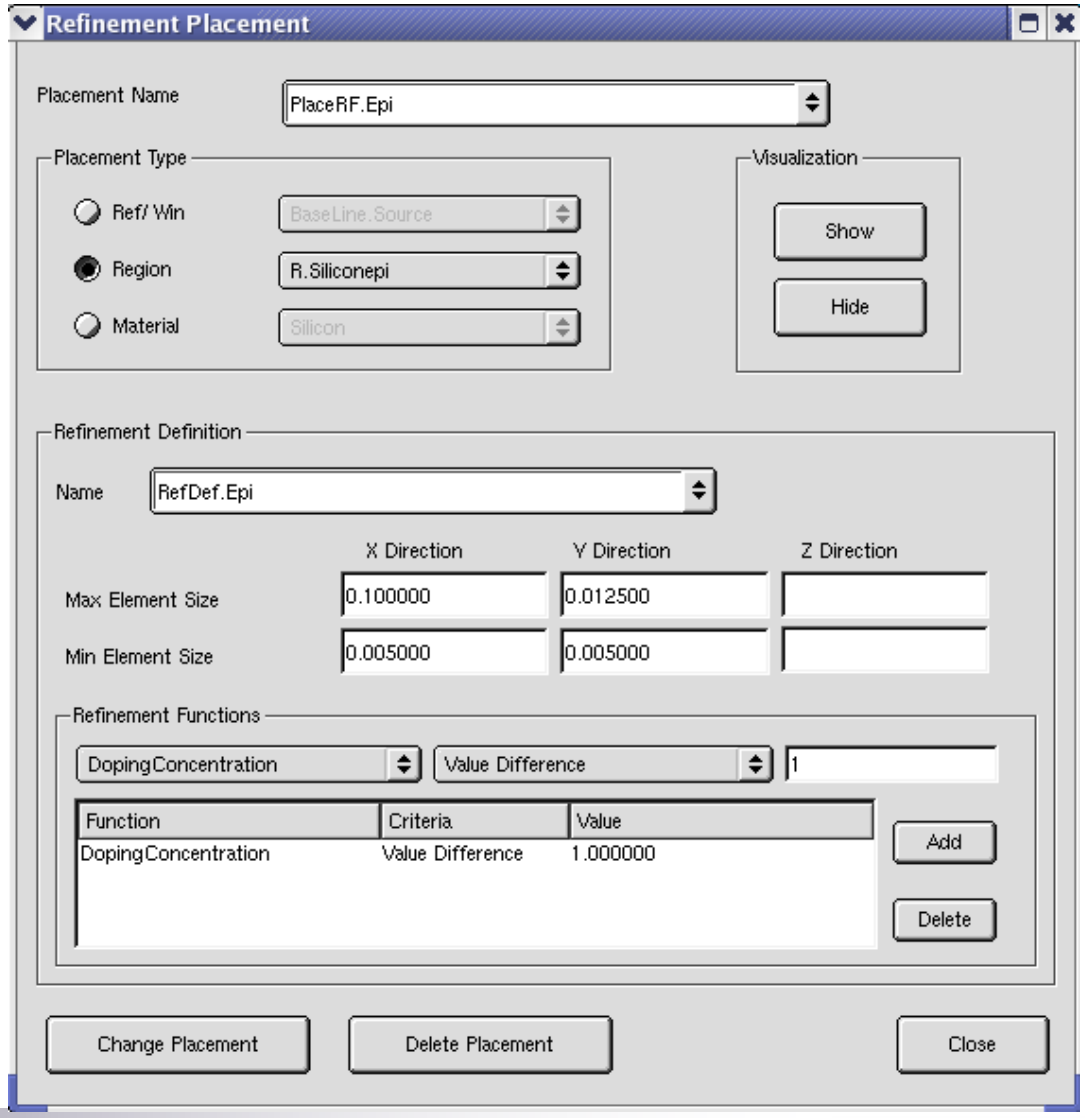
# Analytical Profile Placement

```
(sedr:define-gaussian-profile
 "AProfileDef_1" "BoronActiveConcentration"
 "PeakPos" 0
 "PeakVal" 1e20
 "ValueAtDepth" 1e15
 "Depth" 0.1
 "Erf" "Factor" 0.8
)
(sedr:define-analytical-profile-placement
 "AProfilePlacement_1"
 "AProfileDef_1"
 "Myline"
 "Both" "NoReplace" "Eval"
)
```

© 2007 Synopsys, Inc. (126)

# Defining Mesh Strategies in Regions

- To define a meshing strategy in a device region:
- Mesh > Refinement Placement.
- The Refinement Placement dialog box is displayed. It includes two input files:
- Refinement Definition group box includes fields for defining the meshing strategy and
- Placement Type group box deals with the placement.




© 2007 Synopsys, Inc. (127)

# Doping Refinement Placement

```
(sedr:define-refinement-size
 "RDef_1"
 0.1 0.1 0.1
 0.01 0.01 0.01
)
(sedr:define-refinement-function
 "RDef_1"
 "DopingConcentration"
 "MaxTransDiff" 1
)
(sedr:define-refinement-region
 "RPlacement_1"
 "RDef_1"
 "region_3"
)
```

© 2007 Synopsys, Inc. (128)

# Defining Refinement Windows

- In SDE, a meshing strategy can be restricted to only selected areas.
- The areas (refinement windows) need to be defined first before a meshing strategy can be applied.
- Mesh > Define Ref/Eval Window > Rectangle.

© 2007 Synopsys, Inc. (129)

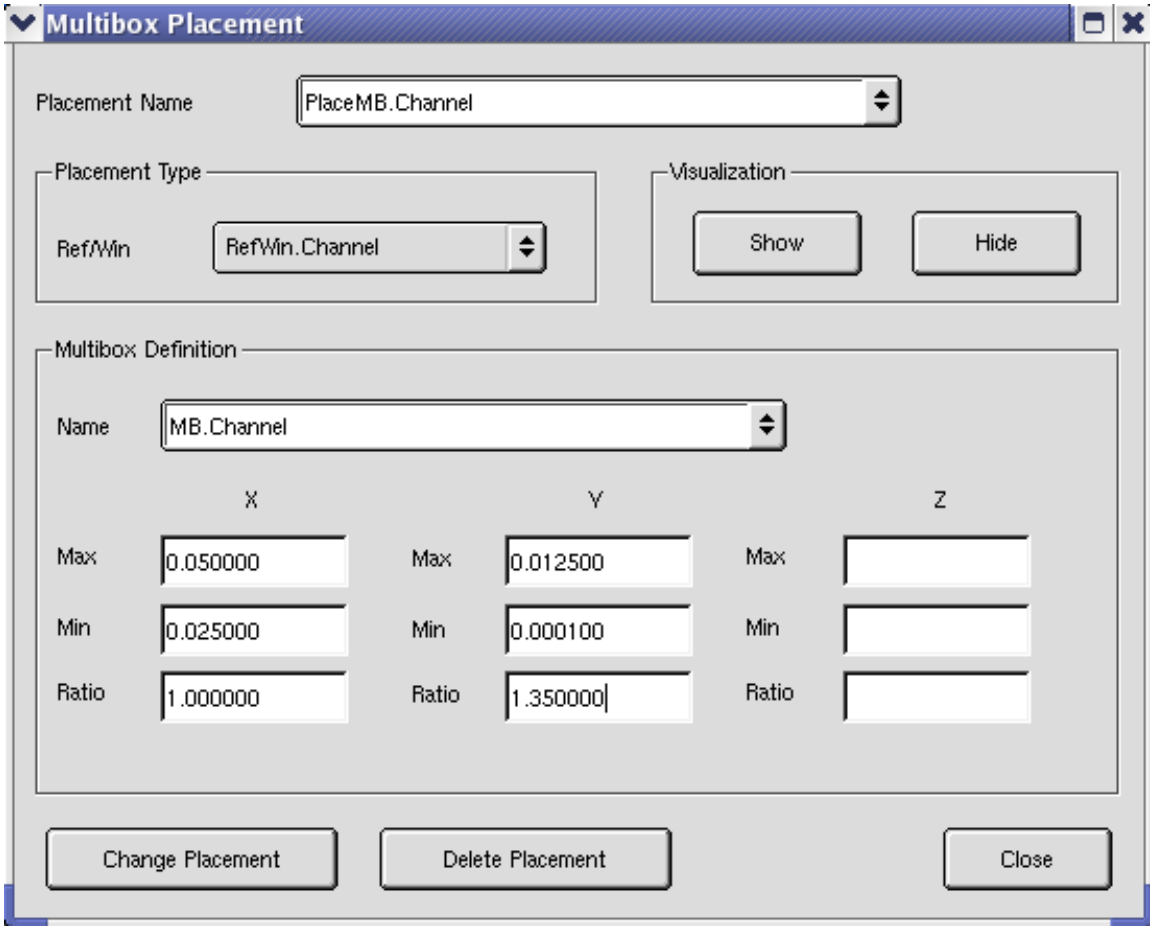
# Refinement windows

```
(sdedr:define-refinement-window
 "Rwin1" "Rectangle"
 (position 0 0 0)  (position 1 1 0)
)
(sdedr:define-refinement-size
 "RDef_1"
 0.1 0.1 0.1
 0.01 0.01 0.01
)
(sdedr:define-refinement-placement
 "RPlacement_1" "RDef_1" "Rwin1"
)
```

© 2007 Synopsys, Inc. (130)

# Defining a Multibox Mesh Strategy

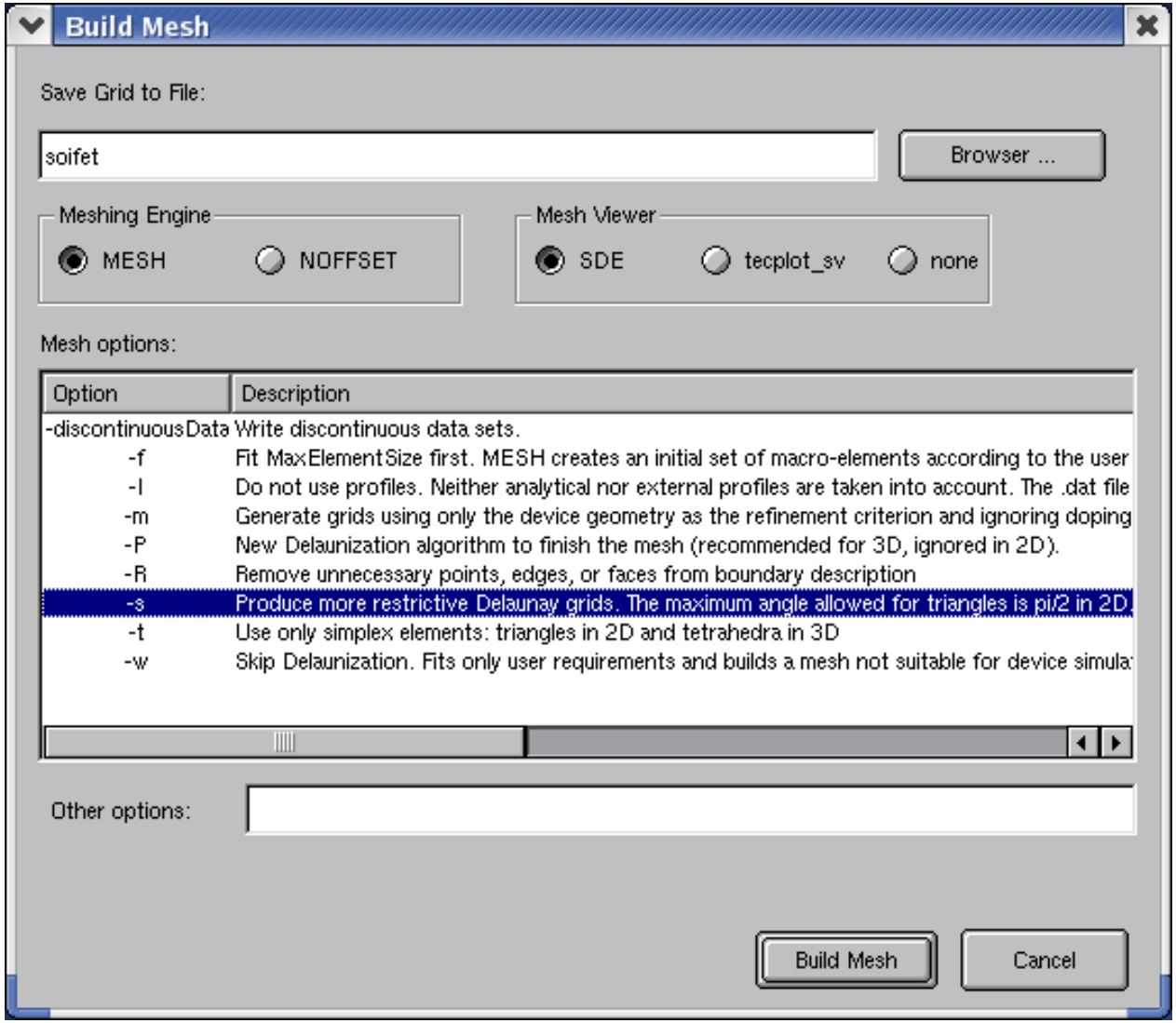
- Some applications require meshing strategies in which meshing line densities are gradually changed.
- SDE supports another type of meshing strategy called multibox refinement strategy:
- Mesh > Multibox Placement.




© 2007 Synopsys, Inc. (131)

# Meshing the Device Structure

- To call the meshing engine:
- Mesh > Build Mesh.




© 2007 Synopsys, Inc. (132)

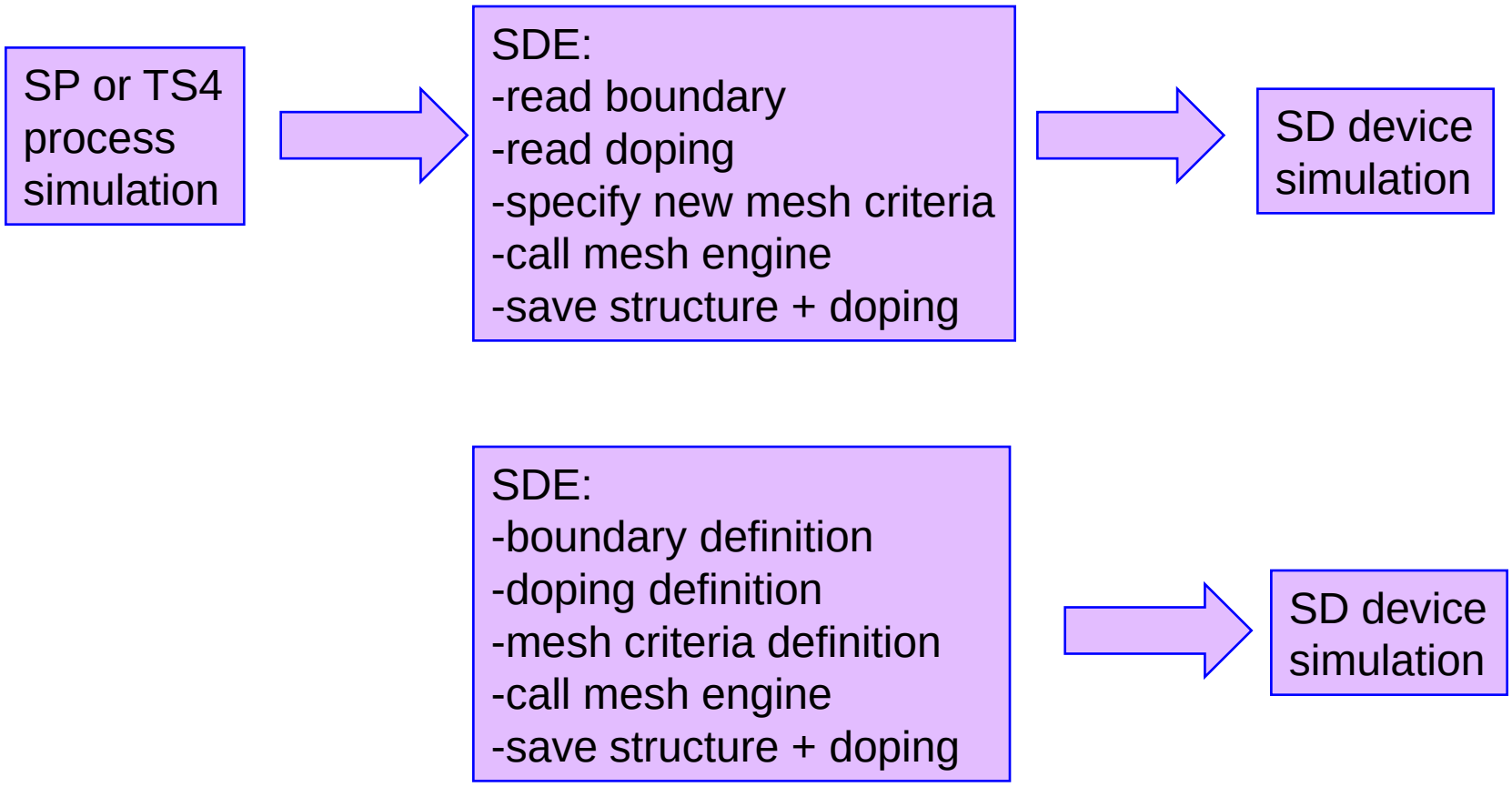
# Basic Flow
# with result of process simulation

- The SSE flow is as follows:
  1. Read boundary
  2. Contact definition
  3. Ref/Eval Window definition
  4. Read profile
  5. Mesh definition
  6. Build Mesh
- These operations and commands will be introduced from next slide.

© 2007 Synopsys, Inc. (133)

# Meshing for Device Simulation

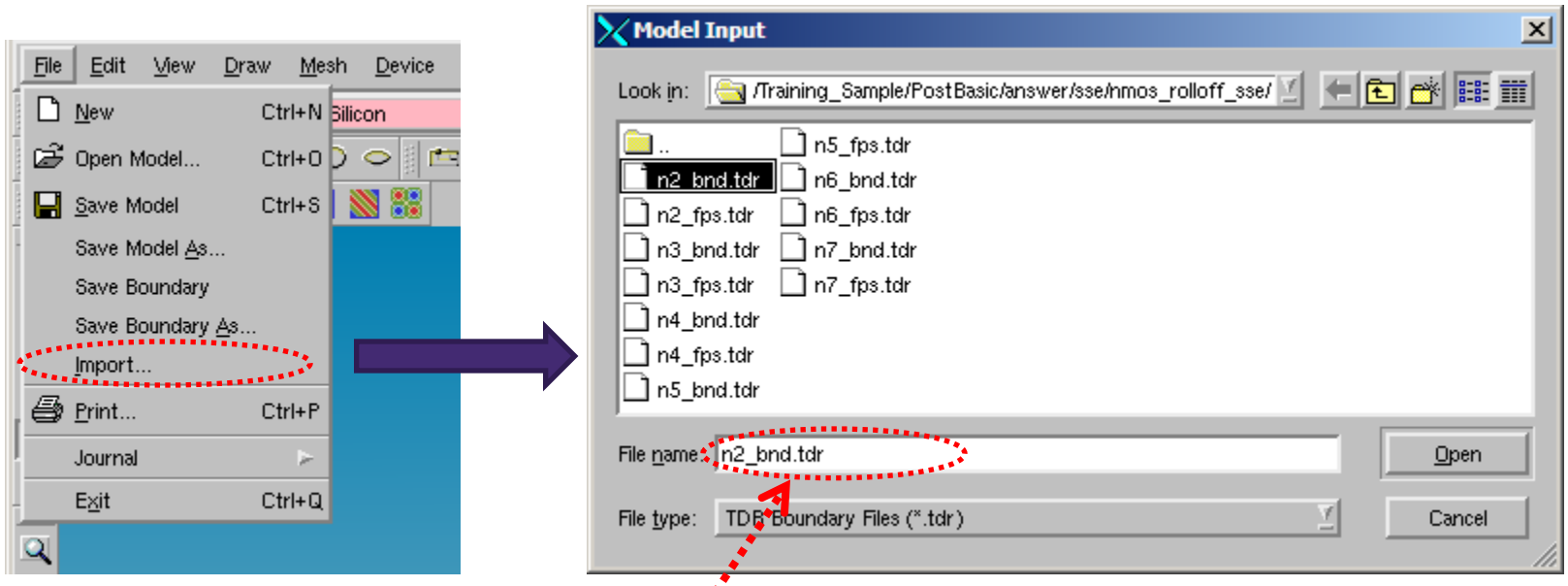
- possible flows:




© 2007 Synopsys, Inc. (134)

# Read boundary

- Read the boundary to be modified.

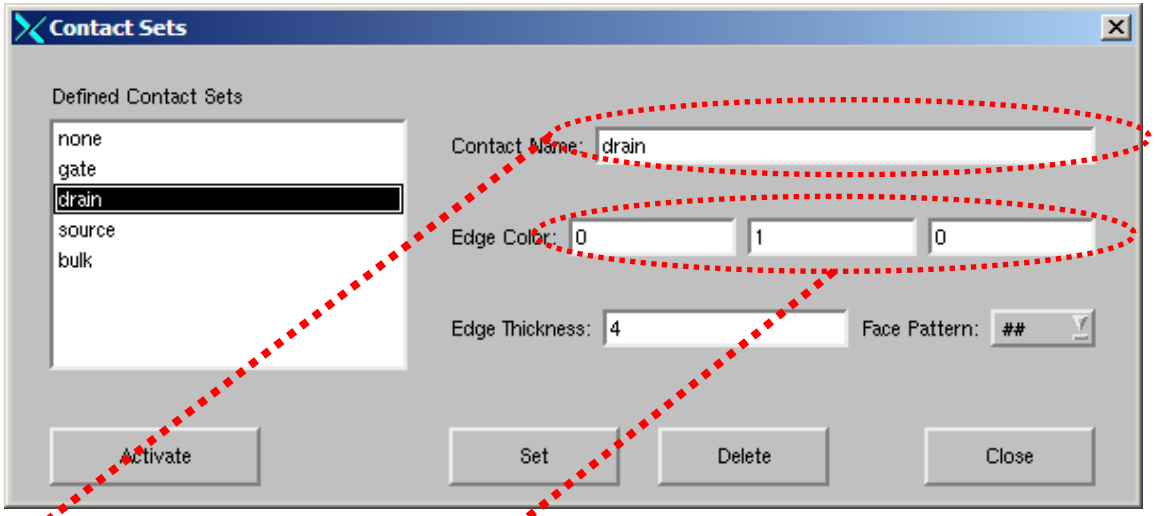



```
(sdeio:read-tdr-bnd "n2_bnd.tdr")
```

© 2007 Synopsys, Inc. (135)

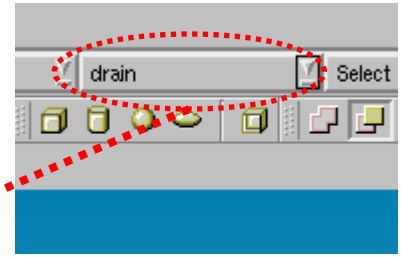
# Contact definition (1/2)

- Define contacts for the device simulation.




1. Define contact name.

```
(sdegeo:define-contact-set "drain" 4.0 (color:rgb 0.0 1.0 0.0 ) "##" )
```



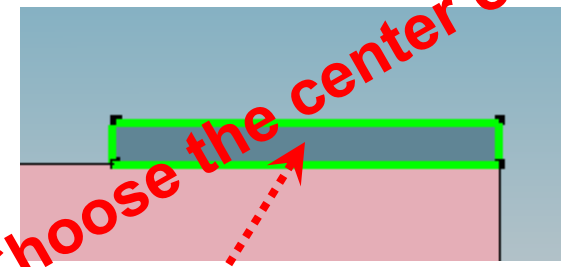

2. Select contact.

```
(sdegeo:set-current-contact-set "drain")
```

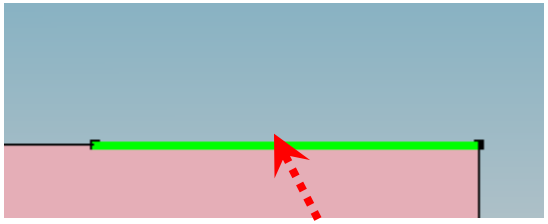
© 2007 Synopsys, Inc. (136)

# Contact definitions (2/2)

3. Select contact region (metal) and set region boundary edge from contacts menu.




```
(sdegeo:set-contact-boundary-edges
       (list(car(find-body-id(position 0.29 -0.002 0.0)))) "drain" )
```



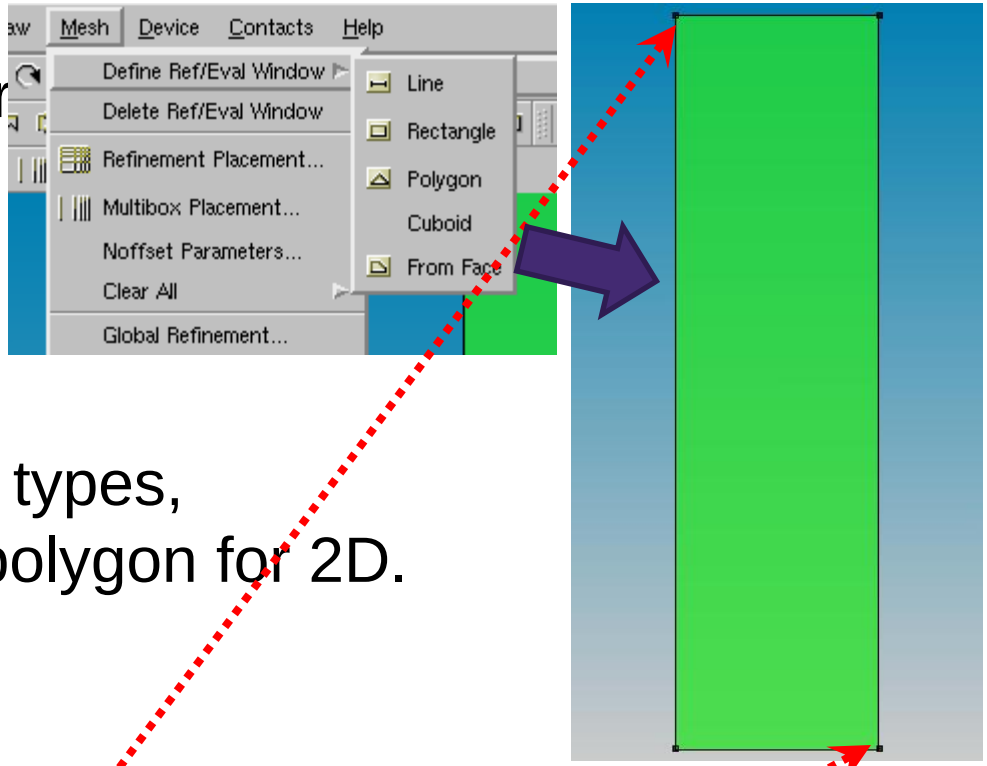
4. Delete contact region (metal).

```
(sdegeo:delete-region
                  (list(car(find-body-id (position 0.29 -0.002 0)))))
```

© 2007 Synopsys, Inc. (137)

# Ref/Eval window definition

- Define the region for mesh and profile.



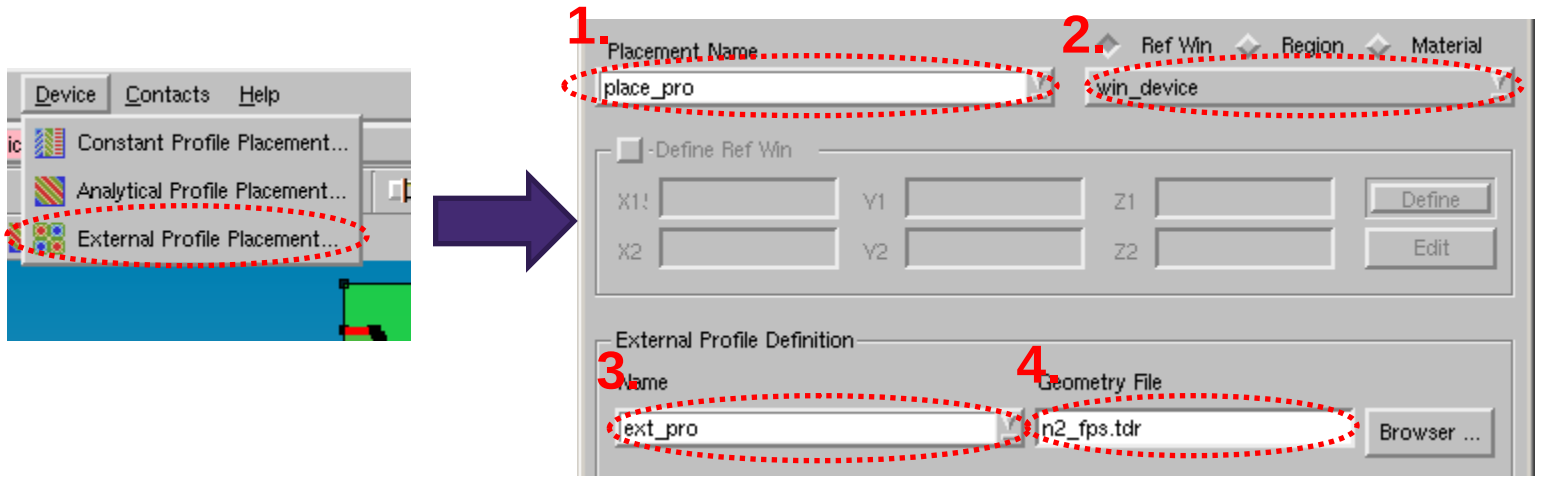

- There are 3 window types, rectangle, line and polygon for 2D.

```
(sdedr:define-refeval-window "win_device" "Rectangle"
        (position 0 -0.2 0) (position 0.3325 1 0) )
```

© 2007 Synopsys, Inc. (138)

# Read profile

- Read the profile data generated by process simulator.

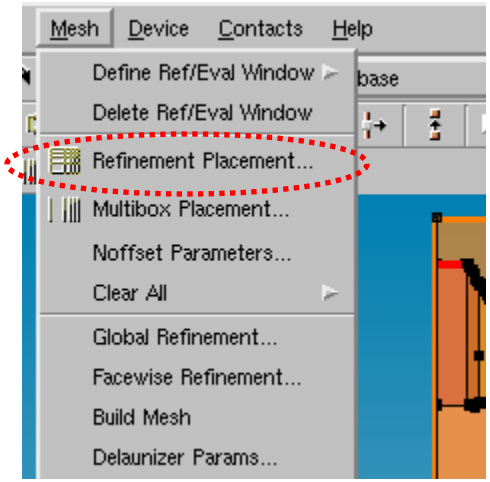


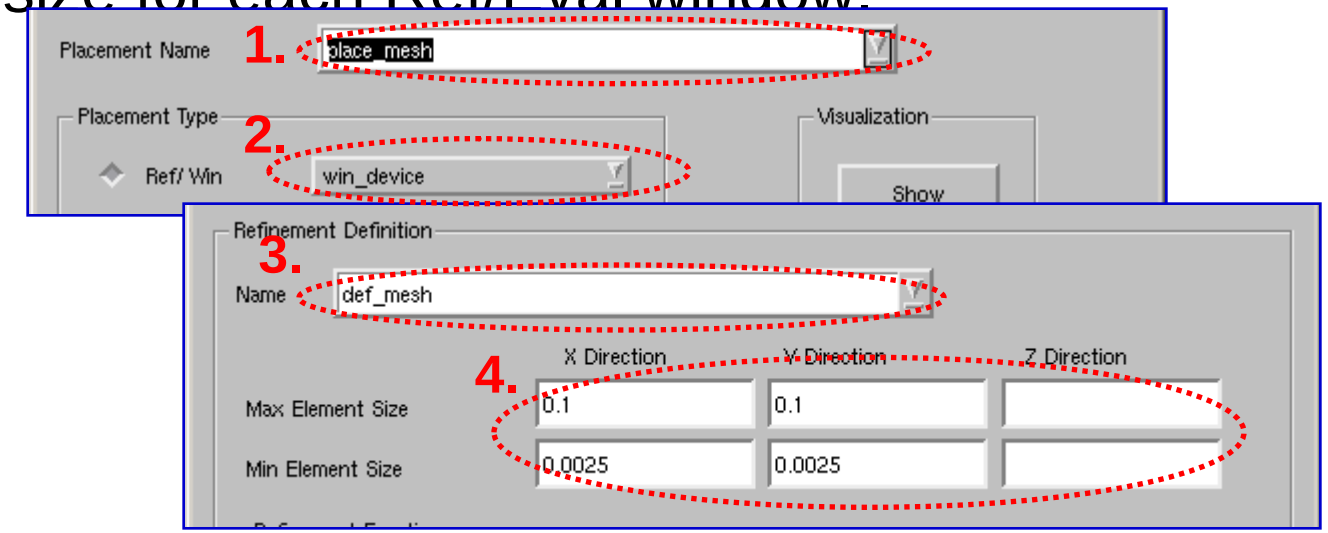

```
(sdedr:define-submesh "ext_pro" "n2_fps.tdr")
                       3.        4.

(sdedr:define-submesh-placement
  "place_pro" "ext_pro" "win_device"
      1.        3.          2.  "DecayLength" 0 "NoReplace")
```

© 2007 Synopsys, Inc. (139)

# Mesh definition (1/2)

- Define mesh size for each Ref/Eval window.

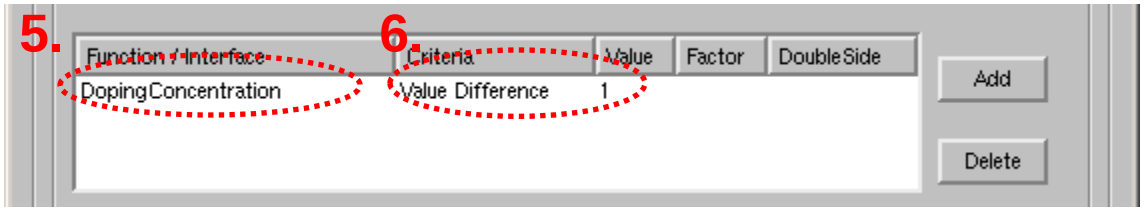



```
(sedr:define-refinement-size
              "def_mesh" 0.1 0.1 0.0025 0.0025)
                    3.          4.
(sedr:define-refinement-placement
                        "place_mesh" "def_mesh" "win_device")
                              1.         3.          2.
```

© 2007 Synopsys, Inc. (140)

# Mesh definition (2/2)



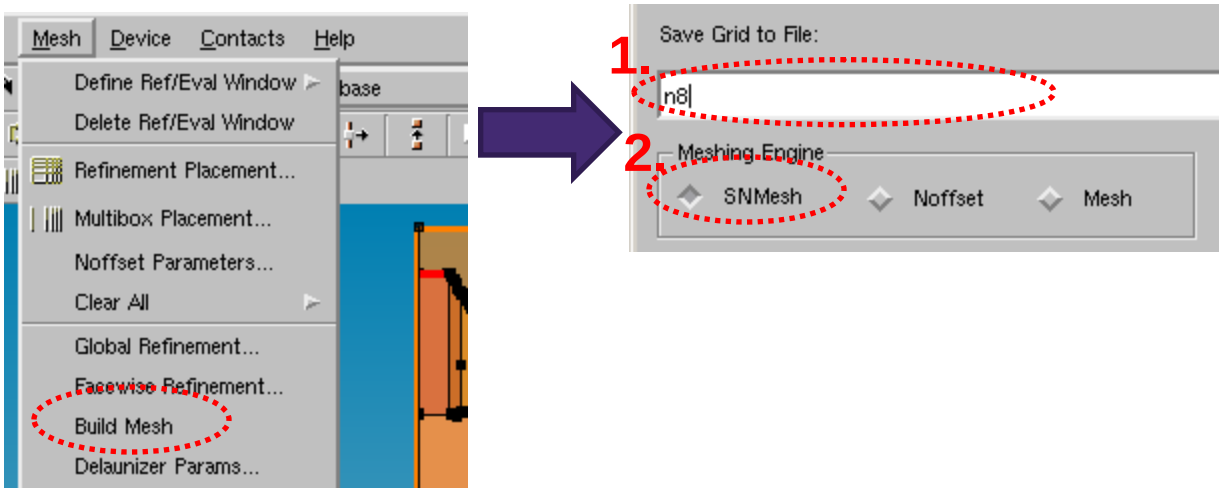

```
(sedr:define-refinement-function
          "def_mesh" "DopingConcentration" "MaxTransDiff" 1)
```

3. 5. 6.

© 2007 Synopsys, Inc. (141)

# Build Mesh

- Build mesh for device simulation.



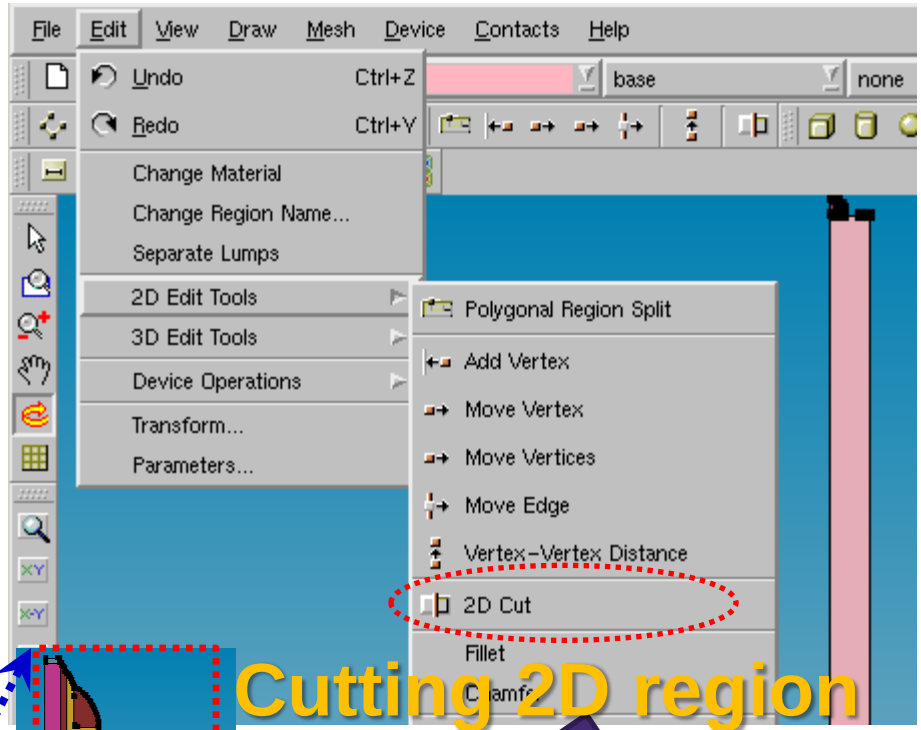

```
(sde:build-mesh "snmesh" "-a -c boxmethod" "n8")
```

2. 1.

© 2007 Synopsys, Inc. (142)

# Appendix
# ~Cut structure~

- Cut the unnecessary region for device simulation




```
(sdegeo:2d-cut
  (position -0.01 -0.2 0)
  (position 0.34 1 0) )
```

© 2007 Synopsys, Inc. (143)

# Meshing with SDE + SMesh

- example: project SimpleMesh
  - refinement criteria for a region

```
<structure, doping …>

(sdedr:define-refinement-size "Sub1.Def"
     0.2    0.2   0.05
     0.05   0.1  0.05
     )
(sdedr:define-refinement-region
        "Sub1.Pl"
        "Sub1.Def“
        "Substrate")
```

← a name for this criterion
← maxx maxy maxz
←minx miny minz

←placement_name
←refinement_name,
← region_name

© 2007 Synopsys, Inc. (144)

# Meshing with SDE + SMesh

- refinement in a box

```
(sedr:define-refinement-window
     "Sub2.Win" "Cuboid"
      (position xboxr 0.15 0.0)
      (position xbox 0.0 zbox) )
(sedr:define-refinement-size "Sub2.Def"
     0.05    0.1   0.025
     0.02    0.05  0.025
     )
(sedr:define-refinement-placement
               "Sub2.Pl“
               "Sub2.Def"
               "Sub2.Win")
```

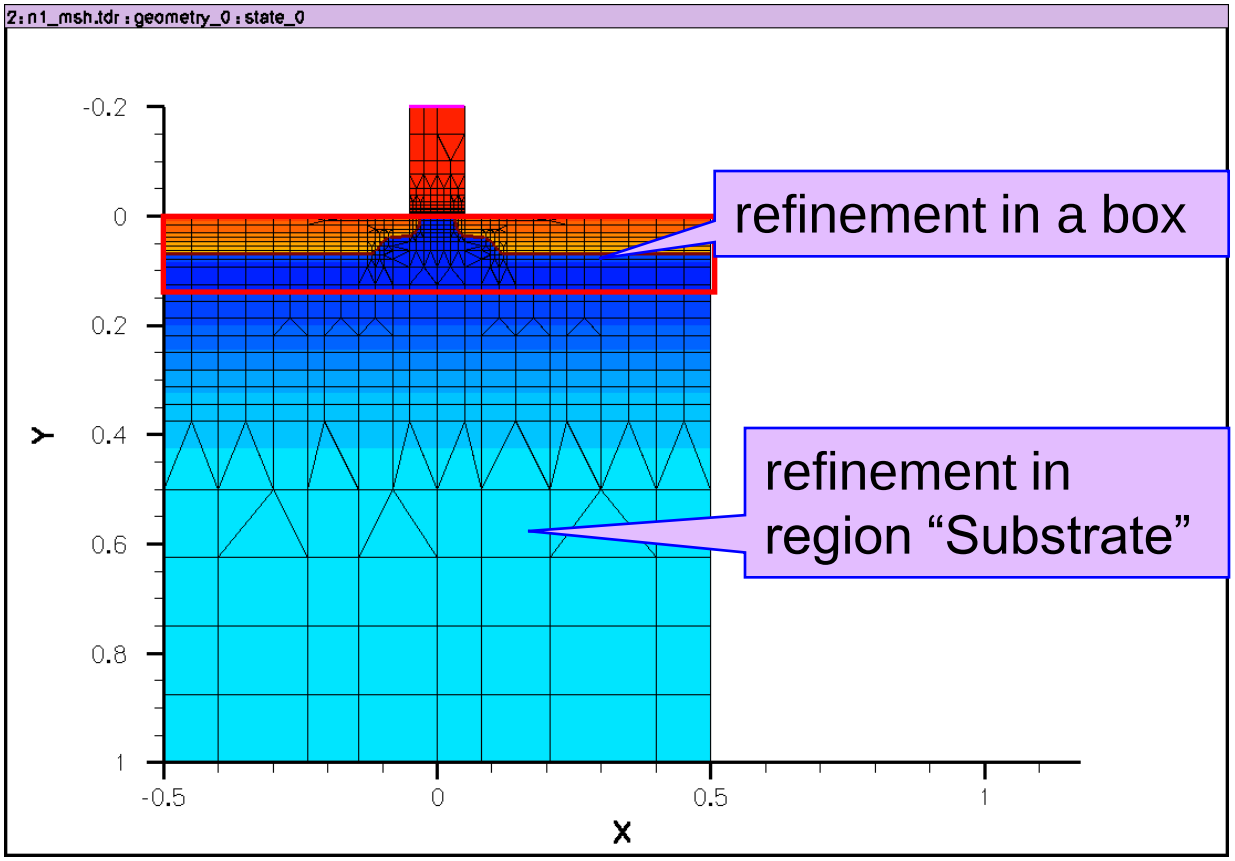
← a name for this criterion
← first corner
← opposite corner

← maxx maxy maxz
← minx miny minz

← placement_name
← refinement_name,
← region_name

© 2007 Synopsys, Inc. (145)

# Meshing with SDE + SMesh




© 2007 Synopsys, Inc. (146)

# Meshing with SDE + SMesh

- graded mesh (“multibox method” )

```
(sedr:define-refinement-window
      "Channel.Win" "Cuboid"
       (position GateLr 0.05 0.0)
       (position GateL 0.0 zbox) )
(sedr:define-multibox-size "Channel.Def"
      0.02    0.01    0.025
      0.02    0.001   0.025
      1.0     2.0     1.0)
(sedr:define-multibox-placement
         "Channel.Pl“
         "Channel.Def“
         "Channel.Win")
```

← graded mesh in y-direction, spacing from 0.01 to 0.001

© 2007 Synopsys, Inc. (147)

# Meshing with SDE + SMesh

- refinement on a field (doping)

```
(sedr:define-refinement-size "Doping.Ref"
      0.5      0.5     0.2
      0.004    0.004 0.2)
(sedr:define-refinement-function
"Doping.Ref" "DopingConcentration"
"MaxTransDiff" 1)
(sedr:define-refinement-region
       "Doping.Pl“
       "Doping.Ref"
       "Substrate")
```

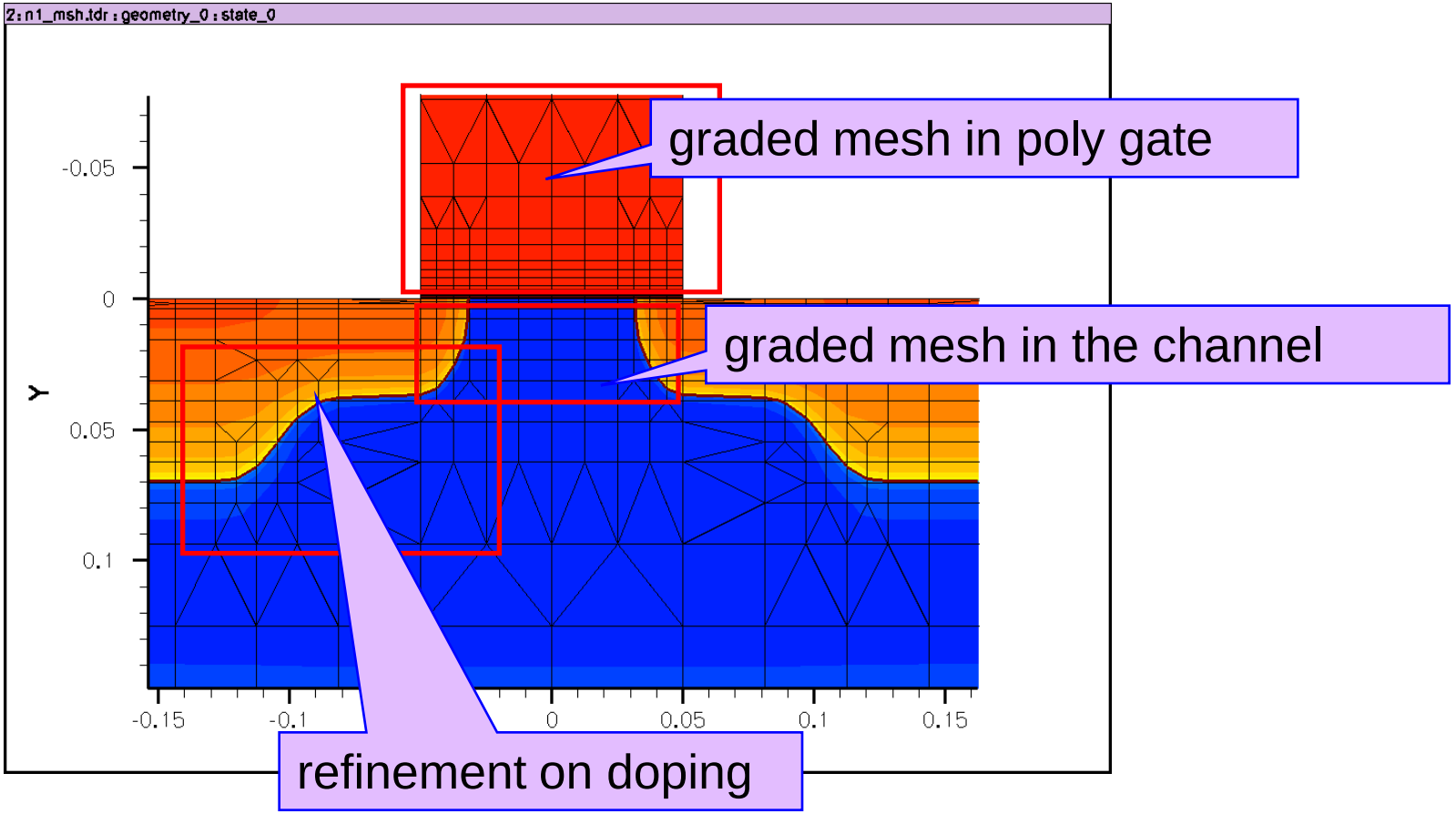
← min/max desired spacing in x,y,z

← apply criteria to field “DopingConcentration”
← criterion: maximum difference

← apply criteria in region “Substrate”

© 2007 Synopsys, Inc. (148)

# Meshing with SDE + SMesh




© 2007 Synopsys, Inc. (149)

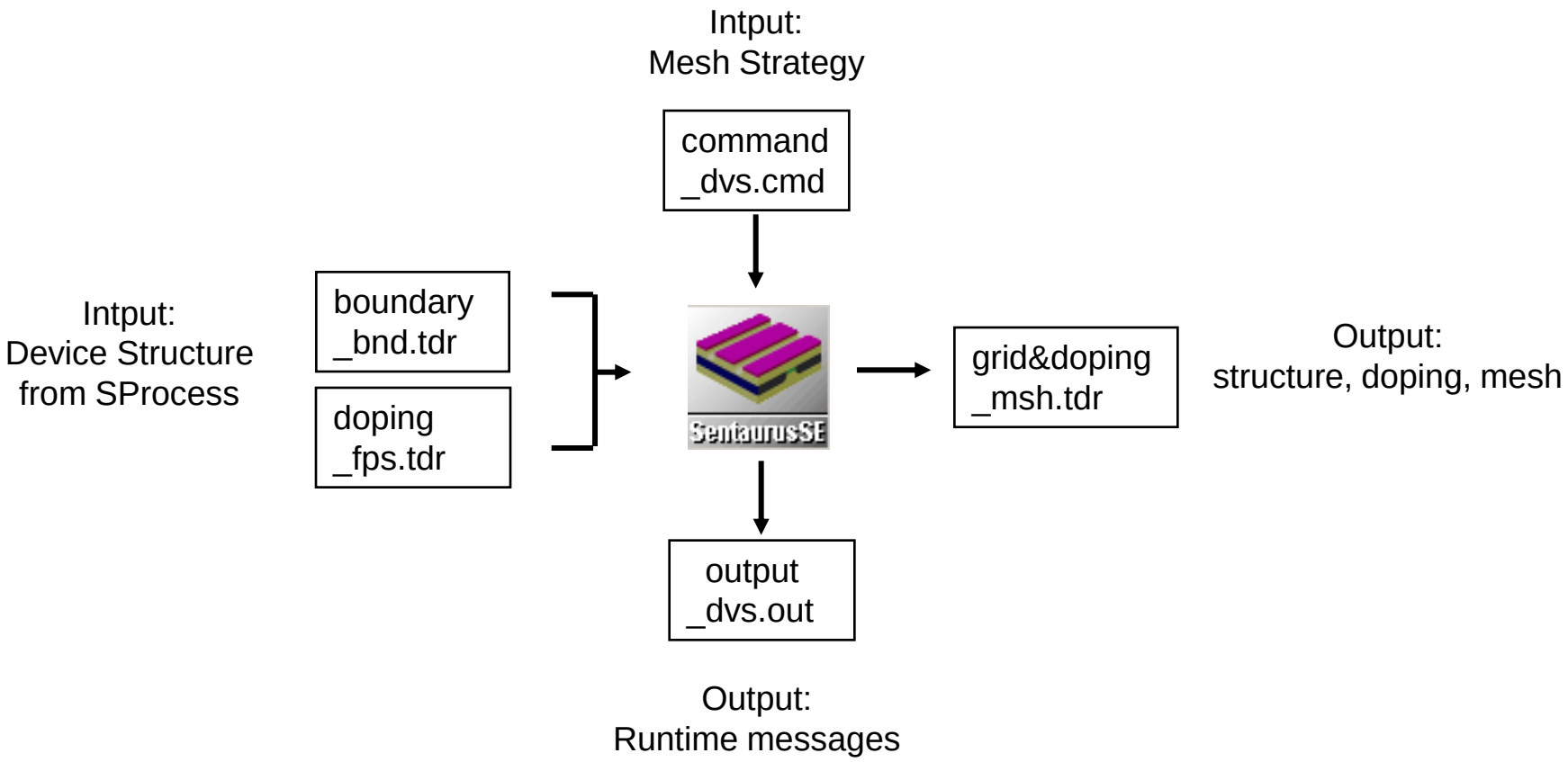
# Meshing with SDE + SMesh

- finally: save boundary, write input file for “smesh”, call the mesh engine

```
(sdeio:save-tdr-bnd (get-body-list) "n@node@_bnd.tdr")

(sdedr:write-cmd-file "n@node@_msh.cmd")

(system:command “smesh n@node@_msh")
```

© 2007 Synopsys, Inc. (150)

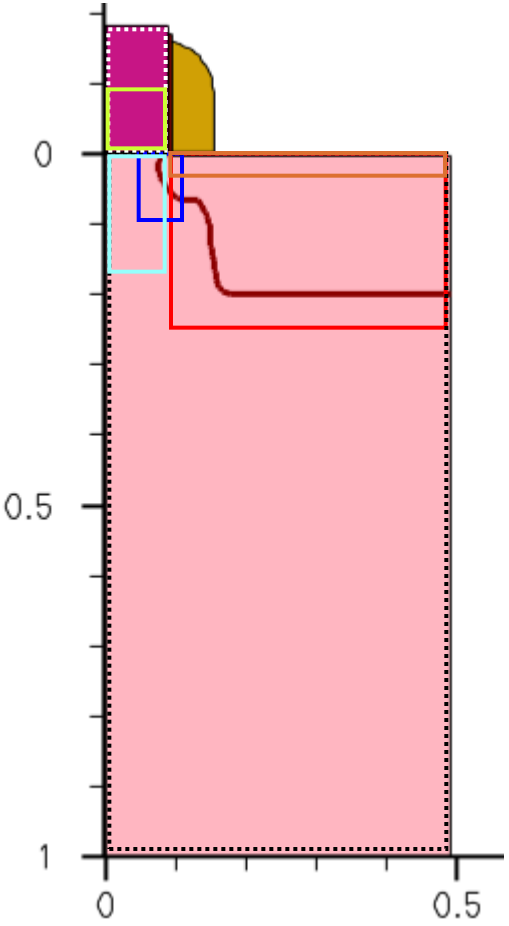
# Flow of Input and Output in SDE




© 2007 Synopsys, Inc. (151)

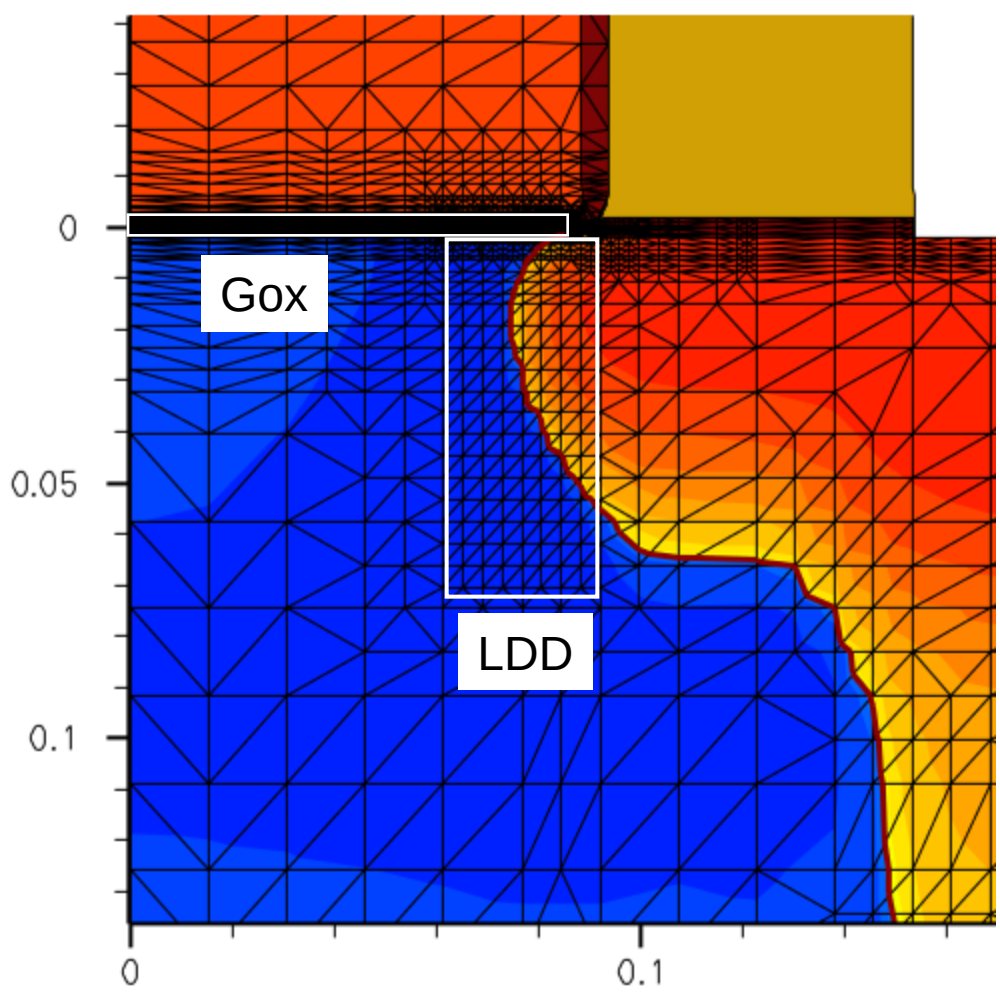
# MOSFET Mesh Setup Strategy (for SDevice Simulation)

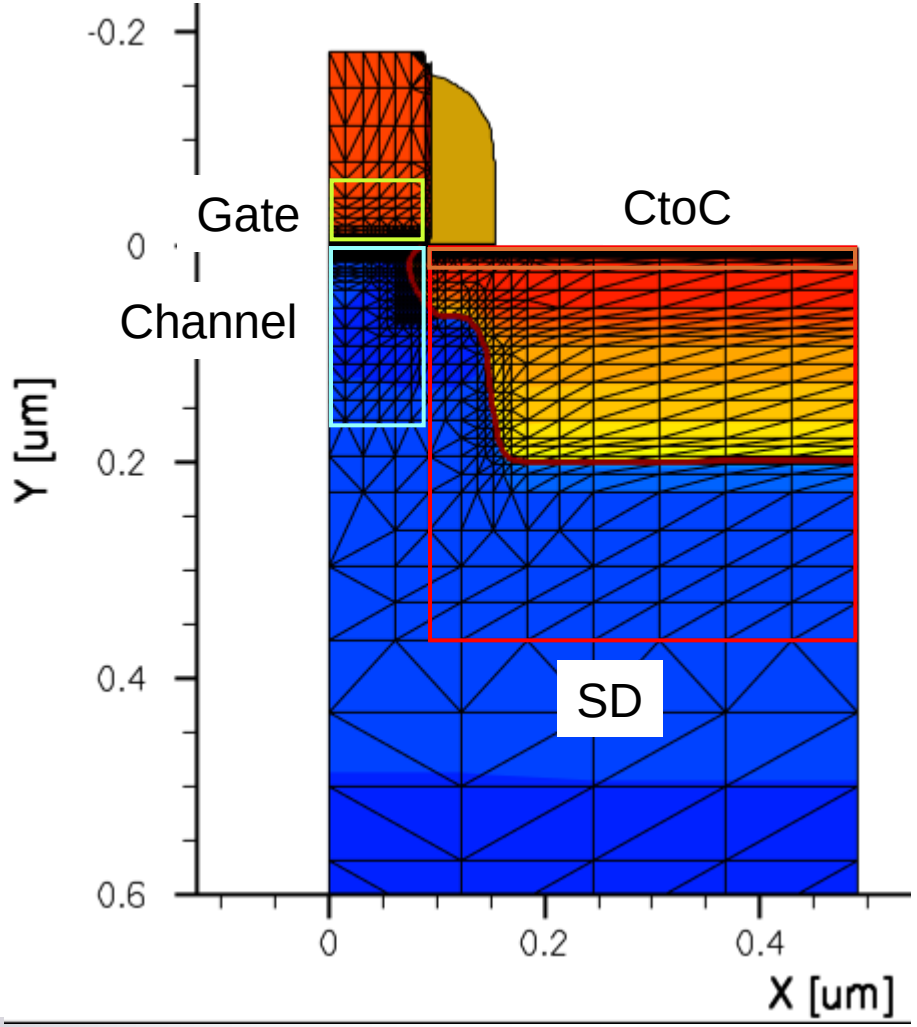
- Load boundary from previous SProcess output
- Define initial mesh for Silicon and PolySilicon materials
- Refine Source/Drain junction mesh
- Refine LDD junction mesh
- Refine silicon surface region between channel to S/D contact
- Define gate oxide mesh
- Refine channel mesh
- Refine PolySilicon mesh
- Load doping profile from previous SProcess output
- Define PolyGate (PolySilicon) doping




© 2007 Synopsys, Inc. (152)

# MOSFET Mesh Setup Result (for SDevice Simulation)




© 2007 Synopsys, Inc. (153)

SYNOPSYS®
Predictable Success

# Interactive Mode

- Start Structure Editor => type “sde”
- Journal on => record the command line of each step
- Import boundary file from n4_bnd.tdr
- turn off “auto region name”, turn on “exact coordinates”
- Step by step setup mesh by GUI
  - Silicon => (0.1, 0.1) / (0.05, 0.1)
  - Poly => (0.02, 0.05) / (0.01, 0.025)
  - SD* => (0.09, 0.0) / (0.49, 0.3) => (0.1, 0.03) / (0.006, 0.006)
  - LDD => (0.05, 0.0) / (0.09, 0.05) => (0.006, 0.006) / (0.005, 0.005)
  - CtoC => (0.09, -0.002) / (0.49, 0.006) => (99, 0.002) / (66, 0.001)
  - Gox => (0.0, -0.002681) / (0.09, 0.00065) => (99, 4e-4) / (66, 4e-4)
  - Channel => (0.0, 0.00065) / (0.09, 0.075) => (0.02, 0.05) / (0.01, 2e-4) / (-1.45, 1.45)
  - Gate => (0.0, -0.18) / (0.09, -0.002681) => (99, 0.04) / (66, 4e-4) / (0.0, -1.75)
- Load sub mesh (geometry and doping data) from n4_fps.tdr
- Define PolyGate doping type and concentration (Arsenic 6e19)
- Save
- Build mesh
- Journal off
- Check .tdr .jrl

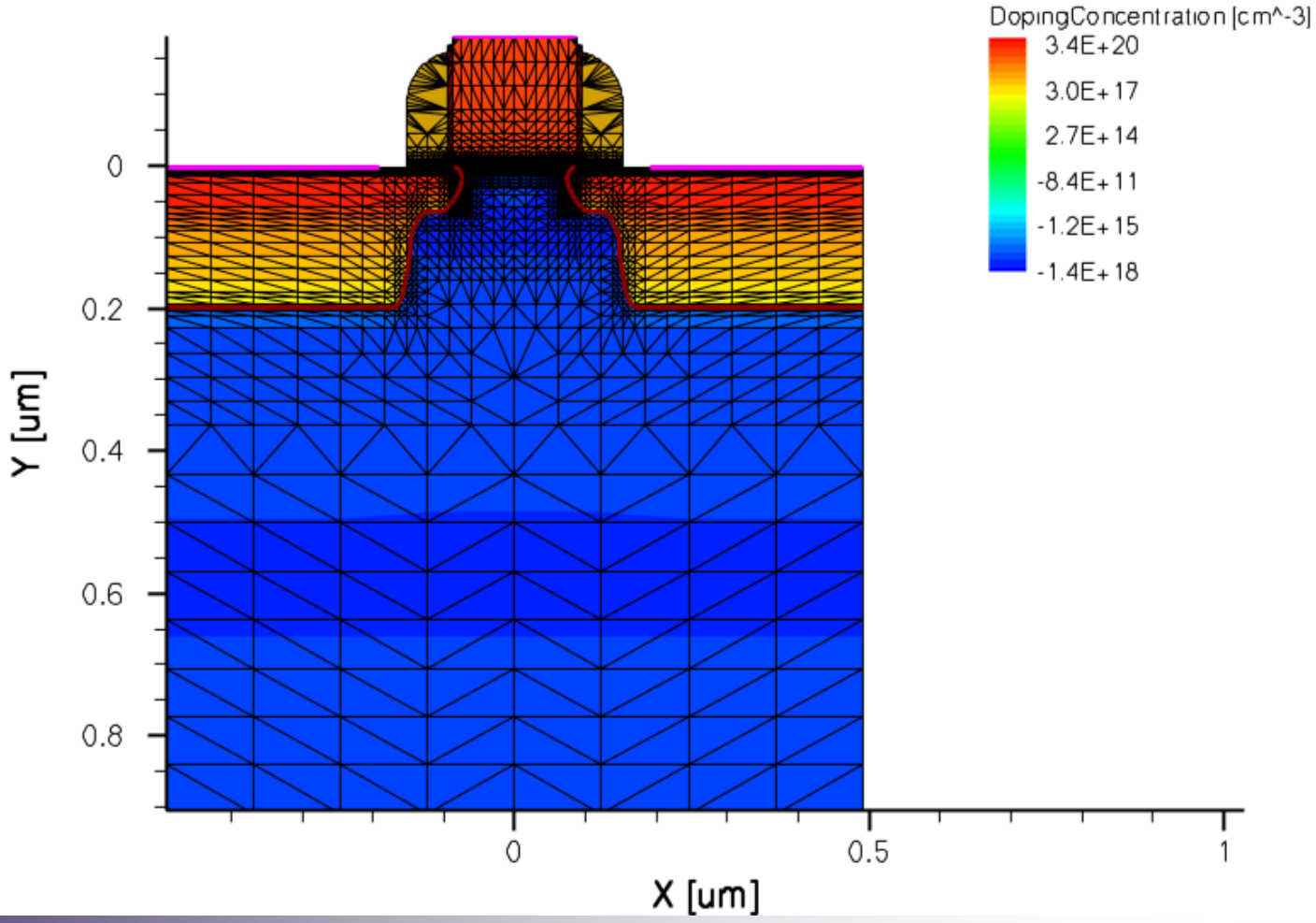
© 2007 Synopsys, Inc. (154)

# Batch Mode

- Refer to command file “sde_dvs.cmd” of example 0.18um_nMOS
- Check output files

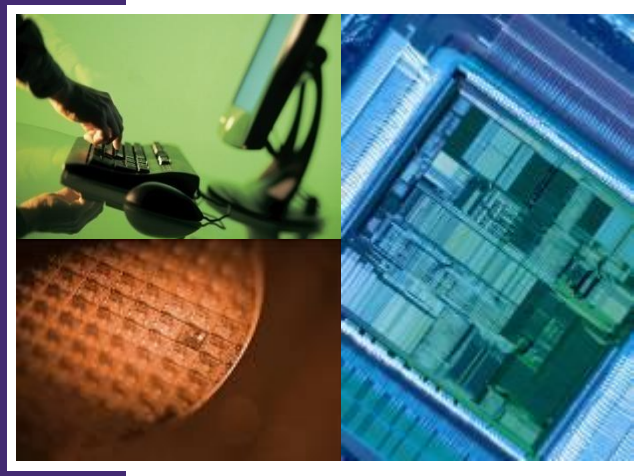
© 2007 Synopsys, Inc. (155)

# The Final Mesh for SDevice




© 2007 Synopsys, Inc. (156)

# Sentaurus Device (SDevice)



SYNOPSYS®

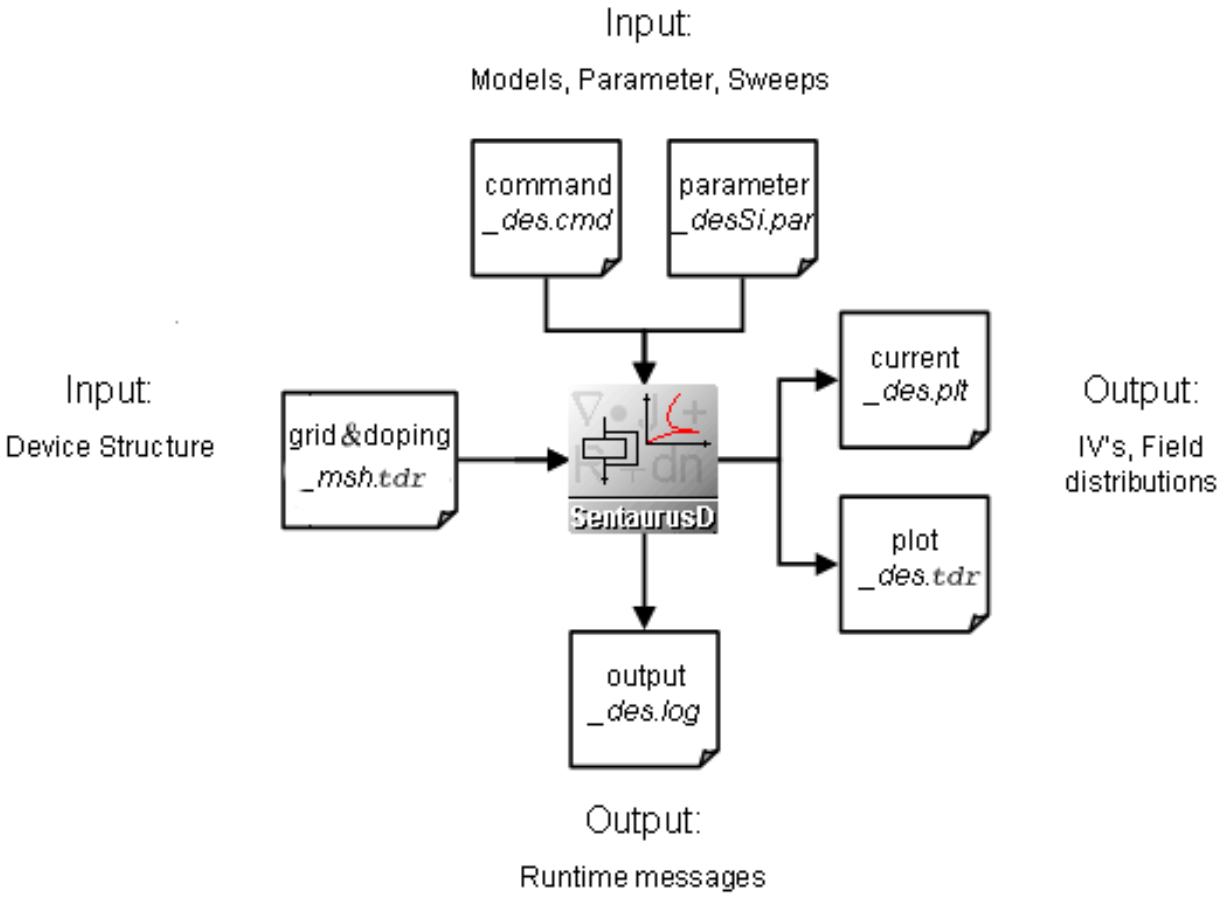
Predictable Success

# Outline

- Flow of input and output in SDevice
- Command file introduction
- Parameter file introduction
- Physical Models in Device Simulation
- Solution Modes in Device Simulation
- 0.18um nMOS Id_Vg example

© 2007 Synopsys, Inc. (158)

# Flow of Input and Output in SDevice




© 2007 Synopsys, Inc. (159)

# Command File

File {
"define the input and output files of the simulation " }

Electrode {
"define electrical (or thermal) contacts, initial bias condition, special boundary condition" }

Physics {
"declare physical models" }

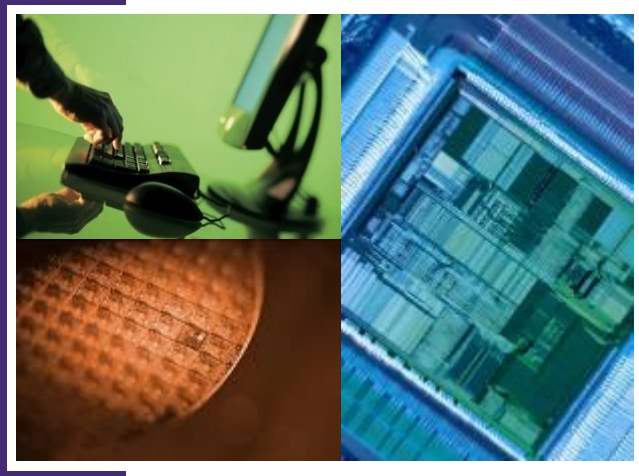
Plot {
"specify the solution variables that are to be saved in the Plot file" }

Math {
"options of numeric solver" }

Solve {
" set bias sweeps sequence and solve transport models" }

© 2007 Synopsys, Inc. (160)

# Overview of SDevice Commands



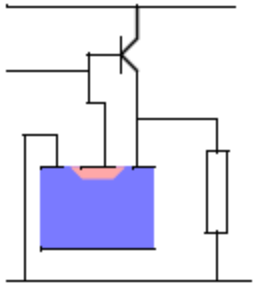
SYNOPSYS®

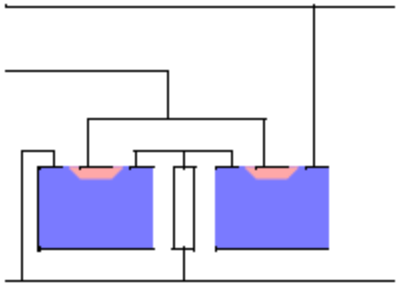
Predictable Success

# What Can Sentaurus Device Do?



Single Device

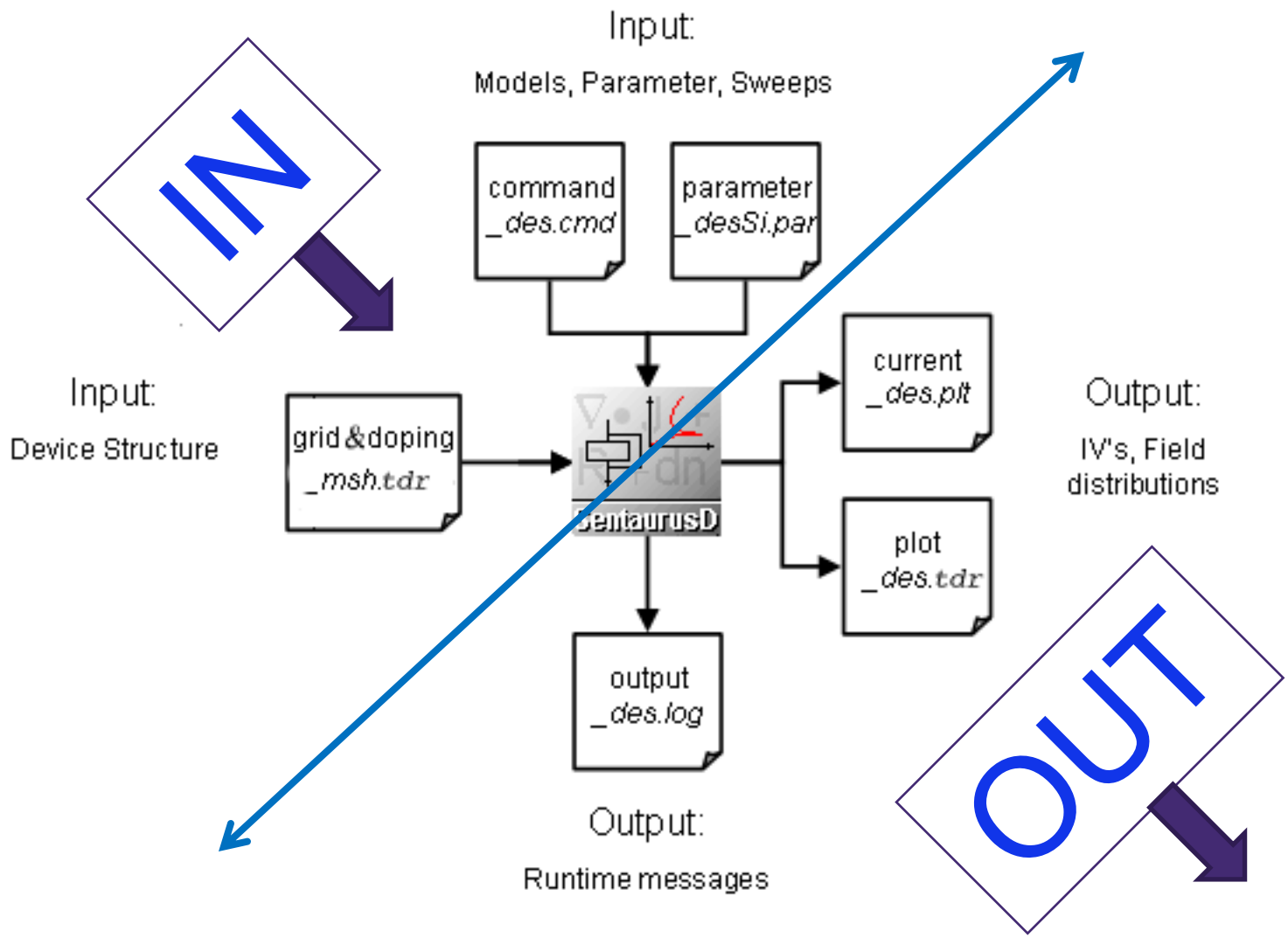

Single Device with Circuit


Multiple Devices with Circuit

- 1D, 2D, and 3D simulations.
- Electrical, thermal, and optical characteristics.
- Silicon-based and compound semiconductor devices.
- Mixed-mode simulations.

© 2007 Synopsys, Inc. (163)

# Sentaurus Device Flow

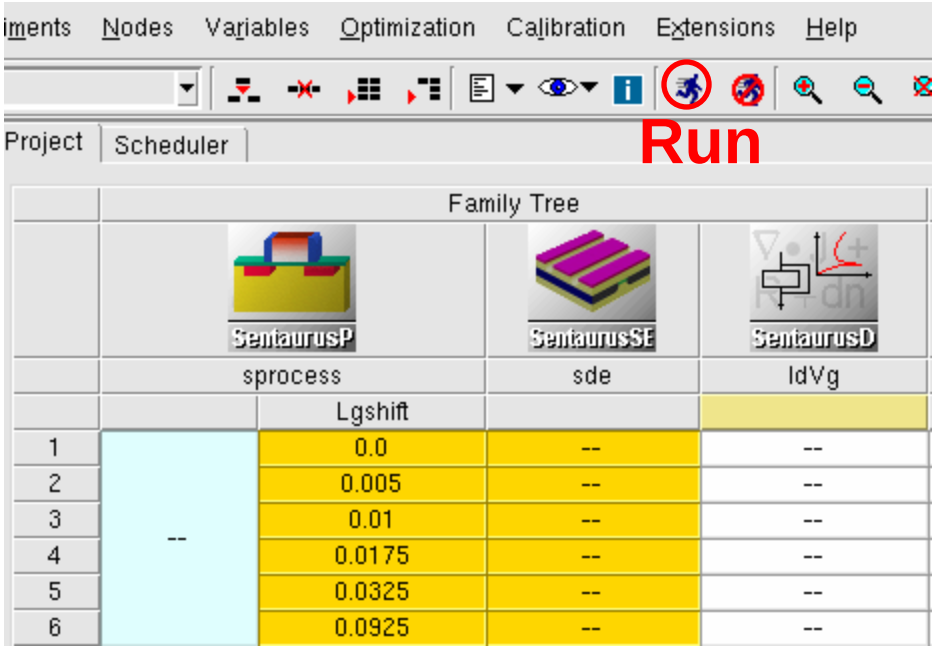



© 2007 Synopsys, Inc. (164)

# Ex 1. Run NMOS Roll Off Project

Project: */PostBasic/question/sdevice/nmos_rolloff_sdevice*

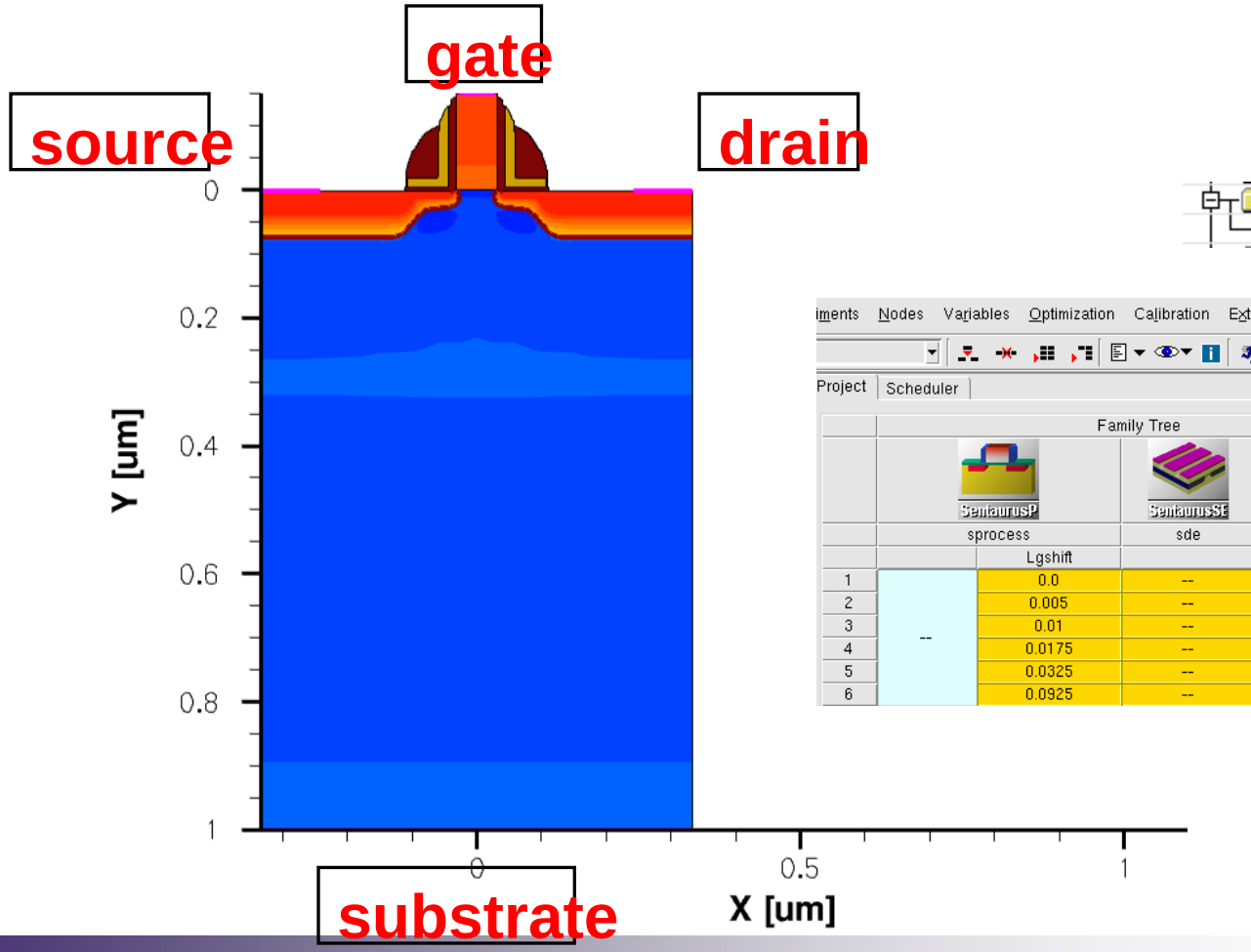
- Open the input command file, and close it.
- Run and observe Id-Vg curves.

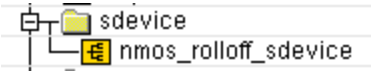



© 2007 Synopsys, Inc. (165)

# NMOS Roll Off Project

Project: */PostBasic/question/sdevice/nmos_rolloff_sdevice*
Description: Simulate NMOS Id-Vg characteristics.







© 2007 Synopsys, Inc. (166)

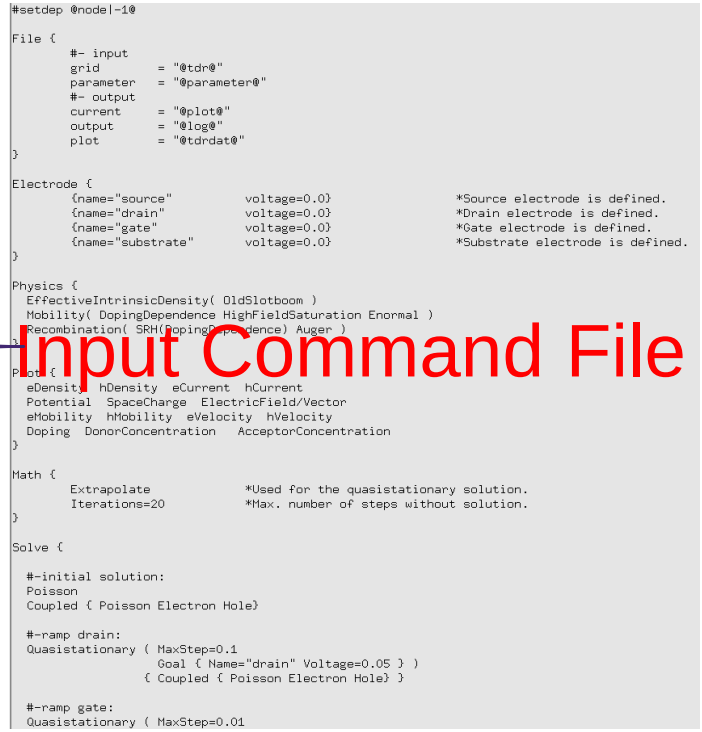
SYNOPSYS®
Predictable Success

# Input Command File

- The default extension of the input command file is _des.cmd, for example, IdVg_des.cmd in this project.

- The input command file typically contains the following sections:

  - File
  - Electrode
  - Physics
  - Plot
  - Solve
  - Math

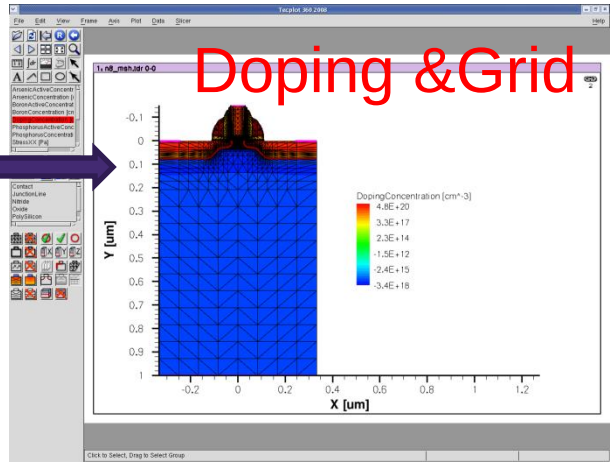


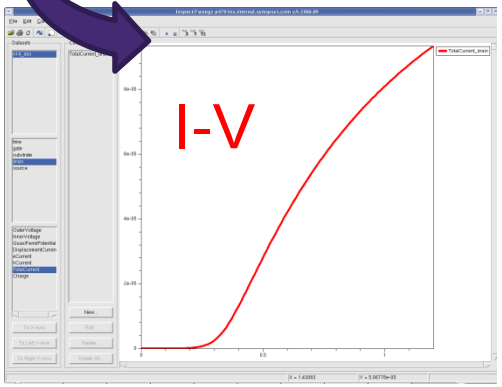

© 2007 Synopsys, Inc. (167)

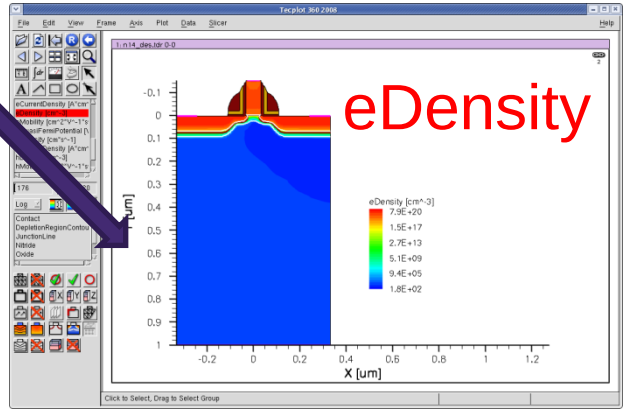
# File Section (1/2)

Define the input and output files of the simulation.

```
File {
      #- input
      grid = "@tdr@"
      parameter= "@parameter@“
      #- output
      plot = "@tdrdat@"
      current = "@plot@"
      output = "@log@"
}
```




© 2007 Synopsys, Inc. (168)

# File Section (2/2)

- **grid =**
  - The grid and doping profile of the device structure.
- **parameter=**
  - The model parameter file.
- **plot =**
  - The final spatial variable data file.
- **current=**
  - The electrical output data file.
- **output =**
  - The output log file.

© 2007 Synopsys, Inc. (169)

# Electrode Section (1/3)

Define the electrical contacts of the device, together with their initial conditions

```
Electrode {
            { Name="source“    Voltage=0.0 }
            { Name="drain“     Voltage=0.0 }
            { Name="gate“      Voltage=0.0 }
            { Name=“substrate“ Voltage=0.0 }
}
```

- Name=
  - Define the contact name.
- Voltage=
  - Define a voltage boundary condition with an initial value.

© 2007 Synopsys, Inc. (170)

# Electrode Section (2/3)

Main Options:

- Barrier=
  - Define the workfunction difference between the intrinsic silicon material and the (highly doped) polysilicon gate when remove polysilicon. For N+ polysilicon, this value is approximately -0.55 eV.
- Current=
  - Define a current boundary condition with initial value [A] (Current=I/AreaFactor).
- Charge=
  - Define a floating electrode with a charge boundary condition and an initial charge value [C] (Charge=C/AreaFactor).

© 2007 Synopsys, Inc. (171)

# Electrode Section (3/3)

Main Options (continued):

- Resistor=
  - Define a series resistance [ Ω ] (Resistor=R*AreaFactor).
- eRecVelocity=
  - Define a recombination velocity at a contact for electrons (hRecVelocity for holes).
- Schottky
  - Define an electrode as a Schottky contact.
- AreaFactor=
  - Specify a multiplication factor for the current in or out of an electrode, 1um by default.

© 2007 Synopsys, Inc. (172)

# Physics Section (1/2)

Declare physical models to be used in the simulation.

```
Physics {
  EffectiveIntrinsicDensity ( OldSlotboom )
  Mobility(DopingDependence  HighFieldSaturation  Enormal)
  Recombination (SRH (DopingDependence)  Auger)
}
```

- Physics section can be set for Material, Region, MaterialInterface and RegionInterface.

- The EffectiveIntrinsicDensity statement activates the silicon band-gap narrowing effect in highly doped regions. The model directly affects the calculation of the intrinsic carrier density in silicon.

© 2007 Synopsys, Inc. (173)

# Physics Section (2/2)

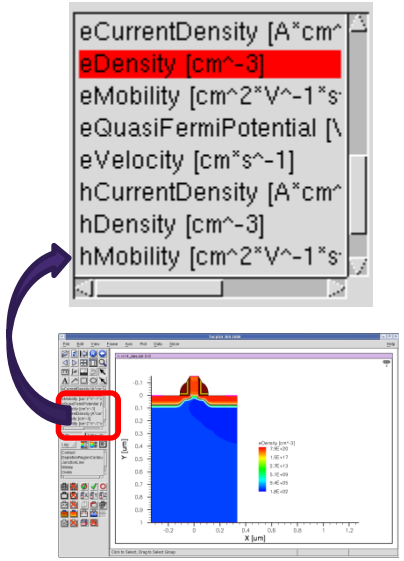
- Mobility models include doping dependence, high-field saturation (velocity saturation), and transverse field dependence.
- Recombination models are specified.
- About mobility models and recombination models, more explanations will be done in the later training.

© 2007 Synopsys, Inc. (174)

# Plot Section

Specify the solution variables that are to be saved in the Plot file.

```
Plot {
  eDensity  hDensity  eCurrent  hCurrent
  Potential  SpaceCharge  ElectricField/Vector
  eMobility  hMobility  eVelocity  hVelocity
  Doping  DonorConcentration
  AcceptorConcentration
}
```




- An extensive list of optional plot variables is in Appendix F of SDevice manual (A-2008.09).
- To save a variable as a vector, append /Vector to the keyword.

© 2007 Synopsys, Inc. (175)

# Solve Section (1/10)

Defines a sequence of solutions to be obtained by the solver.

```
Solve {
  #-initial solution:
  Poisson
  Coupled { Poisson Electron Hole}

  #-ramp drain:
  Quasistationary ( MaxStep=0.1
                    Goal { Name="drain" Voltage=0.05 } )
                    { Coupled { Poisson Electron Hole} }

  #-ramp gate:
  Quasistationary ( MaxStep=0.05
                    Goal { Name="gate" Voltage=1.2 } )
                    { Coupled { Poisson Electron Hole} }
}
```

© 2007 Synopsys, Inc. (176)

# Solve Section (2/10)

```
Coupled { Poisson Electron Hole }
```

- The drift-diffusion transport model is used, it solves self-consistently the coupled Poisson and carrier continuity equations.

- Other transport models include thermodynamic, hydrodynamic transport, etc., using these models can improve simulation accuracy, however usually simulation time increases, and sometimes it becomes difficult to converge.

© 2007 Synopsys, Inc. (177)

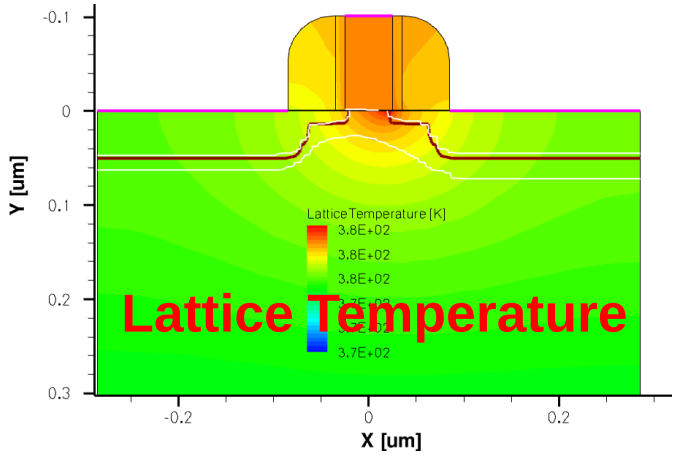
# Solve Section (3/10)

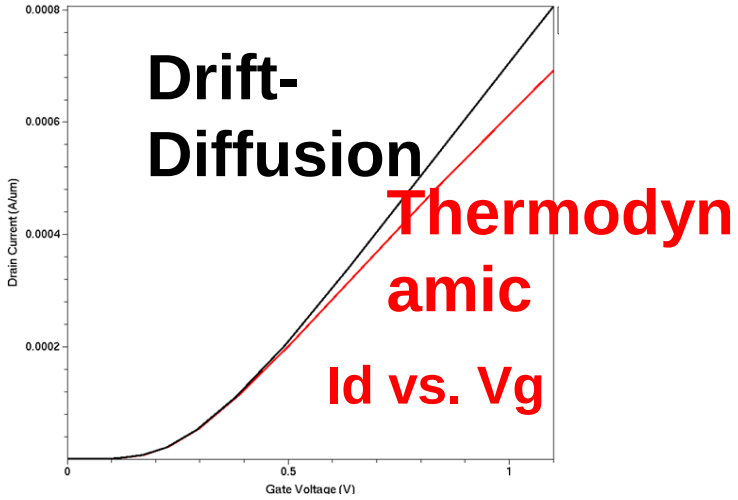
## Thermodynamic Transport

- Solve the lattice temperature equation in addition to the Poisson and carrier continuity equations.

  Coupled { Poisson Electron Hole Temperature }

- Some devices, as power devices, can occur self-heating phenomenon, and need to specify the Thermodynamic transport model.




© 2007 Synopsys, Inc. (178)

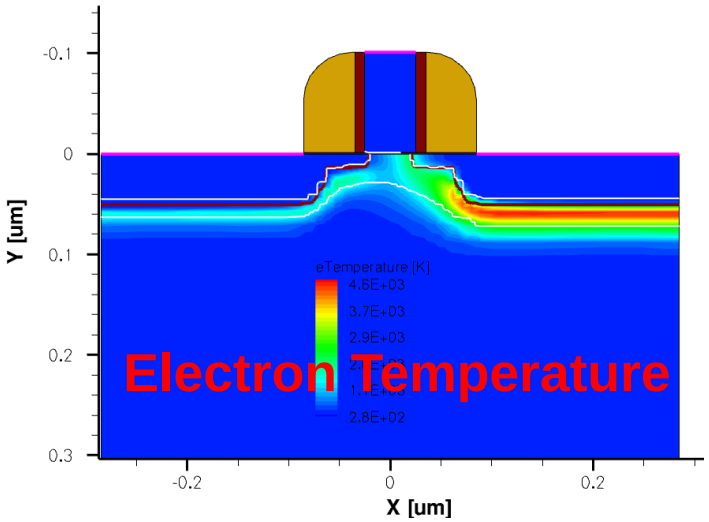
# Solve Section (4/10)

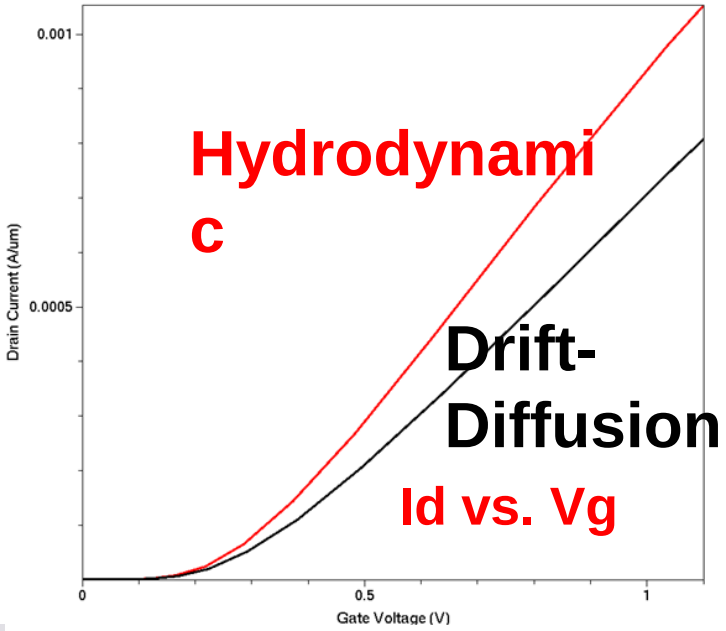
## Hydrodynamic Transport

- Solve the carrier temperature equations in addition to the Poisson and carrier continuity equations.

  Coupled { Poisson Electron Hole eTemperature hTemperature }

- The deep submicron MOSFETs need to specify the hydrodynamic transport model.




© 2007 Synopsys, Inc. (179)

# Solve Section (5/10)

```
#-ramp drain:
Quasistationary ( MaxStep=0.1
                  Goal { Name="drain" Voltage=0.05 } )
                  { Coupled { Poisson Electron Hole} }

#-ramp gate:
Quasistationary ( MaxStep=0.05
                  Goal { Name="gate" Voltage=1.2 } )
                  { Coupled { Poisson Electron Hole} }
```

- There are two Quasistationary statements in this Solve command. The first one ramps the drain voltage to 0.05 V (from 0 as defined in the Electrode section). Similarly, the second sweeps the gate bias from 0 to 1.2 V.

© 2007 Synopsys, Inc. (180)

# Solve Section (6/10)

## Quasistationary Analysis Syntax:

```
Solve{
            …
            Quasistationary   ( InitialStep=…
                                Minstep=…
                                MaxStep=…
                                Increment=…
                                Decrement=…
                                Goal { Name="…" Voltage=… } )
             {
              Coupled{ … }
             }
   }
```

© 2007 Synopsys, Inc. (181)

# Solve Section (7/10)

- Internally, the Quasistationary command works by ramping a variable t from 0.0 to 1.0. $V=V_0+t(V_1-V_0)$, where $V_0$ is the initial voltage and $V_1$ is the final voltage, which is specified in the Goal statement.

- InitialStep controls the size of the first step of the ramping (0.1 by default). 0.01 is recommended.

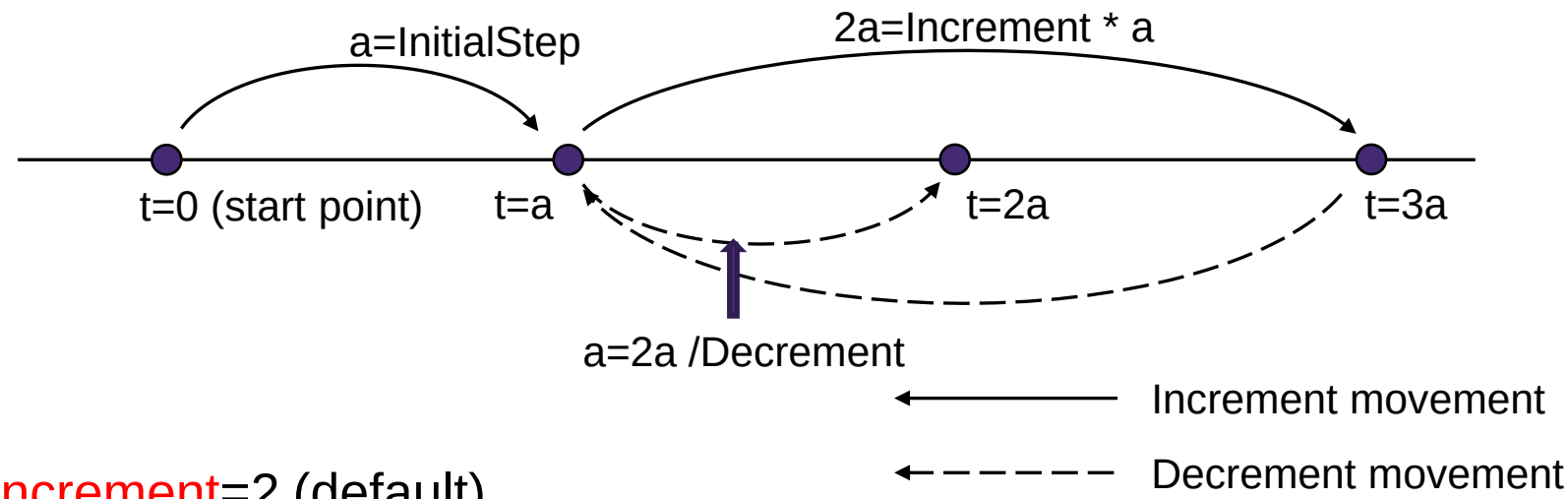
- MaxStep (1 by default) and MinStep (0.001 by default) limit this change.

- The rate of increase is controlled by the factor Increment (2 by default).

- The rate of decrease is controlled by the factor Decrement (2 by default).

© 2007 Synopsys, Inc. (182)

# Solve Section (8/10)




- Increment=2 (default)
  - If it converges at t=0 and t=a points, then it will go to t=3a point.
- Decrement=2 (default)
  - If it fails at t=3a point, then it comes back to t=a point, and this time it will try x=2a point.
- MinStep and MaxStep can forcibly limit their movements.
- When the step is smaller than MinStep, then it fails to converge.

© 2007 Synopsys, Inc. (183)

# Solve Section (9/10)

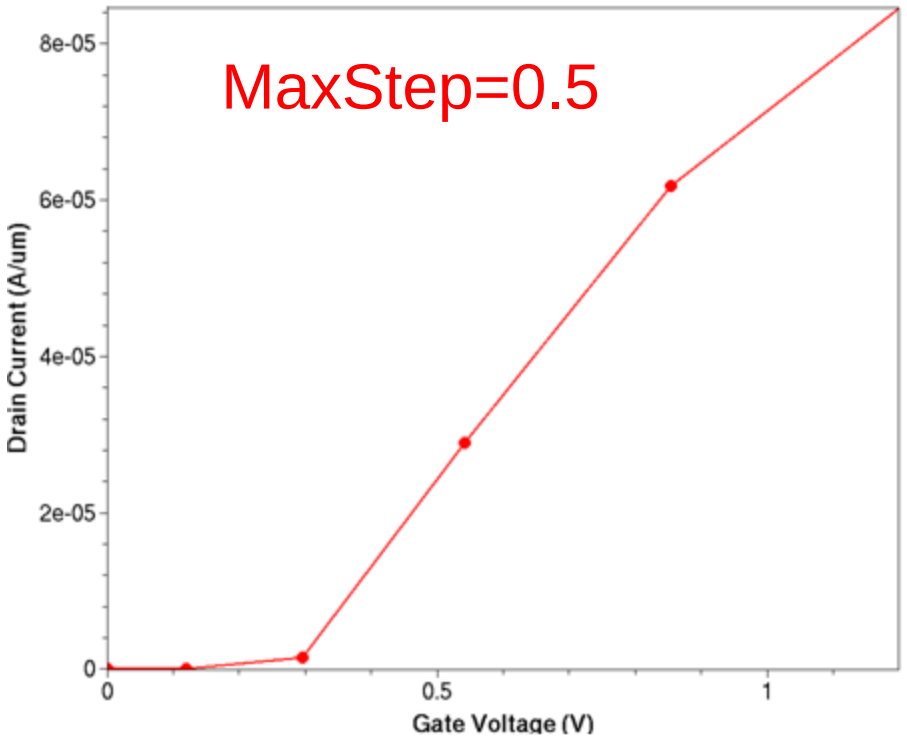
- This is a part of out file:

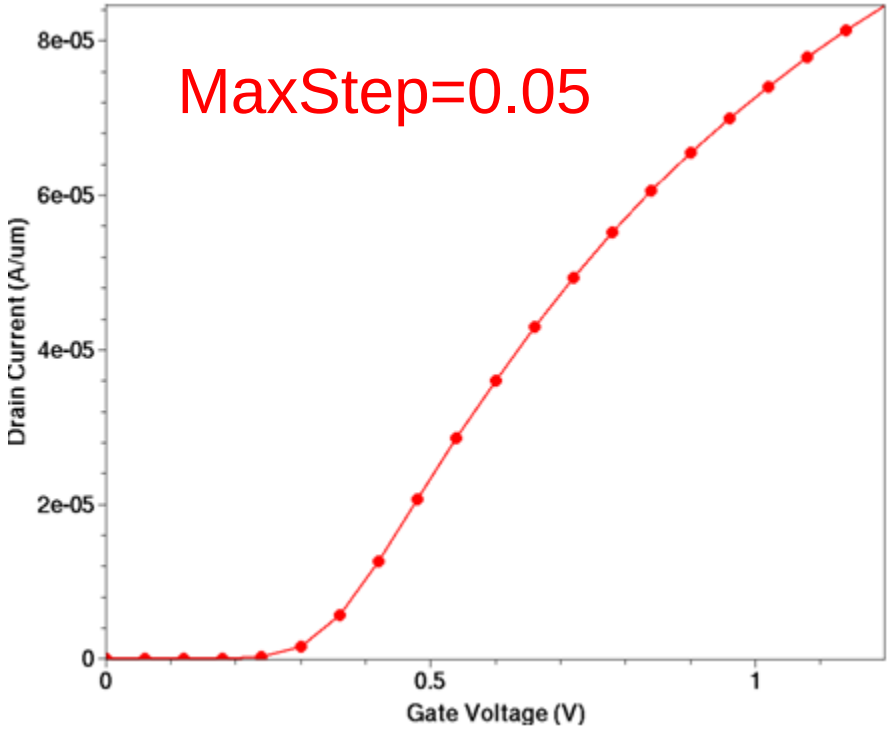
```
…
Computing step from t=0.0000e+00 to t=0.1 (Stepsize: 0.1) :
Computing Coupled( 1 poisson-equation(s) , 1 electron-equation(s) ,
                   1 hole-equation(s) )
using Bank/Rose nonlinear solver.
…
contact       voltage     electron current    hole current  conduction current
 gate        0.000E+00       6.408E-36        -6.408E-36        0.000E+00
 substrate    0.000E+00       2.615E-15         -2.213E-17        2.592E-15
 drain       5.000E-03       4.786E-11         1.357E-21        4.786E-11
 source       0.000E+00      -4.786E-11         -4.701E-27       -4.786E-11

Computing step from t=0.1 to t=0.2 (Stepsize: 0.1) :
 …
```

- The drain voltage changes from 0V to 0.05V.
- When t=0.1, the drain voltage = 0.005V.

© 2007 Synopsys, Inc. (184)

# Solve Section (10/10)



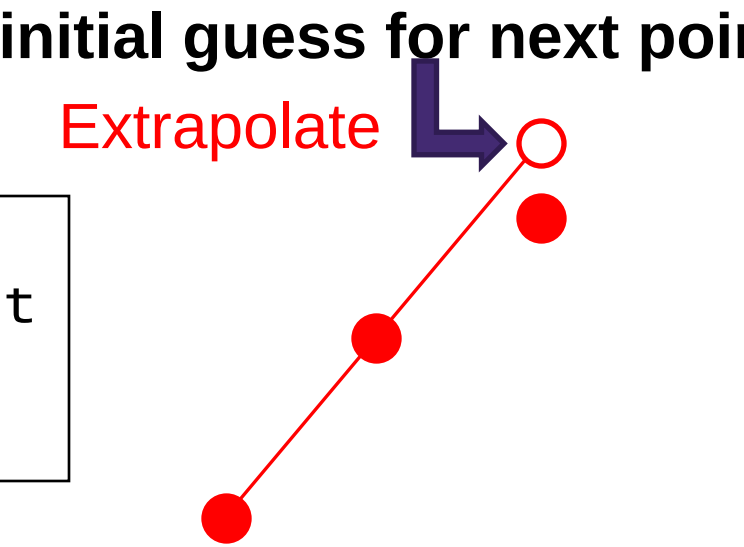




- I-V Curve's smoothness can be improved by decreasing MaxStep.

© 2007 Synopsys, Inc. (185)

# Math Section (1/2)

Control the numeric solver in the simulation.



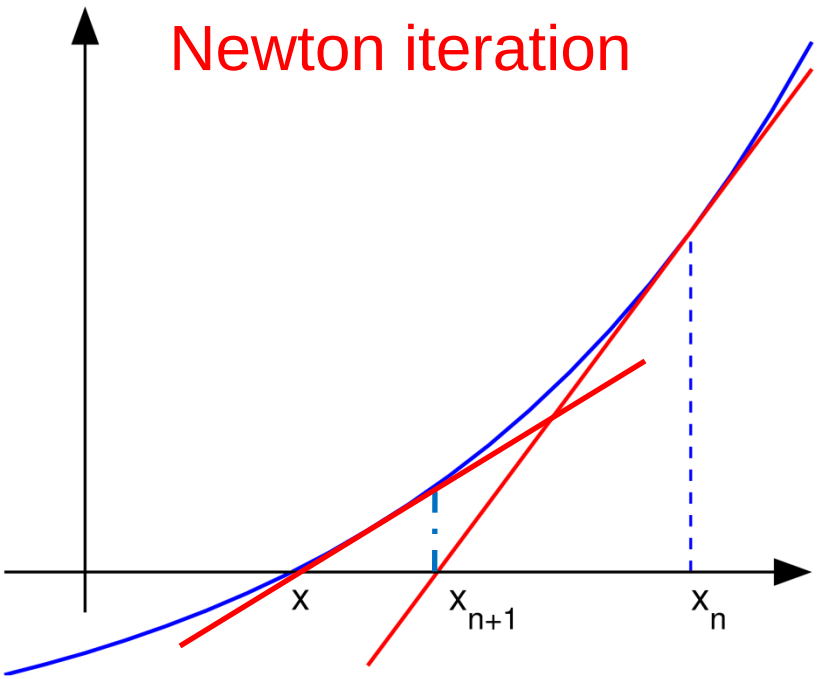

```
Math {
  Extrapolate      *off by default
  Iterations=20    *default = 50
}
```

- Extrapolate
  - In quasistationary bias ramps, the initial guess for a given step is obtained by extrapolation from the solutions of the previous two steps.

- Iterations
  - A maximum of Newton iterations are specified. 20 is recommended.

© 2007 Synopsys, Inc. (186)

# Math Section (2/2)




*If $|\Delta x|$<error, then it converges.*

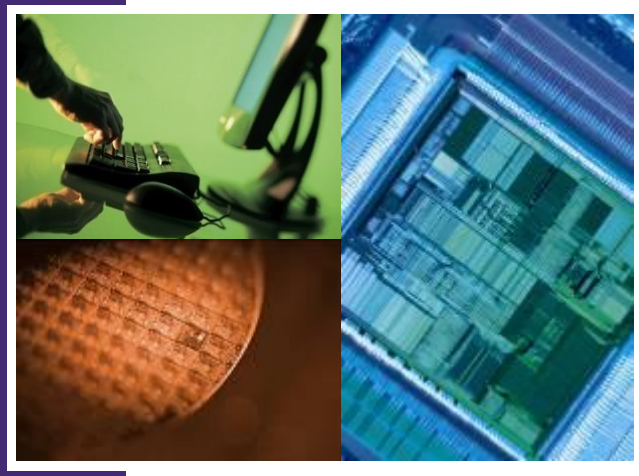
- Given a function *f(x)* and its derivative *f'(x)*, we begin with a first guess $x_0$ . A better approximation $x_1$ is

$$x_1 = x_0 + f(x_0)/ f'(x_0)$$

- The function *f* is shown in blue and the tangent line is in red). We see that $x_{n+1}$ is a better approximation than $x_n$ for the root x of the function f.

© 2007 Synopsys, Inc. (187)

# Additional Features



SYNOPSYS®

Predictable Success

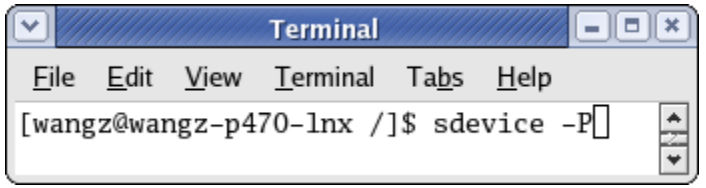
# Edit the Model Parameters

- If you need to use user-defined values for model parameters, you must edit the parameter file.

- To generate a copy of the default parameter file of Silicon, type on the command line:

  $ sdevice -P (for Silicon)

  $ sdevice -P:GaAs (for GaAs)




- The above commands generate the default parameter file models.par.

- If user defines parameter = “@parameter@” in the File Section of the input command file, it will lead to the file sdevice.par.

- User can other parameter file name, for example, parameter = “silicon.par”

- The user-defined parameter file should only contain the user-defined values for model parameters.

© 2007 Synopsys, Inc. (189)

# Define Multi-Material Model Parameters In a Parameter File

- User can define multi-material model parameters in a parameter file as follows.

```
Material="AlGaAs" {
   Epsilon {
                …
    }
}
Material="GaAs" {
   Epsilon {
                …
    }
}
```

© 2007 Synopsys, Inc. (190)

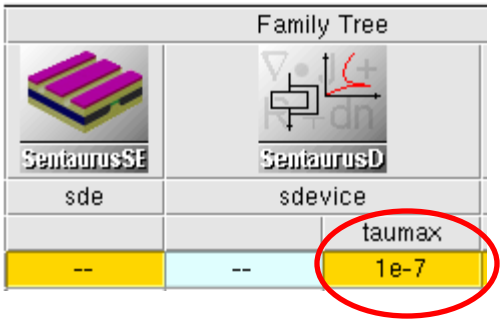
# Ex 2. Change the Electron Lifetime

Project: */PostBasic/question/sdevice/nmos_parameter*

- Open the parameter file. Because parameter = "@parameter@" is defined in the File Section, then it leads to the file sdevice.par.

- Edit sdevice.par as follows:

```
Scharfetter
      { taumin = 0.0000e+00 , 0.0000e+00 # [s]
        #- taumax = 1.0000e-05 , 3.0000e-06 # [s]
        taumax = @taumax@ , 3.0000e-06 # [s]
        Nref = 1.0000e+16 , 1.0000e+16 # [cm^(-3)]
        gamma = 1 , 1 # [1]
        Talpha = -1.5000e+00 , -1.5000e+00 # [1]
        Tcoeff = 2.55 , 2.55 # [1]
        Etrap = 0.0000e+00 # [eV] }
```




- Define the parameter taumax in the SWB, set its value as 1e-07. Run the nmos_parameter project.

© 2007 Synopsys, Inc. (191)

# Plot Statement In the Solve Section (1/2)

```
Solve{
  …
  Quasistationary(
  …
  ){ Coupled{ … }
       Plot(
          FilePrefix="n@node@"  NoOverWrite
          Time=(0.1; 0.3; 0.8)
       )
   }
…
}
```

- Plotting or saving a solution during a (DC or transient) sweep at a set of given values of the t variable. for example, 10%, 30%, 80% of the goal.

© 2007 Synopsys, Inc. (192)

# Plot Statement In the Solve Section (2/2)

```
Solve{
  …
  Quasistationary(
  …
  ){ Coupled{ … }
       Plot(
           FilePrefix="n@node@" NoOverWrite
           Time=(Range=(0 1) Intervals=10)
        )
    }
…
}
```

- Plotting at regular intervals, for example, 0%, 10%, 20%, ... , 100% of the goal.

© 2007 Synopsys, Inc. (193)

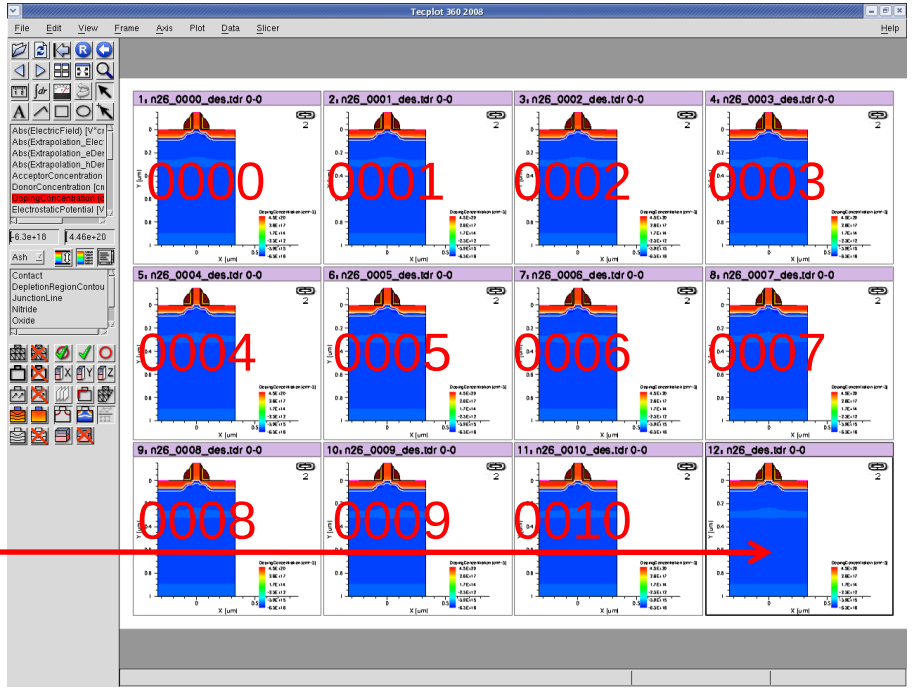
# Ex 3. Add Plot Statement

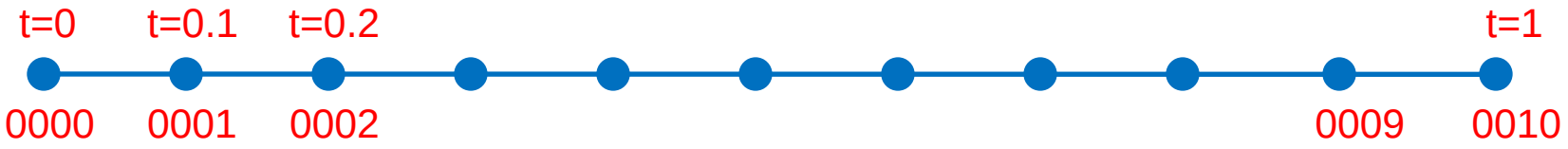
Project: */PostBasic/question/sdevice/nmos_plot*

- Add the following statement to the second goal of Solve section.

```
Plot(
     FilePrefix="n@node@" NoOverWrite
     Time=(Range=(0 1) Intervals=10)
  )
```

- Run and observe tdr files.
  These tdr files are
  *n@node@_0000_des.tdr,*
  *n@node@_0001_des.tdr,*
  *…,*
  *n@node@_0010_des.tdr,*
  *n@node@_des.tdr*




© 2007 Synopsys, Inc. (194)

# AreaFactor

- For the two-dimensional simulation, by default, SDevice assumes a ‘thickness’ (effective gate width along the z-axis) of 1um.

- The effective width is adjusted by specifying
  - an AreaFactor in the Physics section, or
  - an AreaFactor for each electrode individually.

- An AreaFactor is a multiplier for the electrode currents and charges.

© 2007 Synopsys, Inc. (195)

# CurrentPlot Section

- Save variables at specified nodes or coordinates to the current file _des.plt. The same variables can be selected as in the Plot Section

- It is possible to output averages, integrals, and the maximum and minimum of quantities over specified domains.
  - To do this, specify the keyword Average, Integrate, Maximum, or Minimum.
  - Refer to SDevice manual for more information.

```
CurrentPlot {
          hDensity( (0 1) ) * hole density at position (0um, 1um)
          Potential (
                Average(Region="Channel") * average over a region
                Maximum(Material="Oxide") * maximum in a material
          )
}
```

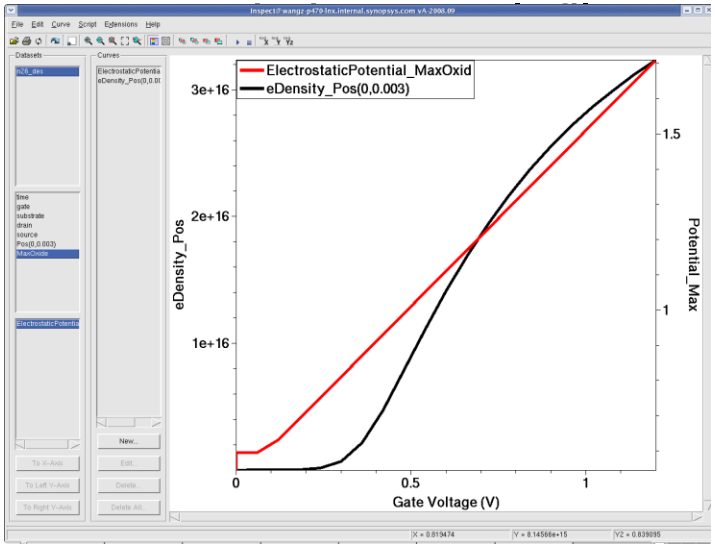
© 2007 Synopsys, Inc. (196)

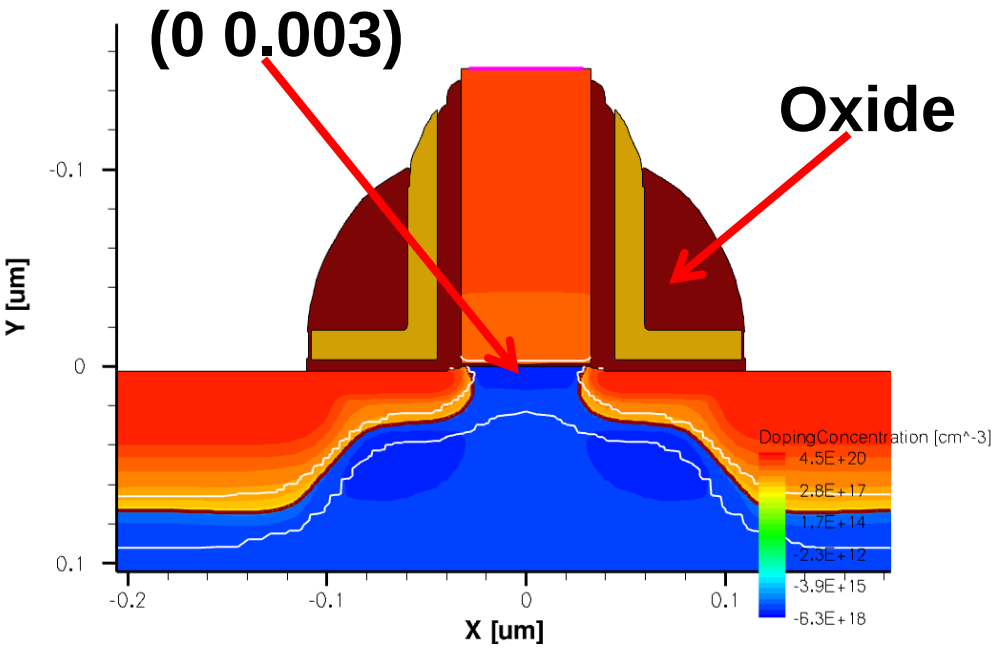
# Ex 4. Add CurrentPlot Section

Project: */PostBasic/question/sdevice/nmos_currentplot*

- Add the following section to the second goal of Solve section.

```
CurrentPlot {
        eDensity( (0 0.003) )
        Potential ( Maximum(Material="Ox
}
```




© 2007 Synopsys, Inc. (197)

SYNOPSYS®
Predictable Success

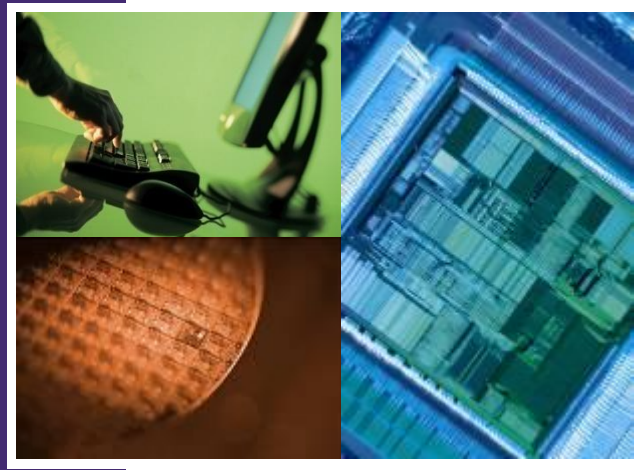
# Sweep Multi Voltages Simultaneously

```
Electrode {
                { Name="A"      Voltage=VA0 }
                { Name="B"      Voltage=VB0 }
      …
      }
      Solve{
                 …
                 Quasistationary(
                          …
                          Goal { Name="A"       Voltage=VA }
                          Goal { Name="B"       Voltage=VB }
                          …
                          )
                  { Coupled{ … }
   }
```

© 2007 Synopsys, Inc. (198)

# Physics Models

- Mobility
- Generation–recombination



SYNOPSYS®

Predictable Success

# Mobility Models (1/3)

```
Physics{
            …
            Mobility( <arguments> )
            …
          }
```

- SDevice uses a modular approach for the description of the carrier mobility models. Silicon MOSFETs usually specify the following models.
  - Doping-dependent mobility model
  - Mobility degradation at interfaces
  - High-field saturation model

$$\frac{1}{\mu} = \frac{1}{\mu_{b1}} + \frac{1}{\mu_{b2}} + \ldots + \frac{1}{\mu_{s1}} + \frac{1}{\mu_{s2}} + \ldots$$

- Refer to SDevice manual (A-2008.09) for more information.
  - *Part II: Physics in Sentaurus Device*
    *CHAPTER 8: Mobility models*

© 2007 Synopsys, Inc. (200)

# Mobility Models (2/3)

## Doping-Dependent Mobility Model

- Activated by

```
Physics{Mobility(DopingDependence … ) … }
```

- For doped materials, the carriers scatter with the impurities. This leads to a degradation of the mobility.

## Mobility Degradation At Interfaces

- Activated by

```
Physics{Mobility(Enormal … ) … }
```

- In the channel region of a MOSFET, the high transverse electric field forces carriers to interact strongly with the semiconductor–insulator interface.

© 2007 Synopsys, Inc. (201)

# Mobility Models (3/3)

## High-Field Saturation Model

- Activated by

```
Physics{Mobility(HighFieldSaturation … ) … }
```

- Describe mobility degradation in high electric fields.
  In high electric fields, the carrier drift velocity is no longer proportional to the electric field.

© 2007 Synopsys, Inc. (202)

# Generation–recombination Models (1/3)

```
Physics{
        …
        Recombination( <arguments> )
        …
      }
```

- Generation–recombination processes are processes that exchange carriers between the conduction band and the valence band.
- Usually, Silicon MOSFETs need to be specified the following models:
  - Shockley–Read–Hall (SRH) recombination
  - Auger recombination
- The breakdown analysis must specify avalanche generation model.
- Refer to SDevice manual (A-2008.09) for more information.
  - *Part II: Physics in Sentaurus Device*
    *CHAPTER 9: Generation–recombination*

© 2007 Synopsys, Inc. (203)

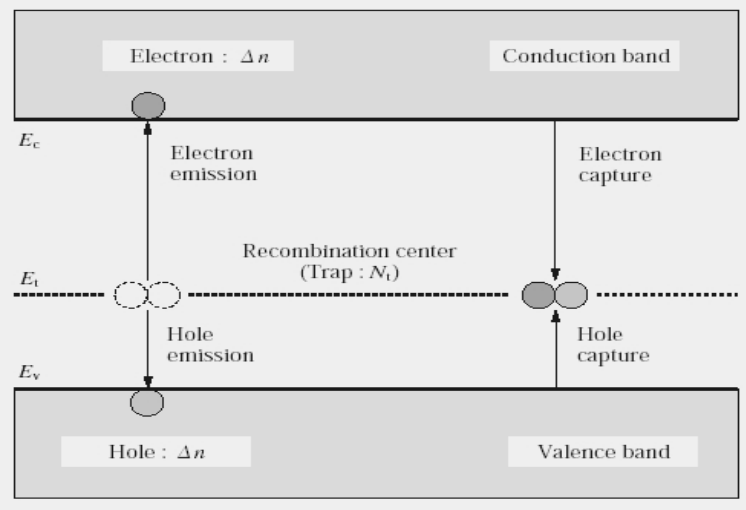
# Generation–recombination Models (2/3)

## SRH Recombination

- Activated by

```
Physics{ Recombination(SRH …) …}
```

- SRH recombination through deep defect levels.




- The doping dependence of the SRH lifetimes is activated by

```
Physics{Recombination(SRH(DopingDependence… ) …) …}
```

© 2007 Synopsys, Inc. (204)

# Generation–recombination Models (3/3)

## Auger Recombination

- The Auger model is activated by

```
Physics{Recombination(Auger …) …}
```

- Band-to-band recombination model. Typically important at high carrier densities.

## Avalanche Generation

- Activated by

```
Physics{Recombination(Avalanche …) …}
```

- Avalanche generation (impact ionization) requires a certain threshold field strength and the possibility of acceleration.

- The breakdown analysis must specify avalanche model.

© 2007 Synopsys, Inc. (205)

# Ex 5. Change Mobility Models

Projects: */PostBasic/question/sdevice/nmos*

- Copy nmos project to a new project named as nmos_mobility.
- Edit nmos_mobility project's Physics section as follows:

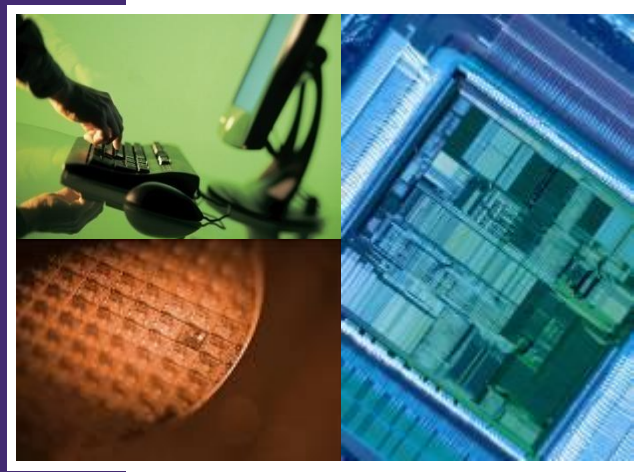
```
#-Mobility( DopingDependence HighFieldSaturation Enormal)
Mobility( HighFieldsaturation Enormal)
```

- Run nmos_mobility project, and compare its Id-Vg curve with nmos project's.

© 2007 Synopsys, Inc. (206)

# Examples of Analysis

- Mixed-Mode Transient Analysis
- AC Analysis



SYNOPSYS®

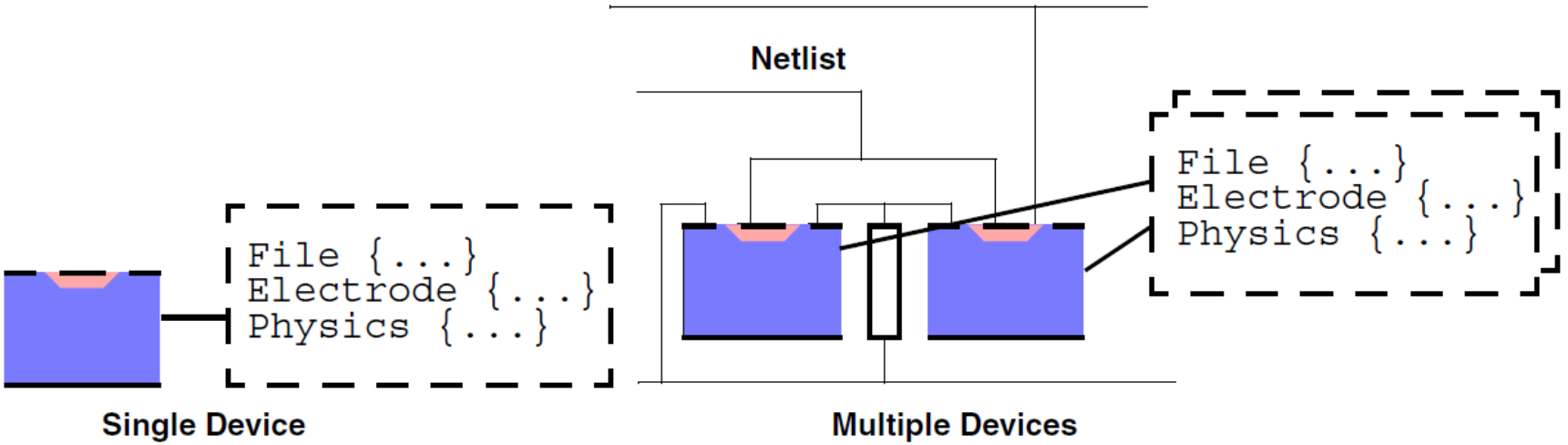
Predictable Success

# Transient and AC Analyses

- The transient analysis is usually done by using the mixed-mode simulation.
- Under the following conditions, the transient analysis can be done without the mixed-mode simulation.
  - There is only a physical device;
  - The piecewise linear source is used.
- The AC analysis must to use the mixed-mode simulation.
- The C-V characteristics can be obtained by the AC analysis.

© 2007 Synopsys, Inc. (208)

# Mixed-Mode Simulation




- SDevice is a single-device simulator, and a mixed-mode device and circuit simulator.

- Provides a number of compact models for use in mixed-mode simulations.
  - SPICE, HSPICE, built-in models, etc.

© 2007 Synopsys, Inc. (209)

# Input Command File of Mixed-Mode

- Three levels of device definition
  - Global
  - Device
  - Instance

```
File {...}
Electrode {...}
  ...

Device <type> {
  File {...}
  Electrode {...}
  ...
}

System {
  <type> <name> {
  File {...}
  Electrode {...}
  }
}
```

Global

Device

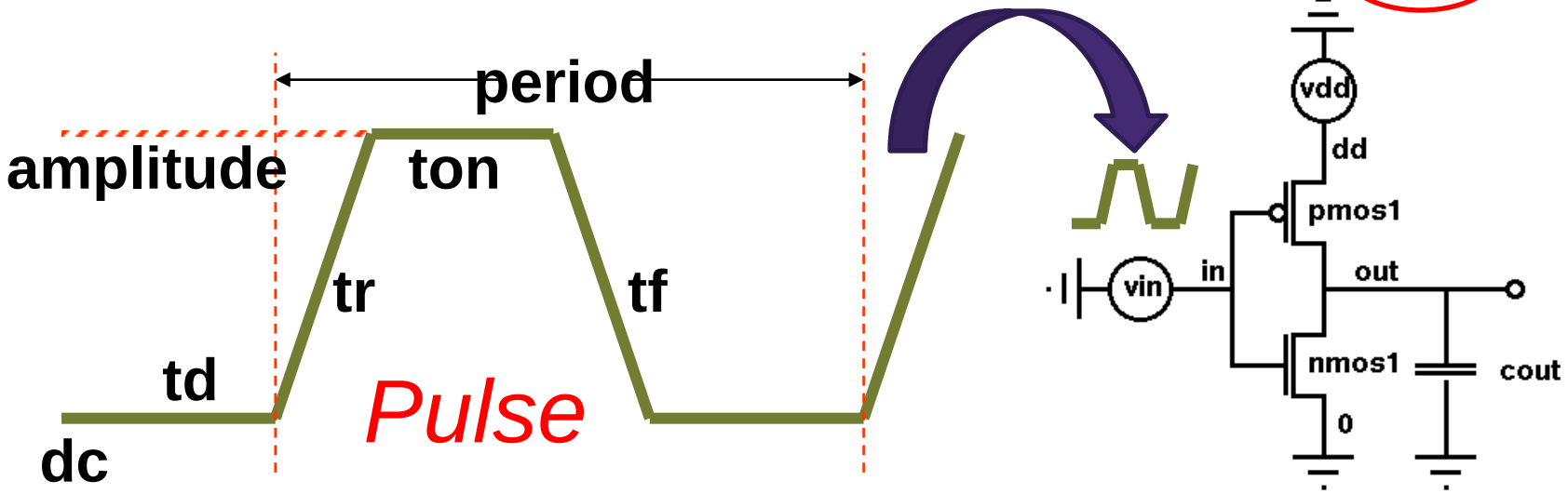
Instance

- Mixed-Mode simulation need to specify two new sections:
  - Device section
    - ✓ Device section defines physical devices used in the sys
  - System section
    - ✓ System section defines the netlist of physical devices ar circuit elements to be solved.

© 2007 Synopsys, Inc. (210)

# Example: CMOS Inverter Transient Analysis

Project: */PostBasic/example/sdevice/cmos_inverter_transient*
Description: Simulate the transient characteristics.


- CMOS inverter consists of both NMOS and PMOS devices.

© 2007 Synopsys, Inc. (211)

# Device Section (1/2)

```
Device NMOS {

  Electrode{
    { Name="source"    Voltage=0.0 }
    { Name="drain"     Voltage=0.0 }
    { Name="gate"      Voltage=0.0 }
    { Name="substrate" Voltage=0.0 }
  }

  File{
     Grid        = "@tdr|nmos@"
     Plot        = "@tdrdat@"
     Current     = "@plot@"
     Parameter   = "@parameter@"
  }

  Physics{
AreaFactor=5
Mobility( DopingDep HighFieldSaturation
Enormal )
EffectiveIntrinsicDensity(oldSlotboom )
  }
}
```

```
Device PMOS{

  Electrode{
    { Name="source"    Voltage=0.0 }
    { Name="drain"     Voltage=0.0 }
    { Name="gate"      Voltage=0.0 }
    { Name="substrate" Voltage=0.0 }
  }

  File{
     Grid        = "@tdr|pmos@"
     Plot        = "@tdrdat@"
     Current     = "@plot@"
     Parameter   = "@parameter@"
  }

  Physics{
AreaFactor=10
Mobility( DopingDep HighFieldSaturation
Enormal )
EffectiveIntrinsicDensity(oldSlotboom )
  }
}
```

© 2007 Synopsys, Inc. (212)

# Device Section (2/2)

- Each Device section contains an Electrode, a File, and a Physics section.

- In mixed mode, currents from different devices interact need to be scaled according to the actual device widths. The AreaFactor specifications in the Physics section are used to define the width of the devices.

© 2007 Synopsys, Inc. (213)

# System Section (1/5)

```
System{
  Vsource_pset  vdd (dd 0) {  dc = 0.0 }
  Vsource_pset  vin (in 0) {  pulse = (0.0          # dc
                                       1.5          # amplitude
                                       10e-12       # td
                                       10e-12       # tr
                                       10e-12       # tf
                                       60e-12       # ton
                                       140e-12)     # period
  }

  NMOS nmos1 ( "source"=0  "drain"=out "gate"=in "substrate"=0 )
  PMOS pmos1 ( "source"=dd "drain"=out "gate"=in "substrate"=dd )
  Capacitor_pset cout ( out 0 ){ capacitance = 3e-14 }
  Plot "n@node@_sys_des.plt" (time() v(in) v(out)
                              i(nmos1,out) i(pmos1,out)
i(cout,out) )
}
```

- The node 0 is predefined as the ground node.

© 2007 Synopsys, Inc. (214)

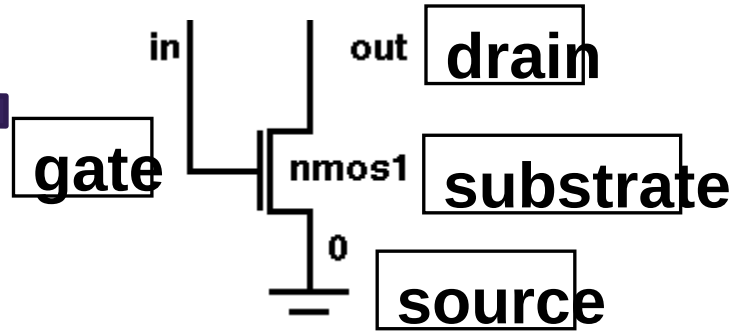
# System Section (2/5)

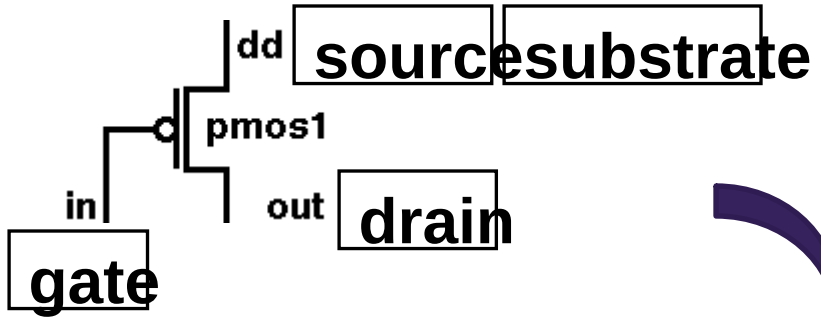
- The keyword Vsource_pset represents a compact model voltage source.
- The keyword Capacitor_pset represents a compact model capacitor.
- NMOS and PMOS, defined in the previous Device sections are used here to define the two circuit elements, nmos1 and pmos1, respectively.
- The Electrode block of the Device section are all connected to a node of the circuit. For example, "source"=0, "drain"=out.

© 2007 Synopsys, Inc. (215)

# System Section (3/5)

```
NMOS nmos1 (“source”= 0  “drain”=out “gate”=in  "substrate"=0 )
```




```
PMOS pmos1 ( "source"=dd "drain"=out "gate"=in "substrate"=dd )
```

© 2007 Synopsys, Inc. (216)

# System Section (4/5)

- vin is defined as a time-dependent voltage source that generates a pulse train. Available time-dependent sources are as follows:

| | |
|---|---|
| `pulse` | Pulse description |
| `sine` | Sinusoidal source description |
| `sin`[a] | Sinusoidal source description |
| `exp` | Exponential source description |
| `pwl` | Piecewise linear description |
| `sffm` | Single-frequency FM description |

a. Equivalent to the `sine` parameter.

- Refer to the following manual for more information.
  - *Compact Models User Guide* (cm_ug.pdf)

© 2007 Synopsys, Inc. (217)

# System Section (5/5)

```
System{
  …
  Plot "n@node@_sys_des.plt" (  time()   v(in)   v(out)
                                i(nmos1,out)   i(pmos1,out)
                                i(cout,out) )
}
```

- The Plot statement inside the System section allows voltages and currents at selected nodes of the circuit to be recorded.
- To record voltage, specify the node name. For example, v(in), v(out).
- To record current, specify both the element and the node name through which the current flows. For example, i(nmos1,out), i(pmos1,out).

© 2007 Synopsys, Inc. (218)

# Global File and Plot Sections

- Global File section

```
File{
      Output = "@log@"
}
```

- Global Plot section

```
Plot{
      eDensity
      hDensity
      …
}
```

- Used by all Device sections.
- The global File (Plot) section can be placed either inside a Device definition.

© 2007 Synopsys, Inc. (219)

# Solve Section (1/2)

```
Solve{
  NewCurrentFile="init"
  Coupled(Iterations=100){ Poisson }
  Coupled{ Poisson Electron Hole Contact Circuit }
  Quasistationary(
     InitialStep=1e-3 Increment=1.35
     MinStep=1e-5 MaxStep=0.05
     Goal{ Parameter=vdd.dc Voltage= 1.5 }
  ){ Coupled{ nmos1.poisson   nmos1.electron    nmos1.contact
              pmos1.poisson   pmos1.hole        pmos1.contact
              circuit }
  }
  NewCurrentFile=""
  Transient (
     InitialTime=0 FinalTime=200e-12
     InitialStep=1e-12 MaxStep=5e-12 MinStep=1e-15
  ){ Coupled{ nmos1.poisson    nmos1.electron     nmos1.contact
                       pmos1.poisson    pmos1.hole
pmos1.contact
                       circuit }
  }
}
```

© 2007 Synopsys, Inc. (220)

# Solve Section (2/2)

## Transient Analysis Syntax:

```
Solve{
        …
        #-Transient sweep
        Transient( InitialTime=   FinalTime=…
                   InitialStep=…  Minstep=…    MaxStep=…
                 )
        { Coupled{ …  } }
}
```

- InitialTime (0s by default) is start time; FinalTime is final time.
- InitialStep controls the size of the first step of the ramping (0.1s by default). MaxStep (1s by default) and MinStep (0.001s by default) limit this change.

© 2007 Synopsys, Inc. (221)

# Output Files of Mixed-Mode

nmos1_n3_des.tdr
initnmos1_n3_des.plt
nmos1_n3_des.plt

pmos1_n3_des.tdr
initpmos1_n3_des.plt
pmos1_n3_des.plt


```
Device NMOS {…
  File{…
     Plot        = "@tdrdat@"
     Current     = "@plot@"
     … } }
```

```
Device PMOS {…
  File{…
     Plot        = "@tdrdat@"
     Current     = "@plot@"
     … } }
```

```
Solve{

NewCurrentFile="init“
…

NewCurrentFile=""
…
}
```

```
System{…
  NMOS nmos1 ( "source"=0  "drain"=out "gate"=in "substrate"=0 )
  PMOS pmos1 ( "source"=dd "drain"=out "gate"=in "substrate"=dd )
  …

  Plot "n@node@_sys_des.plt" (time() v(in) v(out)
                              i(nmos1,out) i(pmos1,out)
                              i(cout,out) )
}
```

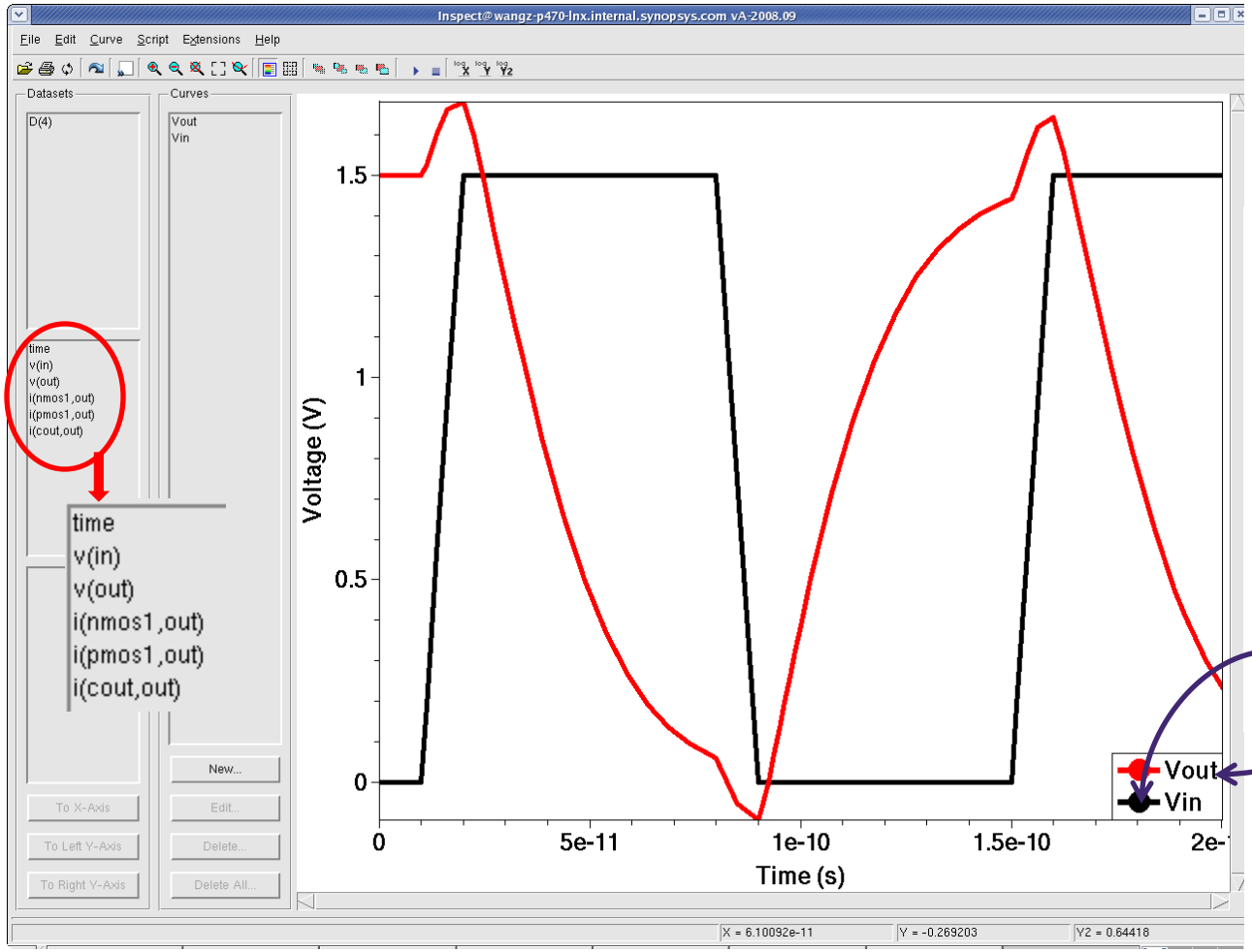
n3_sys_des.plt

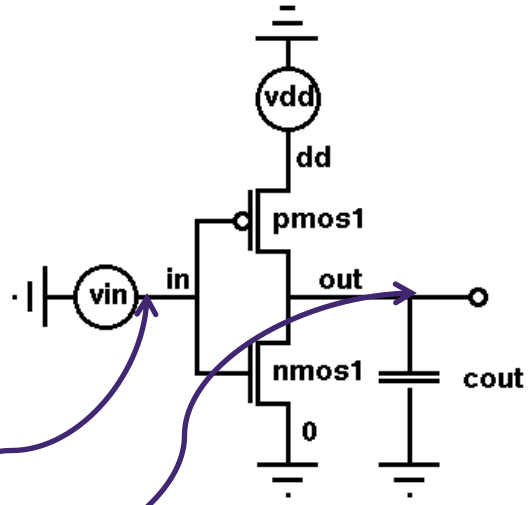
```
File{ Output = "@log@“ }
```

n3_des.out

© 2007 Synopsys, Inc. (222)

# CMOS Transient Characteristic



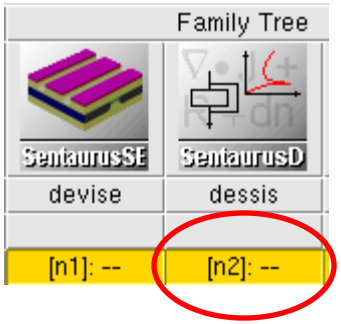

Vin=v(in)

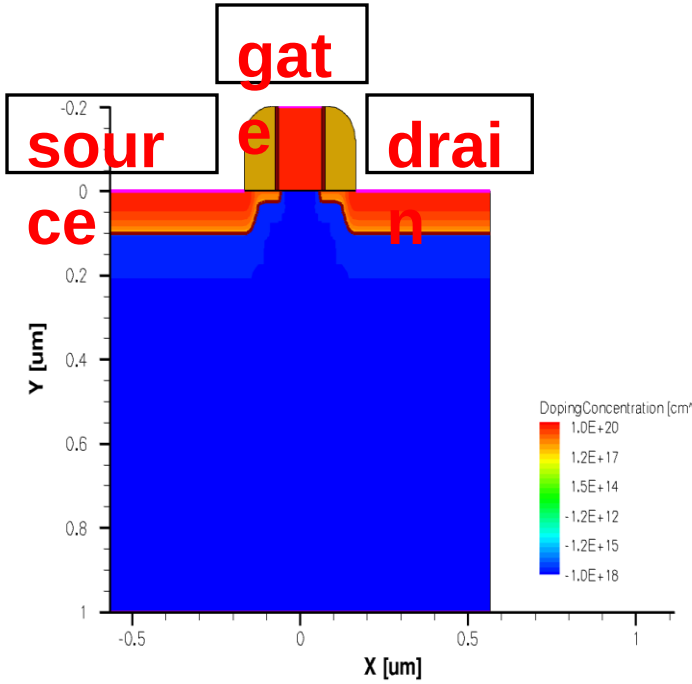
Vout=v(out)

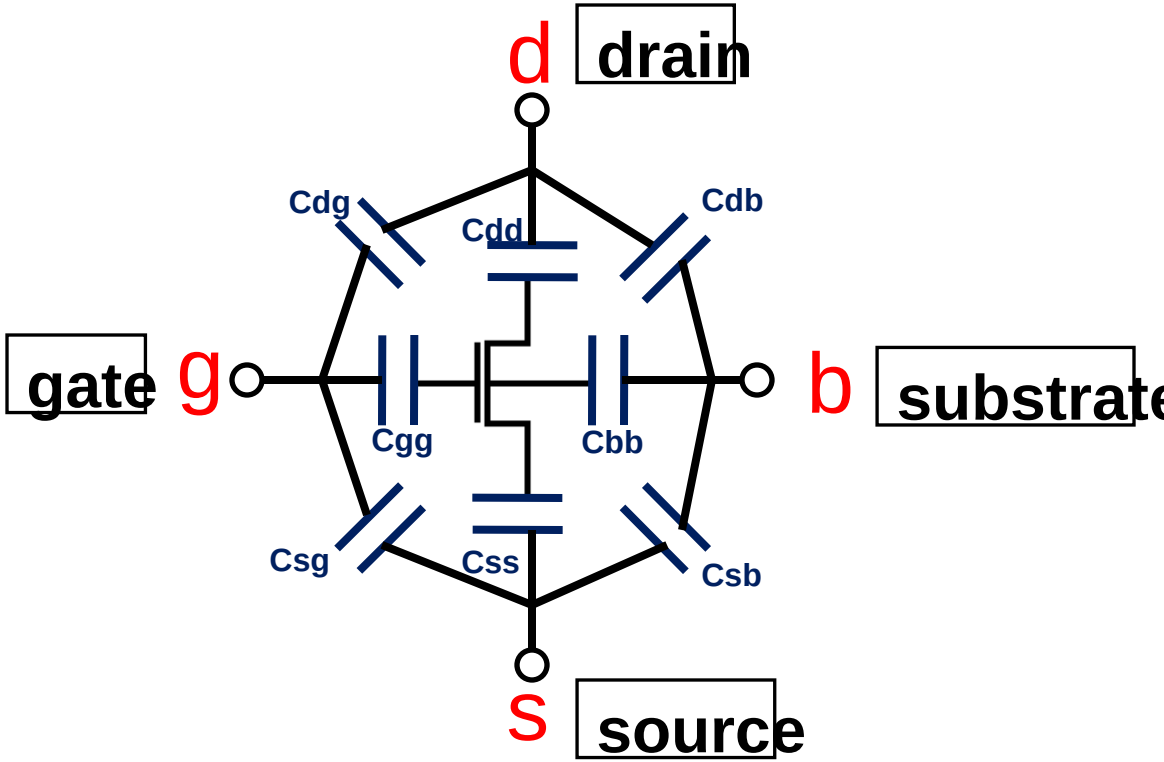
© 2007 Synopsys, Inc. (223)

# Example: NMOS Small-Signal AC Analysis

Project: */PostBasic/example/sdevice/ac*
Description: Calculate C-V characteristics.




© 2007 Synopsys, Inc. (224)

SYNOPSYS®
Predictable Success

# Device Section

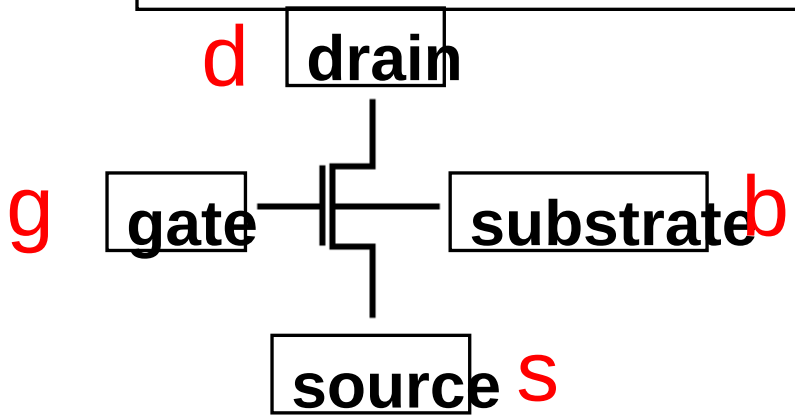
```
Device NMOS{
  Electrode {
    { name="source"    Voltage=0.0 }
    { name="drain"     Voltage=0.0 }
    { name="gate"      Voltage=0.0 }
    { name="substrate" Voltage=0.0 }
  }
  File{
     Grid          = "@tdr@"
     Plot          = "@tdrdat@"
     Current     = "@plot@"
     Param      = "@parameter@"
  }
  Physics{
    Mobility( DopingDep HighFieldSaturation Enormal )
    EffectiveIntrinsicDensity( oldSlotboom )
  }
}
```

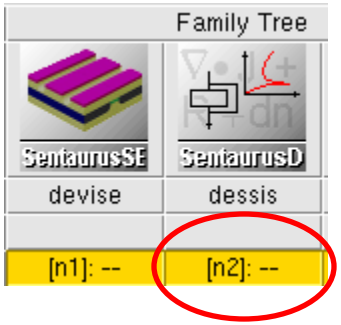
- The AC analysis needs to be done by the mixed-mode simulation. One or more Device sections are required to define the physical device(s).

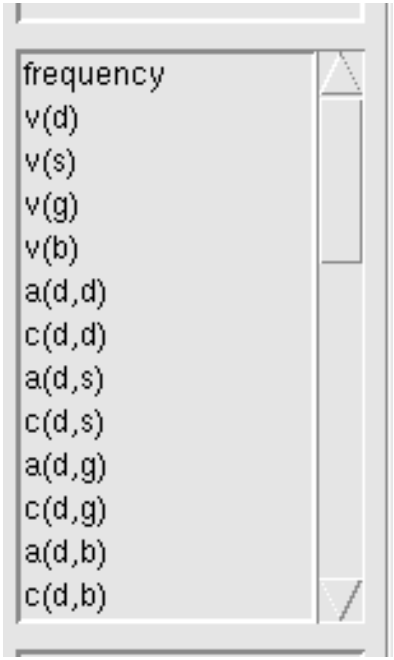
© 2007 Synopsys, Inc. (225)

# Global File Section

```
File{
    Output     = "@log@"
    ACExtract = "@acplot@"
}
```




- The file name for the AC analysis output (n2_ac_des.plt), is declared in the global File section.

- It contains the components of the conductance matrix A and the capacitance matrix C. $Y = A + j\omega C$, where $\omega = 2\pi f$.

© 2007 Synopsys, Inc. (226)

# System Section

```
System {
  NMOS nmos1 ("drain"=d "source"=s "gate"=g“ substrate"=b )
  Vsource_pset vd ( d 0 ){ dc = 0 }
  Vsource_pset vs ( s 0 ){ dc = 0 }
  Vsource_pset vg ( g 0 ){ dc = 0 }
  Vsource_pset vb ( b 0 ){ dc = 0 }
}
```

- For small-signal analysis simulations, an attached voltage source is required for each terminal if the terminal is to be included in the AC analysis.

© 2007 Synopsys, Inc. (227)

# Solve Section (1/3)

```
Solve{
  NewCurrentFile="init"
  Coupled(Iterations=100){ Poisson }
  Coupled{ Poisson Electron Hole }

  Quasistationary (
     InitialStep=0.1 Increment=1.3
     MaxStep=0.5 Minstep=1.e-5
     Goal { Parameter=vg.dc Voltage=-3}
  ){ Coupled { Poisson Electron Hole } }

  #-ramp gate
  NewCurrentFile=""
  Quasistationary (
     InitialStep=0.01 Increment=1.3
     MaxStep=0.05 Minstep=1.e-5
     Goal { Parameter=vg.dc Voltage=3}
  ){ ACCoupled (
       StartFrequency=1e6    EndFrequency=1e6    NumberOfPoints=1
       Decade
       Node(d s g b)    Exclude(vd vs vg vb)
       ACCompute (Time = (Range = (0 1)  Intervals = 20))
     ){ Poisson Electron Hole }
  }
}
```

© 2007 Synopsys, Inc. (228)

# Solve Section (2/3)

- The initial Quasistationary block biases the device to the starting gate voltage for the CV sweep.

- The keyword ACCoupled activates the small-signal analysis during the next Quasistationary sweep. In this example, the AC analysis is performed at a single frequency (1 MHz) only.

**AC Analysis Syntax**:

```
Solve { ...
        Quasistationary (
            …
        ){ ACCoupled (
                    StartFrequency=…    EndFrequency=…
                    NumberOfPoints=…  Decade or Linear
                    Node(node list)   Exclude(list of circuit element)
                    ACCompute (…)
      ){ … }
  }
}
```

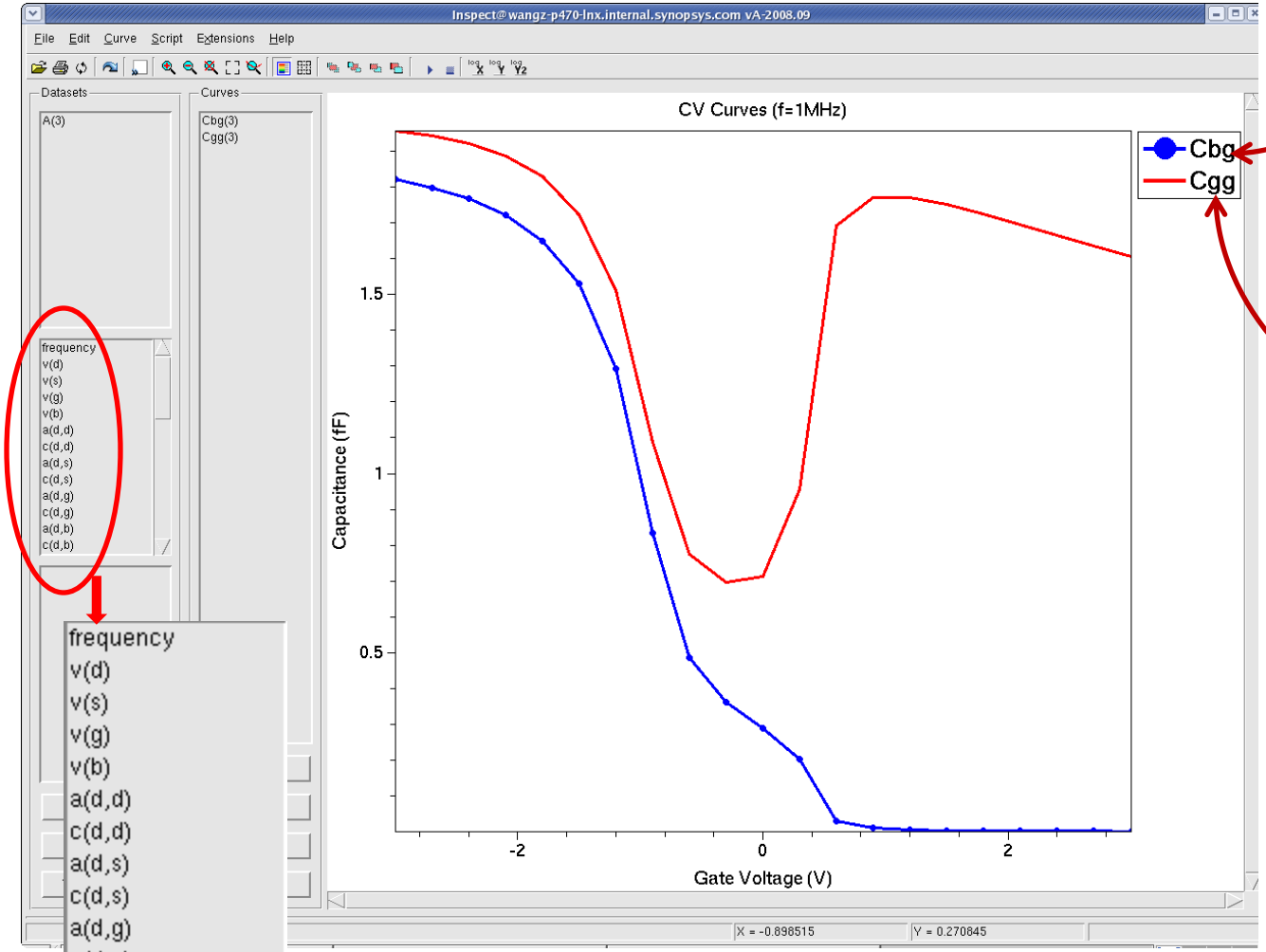
© 2007 Synopsys, Inc. (229)

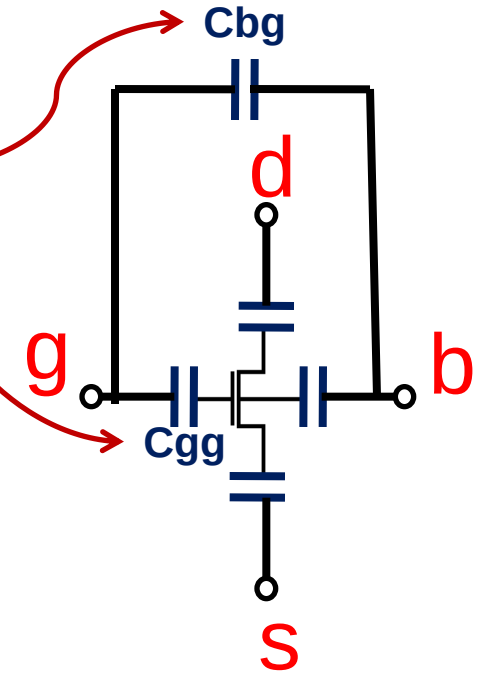
# Solve Section (3/3)

- The AC analysis is performed in a mixed-mode environment.
- The AC analysis computes the complex admittance Y matrix.
- The options StartFrequency, EndFrequency, NumberOfPoints, and Decade (or Linear) are used to select the frequencies at which the analysis is performed and the frequency distribution.
- A Node list must be given, which nodes are to be included in the **Y**-matrix calculation. With it, for each frequency, the compact equivalent small signal model is generated between the given nodes.
- The Exclude list is used to remove a set of circuit or physical devices from the AC analysis.
- The (optional) ACCompute statement enforces that the AC analysis is performed on a set of predefined bias points.

© 2007 Synopsys, Inc. (230)

# NMOS C-V Characteristic




Cbg=c(b.g)

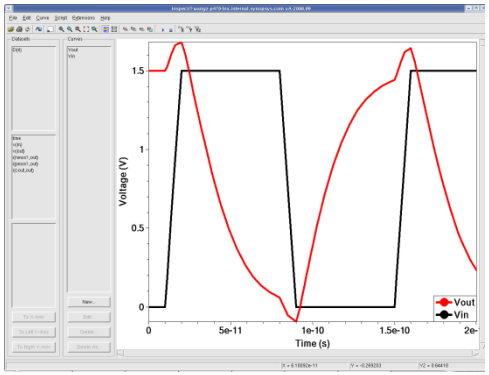
Cgg=c(g,g)

© 2007 Synopsys, Inc. (231)

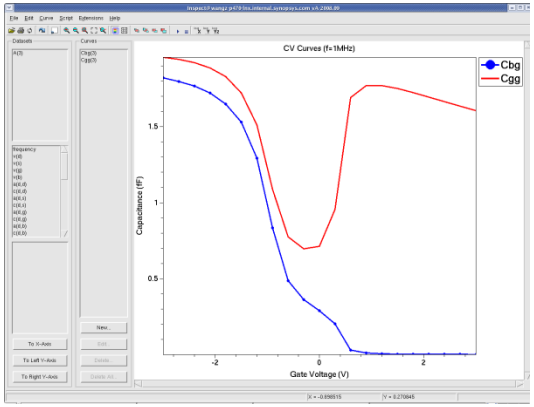
# Ex 6. Observe Transient and AC's Results

- Observe the following projects' results.
  - ➢*/PostBasic/example/sdevice/cmos_inverter_transient*

time
v(in)
v(out)
i(nmos1,out)
i(pmos1,out)
i(cout,out)




  - ➢*/PostBasic/example/sdevice/ac*




frequency
v(d)
v(s)
v(g)
v(b)
a(d,d)
c(d,d)
a(d,s)
c(d,s)
a(d,g)
c(d,g)
a(d,b)
c(d,b)

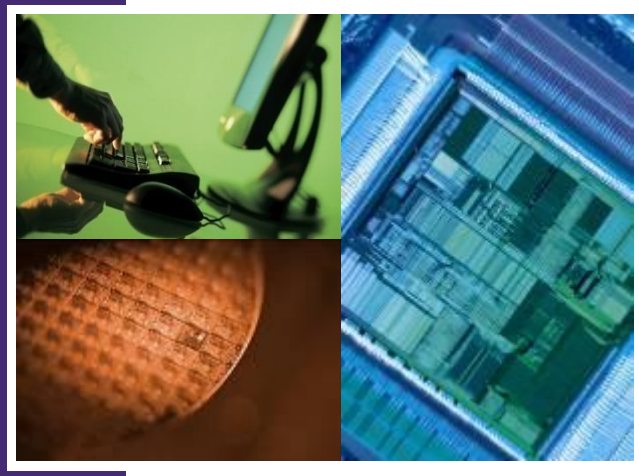
© 2007 Synopsys, Inc. (232)

# Quasistationary, Transient and AC Analysis

| | Quasistationary | Transient | AC (Y-parameters) |
|---|---|---|---|
| **Input Signals** | DC | Time-dependent sources | Small-signal AC sources |
| **Simulation Domain** | ----- | Time | Frequency |
| **Need Quasistationary Analysis?** | ----- | Yes | Yes |
| **Mixed-Mode Necessary?** | No | No | Yes |
| **“t” Variable** | “t” is the internal variable. (0->1) | “t “ is the real time [s]. | ----- |
| **Applications** | DC operating point ,DC sweep | Transient response | C-V, FT |

© 2007 Synopsys, Inc. (233)

# Inspect



SYNOPSYS®

Predictable Success

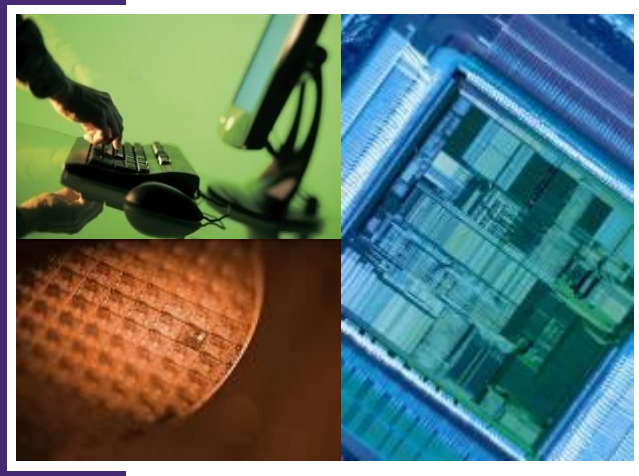
# Outline

- GUI interactive mode
  - Load curve file, plot curves
  - Edit Plot area, Axes
  - Create new curve
  - Formula library - refer to manual 2007.12 P.25,48
  - Macros - refer to manual 2007.12 P.26,50
  - Save, Export
- Script file batch mode
  - Record each operation step on interactive mode to generate a script file.
  - Script a command file – refer to example 0.18um nMOS
  - Extract standard parameters of the extraction library

© 2007 Synopsys, Inc. (235)

# Inspect Script Basic

Version A-2008.09
Rev0.0



SYNOPSYS®

Predictable Success

# Agenda

- Basic Operations and Script Commands
  - Load plt File: proj_load
  - Create Curve : cv_createDS
  - Set Attributes: gr_setAxisAttr; cv_setCurveAttr
  - Extract Variables : cv_compute; ft_scalar
- Mathematical Formulas and Macros
  - Mathematical Formulas
  - Macros
- Threshold Voltage Extraction
  - Definitions of Threshold Voltage
  - Inspect Script

© 2007 Synopsys, Inc. (237)

# Inspect Script

- Inspect can be controlled by using a simple scrip language. For example, a script can load a project (data file), draw curves, and perform mathematical computations on curves.

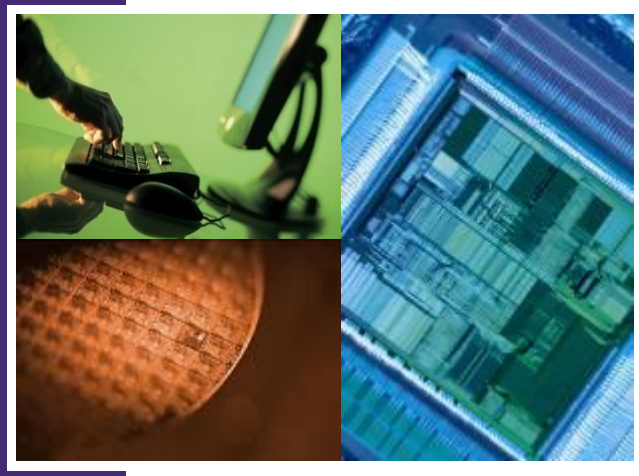
- A script can be written manually or created automatically by recording actions performed interactively through the graphical user interface.

- Inspect script can extract variables to SWB family table.

| | Family Tree | | | | | Variable Values | | |
|---|---|---|---|---|---|---|---|---|
| |  SentaurusP | |  SentaurusSE |  SentaurusD |  Inspect | | | |
| | sprocess | | sde | IdVg | rolloff | | | |
| | | Lgshift | | | | Lg | vt | vt1 |
| 1 | -- | 0.0 | -- | -- | -- | 0.065 | 0.316 | 0.183 |
| 2 | | 0.005 | -- | -- | -- | 0.075 | 0.340 | 0.215 |
| 3 | | 0.01 | -- | -- | -- | 0.085 | 0.327 | 0.206 |
| 4 | | 0.0175 | -- | -- | -- | 0.100 | 0.315 | 0.199 |
| 5 | | 0.0325 | -- | -- | -- | 0.130 | 0.298 | 0.188 |
| 6 | | 0.0925 | -- | -- | -- | 0.250 | 0.257 | 0.158 |

© 2007 Synopsys, Inc. (238)

# Basic Operations and Script Commands



SYNOPSYS®

Predictable Success

# Basic Operations and Script Commands

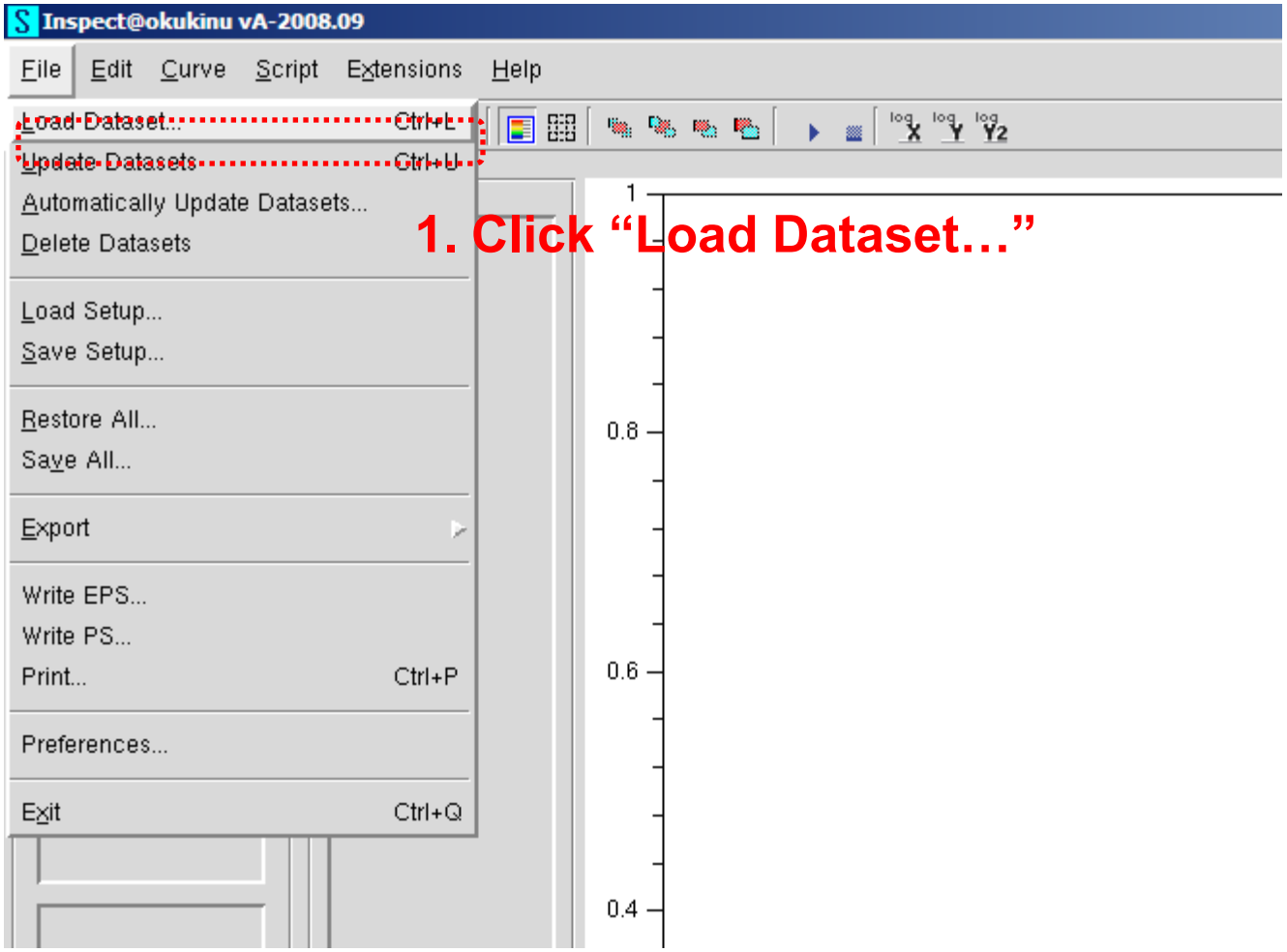
- The basic operations and script commands are explained:
  - Load plt File: proj_load
  - Create Curves : cv_createDS
  - Set Attributes: gr_setAxisAttr; cv_setCurveAttr
  - Extract Variables : cv_compute; ft_scalar

- A backslash ‘\’ is used to extend a command to multiple

  lines if it appears as the last character on the line.

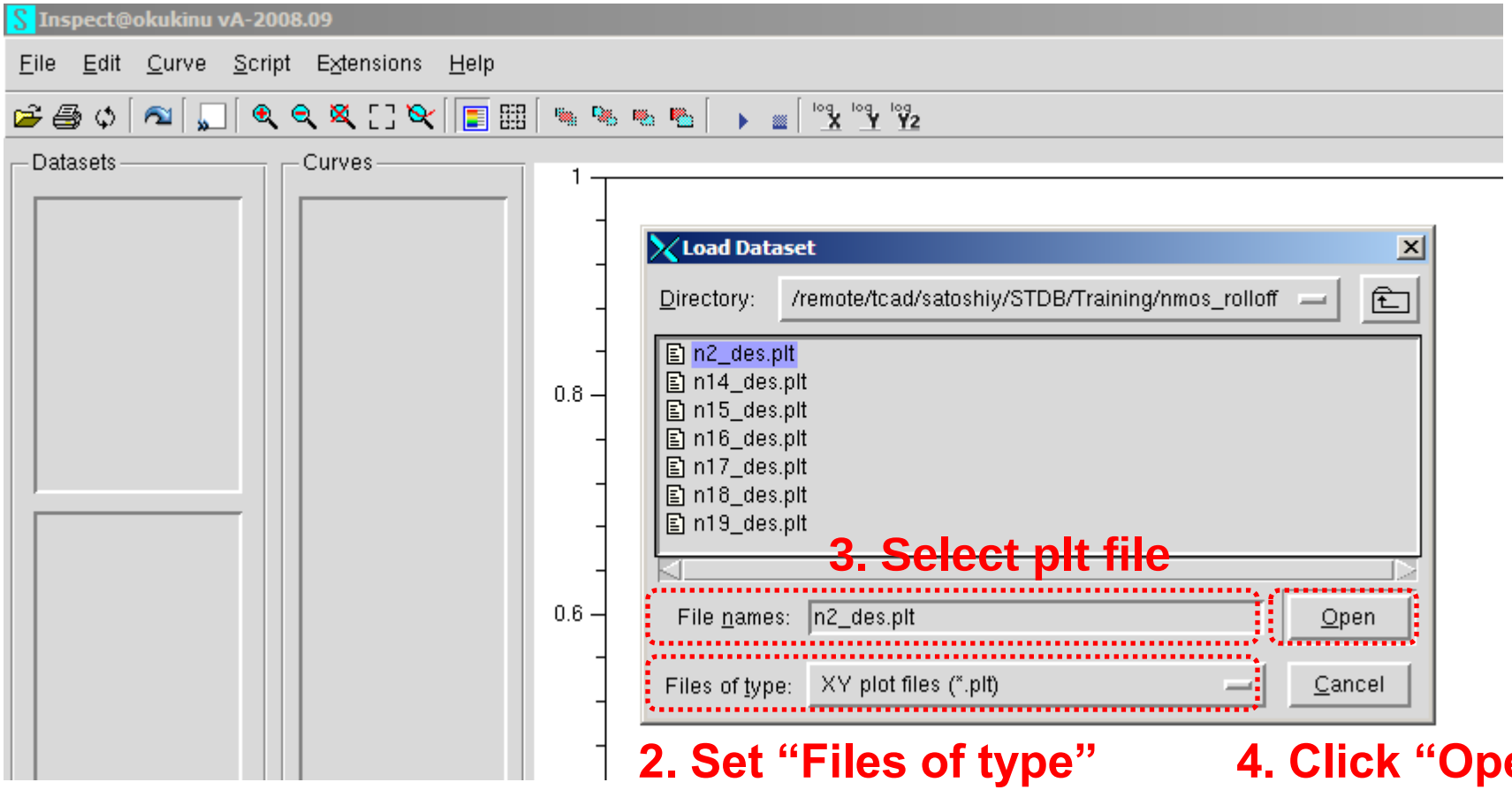
© 2007 Synopsys, Inc. (240)

# Load plt File ~ Operation ~ (1/3)




© 2007 Synopsys, Inc. (241)

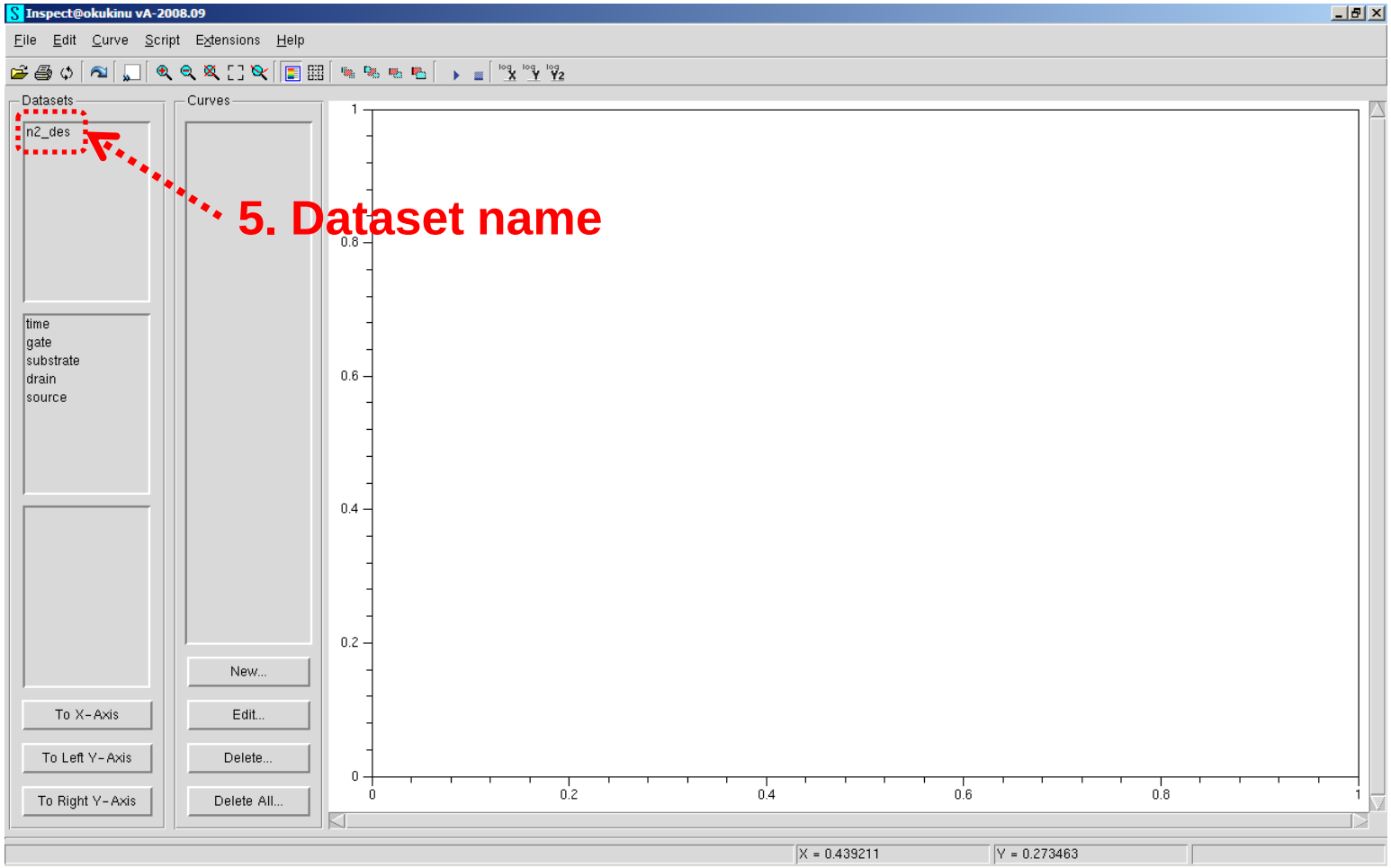
# Load plt File ~ Operation ~ (2/3)




© 2007 Synopsys, Inc. (242)

SYNOPSYS®
Predictable Success

# Load plt File ~ Operation ~ (3/3)




© 2007 Synopsys, Inc. (243)

# Load plt File ~ Command ~

- Load plt file

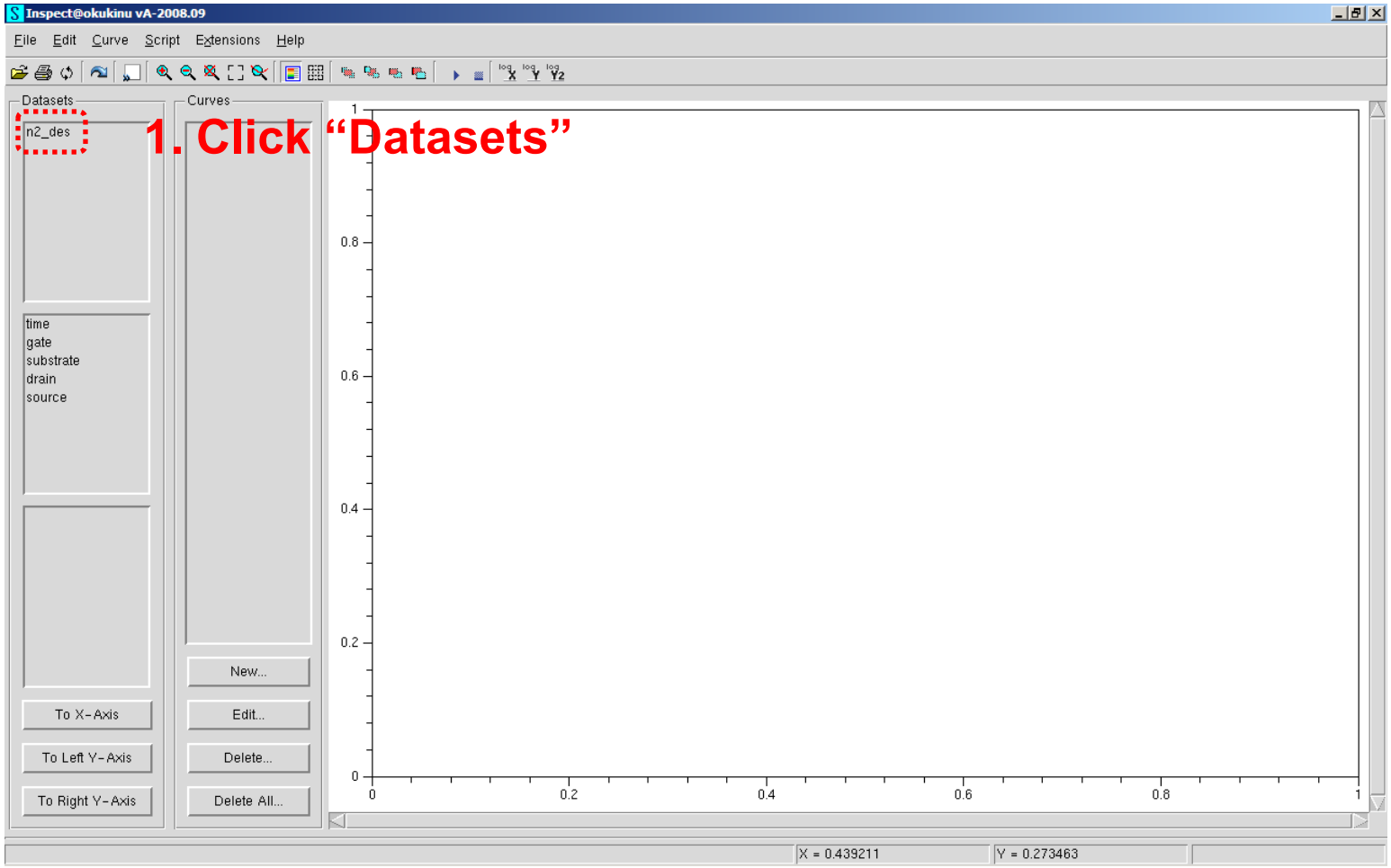
```
proj_load n2_des.plt n2_des
```

**3.** **5.**

  - n2_des.plt **:** the name of the file to load.
  - n2_des **:** the arbitrary name user can define for this dataset.

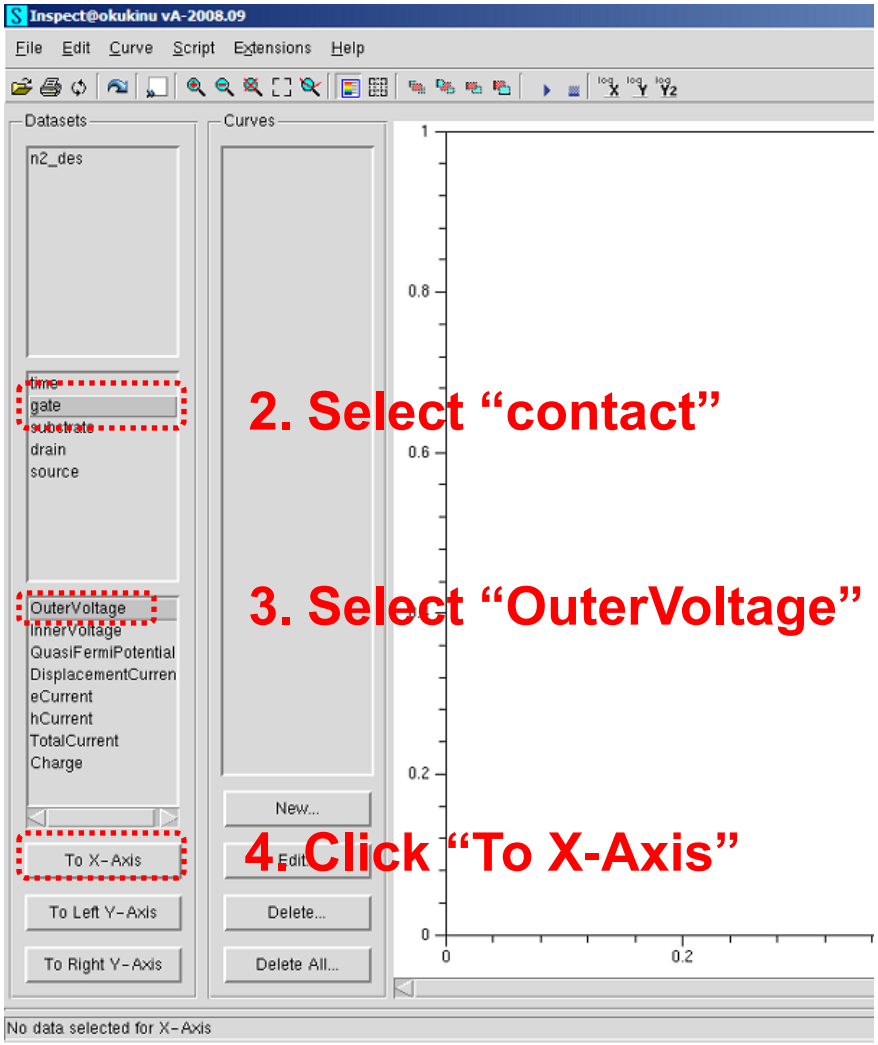
© 2007 Synopsys, Inc. (244)

# Create Curve ~ Operation ~ (1/4)



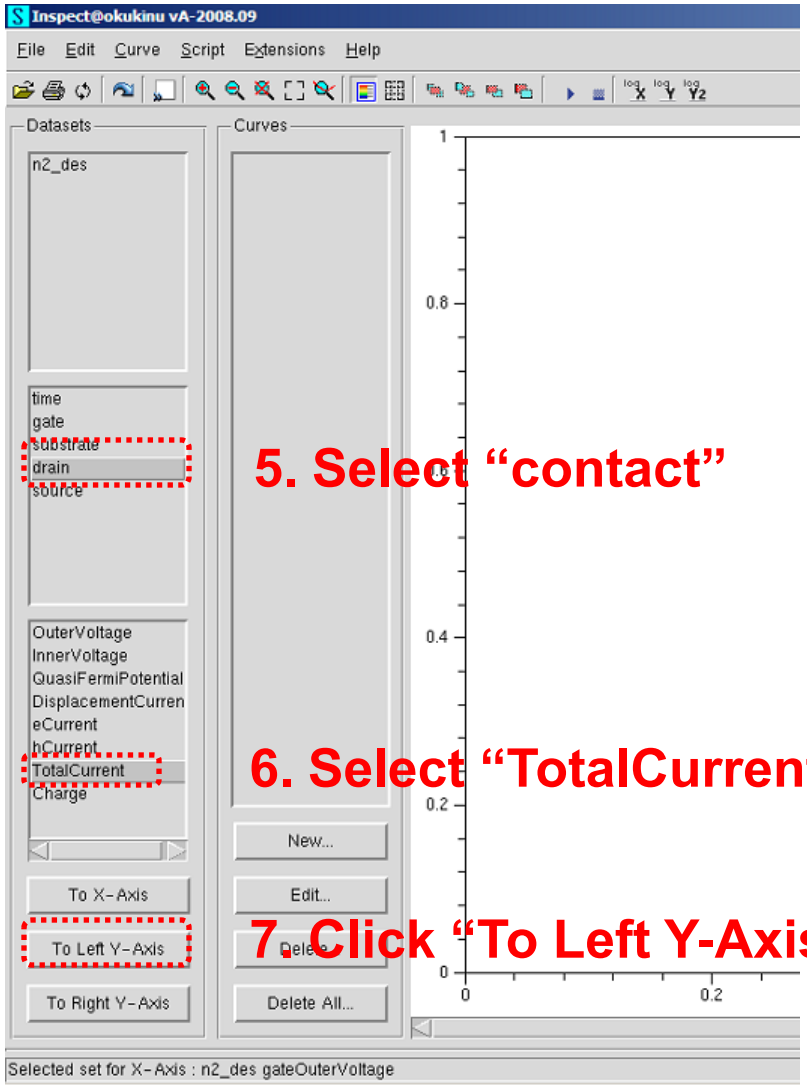

© 2007 Synopsys, Inc. (245)

# Create Curve ~ Operation ~ (2/4)




© 2007 Synopsys, Inc. (246)

# Create Curve ~ Operation ~ (3/4)



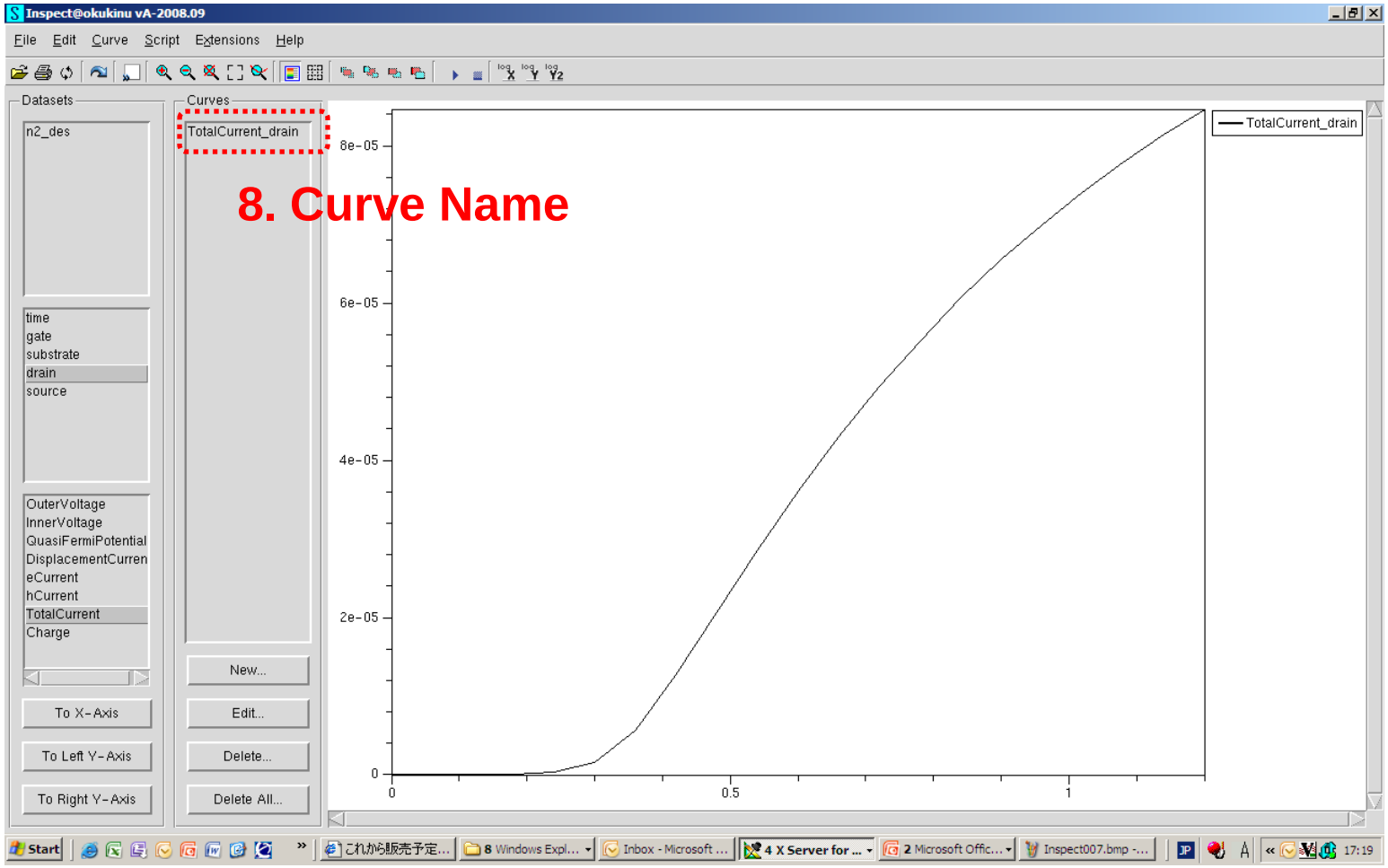

5. Select “contact”

6. Select “TotalCurrent”

7. Click “To Left Y-Axis”

© 2007 Synopsys, Inc. (247)

# Create Curve ~ Operation ~ (4/4)




© 2007 Synopsys, Inc. (248)

# Create Curve ~ Command ~

```
cv_createDS TotalCurrent_Drain \
\                    8.
   {n2_des gate OuterVoltage} \
\     1.     2.       4.
   {n2_des drain TotalCurrent} y
     1.     5.       6.          7.
```

- TotalCurrent_Drain **:** a unique name for the new curve.
- {n2_des gate OuterVoltage} **:** a list of data to use for the x-dataset.
- {n2_des drain TotalCurrent} **:** a list of data to use for the y-dataset.
- y **:** optional parameter specifying the axis to use; the default is y; the options are y or y2.

© 2007 Synopsys, Inc. (249)

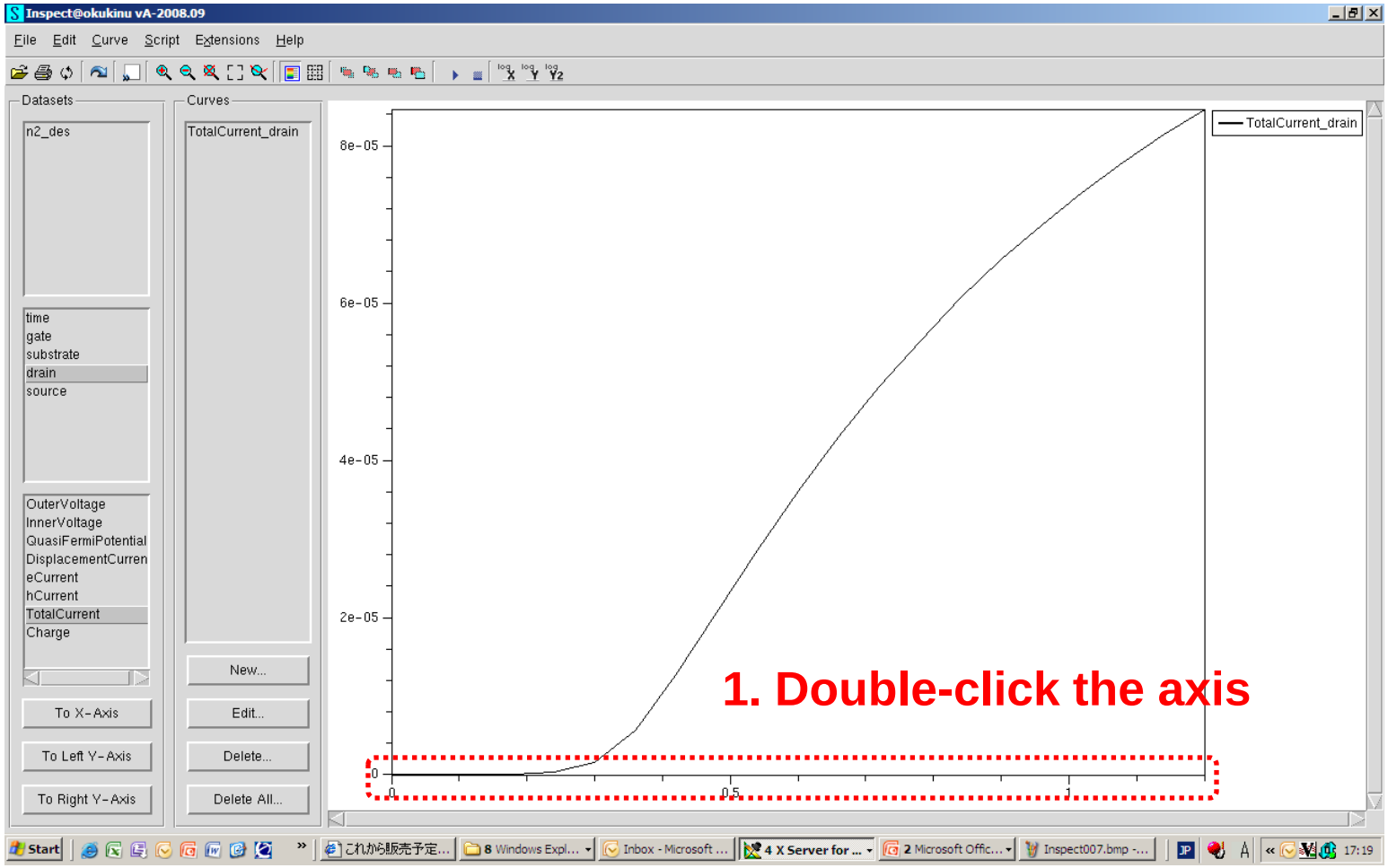
# Set Attributes

- This section describes the functions that change the attributes of the axes, curves, and legend.

© 2007 Synopsys, Inc. (250)

# Set the Axis Attributes

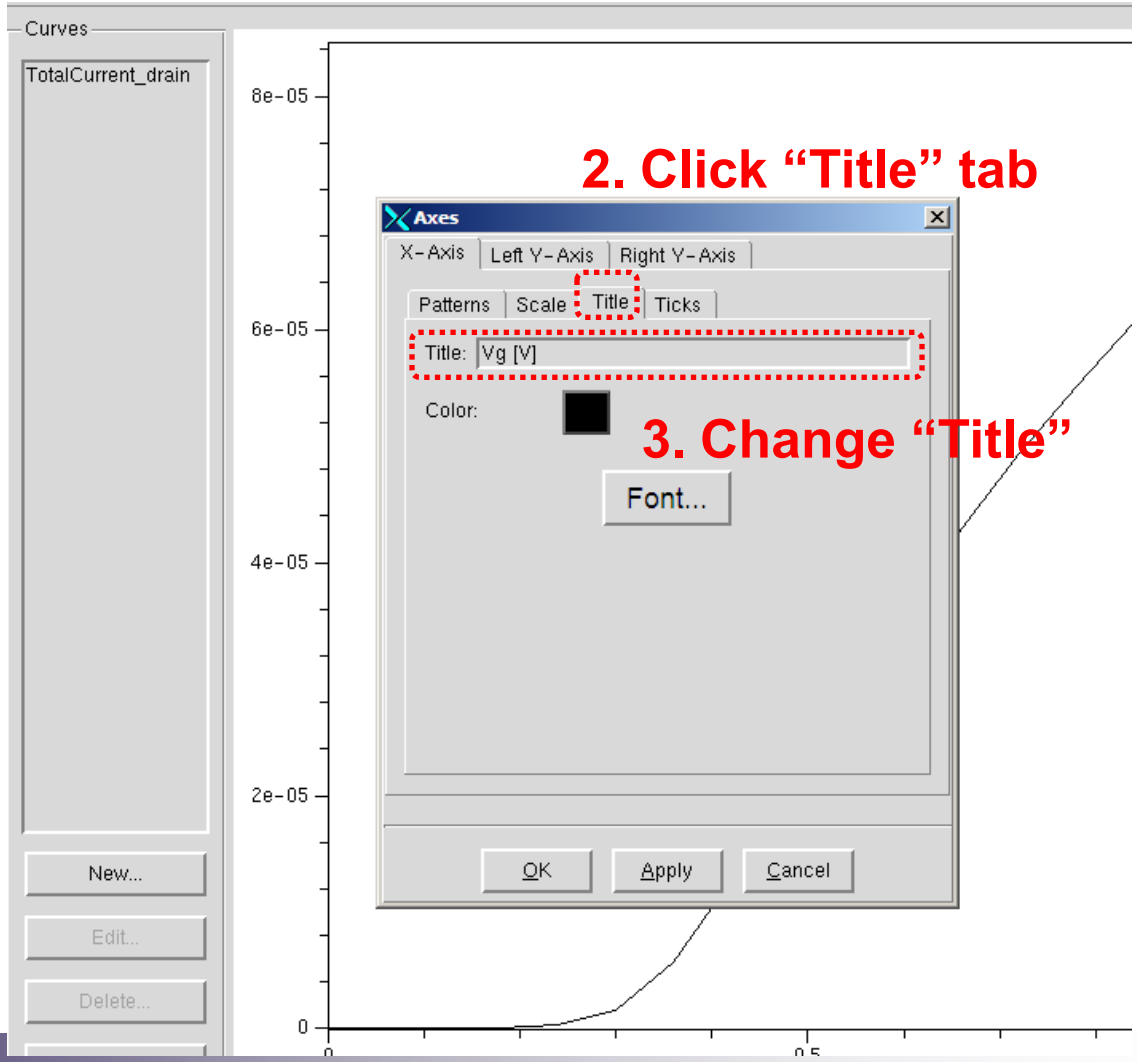
## ~ Operation : Start ~




© 2007 Synopsys, Inc. (251)

SYNOPSYS®
Predictable Success

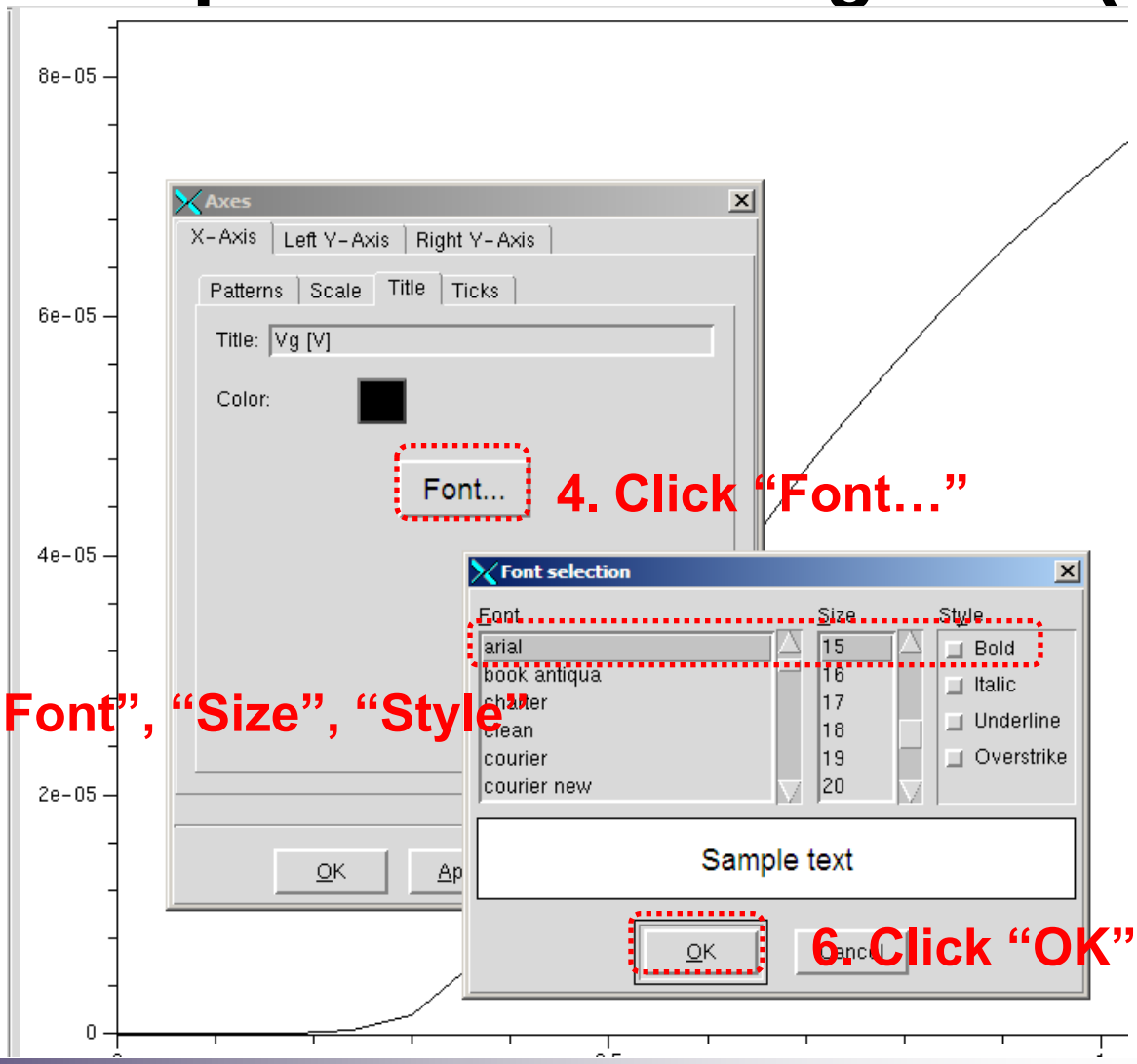
# Set the Axis Attributes
## ~ Operation : Editing Title (1/2) ~




© 2007 Synopsys, Inc. (252)

SYNOPSYS®
Predictable Success

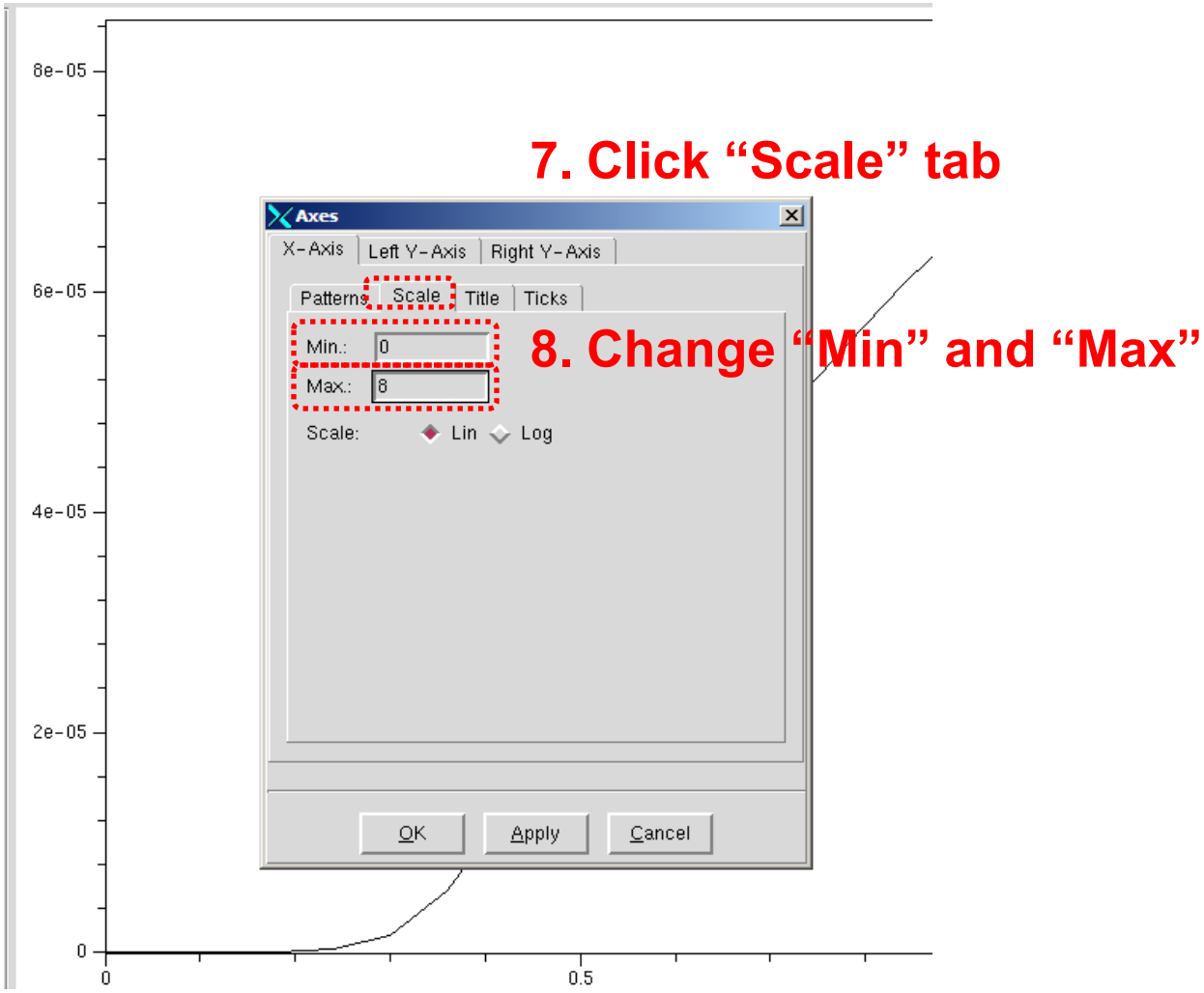
# Set the Axis Attributes
## ~ Operation : Editing Title (2/2) ~




© 2007 Synopsys, Inc. (253)

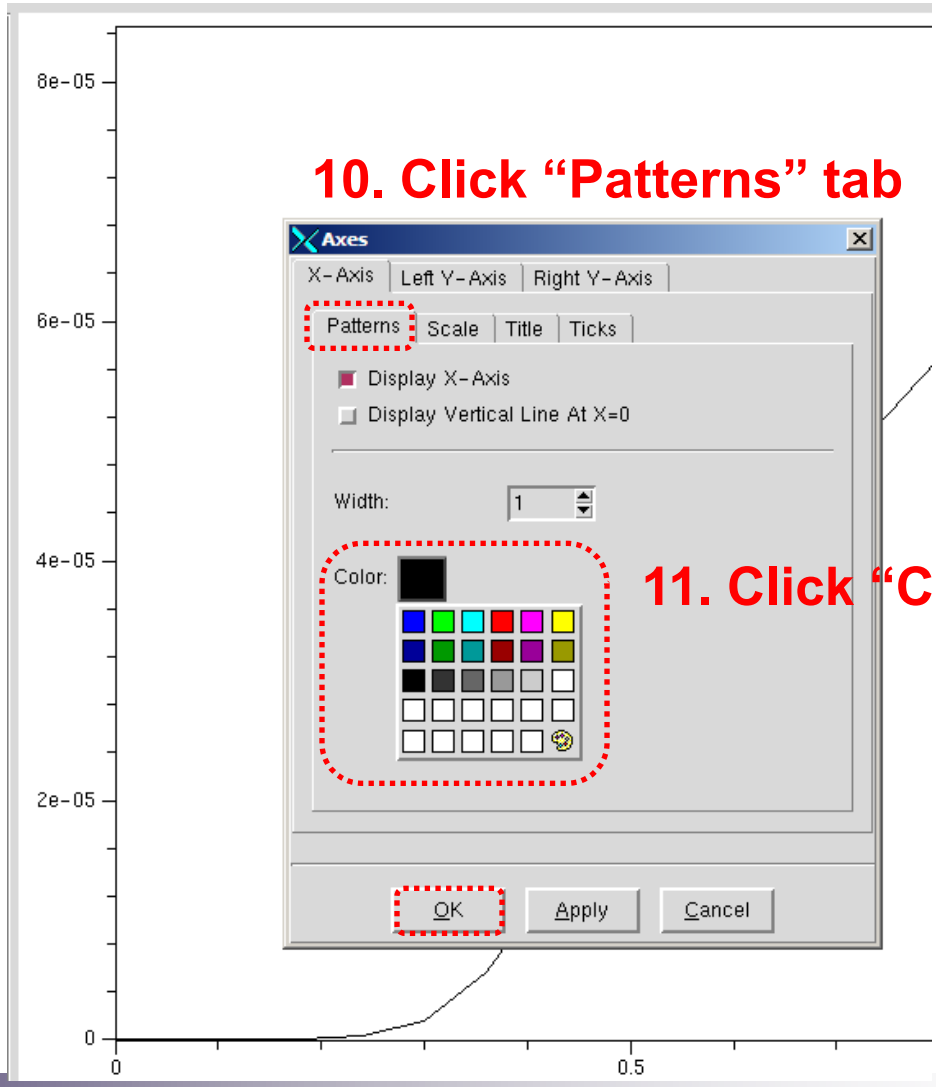
SYNOPSYS®
Predictable Success

# Set the Axis Attributes
## ~ Operation : Editing Scale~




© 2007 Synopsys, Inc. (254)

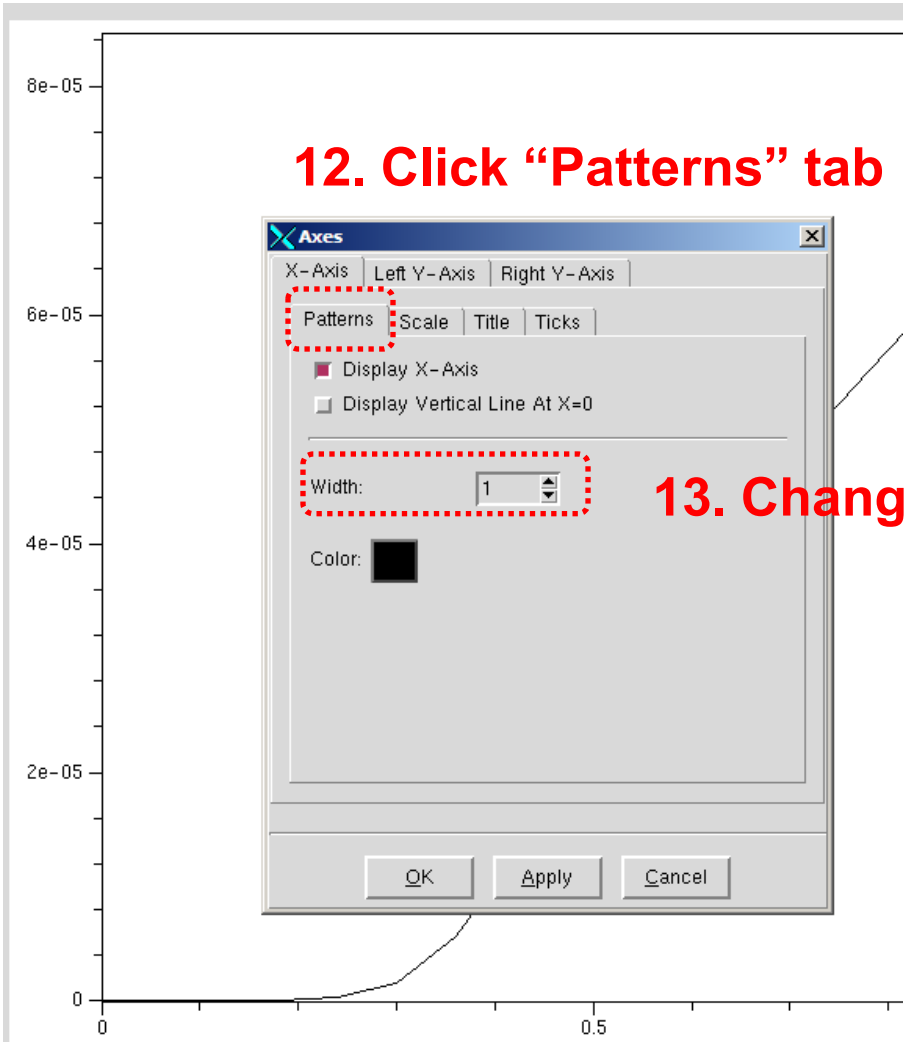
# Set the Axis Attributes
## ~ Operation : Editing Pattern (1/2) ~




© 2007 Synopsys, Inc. (255)

SYNOPSYS®
Predictable Success

# Set the Axis Attributes
## ~ Operation : Editing Pattern (2/2) ~




© 2007 Synopsys, Inc. (256)

# Set the Axis Attributes ~ Command ~

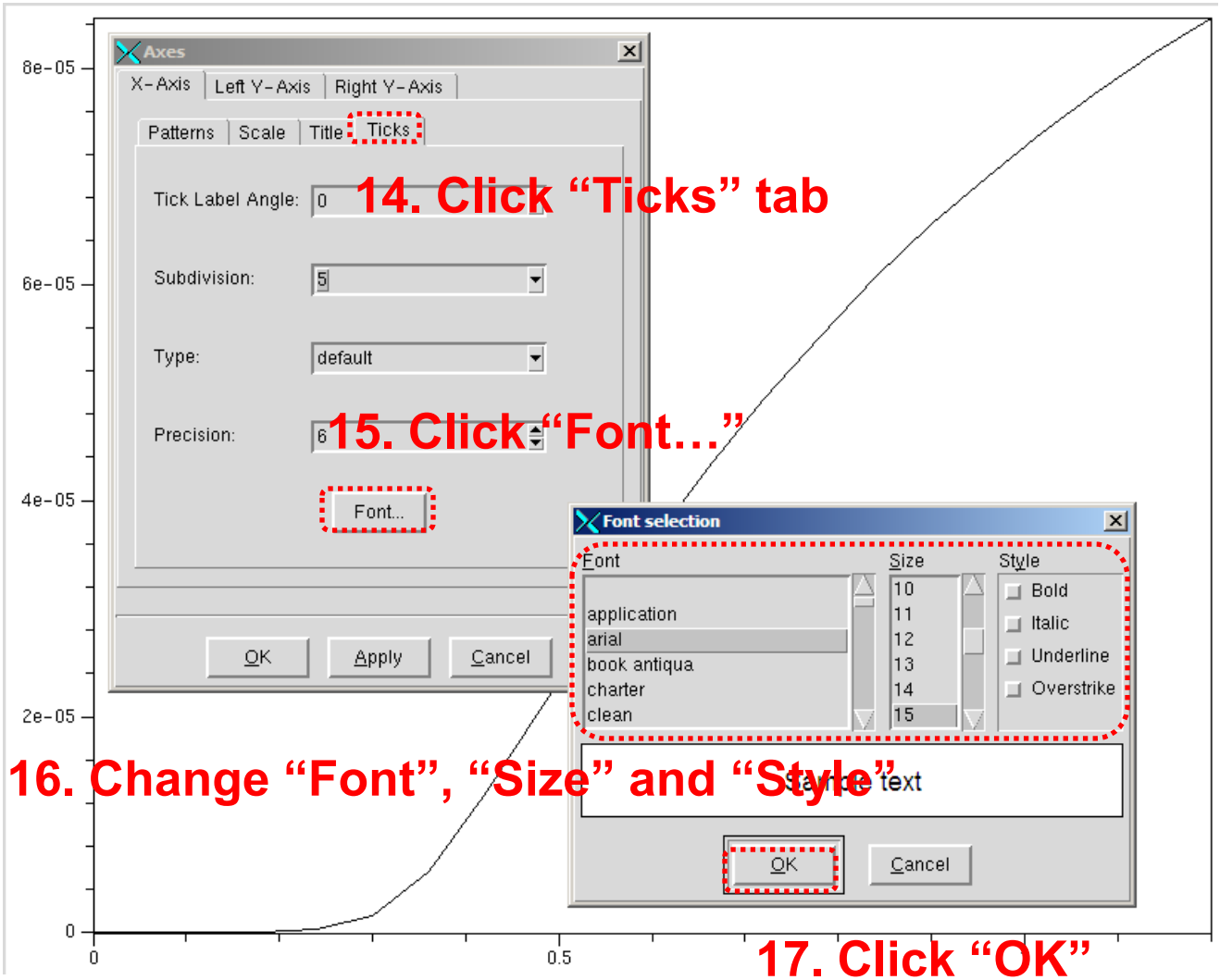
```
gr_setAxisAttr X {Vg [V]} {arial 15} 0 8 black 1 \
\                 1.    3.        5.      8.   11. 13.
      {arial 15} 0 5 0 black
```

- X : a keyword (X, Y, or Y2) specifying an axis.
- Vg [V] : the axis title.
- {arial 15} : the font size of the axis title.
- 0 8 : minimal and maximal values of the axis.
- black : the color of the axis.
- 1 : the width of the axis line.

© 2007 Synopsys, Inc. (257)

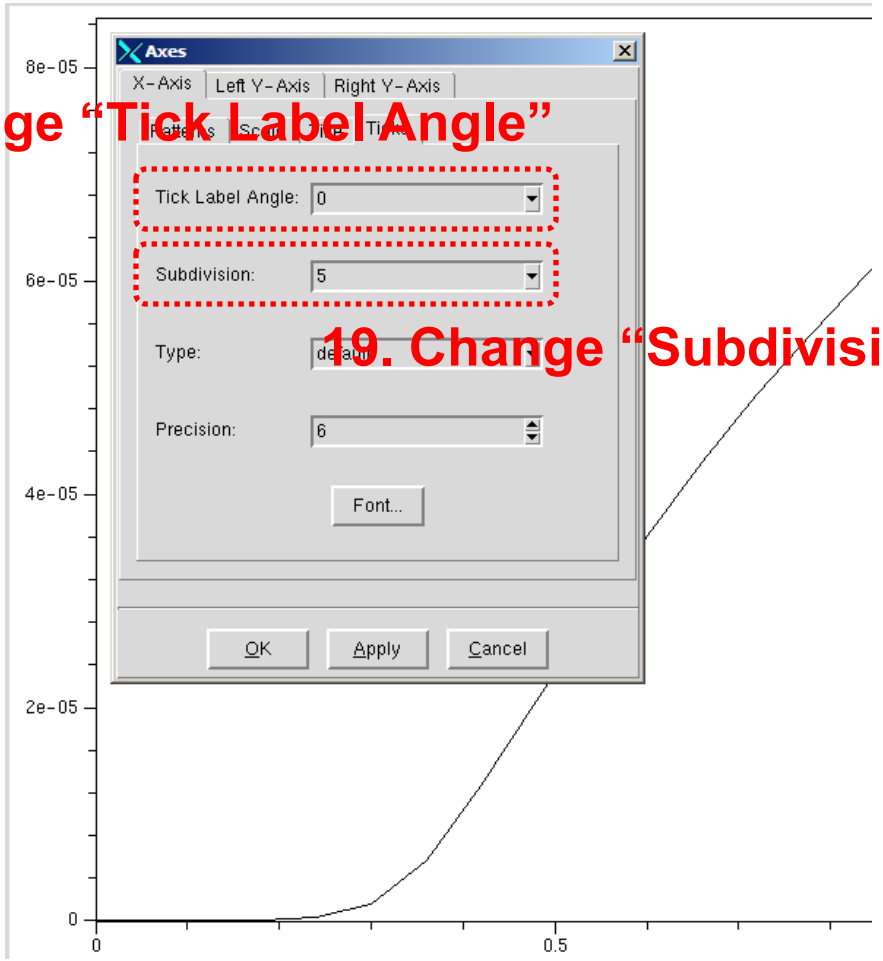
# Set the Axis Attributes

## ~ Operation : Editing Ticks (1/2) ~




© 2007 Synopsys, Inc. (258)

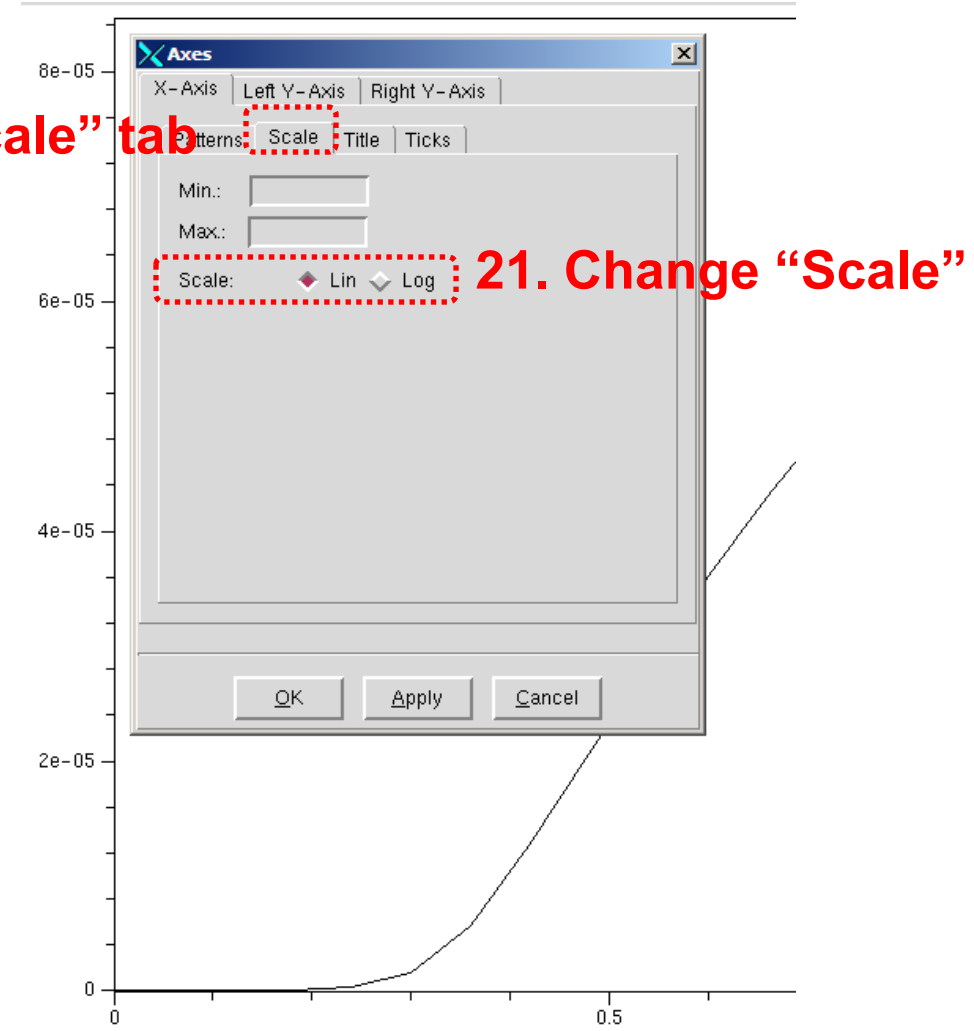
# Set the Axis Attributes
## ~ Operation : Editing Ticks (2/2) ~




© 2007 Synopsys, Inc. (259)

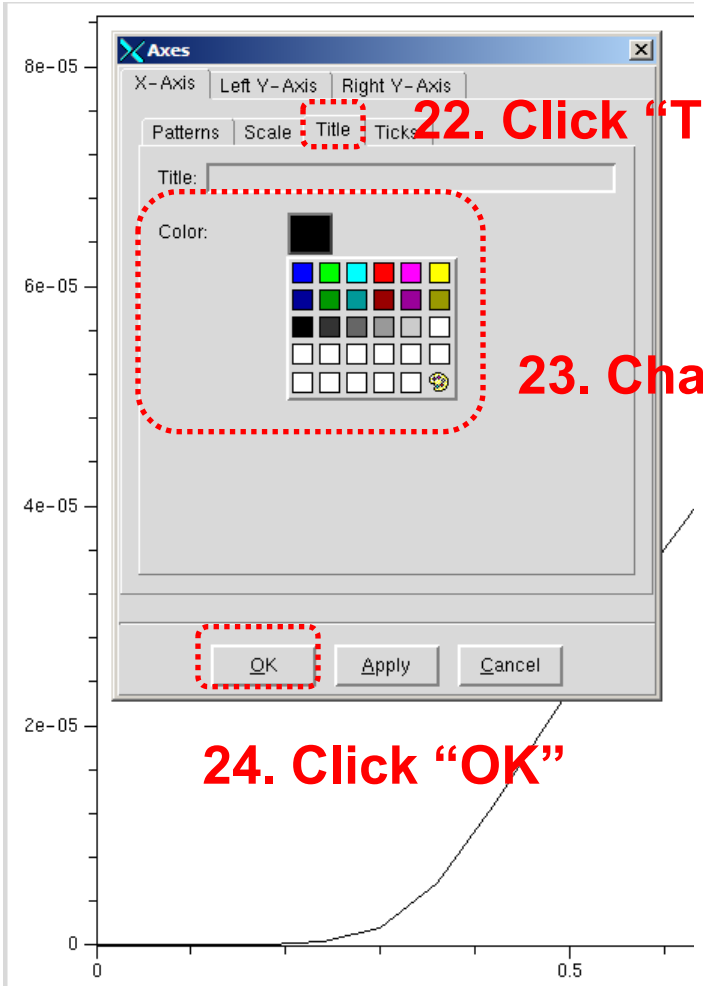
# Set the Axis Attributes

## ~ Operation : Changing Log/Lin ~




© 2007 Synopsys, Inc. (260)

# Set the Axis Attributes
## ~ Operation : Editing Font of Title ~




© 2007 Synopsys, Inc. (261)

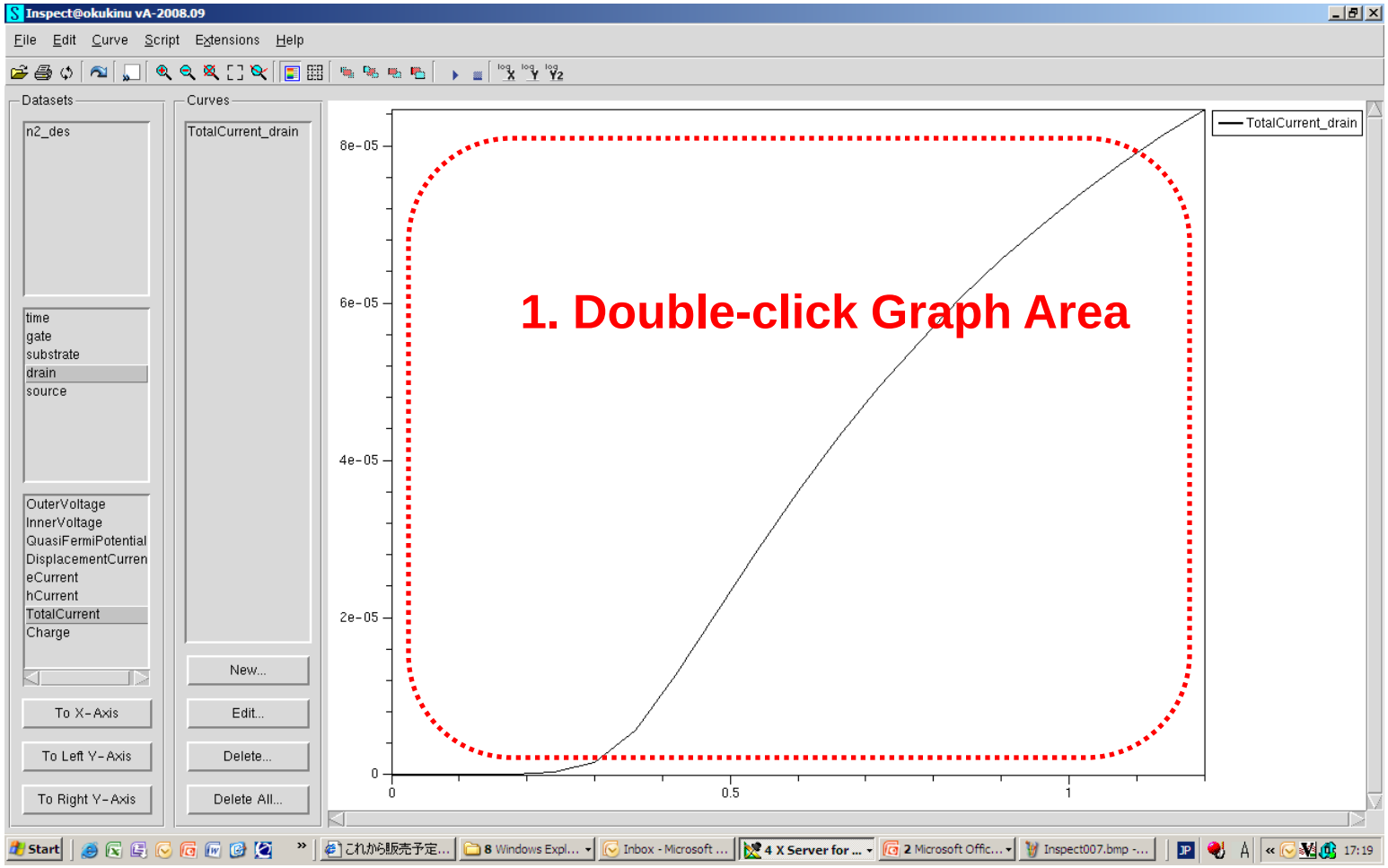
# Set the Axis Attributes ~ Command ~

```
gr_setAxisAttr X {Vg [V]} {arial 15} 0 8 black 1 \
\
      {arial 15}  0  5  0 black
           16.   18.  19. 20.  23.
```

- X : a keyword (X, Y, or Y2) specifying an axis
- {arial 15} :the font size of the tick label
- 0 : the angle at which the tick labels are drawn
- 5 **:** the number of secondary ticks between the main ticks
- black : the color of the axis title

© 2007 Synopsys, Inc. (262)

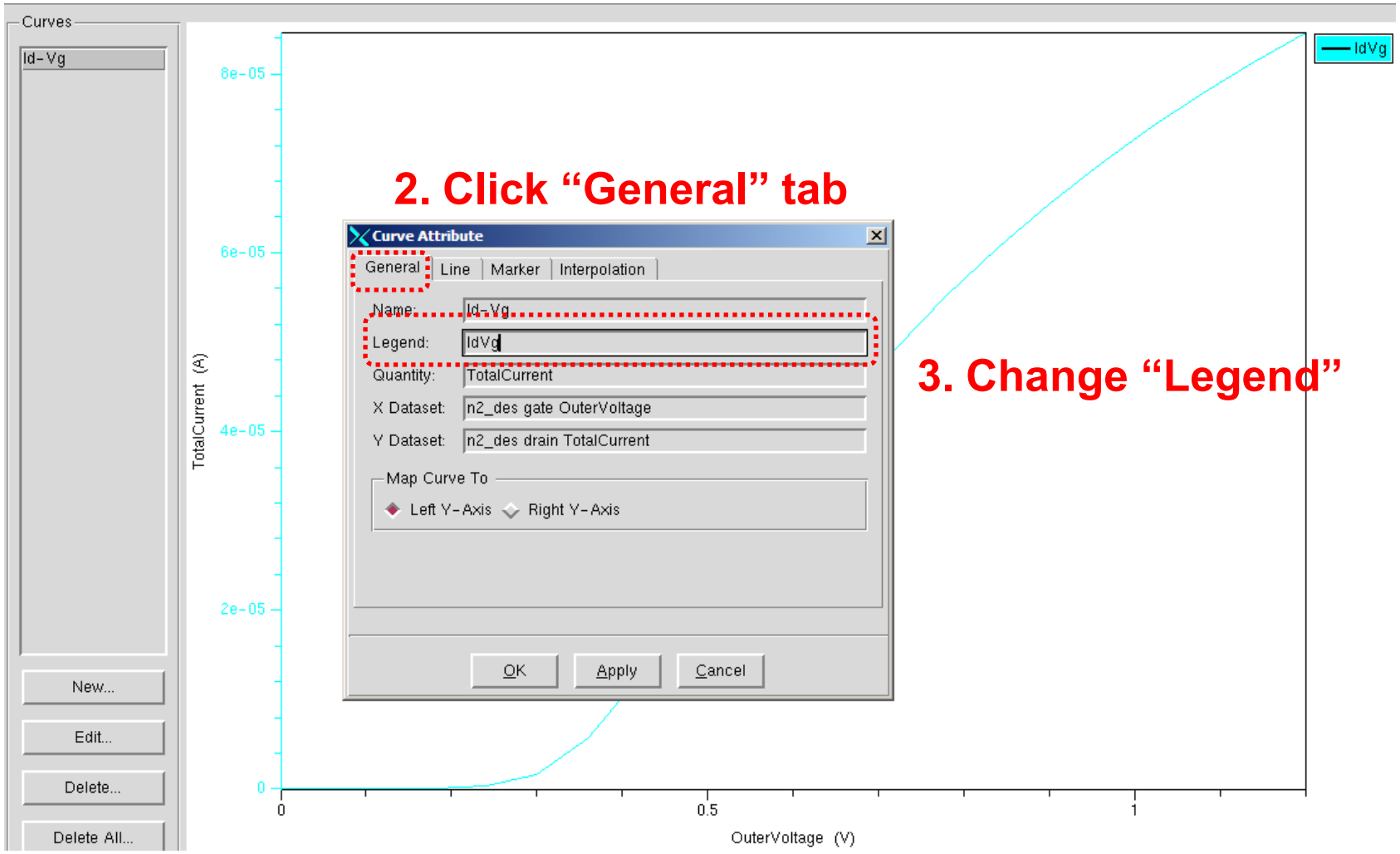
# Set Curve Drawing Attributes
## ~ Operation : Start ~




© 2007 Synopsys, Inc. (263)

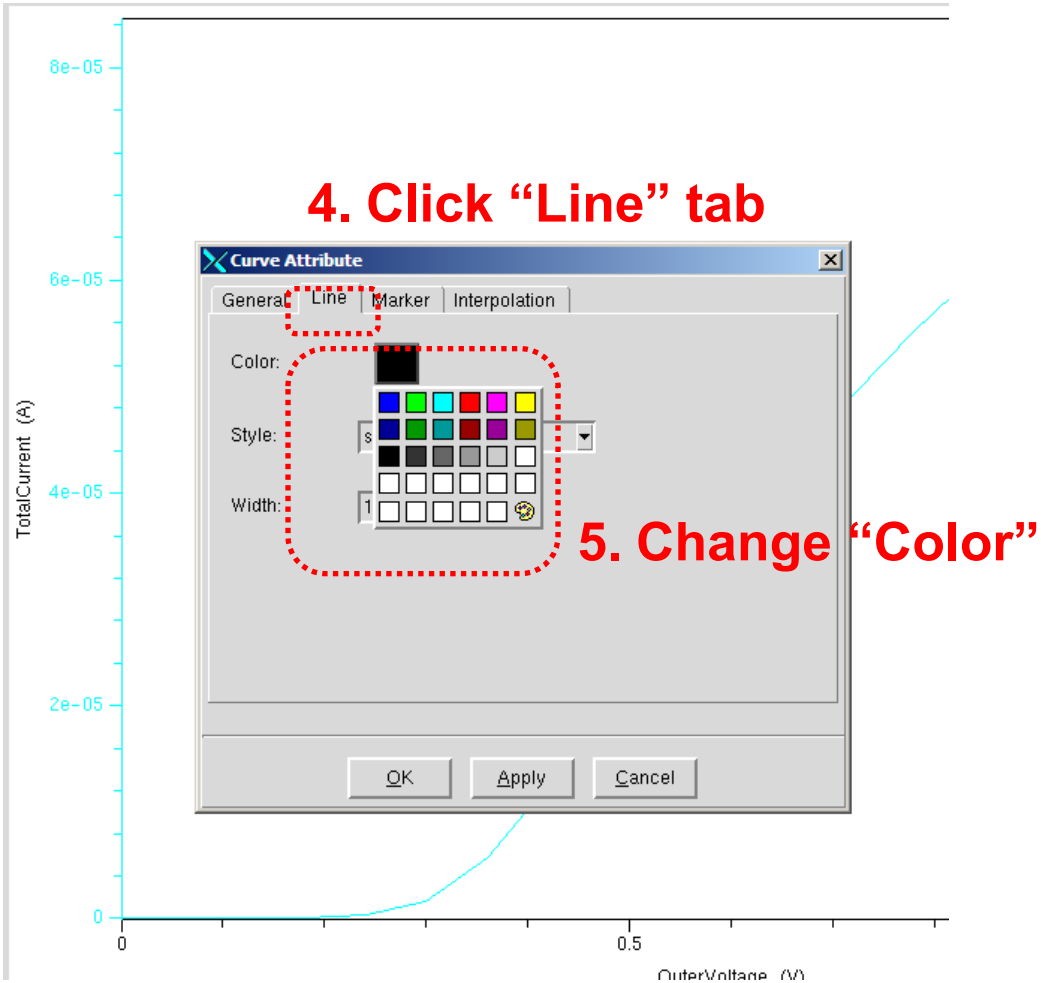
SYNOPSYS®
Predictable Success

# Set Curve Drawing Attributes
## ~ Operation : Editing General Part ~



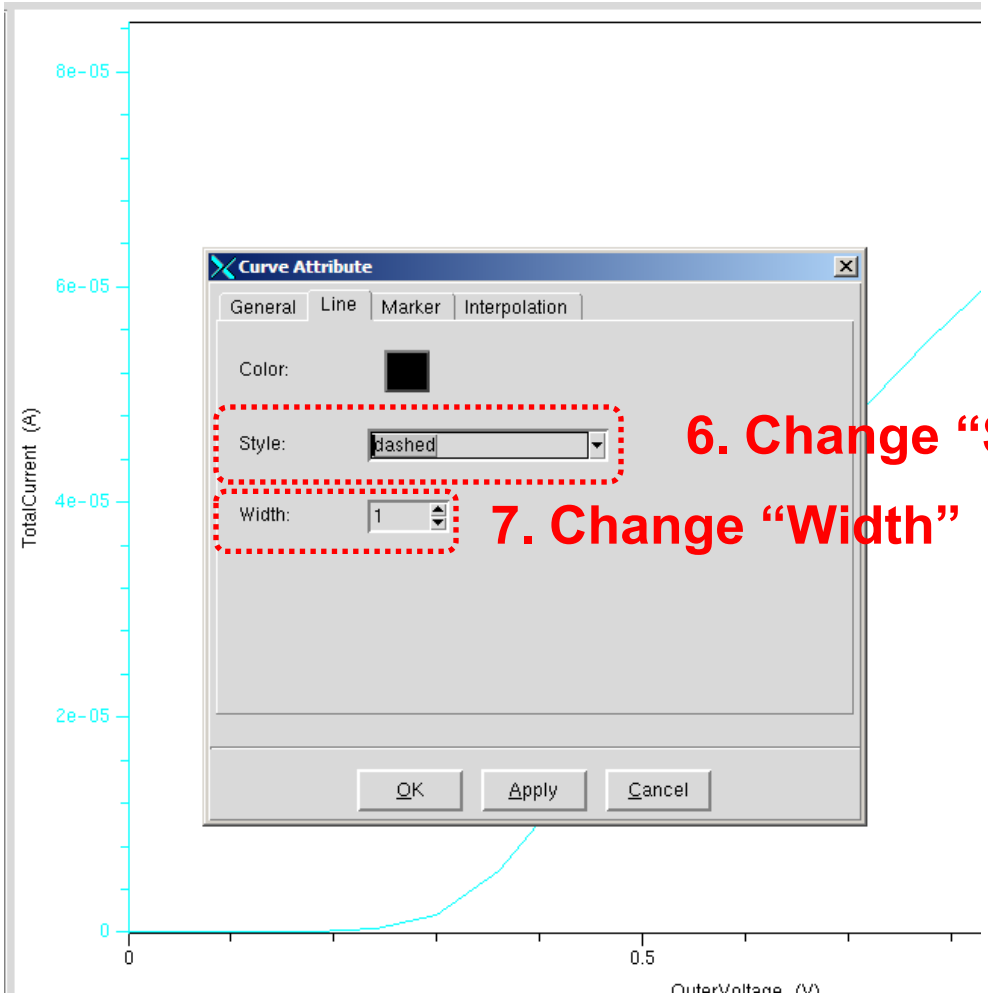

© 2007 Synopsys, Inc. (264)

# Set Curve Drawing Attributes
## ~ Operation : Editing Line Part (1/2)~




© 2007 Synopsys, Inc. (265)

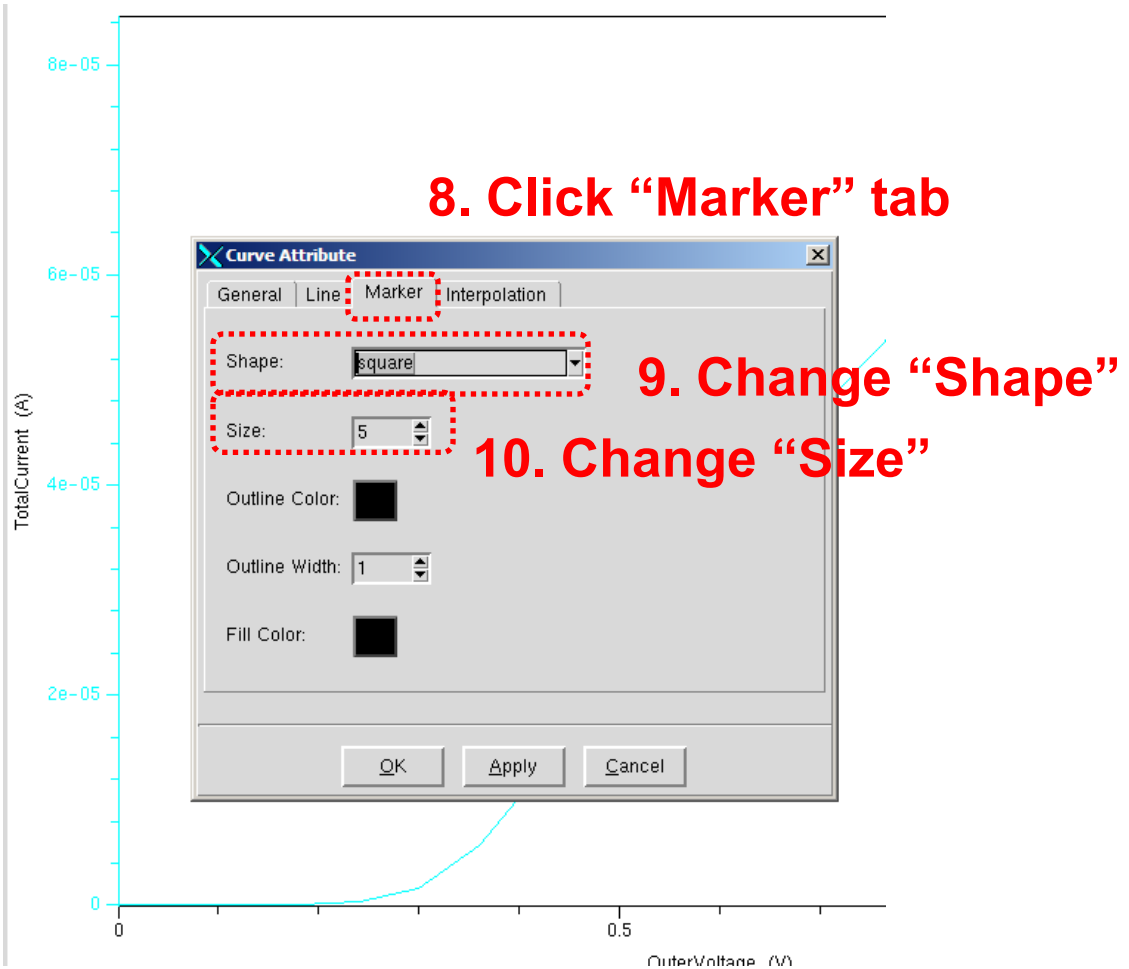
# Set Curve Drawing Attributes
## ~Operation : Editing Line Part (2/2)~




© 2007 Synopsys, Inc. (266)

SYNOPSYS®
Predictable Success

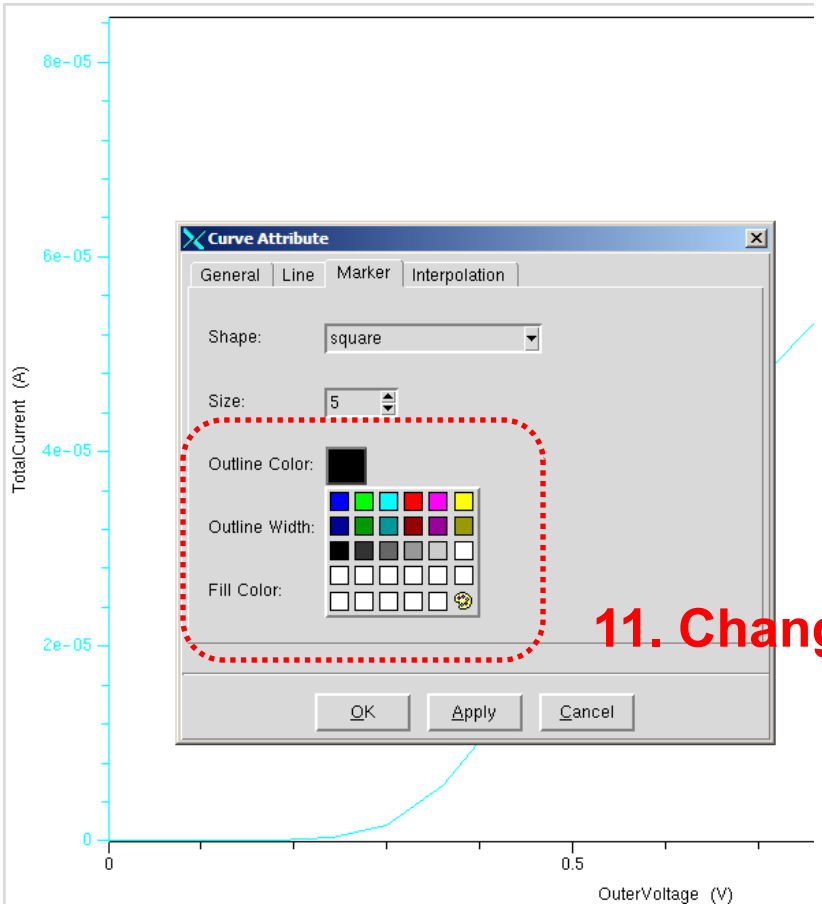
# Set Curve Drawing Attributes
## ~ Operation : Editing Marker (1/4) ~




© 2007 Synopsys, Inc. (267)

SYNOPSYS®
Predictable Success

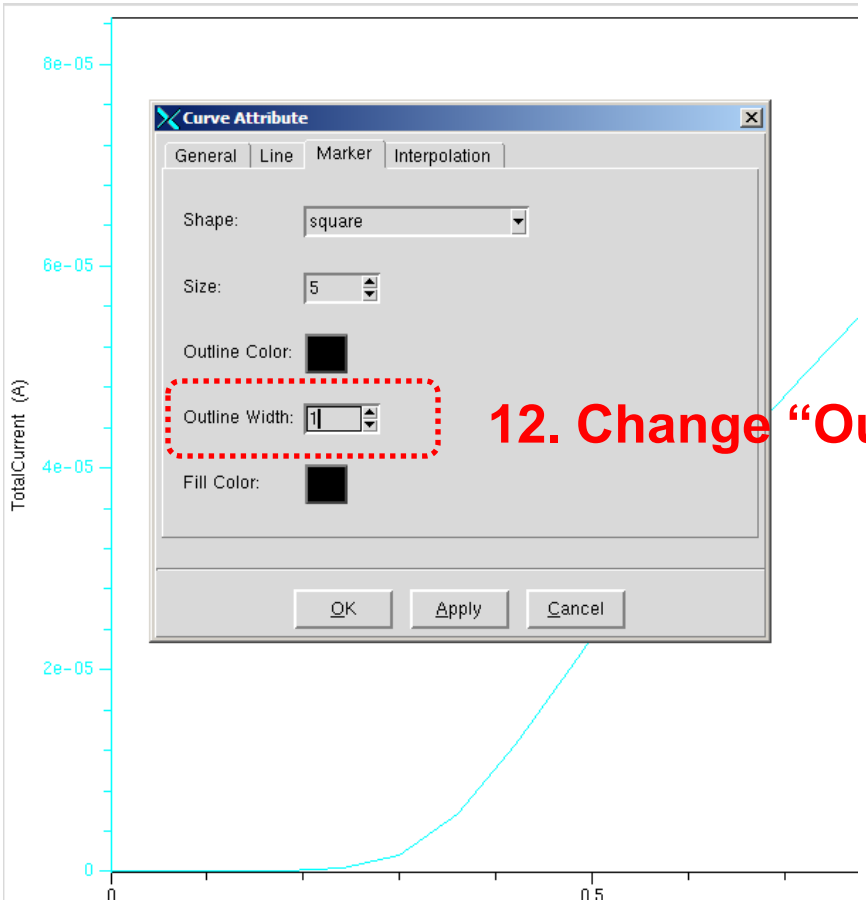
# Set Curve Drawing Attributes
## ~ Operation : Editing Marker (2/4) ~




11. Change “Outline Color”

© 2007 Synopsys, Inc. (268)

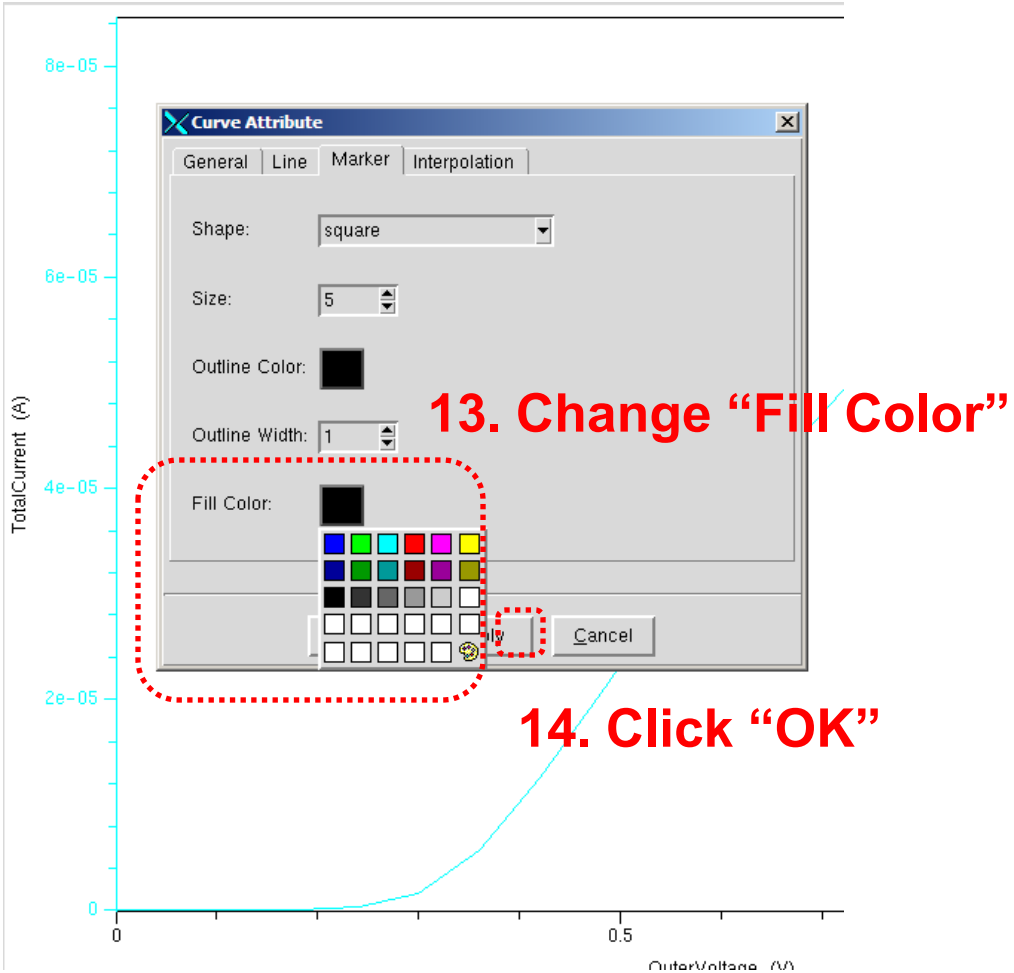
# Set Curve Drawing Attributes
## ~ Operation : Editing Marker (3/4)~




**12. Change "Outline Width"**

© 2007 Synopsys, Inc. (269)

# Set Curve Drawing Attributes
## ~ Operation : Editing Marker (4/4) ~




© 2007 Synopsys, Inc. (270)

SYNOPSYS®
Predictable Success

# Set Curve Drawing Attributes ~ Command ~

```
cv_setCurveAttr TotalCurrent_Drain IdVg black solid 3 \
\                                   3.    5.    6.   7.
 square 5 black 1 black
    9.  10. 11. 12. 13.
```

- `TotalCurrent_Drain` : the curve name.
- `IdVg` : the curve legend.
- `black` : the color of the curve line.
- `solid` : the drawing style of the curve line.
- `3` : the width of the curve line.
- `square` : a keyword for the marker shape.
- `5` : the marker size.
- `black` : the color of the marker outline.
- `1` : the width of the marker outline.
- `black` : the fill-in color of the marker.

© 2007 Synopsys, Inc. (271)

# Compute a Scalar Using the Formula

- The formula library allows some basic calculations to be performed on one or more selected curves.

```
cv_compute "formula" xmin xmax ymin ymax
```

  - *formula* : the string with the formula to evaluate.
  - *xmin, xmax, ymin, ymax* : the range for which the formula is applied.

© 2007 Synopsys, Inc. (272)

# Extract Variables

- The Inspect command `ft_scalar` prints the extracted value and passes it to Sentaurus Workbench.

- Examples:

```
set maxid [cv_compute "vecmax(<Id>)" 0 0.5 0 1.0]
ft_scalar id_max $maxid
```

```
set vt [cv_compute "vecvalx(<Id>, 1e-5)" 0 1.0 0 1.0]
ft_scalar Vt $vt
```

© 2007 Synopsys, Inc. (273)

# Extraction Library

- The commands provided by this library are used to extract various parameters from I–V curves. The library is loaded with the command:

```
load_library EXTRACT
```

- Refer to Inspect manual for more information.

© 2007 Synopsys, Inc. (274)

# RF Extraction Library

- The commands provided in this library are used to extract RF parameters from small-signal data. The library is loaded with the command:

```
load_library RFX
```

- Refer to Inspect manual for more information.

© 2007 Synopsys, Inc. (275)

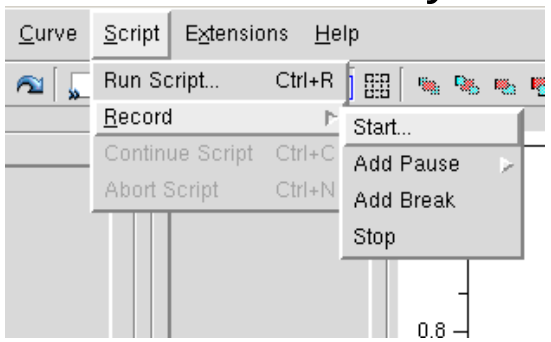
# Recording Script

- A script can be created automatically by recording actions performed interactively through the graphical user interface.

- To create a script:
  - A Script Name dialog box is displayed, which prompts you to select or create a script file.

    **Script > Record > Start**

  - After selecting the file name, Inspect starts to store every operation until recording is stopped.
    **Script > Record > Stop**

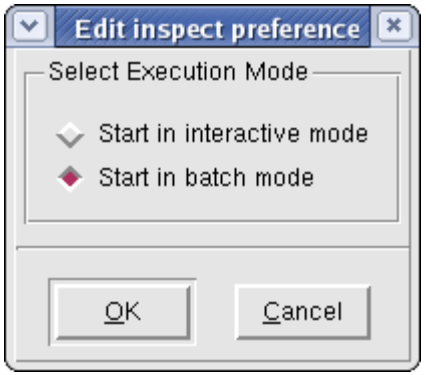


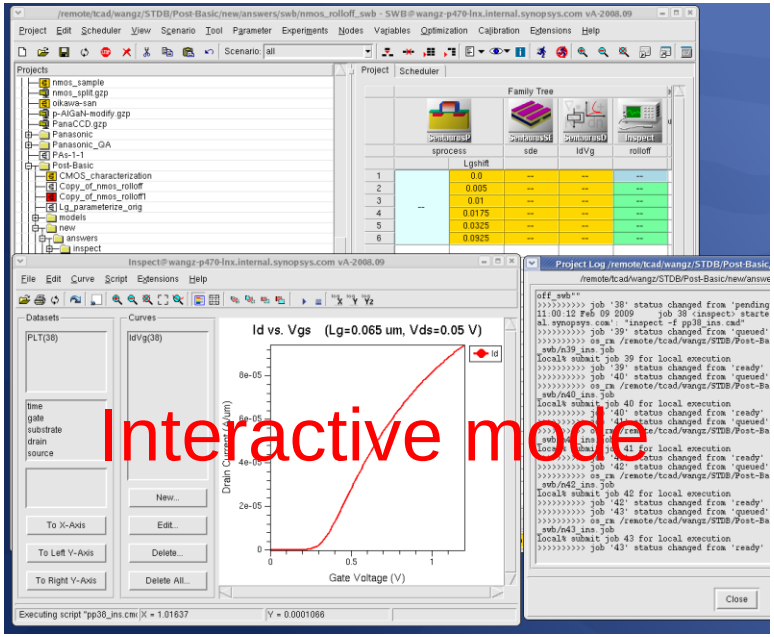

© 2007 Synopsys, Inc. (276)

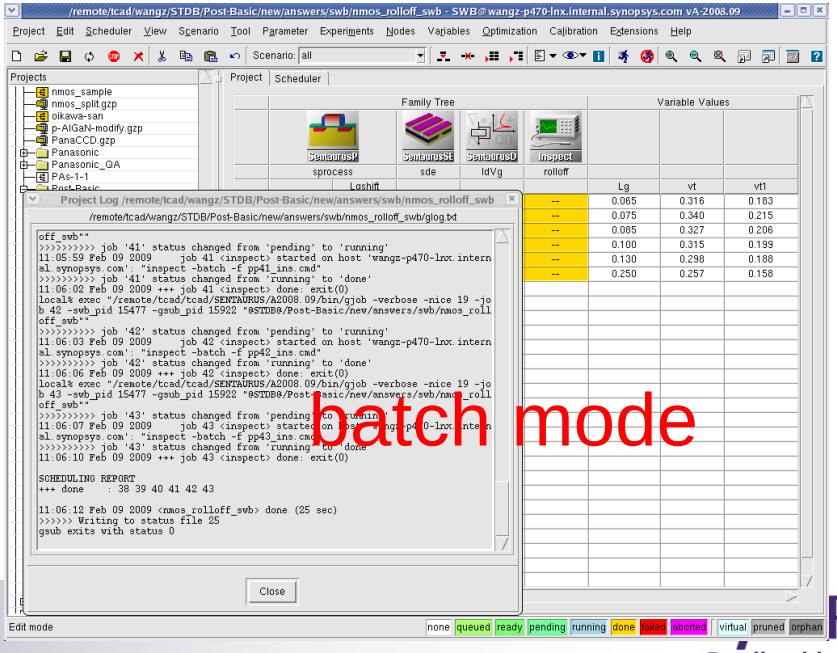
# Inspect Execution Mode

- Inspect has two execution modes: interactive and batch mode. You can change mode by:

  **Right button>Edit  Input>Preferences…**



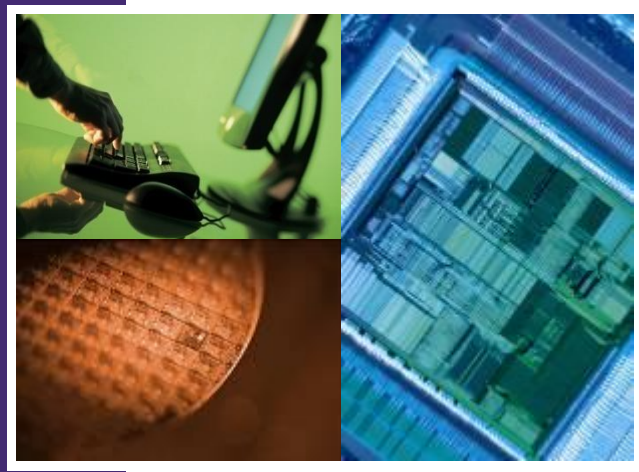




© 2007 Synopsys, Inc. (277)

Predictable Success

# Mathematical Formulas and Macros



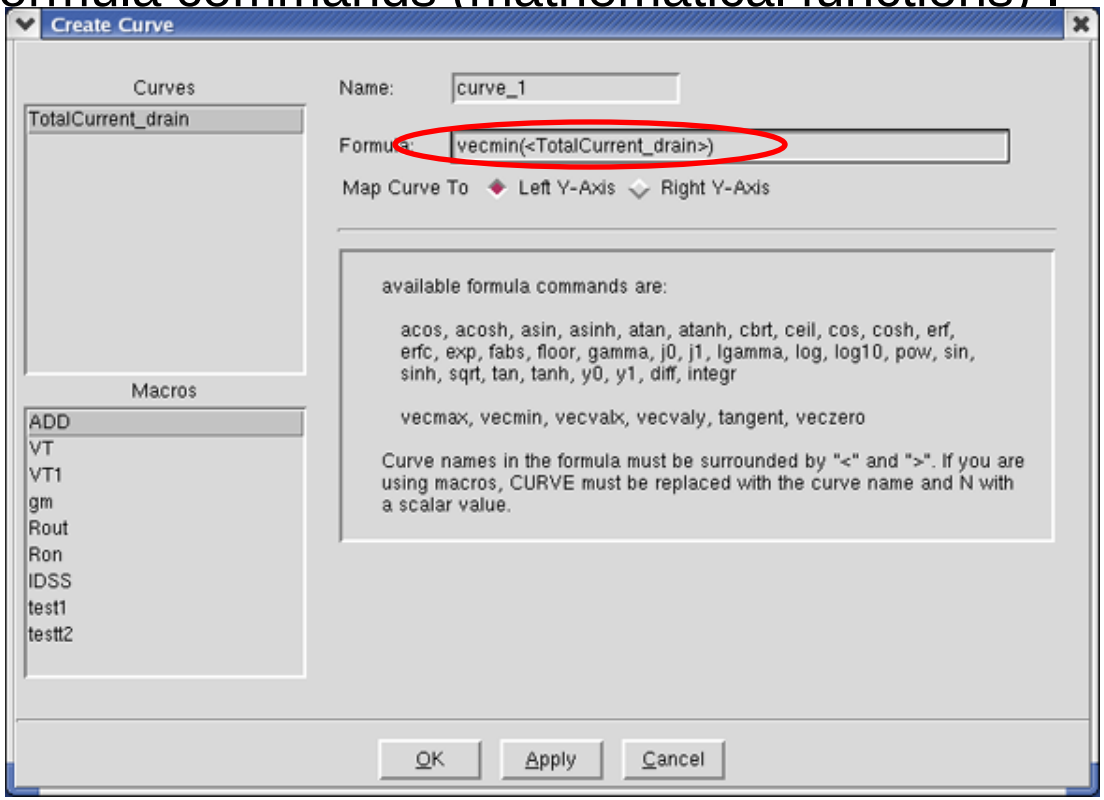
SYNOPSYS®

Predictable Success

# Mathematical Formulas (1/2)

- Click the **New** button below the **Curves** area in the Inspect main window.

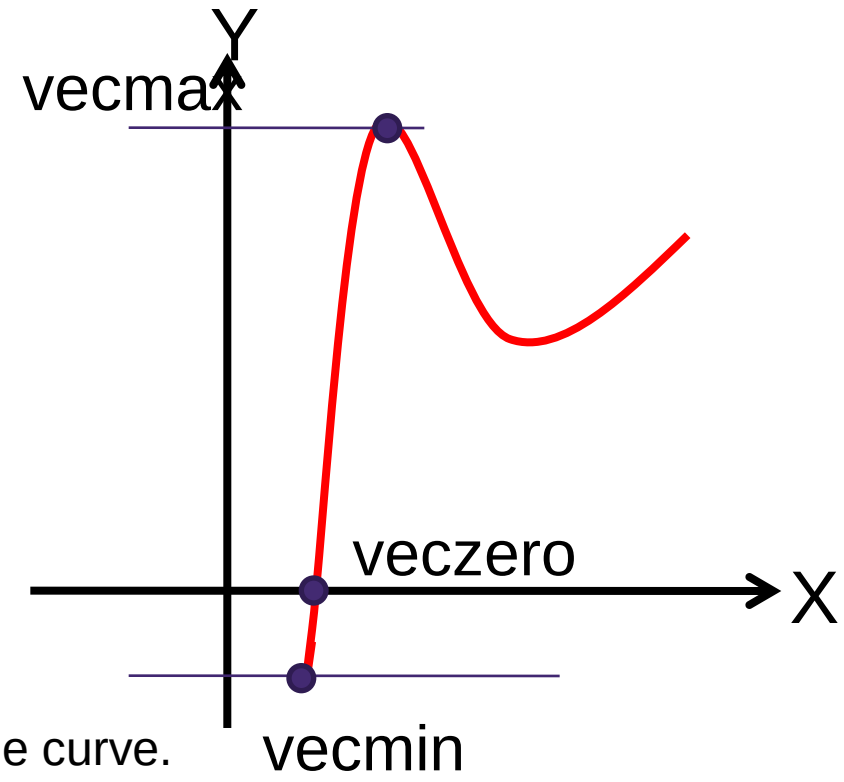
- The **Create Curve** dialog box is displayed. The right pane lists available formula commands (mathematical functions) .




© 2007 Synopsys, Inc. (279)

# Mathematical Formulas (2/2)

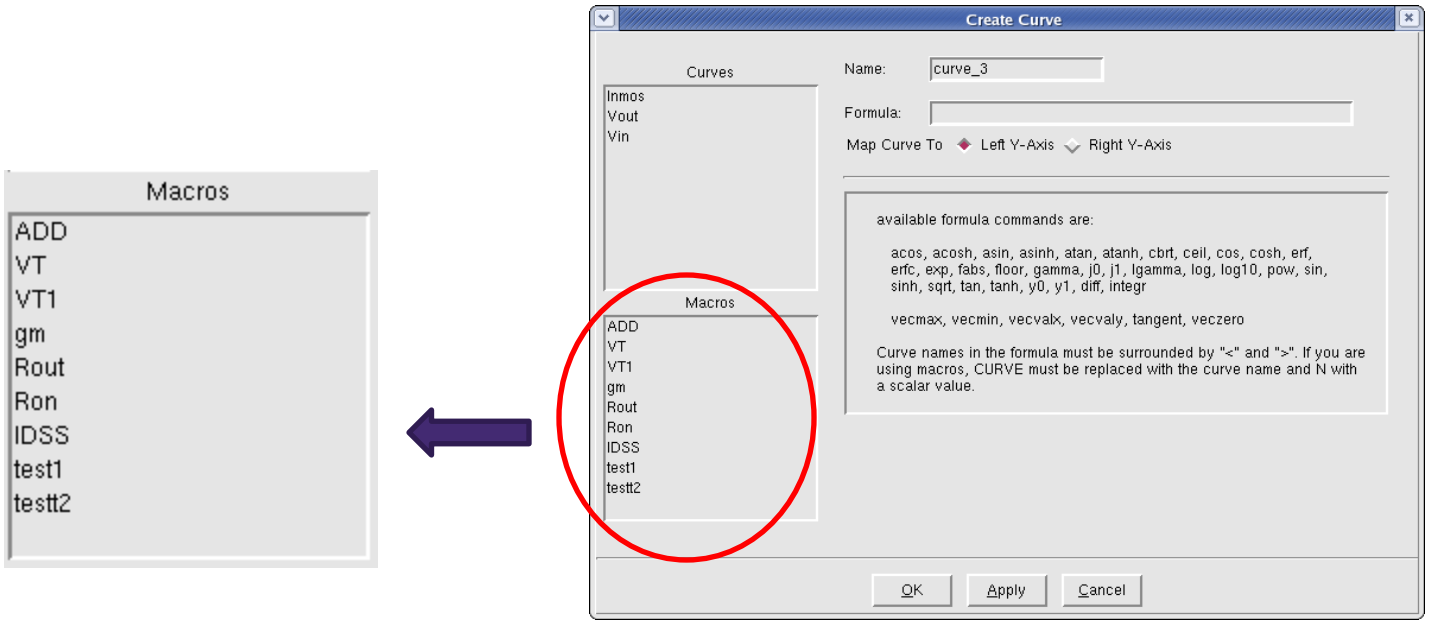
- vecmax (curve)
  - Maximum y-value.
- vecmin (curve)
  - Minimum y-value.
- vecvalx (curve, scalar)
  - x-value at a given y
- vecvaly (curve, scalar)
  - y-value at a given x
- veczero (curve)
  - x-value at y = 0.
- diff (curve)
  - Returns the first derivative of the curve.
- integr (curve)
  - Returns the integral of the curve.
- tangent (curve)
  - Returns the tangent.




© 2007 Synopsys, Inc. (280)

# Macros (1/3)

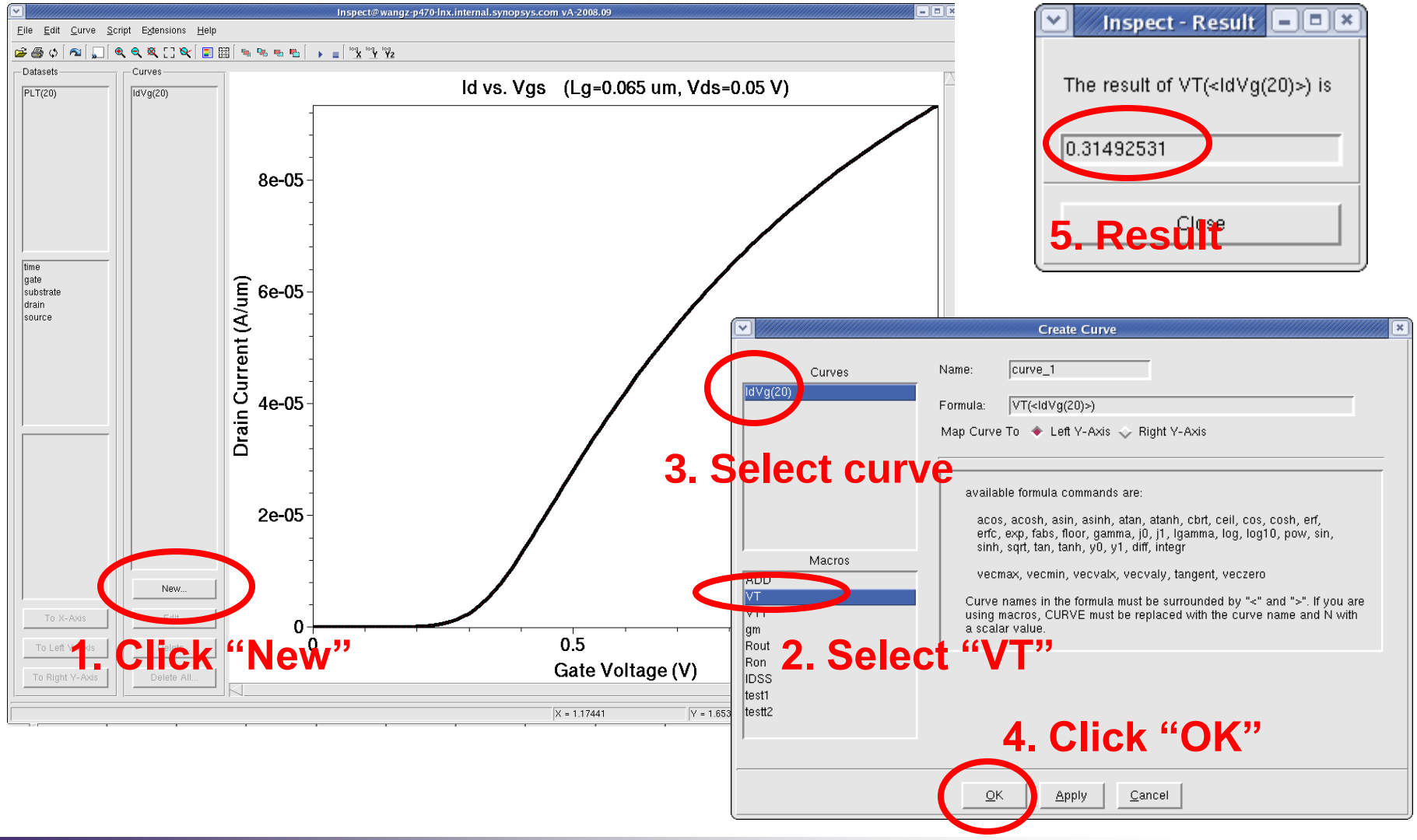
- Macros are predefined commands that can be later recalled. For example, in Inspect, there is a predefined macro, VT, which performs the threshold voltage extraction.




© 2007 Synopsys, Inc. (281)

# Macros (2/3)

- Macro usage:



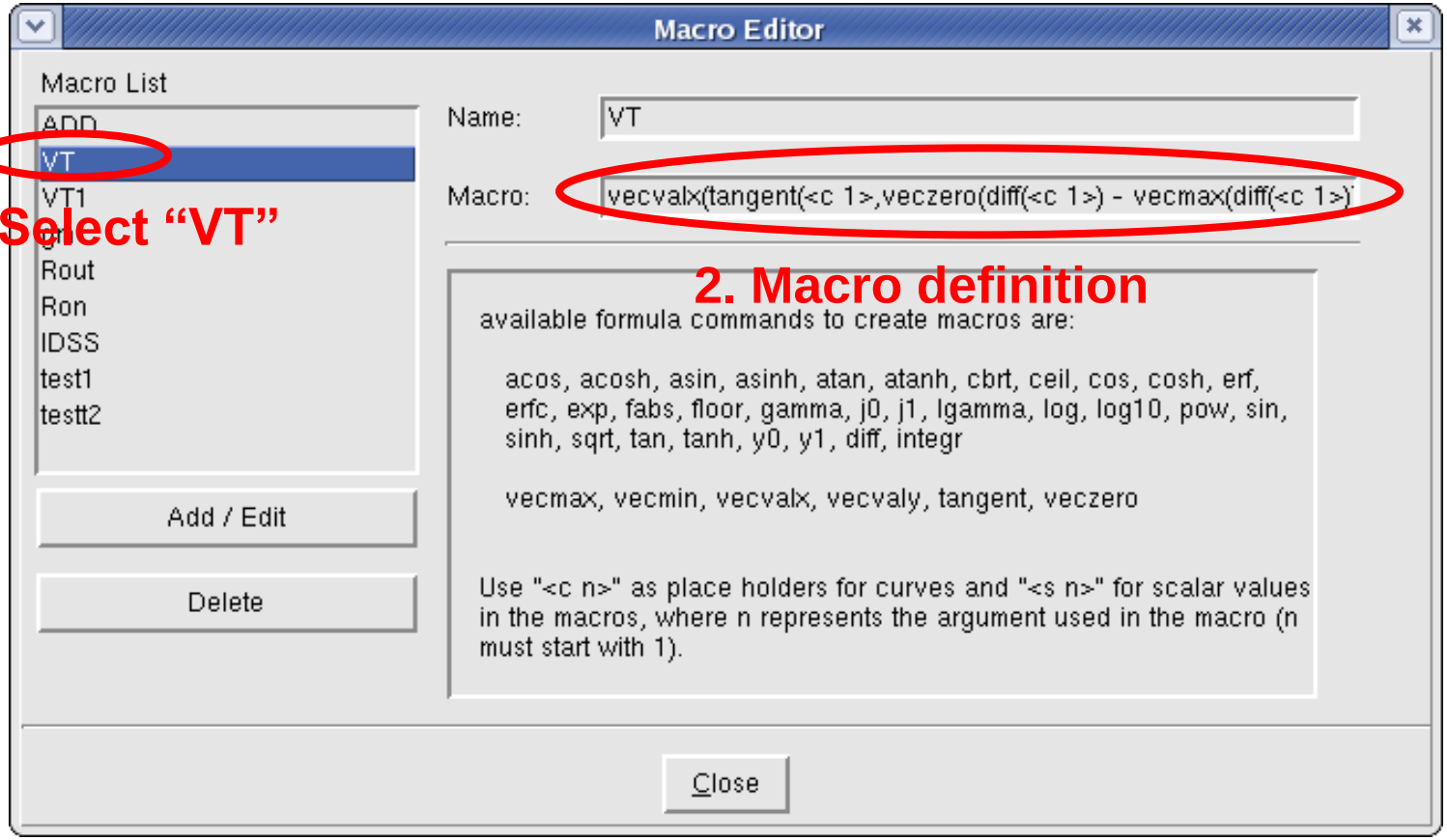

© 2007 Synopsys, Inc. (282)

SYNOPSYS®
Predictable Success

# Macros (3/3)

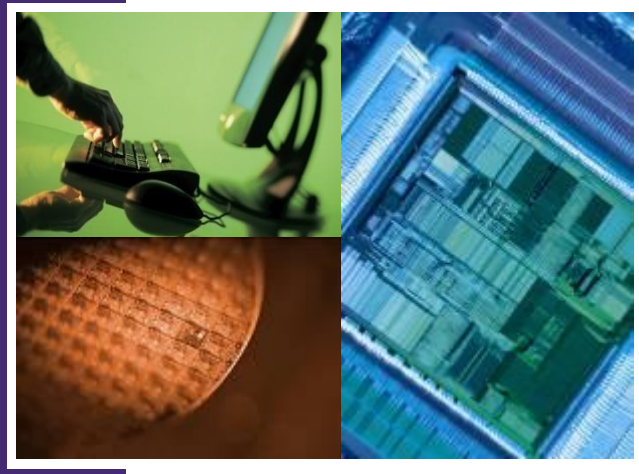
- Macro definition can be edited by

  **Edit>Define Macros…**




© 2007 Synopsys, Inc. (283)

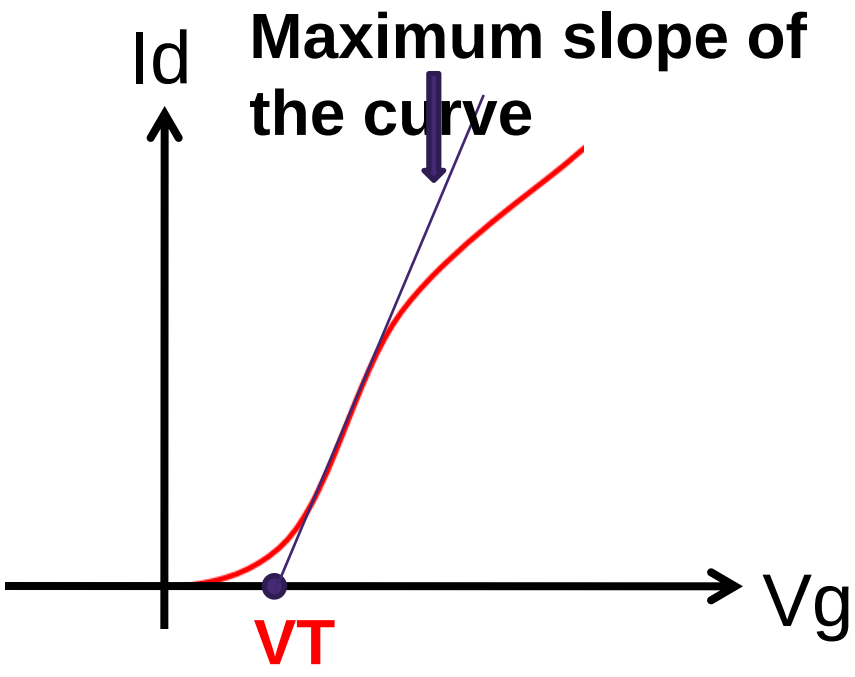
# Threshold Voltage Extraction



SYNOPSYS®

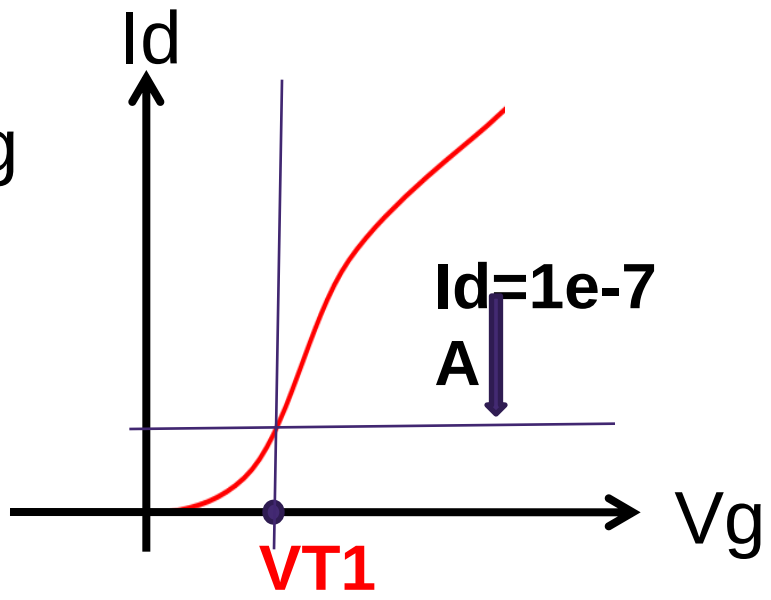
Predictable Success

# Definitions of Threshold Voltage (1/2)




Definition 1

Definition 2




© 2007 Synopsys, Inc. (285)

# Definitions of Threshold Voltage (2/2)

- Macros

  **VT:**

```
vecvalx(tangent(<c1>, veczero(diff(<c1> - vecmax(diff(<c1>)))), 0.0)
```

  **VT1:**

```
vecvalx(<c1>, 1.0e-7)
```

- Script functions

  **VT:**

```
f_VT curveName xmin xmax ymin ymax
```

  **VT1:**

```
f_VT1 curveName xmin xmax ymin ymax
```

➢xmin, xmax, ymin, ymax, the range for computing the result; default values correspond to the full curve range.

© 2007 Synopsys, Inc. (286)

# Inspect Script (1/4)

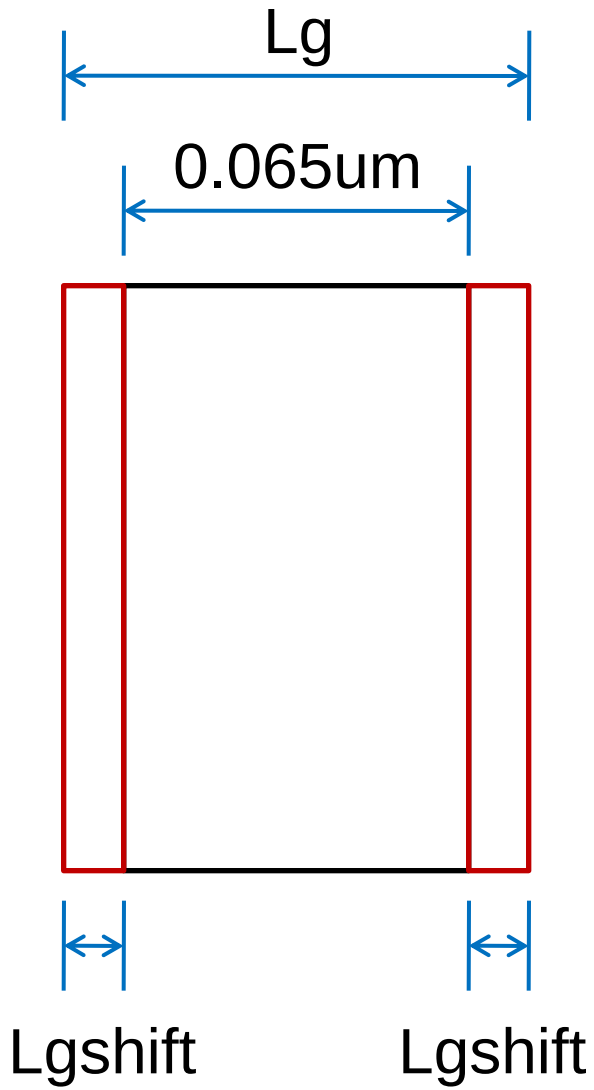
```
set   N     @node@
set   Vd    0.05
set   Lgshift  @Lgshift@
set   Lg   [expr 2.0*$Lgshift+0.065]
ft_scalar Lg  [format %0.3f $Lg]

proj_load @plot@ PLT($N)
cv_createDS IdVg($N) \
 "PLT($N) gate OuterVoltage" "PLT($N) drain TotalCurrent" y
cv_setCurveAttr IdVg($N)  "Id" red  solid 2 circle 0 defcolor 1 defcolor
gr_setAxisAttr X {Gate Voltage (V)} 12 {} {} black 1 10 0 5 0
gr_setAxisAttr Y {Drain Current (A/um)} 12 {} {} black 1 10 0 5 0
gr_setTitleAttr  "Id vs. Vgs    (Lg=$Lg um, Vds=$Vd V)" 14 center

#- Extraction
#- Calculate the threshold voltage [V]
set vt [f_VT IdVg($N)]
ft_scalar vt  [format %0.3f $vt]
set vt1 [f_VT1 IdVg($N)]
ft_scalar vt1  [format %0.3f $vt1]
```

© 2007 Synopsys, Inc. (287)

# Inspect Script (2/4)

Set variables:

```
set   N         @node@
set   Vd        0.05
set   Lgshift   @Lgshift@
set   Lg        [expr 2.0*$Lgshift+0.065]
```




© 2007 Synopsys, Inc. (288)

# Inspect Script (3/4)

Load plot file:

```
proj_load @plot@ PLT($N)
```

Create curve:

```
cv_createDS IdVg($N) \
 "PLT($N) gate OuterVoltage" "PLT($N) drain TotalCurrent" y
```

Set curve attribution:

```
cv_setCurveAttr IdVg($N)  "Id" red  solid 2 circle 0 defcolor 1 defcolor
```

Set axis attributions:

```
gr_setAxisAttr X {Gate Voltage (V)} 12 {} {} black 1 10 0 5 0
gr_setAxisAttr Y {Drain Current (A/um)} 12 {} {} black 1 10 0 5 0
```

Set title:

```
gr_setTitleAttr  "Id vs. Vgs    (Lg=$Lg um, Vds=$Vd V)" 14 center
```

© 2007 Synopsys, Inc. (289)

# Inspect Script (4/4)

Calculate VT:

```
set   vt   [f_VT  IdVg($N)]
```

Calculate VT1:

```
set   vt1   [f_VT1 IdVg($N)]
```

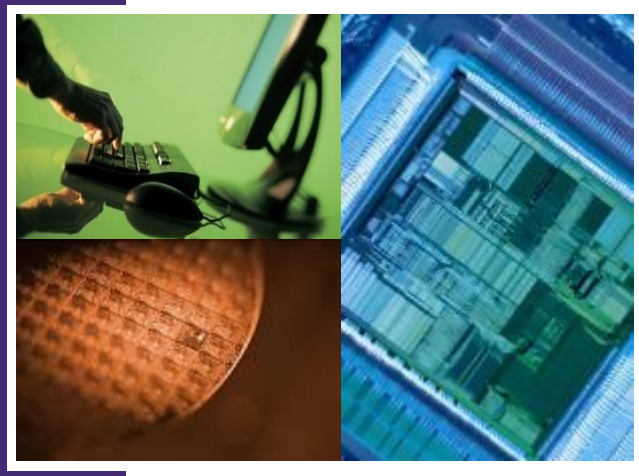
Export the variable to the Family Table of Sentaurus Workbench:

```
ft_scalar   Lg  [format %0.3f $Lg]
…
ft_scalar   vt  [format %0.3f $vt]
ft_scalar   vt1 [format %0.3f $vt1]
```

| | Family Tree | | | | | Variable Values | | |
|---|---|---|---|---|---|---|---|---|
| |  SentaurusP | |  SentaurusSE |  SentaurusD |  Inspect | | | |
| | sprocess | | sde | ldVg | rolloff | | | |
| | | Lgshift | | | | Lg | vt | vt1 |
| 1 | -- | 0.0 | -- | -- | -- | 0.065 | 0.316 | 0.183 |
| 2 | | 0.005 | -- | -- | -- | 0.075 | 0.340 | 0.215 |
| 3 | | 0.01 | -- | -- | -- | 0.085 | 0.327 | 0.206 |
| 4 | | 0.0175 | -- | -- | -- | 0.100 | 0.315 | 0.199 |
| 5 | | 0.0325 | -- | -- | -- | 0.130 | 0.298 | 0.188 |
| 6 | | 0.0925 | -- | -- | -- | 0.250 | 0.257 | 0.158 |

© 2007 Synopsys, Inc. (290)

# 90nm nMOS Exercise



SYNOPSYS®

Predictable Success

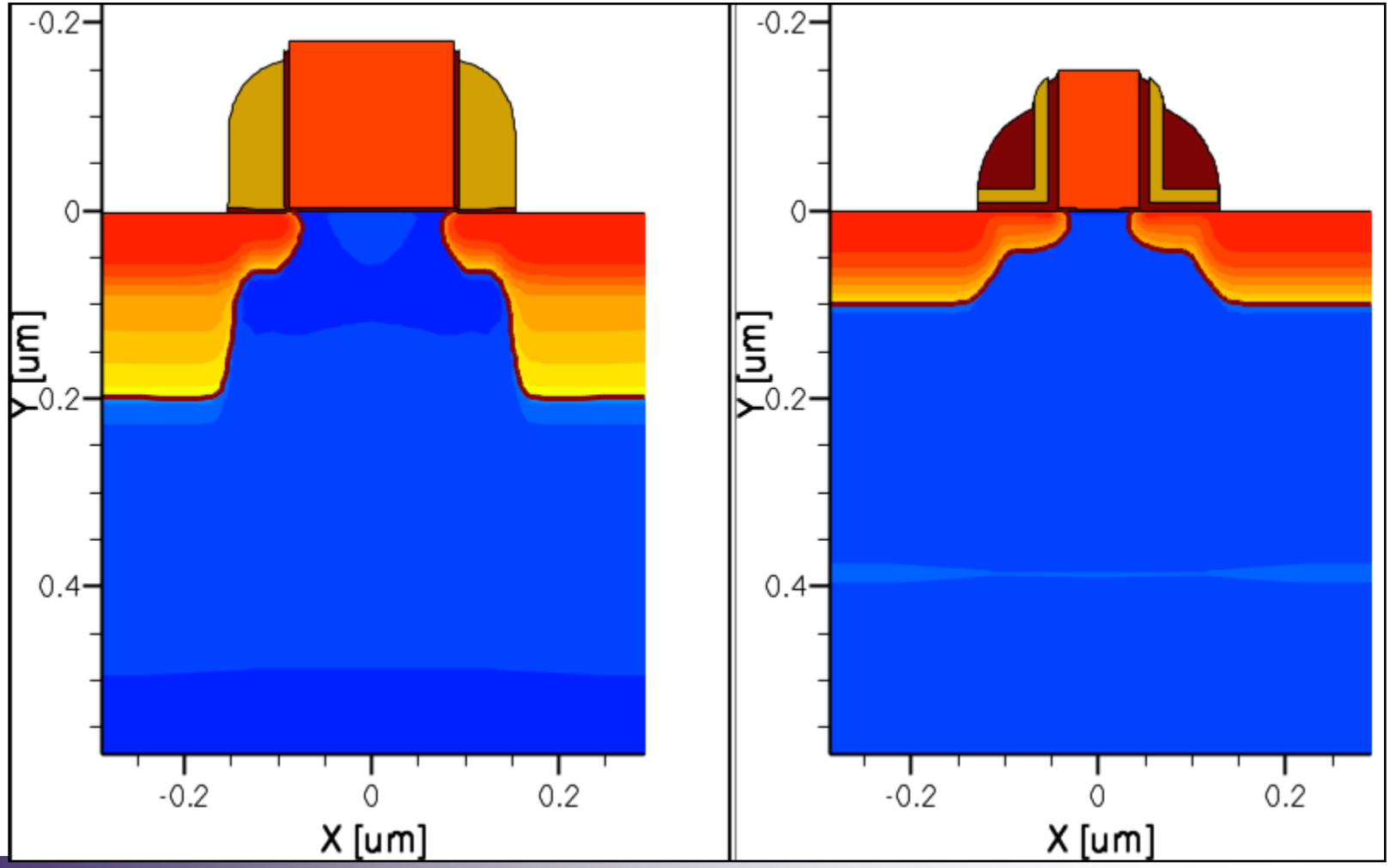
# Preparation

- Copy the 0.18um_nMOS project and rename it as 90nm_nMOS, then clear up all the output files
- SProcess uses the –f option to speed up the simulation time for structure verification
  - -f option skip diffusion and monte carlo implant step
  - Refer to SProcess manual 2007.12 P.52
  - It will cause some parameter extraction failure because of a missing net doping profile. When we are sure the device structure is correct, we should remove the –f option to do a complete simulation.

© 2007 Synopsys, Inc. (292)

# Structure Comparison

- 0.18um nMOS
- 90nm nMOS




© 2007 Synopsys, Inc. (293)

# Process Comparison

- 0.18um nMOS

Well/Vth implant & anneal
Boron dose=2e13 energy=200 tilt=0 rot=0
Boron dose=1e13 energy=80 tilt=0 rot=0
Boron dose=6e12 energy=25 tilt=0 rot=0
diffuse temp=1050 time=0.3<s>

Gox~33A
diffuse temp=650 ramprate=0.6666 time=5
diffuse temp=850 time=10 O2 1atm
diffuse temp=850 ramprate=-0.6666 time=5

POLY height = 0.18um
deposit poly type=isotropic thickness=0.18

Halo/LDD implant & anneal
Boron dose=1e13 energy=15 tilt=30 /
rot=0,90,180,270
Arsenic dose=1e15 energy=20 tilt=0 rot=0
diffuse temp=1050 time=0.02<s>

- 90nm nMOS

Well/Vth implant & anneal
Boron dose=2e13 energy=200 tilt=0 rot=0
Boron dose=1e13 energy=80 tilt=0 rot=0
Boron dose=8e12 energy=30 tilt=0 rot=0
diffuse temp=1050 time=0.3<s>

Gox~18A
diffuse temp=650 ramprate=0.6666 time=5
diffuse temp=850 time=10 O2=0.8<l/min> N2=9.2<l/min> 1atm
diffuse temp=850 ramprate=-0.6666 time=5

POLY height = 0.15um
deposit poly type=isotropic thickness=0.15

Halo/LDD implant & anneal
Boron dose=2e13 energy=12 tilt=30 /
rot=0,90,180,270
Arsenic dose=1e15 energy=5 tilt=0 rot=0
diffuse temp=1020 time=0.03<s>

© 2007 Synopsys, Inc. (294)

# Process Comparison

- 0.18um nMOS

Nitride Spacer

```
deposit  nitride type=isotropic thickness=0.06
etch  nitride type=anisotropic thickness=0.084
etch  oxide type=anisotropic thickness=0.01
```

Source/Drain implant & anneal

```
implant Arsenic dose=4e15 energy=30 tilt=0 rot=0
diffuse temp=1080 time=0.02<s>
```

- 90nm nMOS

ONO Spacer

```
deposit oxide type=isotropic  thickness=0.005
deposit nitride type=isotropic  thickness=0.015
deposit oxide type=isotropic  thickness=0.06
diffuse temperature=720 time=20
etch oxide   type=anisotropic thickness=0.1
etch nitride type=anisotropic thickness=0.03
etch oxide   type=anisotropic thickness=0.015
#--- consider native oxide ---------------------------
deposit oxide type=isotropic  thickness=0.0015
```

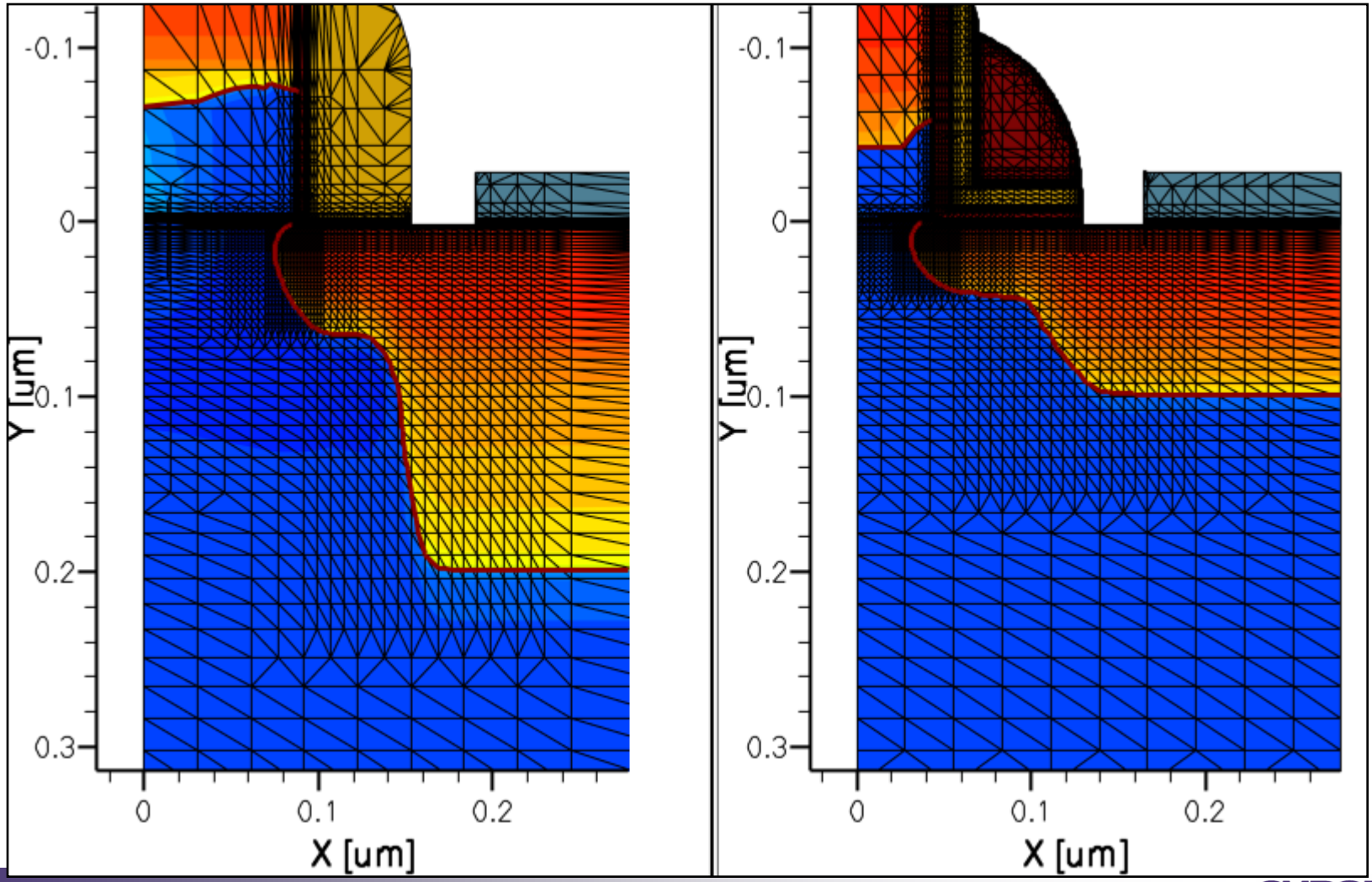
Source/Drain implant & anneal

```
implant Phosphorus dose=1e15 energy=6 tilt=0 rot=0
implant Arsenic dose=5e15 energy=30 tilt=0 rot=0
diffuse temp=1050 time=0.03<s>
```

© 2007 Synopsys, Inc. (295)

# SProcess Mesh Comparison

- 0.18um nMOS
- 90nm nMOS

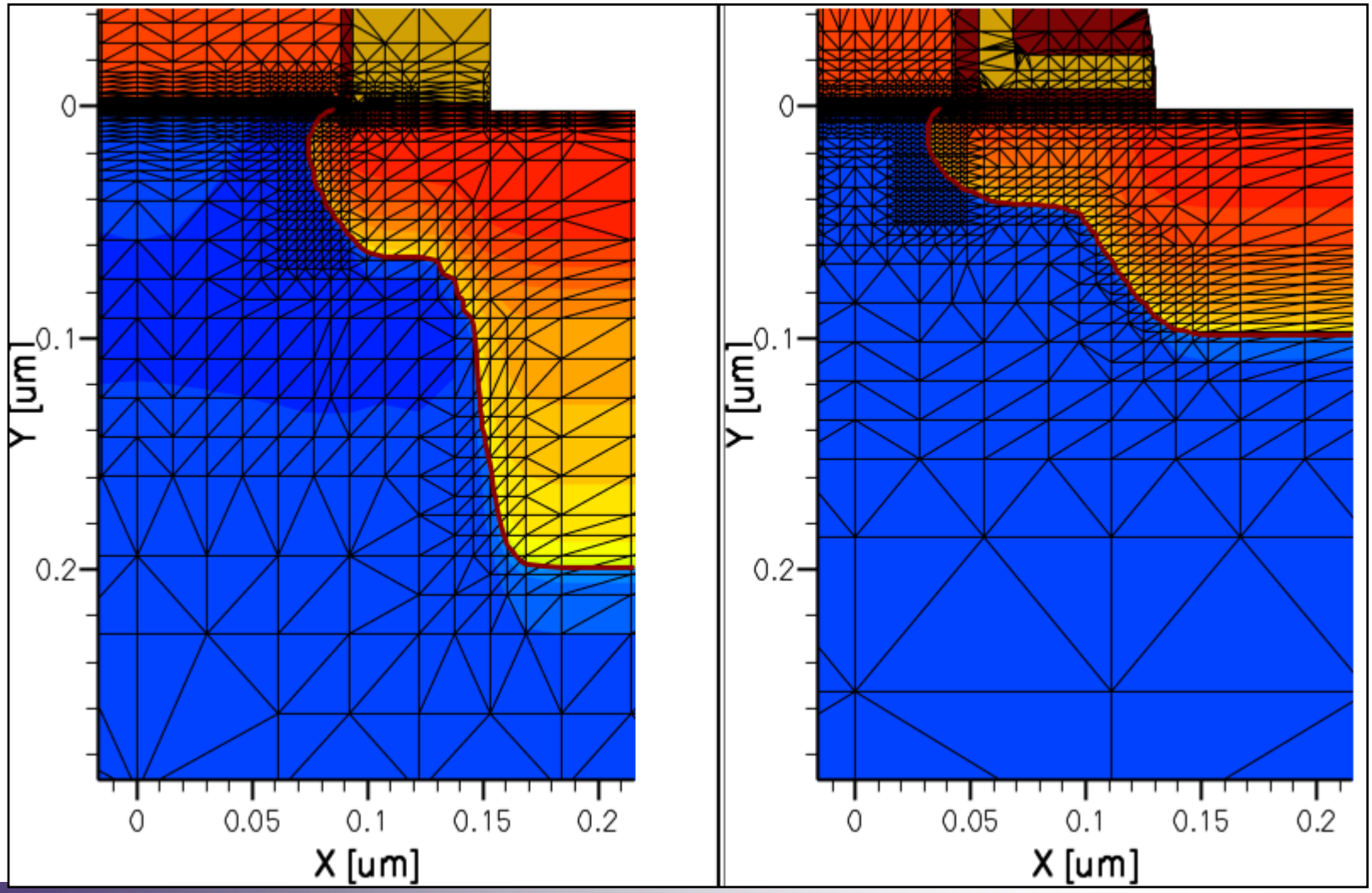



© 2007 Synopsys, Inc. (296)

SYNOPSYS®
Predictable Success

# SDevice Mesh Comparison

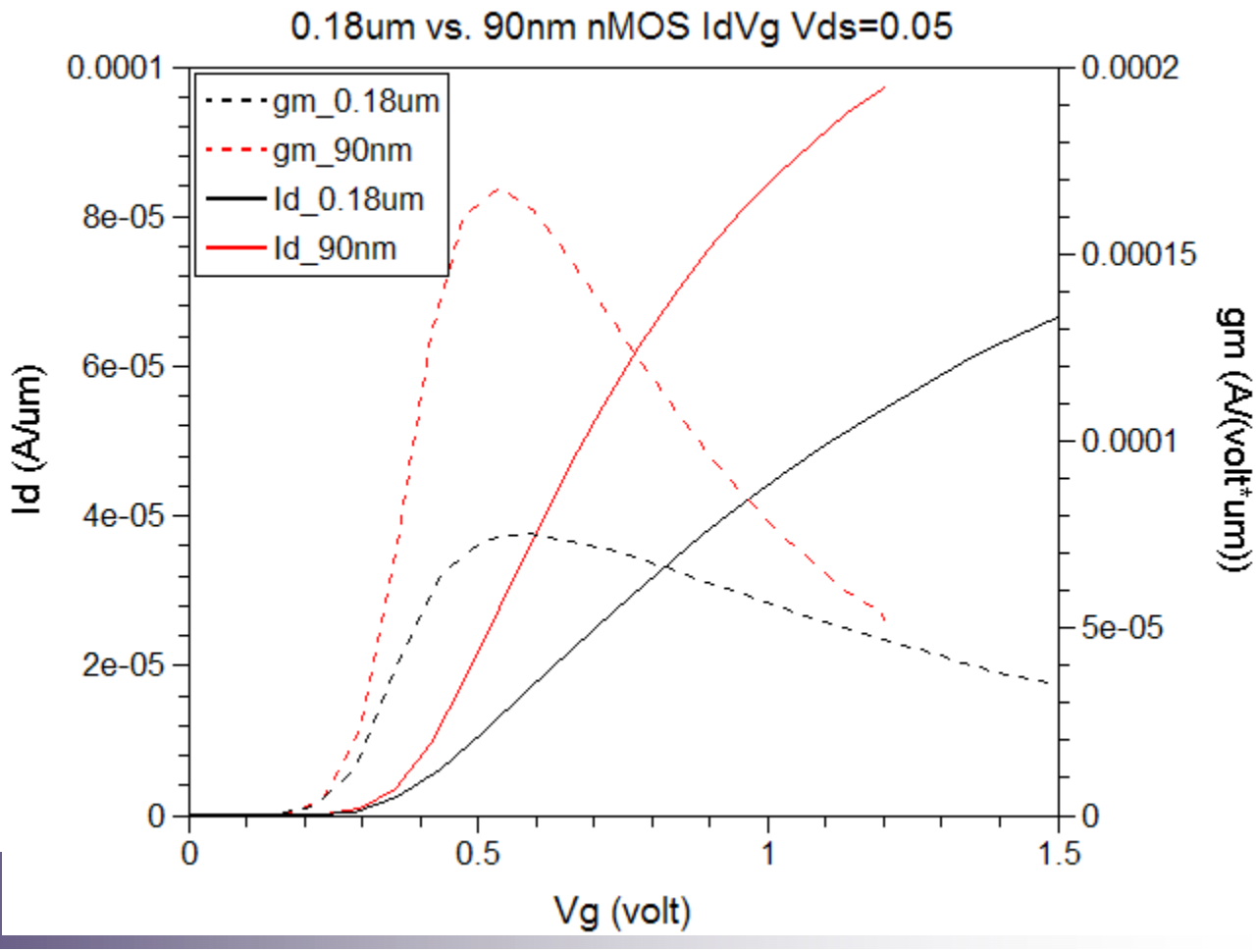
- 0.18um nMOS
- 90nm nMOS




© 2007 Synopsys, Inc. (297)

# Id_Vg Curve Comparison

- 0.18um nMOS
  - Vdd=1.5v; Vds=0.05v
- 90nm nMOS
  - Vdd=1.2v; Vds=0.05v




© 2007 Synopsys, Inc. (298)

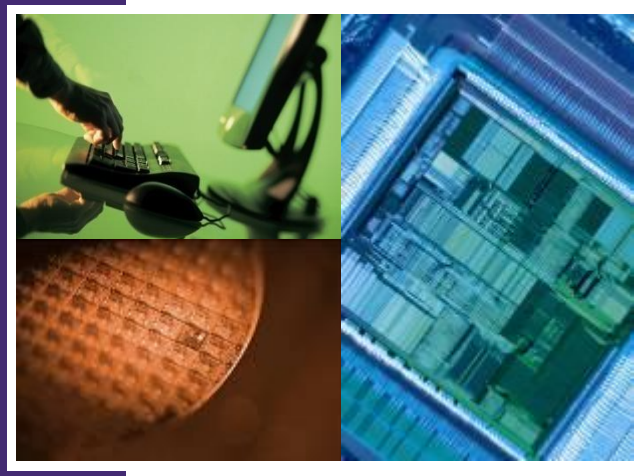
SYNOPSYS®
Predictable Success

# Hints

- Need to translate the process conditions into correct syntax of Sprocess.
- Poly height has been changed from 0.18um to 0.15um, need to update the coordinate of gate contact in Sprocess.
- After finish the modification of Sprocess, whole simulation flow (Sprocess, SSE, Sdevcie, Inspect) can be executed well.
- In Sprocess, fine tune LDD & SD refine-box to get a more accurate doping profile.
- In SSE, fine tune variable “PNres” to optimize the mesh of LDD and SD junction to get a more accurate Sdevice simulation result.

© 2007 Synopsys, Inc. (299)

# C-V Device Simulation
# Breakdown Device Simulation
# pMOS Simulation



SYNOPSYS®

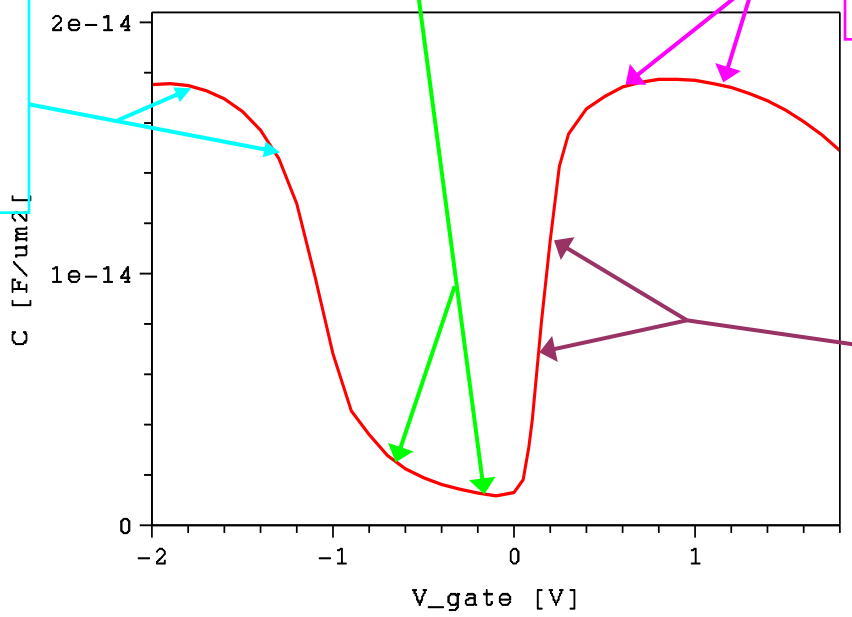
Predictable Success

# CV-curve: long channel transistor

**Bottom region:**
Good fit indicates, that the channel profile is correct

**Inversion region:**
Used to adjust the doping concentration in poly-Silicon.

**Accumulation region:**
Good fit indicates correct oxide thickness.

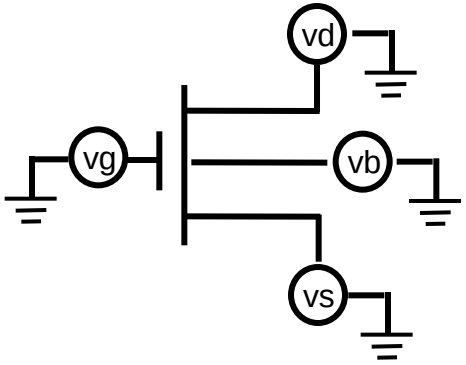



**Threshold voltage:**
- Dopant concentration at the gate oxide interface
- Interface charges

© 2007 Synopsys, Inc. (301)

# C-V Device Simulation

- Small-signal AC analysis (2007.12 P.183)
  - AC simulation are performed in mixed mode (2007.12 P.163).




  - Extract conductance and capacitance of each node, and output matrix A (conductance) and matrix C (capacitance).
    - **j = Yu = Au + iωCu**
    - j is the vector containing the small-signal currents at all nodes.
    - u is the corresponding voltage vector.

© 2007 Synopsys, Inc. (302)

# C-V Simulation Command File

Device “device1” {

File { “define device input and output file” }

Electrode { “same with IV simulation” }

Physics { “same with IV simulation and turn on minority carrier quantum potential equation for accumulation side CV extraction” }

}

File { “define system output file” }

Plot { “same with IV simulation” }

Math { “same with IV simulation” }

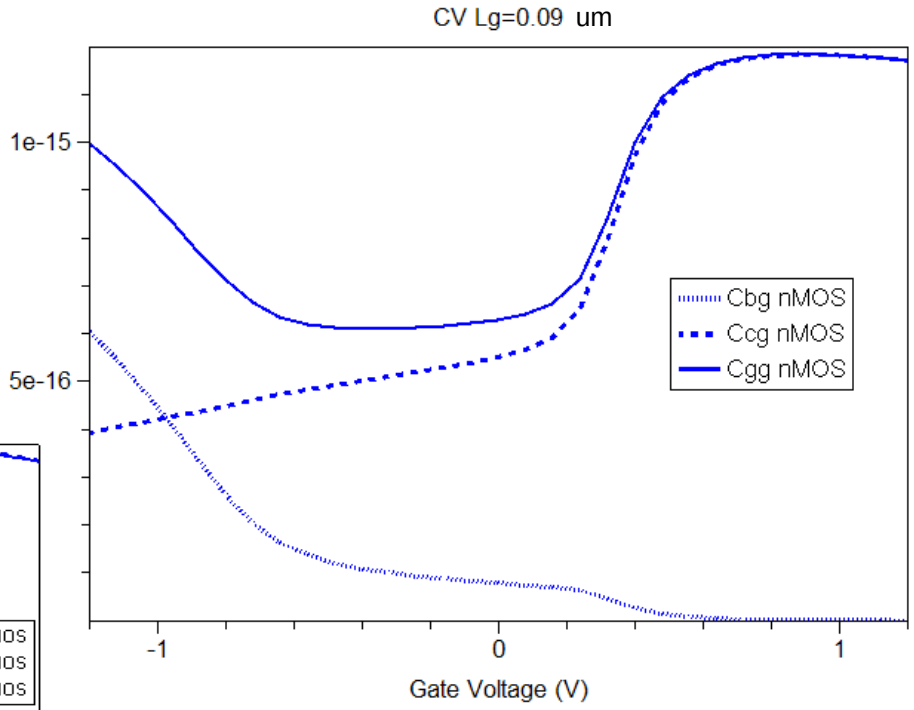
System { “based on previous device1 and the voltage source of each node construct a simple circuit” }

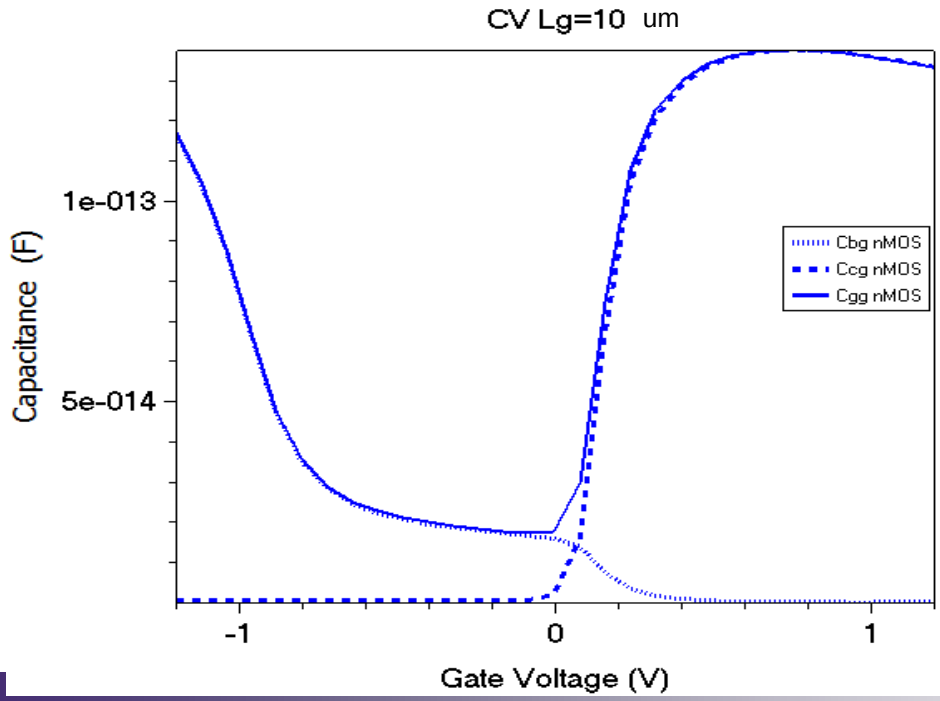
Solve { “set bias sweeps sequence, solve transport models and extract AC parameters” }

© 2007 Synopsys, Inc. (303)

# C-V Device Simulation Result

- Cgg: gate node capacitor
- Ccg: drain node + source node
- Cbg: substrate node
- Cgg = Ccg + Cbg



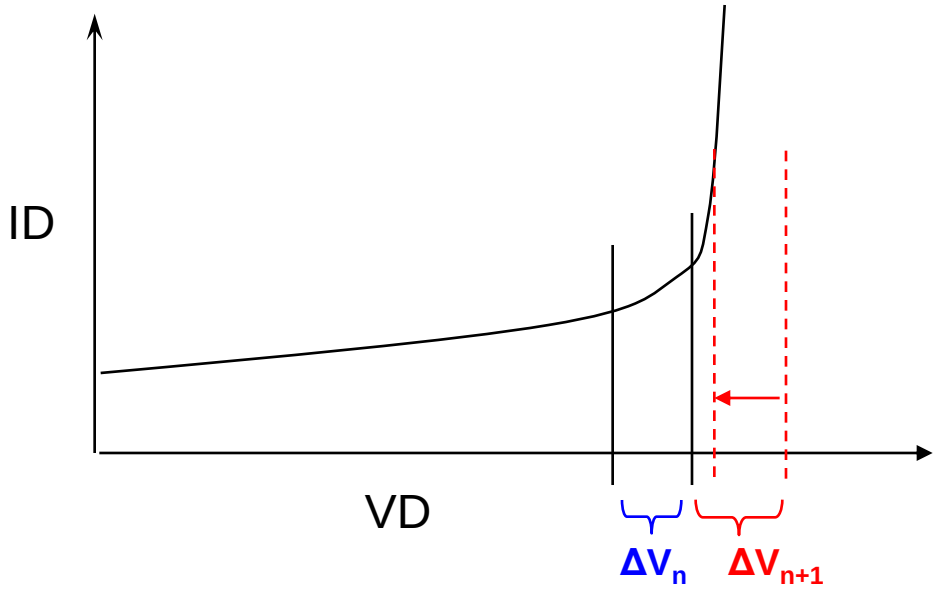




© 2007 Synopsys, Inc. (304)

SYNOPSYS®
Predictable Success

# Breakdown Device Simulation

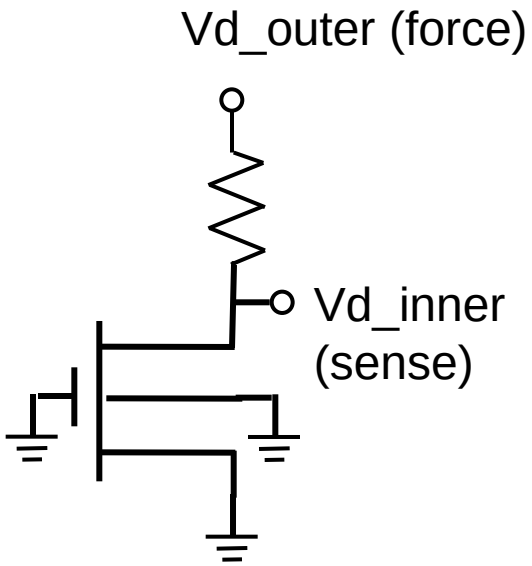
- n-th step converge, next $\Delta V_{n+1} > \Delta V_n$
- n+1-th step diverge, then $\Delta V_{n+1} = \Delta V_n/2$
- The voltage step will be divided by 2 again and again, until it converges or until it reaches a minimum step that we define in the command file.

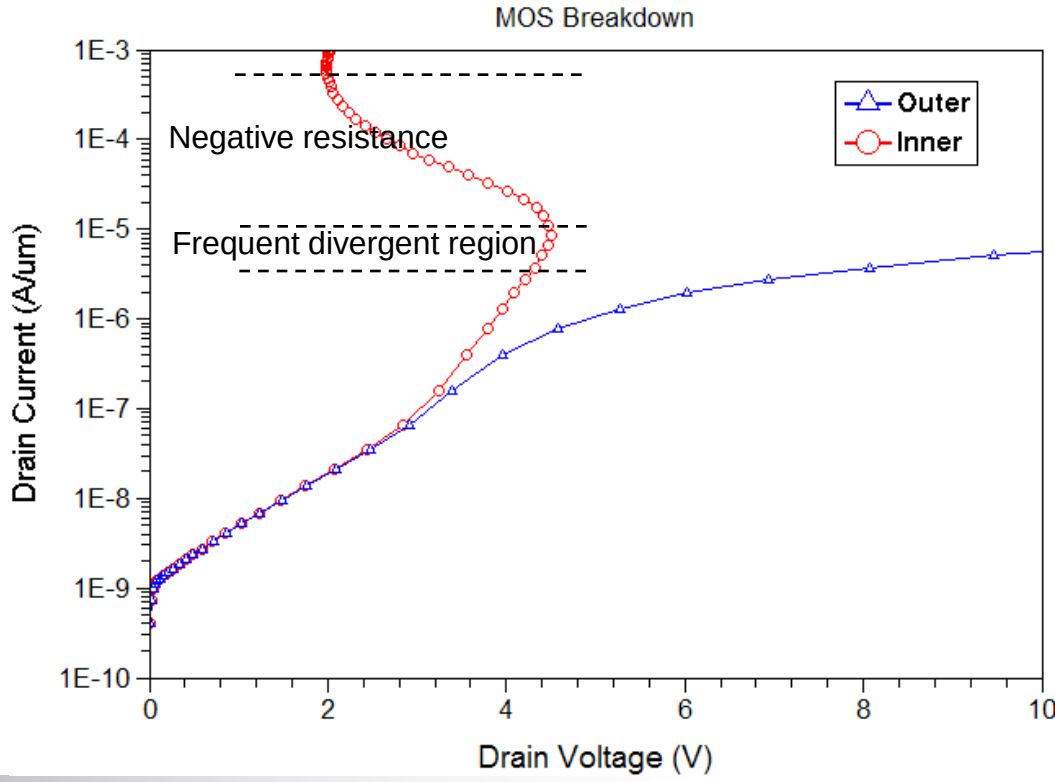



© 2007 Synopsys, Inc. (305)

# Breakdown Device Simulation

- An approach to solve divergent and snapback issues
  - Introduction please refer to on-line training material “SDevice section D.1 MOSFET Breakdown Simulation ”




© 2007 Synopsys, Inc. (306)

SYNOPSYS®
Predictable Success

# Breakdown Device Simulation

- Modify SDevice command file
  - Turn on Avalanche and Band2Band model
  - Solve dual carrier transport
  - Detail please refer to 90nm nMOS breakdown example.

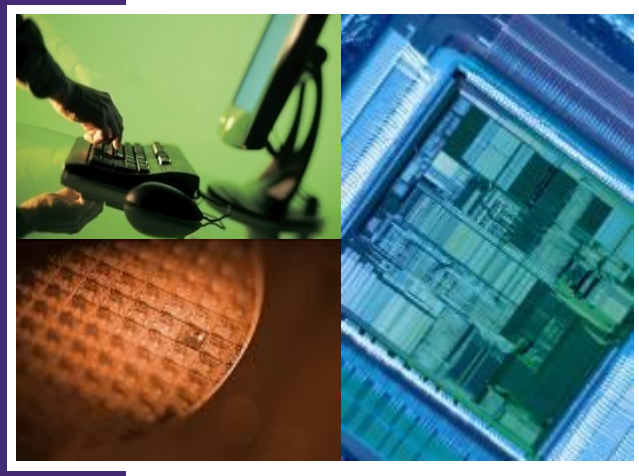
© 2007 Synopsys, Inc. (307)

# Modify nMOS Template to pMOS

- SProcess: change to pMOS process condition
- SDE: change PLOY dopant type from N-type to P-type
- SDevice
  - Change majority carrier from electron to hole and related models. (Exchange e and h in input file in most cases.)
  - Change bias condition
- Inspect: change device type and extraction region for “Extraction library”
- Detail please refer to 90nm pMOS example

© 2007 Synopsys, Inc. (308)

# SolvNet Introduction



SYNOPSYS®

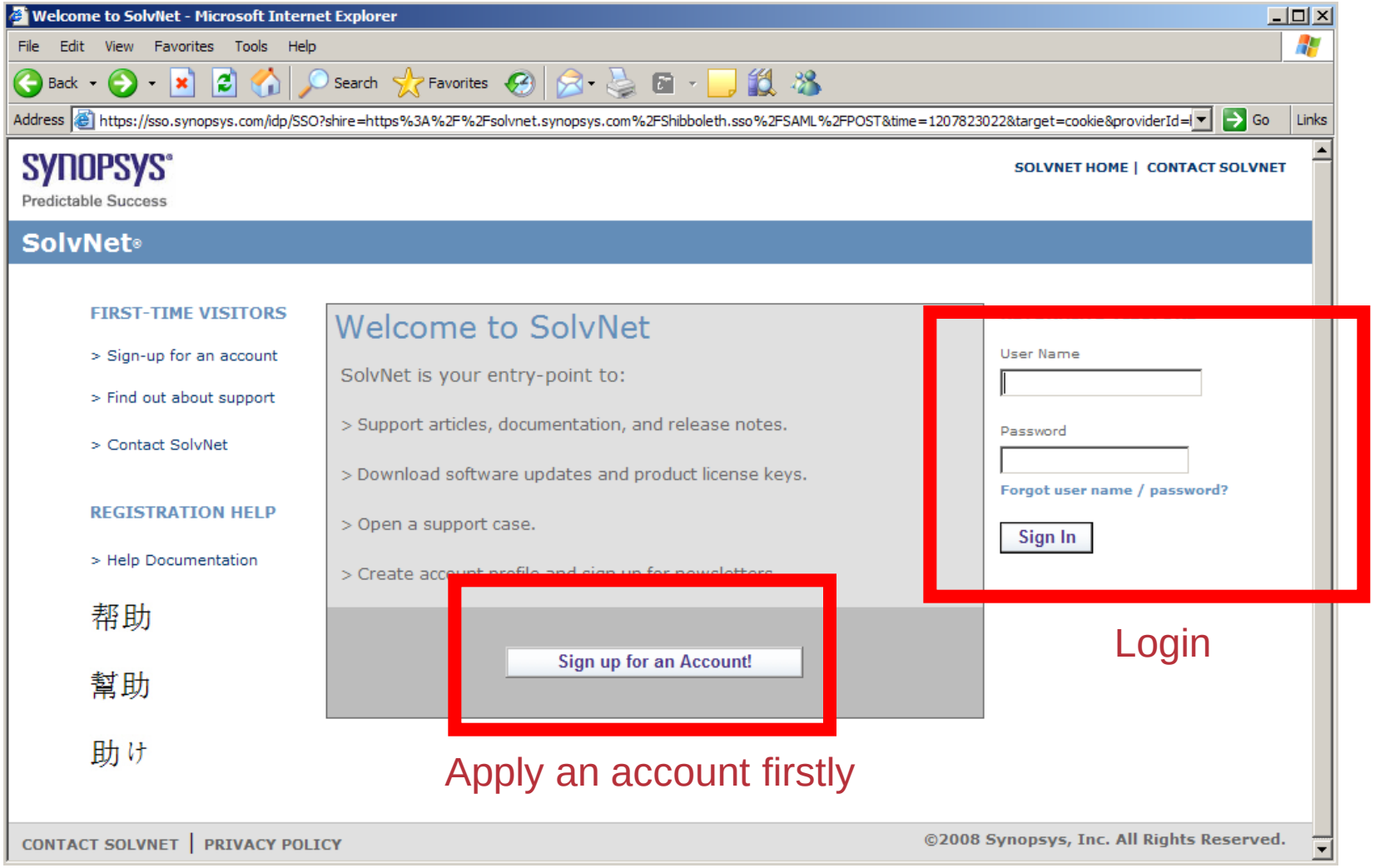
Predictable Success

# Outline

- How to register a SolvNet account
- What kind of TCAD resources we can find on SolvNet
  - Software, Manual, Release Notes, TCAD Example updates
- Two advanced TCAD examples of the SolvNet example library
  - 2D Strained Silicon 45nm CMOS Reference Flow Demo
  - 3D nMOSFET Demo

© 2007 Synopsys, Inc. (310)

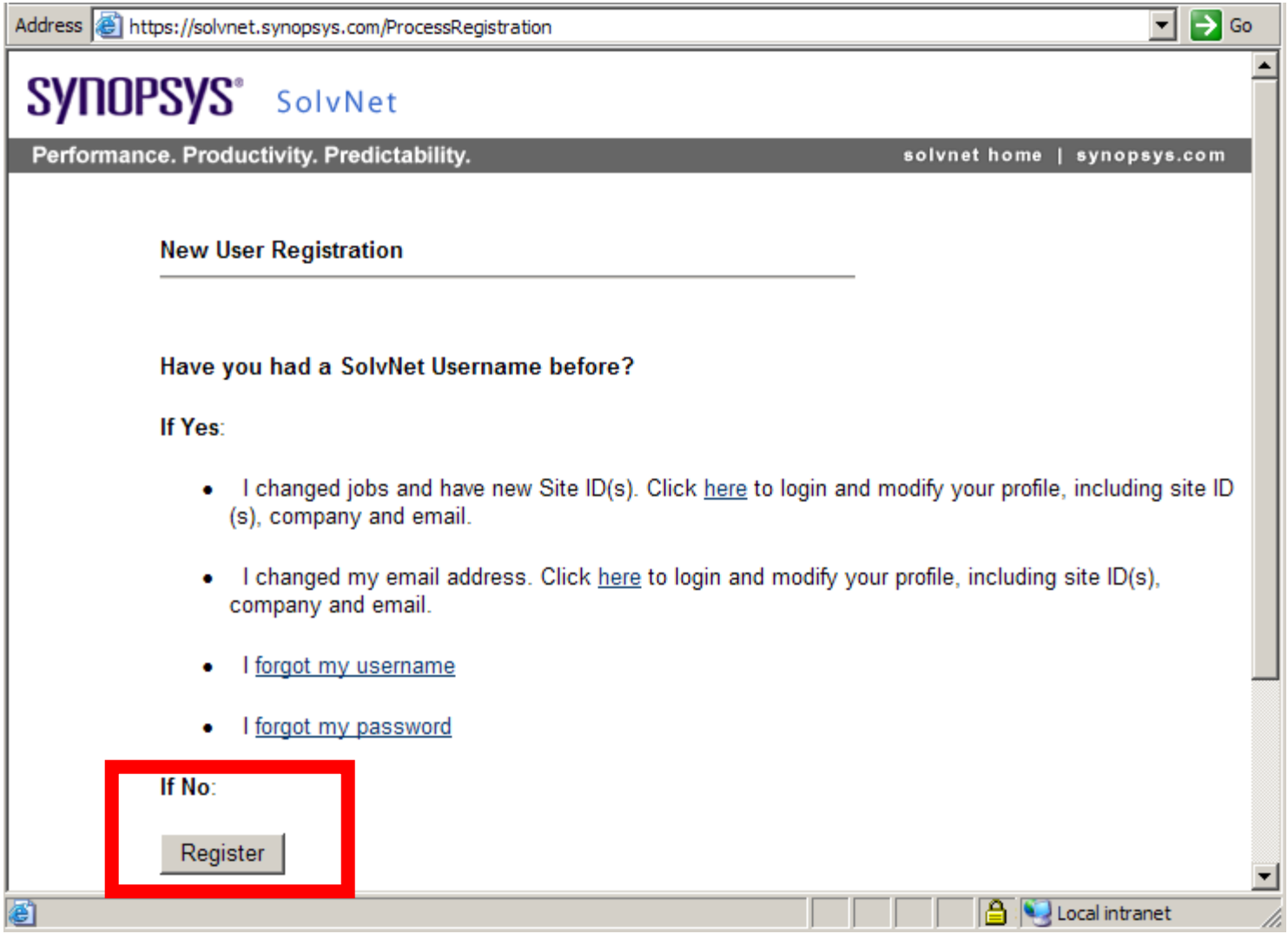
# SolvNet Website http://solvnet.synopsys.com/




© 2007 Synopsys, Inc. (311)

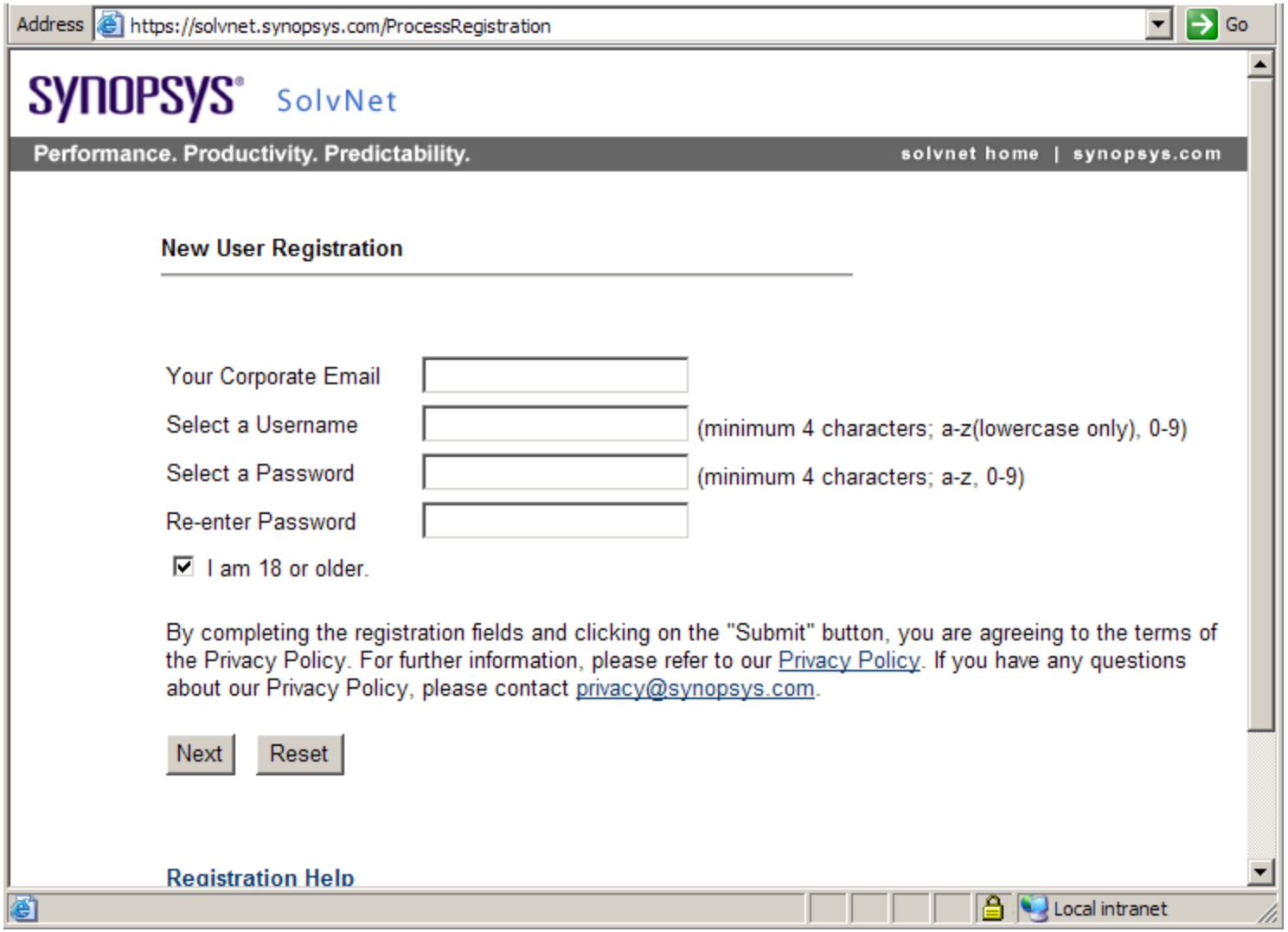
SYNOPSYS®
Predictable Success

# SolvNet Register



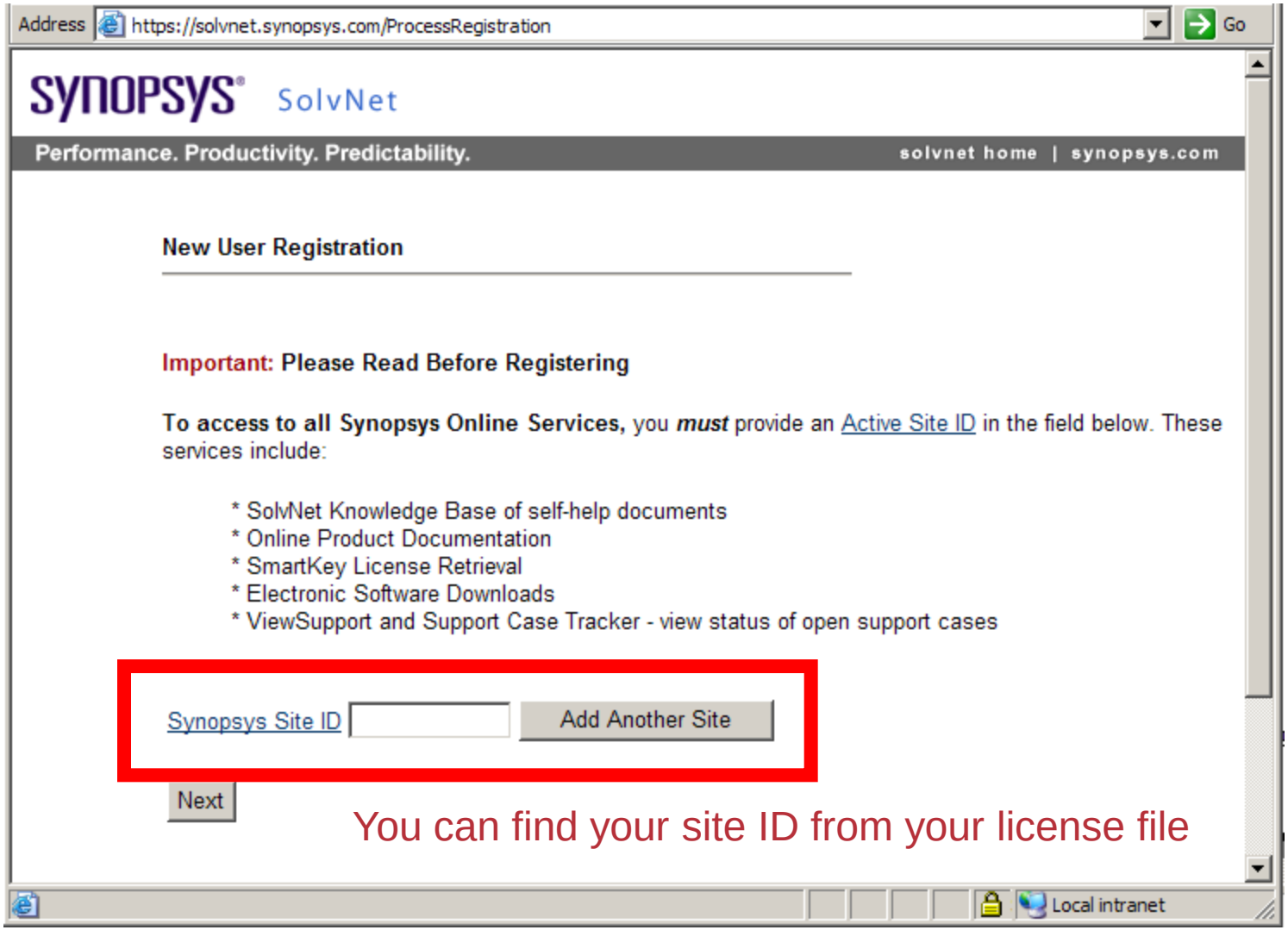

© 2007 Synopsys, Inc. (312)

# SolvNet Register



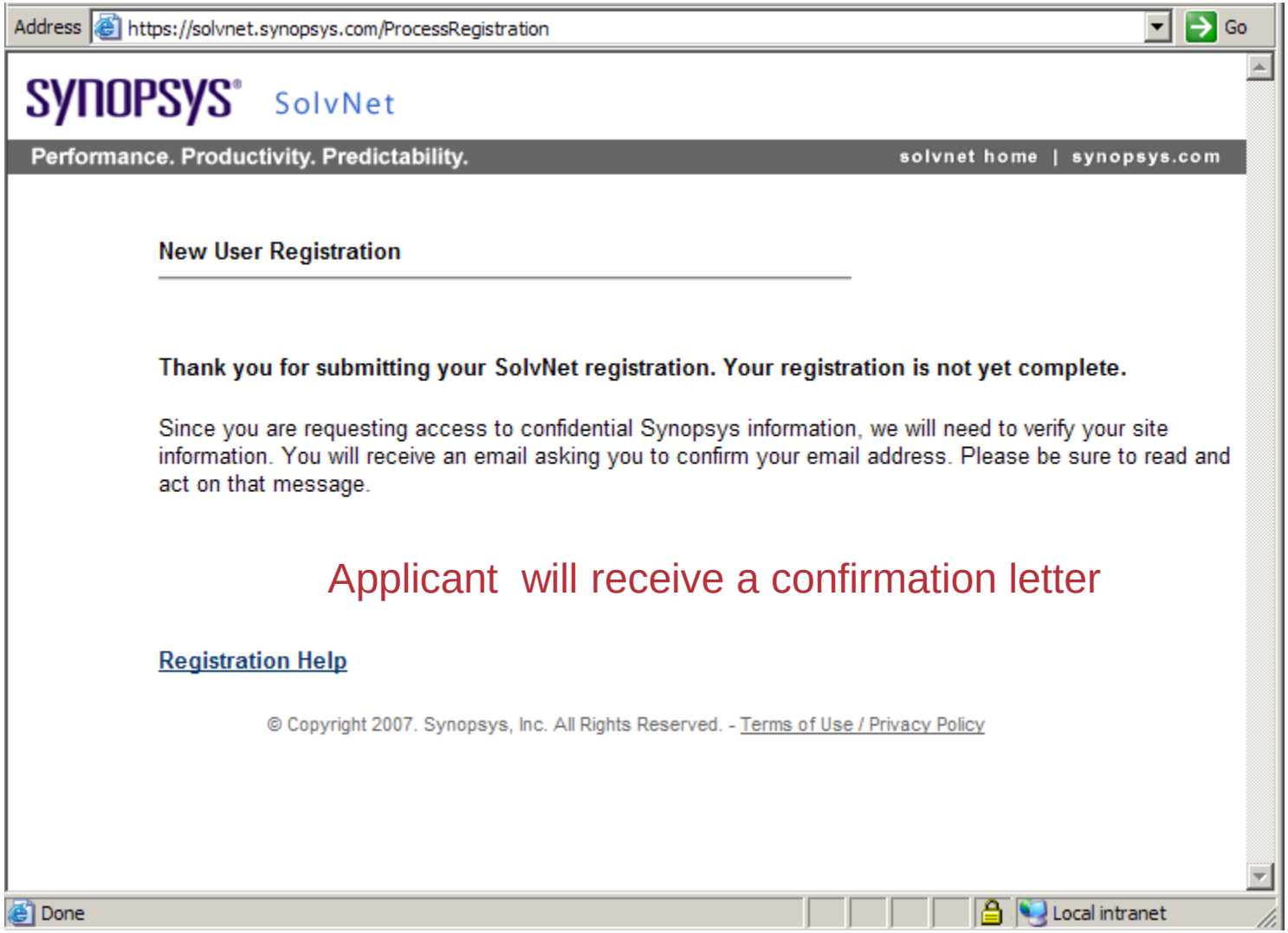

© 2007 Synopsys, Inc. (313)

# SolvNet Register



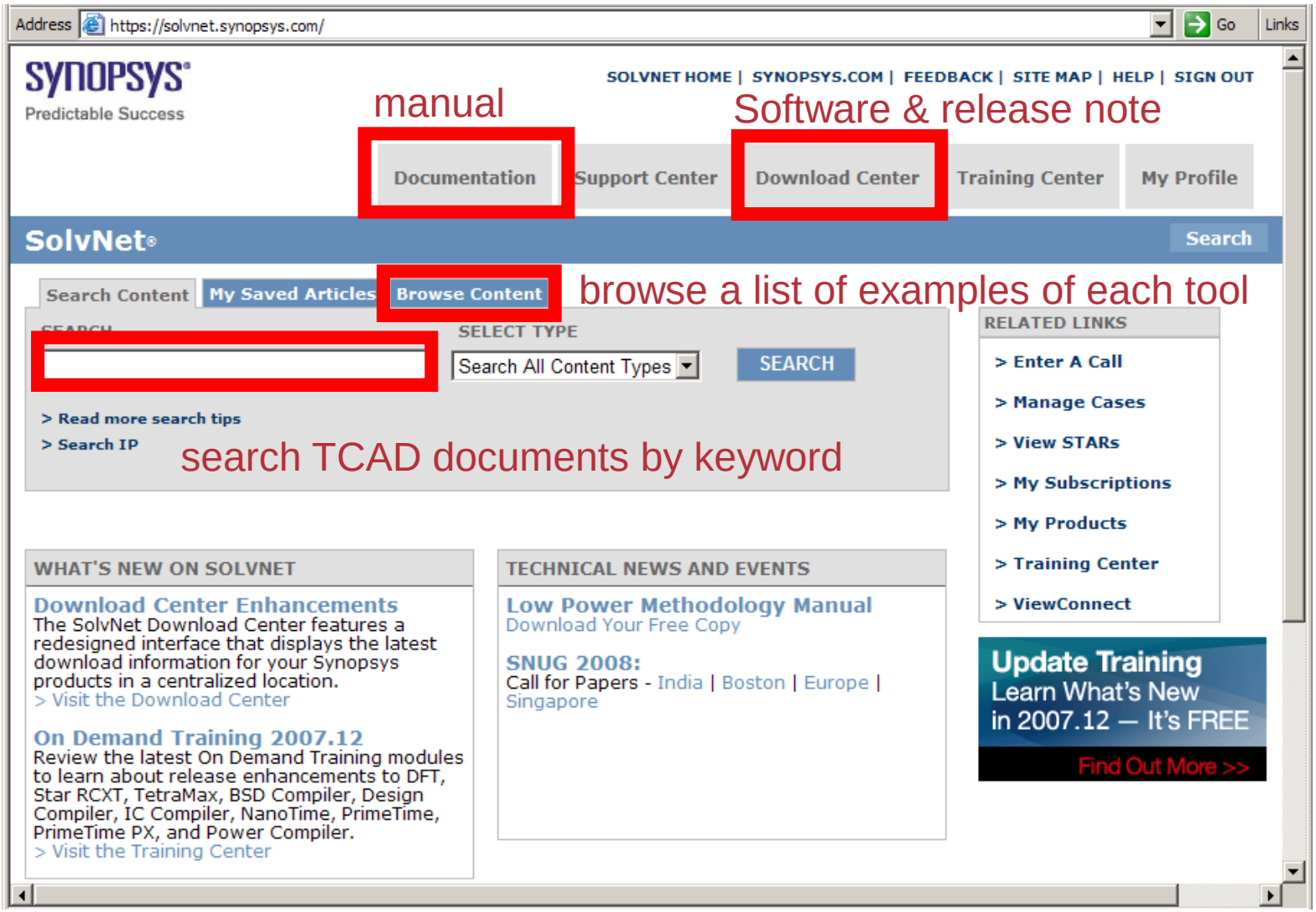

© 2007 Synopsys, Inc. (314)

# SolvNet Register




© 2007 Synopsys, Inc. (315)

# SolvNet HomePage




© 2007 Synopsys, Inc. (316)


# Predictable Success

© 2007 Synopsys, Inc. (317)