Sentaurus TCAD Training for CMOS Application
Course Outline

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

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

CMOS
- Deep submicron transistor
- Sophisticated models
- Atomistic modeling

Memory
- Flash
- DRAM

Power
- Complex structure & process
- Mixed mode simulation

RF
- High-speed device
- Compound semiconductor

TCAD Application Markets

Al_{0.3}Ga_{0.7}As
GaAs
Collector
Emitter
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
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
TCAD Product Architecture

- Sentaurus Workbench
- Sentaurus Device Editor
- Sentaurus Process
- Mesh
- PCM Studio

Drain Current (A/um) vs Drain Voltage (V)
- Strained
- Unstrained
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
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
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
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
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
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
Sentaurus Work Bench (SWB)
Sentaurus Workbench

Ligament

(a GUI tool to assemble process flow and layout data, then translate to the input file of SProcess)

SProcess

Structure Editor

SDevice

SNMesh

Tecplot SV (2D, 3D)

Inspect (1D, XY plot)
Sentaurus Workbench

Menu Bar

Tool Flow

Parameters Bar

Simulation Nodes

Color Chart – indicating the status of each node
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.
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.
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
Editing Tool Command File and Preferences

• Click on the tool icon
• Right click -> Edit Input -> Commands
• To change preferences, Edit Input -> Preferences
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
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.
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.
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.
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.
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.
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.
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
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
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 \( \text{lgate}=0.18, \text{HaloDose}=1e13, \text{HaloEnergy}=25, \) and \( \text{Vds}=0.05 \) 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.
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.
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.
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.
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.
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.
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.
Online Manuals and Training Material
Sentaurus Process (SProcess)
Outline

• Sentaurus TCAD coordinate systems
• Script file setup sequence
• 1D example
• 2D example
• Implement external variable on Sentaurus Workbench
Sentaurus TCAD Coordinate Systems (default setting)

SProcess

Tecplot, SDE, SDevice

Silicon miller indices

[0,0,-1]

[0,0,-1]

[1,-1,0] →

[1,-1,0] →

Y-axis (um)

X-axis (um)

Y-axis (um)

X-axis (um)
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
Ligament commands
Limitation of basic Ligament Flow Editor command

The below table is comparison of supported features.

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

Complicated feature is not supported by basic command.
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!
Basic command rule of Sentaurus Process (1/2)

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

**Example:**

```plaintext
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:

**Example:**

```plaintext
[space]\{value\} 
```

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

→ Error!

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

→ OK!
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**

<table>
<thead>
<tr>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diffuse temperature=1000 &lt;C&gt; time=30 &lt;min&gt;</td>
<td>Error!</td>
</tr>
<tr>
<td>Diffuse temperature=1000&lt;C&gt; time=30&lt;min&gt;</td>
<td>OK!</td>
</tr>
</tbody>
</table>

• Sentaurus Process supports “TCL script” command. Please refer to a reference book or web if you want to know the detail.
Taper etching
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 example is:
  ```
  etch material=PolySilicon type=trapezoidal thickness=0.55 angle=85
  ```
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.
5. Insert “**insert**” command

6. Specify etch of simulator-specific command like below.

7. Finish Ligament Flow Editor and run the project.

---

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

---

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

---

© 2007 Synopsys, Inc. (51)
Ex: taper etching ~the result~
Appendix ~polygon etch~

- Complex figure
  → Use polygon etch to specify coordinates.

The example is:

```plaintext
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 \ }
```
Definition of mesh
Mesh Overview (1/2)

• Mesh is important: Accuracy are dependent on the mesh

• Surplus dense mesh. → Long calculation time.

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

Difference of accuracy by mesh

Dense mesh for strict profile

Spurs mesh

Combination improves the performance!
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.
Mesh Type ~line~

• This is used only for initial mesh (grid) size.

• Mesh size is changed after “mgoals” and “refinebox”.

![Initial grid](image)
Mesh type ~mgoals~ (1/2)

- mgoals means "Mesh and Geometry OperAtions using the Level Set method".

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

- The right example shows after deposition and etching.
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.
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.
Mesh type ~refinebox~ (2/2)

- Strict boundary by oxidation is also appropriate area for refinebox.
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.
Mesh command: mgoals ~usage~ (1/2)

```plaintext
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.
Mesh command: mgoals ~parameter~

\[ \text{min.normal.size} \]

\[ \text{min.normal.size} \times \text{normal.growth.ratio}^n \]

\[ \text{max.lateral.size} \]
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
```
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.
Ex: \texttt{mgoals} \textasciitilde \texttt{the results}~
(min.normal.size)

<table>
<thead>
<tr>
<th></th>
<th>1: n6_ips,ldr 0.0</th>
<th>2: n7_ips,ldr 0.0</th>
<th>3: n8_ips,ldr 0.0</th>
<th>4: n9_ips,ldr 0.0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="#" alt="Graph 1" /></td>
<td><img src="#" alt="Graph 2" /></td>
<td><img src="#" alt="Graph 3" /></td>
<td><img src="#" alt="Graph 4" /></td>
</tr>
<tr>
<td></td>
<td><img src="#" alt="Graph 5" /></td>
<td><img src="#" alt="Graph 6" /></td>
<td><img src="#" alt="Graph 7" /></td>
<td><img src="#" alt="Graph 8" /></td>
</tr>
</tbody>
</table>

0.001 0.002 0.005 0.01
Ex: mgoals ~the results~
(normal.growth.ratio)
Mesh command: refinebox ~usage~

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

grid remesh

\ means that the command continues in next line.

"grid remesh" is necessary just after "refinebox" command.

Mesh size:
- x direction
- y direction

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

13. Add value “1” to the refine parameter. An Experiment with “0” is without refinebox. An Experiment with “1” is with refinebox.
Ex: refinebox ~The results~
Ex: refinebox
~ comparison of 1D profile at y=1.0 ~
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}
```
Extraction
Extraction

• Sentaurus Process supports the extraction, layer thickness, $X_j$ and so on.

• This feature is useful to check these values.

• Some useful command is introduced from next slide.
Extracting oxide thickness

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

```bash
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.
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:

   - `#header` with `title`
   - `substrate` with `dopant`
   - `comment` with `text`
   - `#endheader` with `material`
   - `deposit` with `layer`
   - `etch` with `material`
   - `insert` with `dios`
   - `save` with `basename`
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.
Ex: Extracting oxide thickness
~The result~

• Extracted value is displayed on swb.

<table>
<thead>
<tr>
<th>Project</th>
<th>Scheduler</th>
</tr>
</thead>
<tbody>
<tr>
<td>Family Tree</td>
<td>Variable Values</td>
</tr>
<tr>
<td>sprocess</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>--</td>
</tr>
</tbody>
</table>
Extracting Xj

- Sprocess can extract xj at the specified location with “interpolate” command.

An example for extraction of Xj is:

```bash
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.
Ex: extracting Xj (1/2)

- Examine the following parameters for refinebox

1. Copy the project "/PostBasic/answer/sproces/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:
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.
Ex: extracting $X_j$ ~the result~

- Extracted value is displayed on swb.

<table>
<thead>
<tr>
<th>Project</th>
<th>Scheduler</th>
</tr>
</thead>
<tbody>
<tr>
<td>Family Tree</td>
<td>Variable Values</td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>refine</td>
<td>$X_{j,um}$</td>
</tr>
<tr>
<td>1</td>
<td>--</td>
</tr>
<tr>
<td>2</td>
<td>--</td>
</tr>
</tbody>
</table>
Changing diffusion models
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.
Change Models

The command to change the diffusion model is:

```
pdbSet Silicon Dopant DiffModel ChargedFermi
```

Simulators and available diffusion models

<table>
<thead>
<tr>
<th></th>
<th>SProcess</th>
<th>Dios</th>
<th>TSuprem4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-Stream</td>
<td>ChargedFermi</td>
<td>Equilibrium&lt;Default&gt;</td>
<td>NSTREAMS=1&lt;Default&gt;</td>
</tr>
<tr>
<td></td>
<td>SetDiosEquilibriumModelMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3-Streams</td>
<td>ChargedPair&lt;Default&gt;</td>
<td>PairDiffusion</td>
<td>NSTREAMS=3</td>
</tr>
<tr>
<td></td>
<td>SetDiosPairModelMode</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Caution!**

The default model is different between each simulator. ChargedFermi model is generally appropriate for power device.
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
2D Example

• Copy project “2D_SProcess” from training_library
• Refer to the command file of “2D_SProcess”
2D Example Process Flow

- After POLY re-oxidation
- After HALO implantation
- After Nitride (spacer) deposition

- After spacer etching
- After S/D implantation
- Final structure
2D Example Mesh Setting

**After POLY re-oxidation**

**After HALO implantation**

**After Nitride (spacer) deposition**

**After spacer etching**

**After S/D implantation**

**Final structure**

- Halo/LDD refinebox
- S/D refinebox
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)
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

<table>
<thead>
<tr>
<th>Family Tree</th>
<th>Variable Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1g</td>
</tr>
<tr>
<td></td>
<td>--</td>
</tr>
</tbody>
</table>
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

---

```plaintext
#--- 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@
```
Tecplot SV
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.
Starting Tecplot_SV

• to start Tecplot_SV:

> tecplot_sv &

> tecplot_sv <filename> &

• in SWB: select Visualize → .tdr Files → Tecplot SV
Screen Elements

- field list
- region/material list
- tool buttons
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
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.
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.
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
Generating Multiple Cuts in One Step

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

- In addition to appearance coordination, linking allows synchronized cuts to be made simultaneously on all linked structures.
Sentaurus Structure Editor – Building Meshes (SDE)
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.
Modules of SDE

- Structure creation via primitives
- *Process emulation with PROCEM (3D only)*
- Contact definition
- Doping definition
- Mesh definition
Starting SDE in GUI Mode

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

```
sde
sde -2D
```
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"
)
```
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"
 )
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.
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.
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.
Types of Boolean Expressions

• Merge
  \(\text{sdgeo: set-default-boolean } "AB"\)

• New Replaces Old
  \(\text{sdgeo: set-default-boolean } "ABA"\)

• Old Replaces New
  \(\text{sdgeo: set-default-boolean } "BAB"\)

• New Overlaps Old
  \(\text{sdgeo: set-default-boolean } "ABiA"\)

• Old Overlaps New
  \(\text{sdgeo: set-default-boolean } "ABiB"\)
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
)
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 ) "##")
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")
```
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)
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"
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.
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))
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.
Analytical Profile Placement

(sdedr:define-gaussian-profile
 "AProfileDef_1" "BoronActiveConcentration"
 "PeakPos" 0
 "PeakVal" 1e20
 "ValueAtDepth" 1e15
 "Depth" 0.1
 "Erf" "Factor" 0.8
)

(sdedr:define-analytical-profile-placement
 "AProfilePlacement_1"
 "AProfileDef_1"
 "Myline"
 "Both" "NoReplace" "Eval"
)
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.
Doping Refinement Placement

(sdedr:define-refinement-size
  "RDef_1"
  0.1 0.1 0.1
  0.01 0.01 0.01
)
(sdedr:define-refinement-function
  "RDef_1"
  "DopingConcentration"
  "MaxTransDiff" 1
)
(sdedr:define-refinement-region
  "RPlacement_1"
  "RDef_1"
  "RDef_1"
  "region_3"
)
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.
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"
)
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.
Meshing the Device Structure

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

![Build Mesh dialog box](image_url)
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.
Meshing for Device Simulation

• possible flows:

SP or TS4 process simulation → SDE:
- read boundary
- read doping
- specify new mesh criteria
- call mesh engine
- save structure + doping → SD device simulation

SDE:
- boundary definition
- doping definition
- mesh criteria definition
- call mesh engine
- save structure + doping → SD device simulation
Read boundary

• Read the boundary to be modified.

(sdeio:read-tdr-bnd "n2_bnd.tdr")
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")
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 0.0)))))
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) )
Read profile

• Read the profile data generated by process simulator.

(seddr:define-submesh "ext_pro" "n2_fps.tdr")

(seddr:define-submesh-placement "place_pro" "ext_pro" "win_device"

1. 2. 3. 4.)
Mesh definition (1/2)

- Define mesh size for each Ref/Eval window.

1. Define mesh size for each Ref/Eval window.
2. Place mesh in the device window.
3. Name the mesh as "def_mesh".
4. Set the mesh parameters as follows:
   - Max Element Size: 0.1
   - Min Element Size: 0.0025

```scheme
(sdedr:define-refinement-size
  "def_mesh" 0.1 0.1 0.0025 0.0025)

(sdedr:define-refinement-placement
  "place_mesh" "def_mesh" "win_device")
```
Mesh definition (2/2)

(sdedr:define-refinement-function
  "def_mesh" "DopingConcentration" "MaxTransDiff" 1)
Build Mesh

- Build mesh for device simulation.

```
sde:build-mesh "snmesh" "-a -c boxmethod" "n8"
```
Appendix
~Cut structure~

- Cut the unnecessary region for device simulation

(sdegeo:2d-cut
  (position -0.01 -0.2 0)
  (position 0.34 1 0))
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
Meshing with SDE + SMesh

- refinement in a box

(sdedr:define-refinement-window
   "Sub2.Win" "Cuboid"
   (position xboxr 0.15 0.0)
   (position xbox 0.0 zbox) )
(sdedr:define-refinement-size "Sub2.Def"
  0.05 0.1 0.025
  0.02 0.05 0.025
)
(sdedr: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
Meshing with SDE + SMesh

refinement in a box

refinement in region “Substrate”
Meshing with SDE + SMesh

• graded mesh ("multibox method")

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

graded mesh in y-direction, spacing from 0.01 to 0.001
Meshing with SDE + SMesh

- refinement on a field (doping)

```
(sdedr:define-refinement-size "Doping.Ref"
  0.5  0.5  0.2
  0.004 0.004 0.2)
```

```
(sdedr:define-refinement-function
 "Doping.Ref" "DopingConcentration"
 "MaxTransDiff" 1)
```

```
(sdedr: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”
Meshing with SDE + SMesh

- Graded mesh in poly gate
- Graded mesh in the channel
- Refinement on doping
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”)
Flow of Input and Output in SDE

**Input:**
- Device Structure from SProcess
- Mesh Strategy
  - command_dvs.cmd

**Output:**
- Runtime messages
- structure, doping, mesh
- output_dvs.out
- grid&doping_msh.tdr
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
MOSFET Mesh Setup Result (for SDevice Simulation)
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
Batch Mode

• Refer to command file “sde_dvs.cmd” of example 0.18um_nMOS
• Check output files
The Final Mesh for SDevice
Sentaurus Device (SDevice)
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
Flow of Input and Output in SDevice

Input:
- Models, Parameter, Sweeps
- command _des.cmd
- parameter _desSi.par
- grid & doping _msh.tdr

Output:
- current _des.plt
- plot _des.tdr
- output _des.log
- Runtime messages
- IV's, Field distributions
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”
  }
Overview of SDevice Commands
What Can Sentaurus Device Do?

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

Input:
- Models, Parameter, Sweeps
  - command _des.cmd
  - parameter _desSi.par

Input:
- Device Structure
  - grid & doping _msh.tdr

Output:
- IV’s, Field distributions
  - current _des.pit
  - plot _des.tdr

Output:
- Runtime messages
  - output _des.log
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.
NMOS Roll Off Project

Project: /PostBasic/question/sdevice/nmos_rolloff_sdevice

Description: Simulate NMOS Id-Vg characteristics.
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

```
#set @model=10
File {
  @ input
  grid = "grid";
  parameter = "parametric";
  @ output
  current = "@plot";
  output = "@img";
  plot = "@plot1";
}
Electrode {
  @name=source @ voltage=0.0;
  @name=drain @ voltage=0.0;
  @name=gate @ voltage=0.0;
  @name=substrate @ voltage=0.0;
}
Physics {
  EffectiveElectronDensity ( @ # # )
  Mobility ( # # )
  Saturation Current ( # # )
}
Plot {
  eDensity "# # #" %Current %Current
  Potential SpaceCharge ElctricField Vector
  Mobility %M# %M# %M#
  Doping Dose
  Concentration
}
Math {
  Interpolate %Mesh for the quasistationary solution.
  Iterations=20 %Max. number of steps without solution.
}
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=0.01
```

© 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@"
}
```
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.
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.
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).
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.
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.
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.
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.
Solve Section (1/10)

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

```plaintext
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} }
}
```
Solve Section (2/10)

Coupled \{ \text{Poisson Electron Hole} \}

- The \textit{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.
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.

Drift-Diffusion
Thermodynamic
Id vs. Vg
Solve Section (4/10)

Hydrodynamic Transport

• Solve the carrier temperature equations in addition to the Poisson and carrier continuity equations.
  Coupled \{ Poisson Electron Hole \text{ eTemperature} \text{ hTemperature} \}

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

![Image of hydrodynamic transport model](image-url)

- Hydrodynamic
- Drift-Diffusion
- Id vs. Vg
- 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.
Solve Section (6/10)

Quasistationary Analysis Syntax:

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

    { Coupled{ ... } }
}
```
• 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).
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.
Solve Section (9/10)

- This is a part of out file:

```plaintext
... 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  5.0000E-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.
I-V Curve’s smoothness can be improved by decreasing MaxStep.
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.
Math Section (2/2)

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

$$x_1 = x_0 + \frac{f(x_0)}{f'(x_0)}$$

• 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 $f$.

If $|\Delta x| < \text{error}$, then it converges.
Additional Features
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.
Define Multi-Material Model Parameters In a Parameter File

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

```plaintext
Material="AlGaAs" {
    Epsilon {
        ...
    }
}
Material="GaAs" {
    Epsilon {
        ...
    }
}
```
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.
Plot Statement In the Solve Section (1/2)

```plaintext
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.
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.
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

   t=0    t=0.1    t=0.2
   0000   0001   0002

   0000 0001 0002 0003 0004 0005 0006 0007 0008 0009 0010
   t=1
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.
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
  )
}

Ex 4. Add CurrentPlot Section

Project: /PostBasic/question/sdevice/nmos_currentplot

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

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

![Graph showing current density and potential distribution with Oxide highlighted at (0 0.003)]
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{ ... } }
}
Physics Models

- Mobility
- Generation–recombination
Mobility Models (1/3)

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
Mobility Models (2/3)

Doping-Dependent Mobility Model

- Activated by

\[
\text{Physics}\{\text{Mobility}(\text{DopingDependence} \ldots ) \ldots \}
\]

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

Mobility Degradation At Interfaces

- Activated by

\[
\text{Physics}\{\text{Mobility}(\text{Enormal} \ldots ) \ldots \}
\]

- In the channel region of a MOSFET, the high transverse electric field forces carriers to interact strongly with the semiconductor–insulator interface.
Mobility Models (3/3)

High-Field Saturation Model

• Activated by

\[
\text{Physics}\{\text{Mobility}(\text{HighFieldSaturation} \ldots ) \ldots \}
\]

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

- Activated by

\[
\text{Physics}\{ \text{Recombination(SRH ... ) ...} \}
\]

- SRH recombination through deep defect levels.

- The doping dependence of the SRH lifetimes is activated by

\[
\text{Physics}\{ \text{Recombination(SRH(DopingDependence... ) ...)} ... \}
\]
Auger Recombination

- The Auger model is activated by

\[
\text{Physics}\{\text{Recombination}(\text{Auger} \ldots) \ldots}\]

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

Avalanche Generation

- Activated by

\[
\text{Physics}\{\text{Recombination}(\text{Avalanche} \ldots) \ldots}\]

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

- The breakdown analysis must specify avalanche model.
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:

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

• Run nmos_mobility project, and compare its Id-Vg curve with nmos project’s.
Examples of Analysis
- Mixed-Mode Transient Analysis
- AC Analysis
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.
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.
Input Command File of Mixed-Mode

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

- Mixed-Mode simulation need to specify two new sections:
  - Device section
    - Device section defines physical devices used in the system
  - System section
    - System section defines the netlist of physical devices and circuit elements to be solved.
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.
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 )
  }
}
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.
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.
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.
System Section (3/5)

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

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

<table>
<thead>
<tr>
<th>pulse</th>
<th>Pulse description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sine</td>
<td>Sinusoidal source description</td>
</tr>
<tr>
<td>sin²</td>
<td>Sinusoidal source description</td>
</tr>
<tr>
<td>exp</td>
<td>Exponential source description</td>
</tr>
<tr>
<td>pwl</td>
<td>Piecewise linear description</td>
</tr>
<tr>
<td>sffm</td>
<td>Single-frequency FM description</td>
</tr>
</tbody>
</table>

a. Equivalent to the sine parameter.

• Refer to the following manual for more information.
  ➢ Compact Models User Guide (cm_ug.pdf)
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).
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.
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 }
    }
}
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.
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@"
    } }

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) )
}

File{ Output = "@log@" }
CMOS Transient Characteristic

Vin = v(in)
Vout = v(out)
Example: NMOS Small-Signal AC Analysis

Project: /PostBasic/example/sdevice/ac
Description: Calculate C-V characteristics.
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).
Global File Section

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 $\omega = 2\pi f$. 
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.
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 } }
}
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 (...) 
    ){ ... }
    }
}
```
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.
NMOS C-V Characteristic

\[ C_{bg} = c(b, g) \]

\[ C_{gg} = c(g, g) \]
Ex 6. Observe Transient and AC’s Results

- Observe the following projects’ results.
  - `/PostBasic/example/sdevice/cmos_inverter_transient`
  - `/PostBasic/example/sdevice/ac`
## Quasistationary, Transient and AC Analysis

<table>
<thead>
<tr>
<th>Input Signals</th>
<th>Quasistationary</th>
<th>Transient</th>
<th>AC (Y-parameters)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC</td>
<td>Time-dependent sources</td>
<td>Small-signal AC sources</td>
<td></td>
</tr>
<tr>
<td>Simulation Domain</td>
<td>Time</td>
<td>Frequency</td>
<td></td>
</tr>
<tr>
<td>Need Quasistationary Analysis?</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Mixed-Mode Necessary?</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>“t” Variable</td>
<td>“t” is the internal variable. (0-&gt;1)</td>
<td>“t” is the real time [s].</td>
<td></td>
</tr>
<tr>
<td>Applications</td>
<td>DC operating point ,DC sweep</td>
<td>Transient response</td>
<td></td>
</tr>
<tr>
<td></td>
<td>C-V, FT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Inspect
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
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
Inspect Script

• Inspect can be controlled by using a simple script 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.

<table>
<thead>
<tr>
<th></th>
<th>Family Tree</th>
<th>Variable Values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>sprocess</td>
<td>sde</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>--</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>--</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>--</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>--</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>--</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>--</td>
</tr>
</tbody>
</table>
Basic Operations and Script Commands
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.
Load plt File ~ Operation ~ (1/3)

1. Click “Load Dataset…”
Load plt File ~ Operation ~ (2/3)

2. Set "Files of type"

3. Select plt file

4. Click "Open"
Load plt File ~ Operation ~ (3/3)

5. Dataset name
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.
Create Curve ~ Operation ~ (1/4)

1. Click “Datasets”
Create Curve ~ Operation ~ (2/4)

2. Select “contact”

3. Select “OuterVoltage”

4. Click “To X-Axis”
Create Curve ~ Operation ~ (3/4)

5. Select “contact”

6. Select “TotalCurrent”

7. Click “To Left Y-Axis”
Create Curve ~ Operation ~ (4/4)

8. Curve Name
Create Curve ~ Command ~

cv_createDS TotalCurrent_Drain \ \
{ n2_des gate OuterVoltage } \ \
{ n2_des drain TotalCurrent } y

- 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.
Set Attributes

• This section describes the functions that change the attributes of the axes, curves, and legend.
Set the Axis Attributes

~ Operation : Start ~

1. Double-click the axis
Set the Axis Attributes
~ Operation : Editing Title (1/2) ~

2. Click “Title” tab

3. Change “Title”
Set the Axis Attributes

~ Operation : Editing Title (2/2) ~

4. Click “Font…”

5. Change “Font”, “Size”, “Style”

6. Click “OK”
Set the Axis Attributes
~ Operation : Editing Scale~

7. Click “Scale” tab
8. Change “Min” and “Max”
Set the Axis Attributes

~ Operation: Editing Pattern

(1/2) ~

10. Click “Patterns” tab

11. Click “Color”
Set the Axis Attributes

~ Operation : Editing Pattern

(2/2) ~

12. Click “Patterns” tab

13. Change “Width”
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.
Set the Axis Attributes
~ Operation : Editing Ticks (1/2)

14. Click “Ticks” tab
15. Click “Font…”
16. Change “Font”, “Size” and “Style”
17. Click “OK”
Set the Axis Attributes
~ Operation : Editing Ticks (2/2)

18. Change “Tick Label Angle”

19. Change “Subdivision”
Set the Axis Attributes

~ Operation : Changing Log/Lin

~

20. Click “Scale” tab

21. Change “Scale”
Set the Axis Attributes
~ Operation : Editing Font of Title ~

22. Click “Title” tab
23. Change “Color”
24. Click “OK”
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
Set Curve Drawing Attributes

~ Operation : Start ~

1. Double-click Graph Area
Set Curve Drawing Attributes
~ Operation : Editing General Part

2. Click "General" tab

3. Change "Legend"
Set Curve Drawing Attributes
~ Operation : Editing Line Part
(1/2)~

4. Click “Line” tab

5. Change “Color”
Set Curve Drawing Attributes
~Operation : Editing Line Part (2/2)~

6. Change “Style”
7. Change “Width”
Set Curve Drawing Attributes

~ Operation : Editing Marker (1/4)

~

8. Click “Marker” tab

9. Change “Shape”

10. Change “Size”
Set Curve Drawing Attributes
~ Operation : Editing Marker (2/4)
~
11. Change “Outline Color”
Set Curve Drawing Attributes
~ Operation : Editing Marker (3/4)~

12. Change “Outline Width”
Set Curve Drawing Attributes
~ Operation : Editing Marker (4/4) ~

13. Change “Fill Color”

14. Click “OK”
Set Curve Drawing Attributes
~ Command ~

```
cv_setCurveAttr TotalCurrent_Drain IdVg black solid 3 \ 
          square 5 black 1 black 3. 5. 6. 7. 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.
Compute a Scalar Using the Formula

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

```c
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.
Extract Variables

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

• Examples:

```plaintext
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
```
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.
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.
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.

  \textbf{Script} > \textbf{Record} > \textbf{Start}

  ➢ After selecting the file name, Inspect starts to store every operation until recording is stopped.

  \textbf{Script} > \textbf{Record} > \textbf{Stop}
Inspect Execution Mode

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

  Right button>Edit Input>Preferences…
Mathematical Formulas and Macros
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).
Mathematical Formulas (2/2)

• \texttt{vecmax} (curve)
  ➢ Maximum y-value.
• \texttt{vecmin} (curve)
  ➢ Minimum y-value.
• \texttt{vecvalx} (curve, scalar)
  ➢ x-value at a given y
• \texttt{vecvaly} (curve, scalar)
  ➢ y-value at a given x
• \texttt{veczero} (curve)
  ➢ x-value at y = 0.
• \texttt{diff} (curve)
  ➢ Returns the first derivative of the curve.
• \texttt{integr} (curve)
  ➢ Returns the integral of the curve.
• \texttt{tangent} (curve)
  ➢ Returns the tangent.
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.
Macros (2/3)

- Macro usage:

1. Click “New”
2. Select “VT”
3. Select curve
4. Click “OK”
5. Result
Macros (3/3)

- Macro definition can be edited by Edit>Define Macros…

1. Select “VT”

2. Macro definition

```
vecvalx(tangent(<c 1>), veczero(diff(<c 1>)) - vecmax(diff(<c 1>))
```
Threshold Voltage Extraction
Definitions of Threshold Voltage (1/2)

Definition 1

Maximum slope of the curve

Definition 2

\[ \text{Id} = 1 \times 10^{-7} \text{ A} \]

\[ \text{VT} \]

\[ \text{VT}_1 \]
Definitions of Threshold Voltage (2/2)

• Macros
  VT:
  \[
  \text{vecvalx(tangent(<c1>), veczero(diff(<c1> - vecmax(diff(<c1>))))), 0.0)}
  \]

  VT1:
  \[
  \text{vecvalx(<c1>, 1.0e-7)}
  \]

• Script functions
  VT:
  \[
  \text{f_VT curveName xmin xmax ymin ymax}
  \]
  VT1:
  \[
  \text{f_VT1 curveName xmin xmax ymin ymax}
  \]
  ➤ xmin, xmax, ymin, ymax, the range for computing the result; default values correspond to the full curve range.
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]
Set variables:

<table>
<thead>
<tr>
<th>Command</th>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>set</td>
<td>N</td>
<td>@node@</td>
</tr>
<tr>
<td>set</td>
<td>Vd</td>
<td>0.05</td>
</tr>
<tr>
<td>set</td>
<td>Lgshift</td>
<td>@Lgshift@</td>
</tr>
<tr>
<td>set</td>
<td>Lg</td>
<td>[expr 2.0*$Lgshift+0.065]</td>
</tr>
</tbody>
</table>
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
```
**Inspect Script (4/4)**

Calculate VT:

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

Calculate VT1:

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

Export the variable to the Family Table of Sentaurus Workbench:

```plaintext
ft_scalar Lg [format %0.3f $Lg]
...
ft_scalar vt [format %0.3f $vt]
ft_scalar vt1 [format %0.3f $vt1]
```
90nm nMOS Exercise
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.
Structure Comparison

- 0.18um nMOS
- 90nm nMOS
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 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>
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

- **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
SProcess Mesh Comparison

- 0.18um nMOS
- 90nm nMOS
SDevice Mesh Comparison

- 0.18um nMOS
- 90nm nMOS
Id_Vg Curve Comparison

- **0.18um nMOS**
  - Vdd=1.5v; Vds=0.05v

- **90nm nMOS**
  - Vdd=1.2v; Vds=0.05v
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, Sdevice, 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.
C-V Device Simulation
Breakdown Device Simulation
pMOS Simulation
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
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).
    - $j = Yu = Au + i\omega Cu$
    - $j$ is the vector containing the small-signal currents at all nodes.
    - $u$ is the corresponding voltage vector.
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”
}
C-V Device Simulation Result

- $C_{gg}$: gate node capacitor
- $C_{cg}$: drain node + source node
- $C_{bg}$: substrate node
- $C_{gg} = C_{cg} + C_{bg}$
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.

\[ \Delta V_n \quad \Delta V_{n+1} \]
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”

\[ V_{d_{\text{outer}}} \text{ (force)} \]
\[ V_{d_{\text{inner}}} \text{ (sense)} \]

![MOS Breakdown Graph]

- Negative resistance
- Frequent divergent region

Drain Current (A/μm)

Drain Voltage (V)
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.
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
SolvNet Introduction
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
SolvNet Website http://solvnet.synopsys.com/

Apply an account firstly
SolvNet Register

New User Registration

Have you had a SolvNet Username before?

If Yes:

- I changed jobs and have new Site ID(s). Click here to login and modify your profile, including site ID(s), company and email.
- I changed my email address. Click here to login and modify your profile, including site ID(s), company and email.
- I forgot my username
- I forgot my password

If No:

Register
SolvNet Register

New User Registration

Your Corporate Email

Select a Username
(minimum 4 characters; a-z(lowercase only), 0-9)

Select a Password
(minimum 4 characters; a-z, 0-9)

Re-enter Password

I am 18 or older.

By completing the registration fields and clicking on the “Submit” button, you are agreeing to the terms of the Privacy Policy. For further information, please refer to our Privacy Policy. If you have any questions about our Privacy Policy, please contact privacy@synopsys.com.

Next  Reset

Registration Help
SolvNet Register

You can find your site ID from your license file
SolvNet Register

New User Registration

Thank you for submitting your SolvNet registration. Your registration is not yet complete.

Since you are requesting access to confidential Synopsys information, we will need to verify your site information. You will receive an email asking you to confirm your email address. Please be sure to read and act on that message.

Applicant will receive a confirmation letter
SolvNet HomePage

- search TCAD documents by keyword
- browse a list of examples of each tool
- manual
- Software & release note

WHAT'S NEW ON SOLVN

Download Center Enhancements
The SolvNet Download Center features a redesigned interface that displays the latest download information for your Synopsys products in a centralized location.

On Demand Training 2007.12
Review the latest On Demand Training modules to learn about release enhancements to DFT, Star RCXT, TetraMax, BSD Compiler, Design Compiler, IC Compiler, NanoTime, PrimeTime, PrimeTime PX, and Power Compiler.

Low Power Methodology Manual
Download Your Free Copy

SNUG 2008:
Call for Papers - India | Boston | Europe | Singapore