SYNOPSYS®

Sentaurus Technology Template: CMOS Characterization

Abstract

This TCAD Sentaurus simulation project provides a template setup for far-submicron CMOS device characterization.

I_d–V_{gs} curves for a low drain bias and high drain bias are simulated for NMOS and PMOS structures with various gate lengths. In addition, for selected structures, a family of I_d–V_{ds} curves for various gate biases is simulated and small-signal simulations are performed to obtain the C–V characteristics of the respective devices. For each of the simulated curves, relevant electrical parameters, such as threshold voltages and drain current levels, are extracted. Finally, the extracted parameters are plotted as a function of the gate length.

Version Information

This TCAD simulation template has been designed and verified using TCAD Sentaurus Version H-2013.03.

Introduction

This project provides standard templates for the device simulator Sentaurus Device that can be used to perform the most common types of simulation used in CMOS device characterization and performance assessment. It contains device simulation setups for I_d–V_{gs} simulation at a low drain bias and a high drain bias, setups for the simulation of a family of I_d–V_{ds} curves, and setups for capacitance simulations.

The setup of the Sentaurus Workbench project supports NMOS and PMOS devices, and allows for the simulation of structures with different gate lengths. Other variations can be included by adjusting the template.

The project discusses the extraction library of the visualization tool Sentaurus Visual to extract the most common electrical parameters such as the threshold voltage, transconductance, subthreshold voltage swing, drain saturation current, drain leakage current, on-state resistance, and gate capacitance.

Finally, the project discusses an extraction library procedure for plotting extracted electrical parameters as a function of Sentaurus Workbench input parameters. A typical example is a V_t roll-off curve.

The NMOS and PMOS devices are defined by using a fully parameterized input file to the Sentaurus Structure Editor. The geometric dimensions, such as gate oxide thickness and spacer width, as well as doping profiles parameters, such as peak concentration and junction depth, can be easily adjusted to specific needs or can become Sentaurus Workbench parameters to allow for the creation of new device families or the plotting of other characteristics, such as the dependence of the threshold voltage on the halo implant depth.

The template setup can also be used to investigate the electrical properties of device structures created by the process simulator Sentaurus Process (or others). In this case, Sentaurus Structure Editor must be replaced with Sentaurus Process. An example of such a setup can be found in the Sentaurus Workbench template project *Sentaurus Technology Template: CMOS Processing*.

It is assumed that users are familiar with the Sentaurus tool suite, in particular, Sentaurus Workbench, Sentaurus Structure Editor, Sentaurus Device, and Sentaurus Visual. For an introduction and tutorials, refer to the TCAD Sentaurus Tutorial.

The focus of this project is to provide a setup that can be used as is or adapted to specific needs. The documentation focuses on aspects of the setups. For details about tool uses and specific tool syntax, refer to the respective manuals.

General Simulation Setup

This section describes the tool flow of the Sentaurus Workbench project. For each tool, the associated Sentaurus Workbench input parameters and the extracted parameters are discussed.

Sentaurus Structure Editor

Sentaurus Structure Editor defines the analytic MOS structures, and it uses two Sentaurus Workbench parameters:

- `Type = nMOS|pMOS` defines whether an NMOS or a PMOS device is created.
- `lgate` [µm] defines the gate length for the MOS devices. Here, it assumes the values `0.045`, `0.065`, `0.09`, `0.13`, `0.25`, and `1.0`. You can add or remove values as required.

Sentaurus Device: IdVg_lin

The name of the first instance of Sentaurus Device is `IdVg_lin`. It performs a low drain bias I_d–V_{gs} sweep for the given devices. The supply voltage V_{dd} and the drain bias V_{ds} are defined by the Sentaurus Workbench parameters:

- `Vdd` [V] defines the supply voltage. Here, it is set to `1.2`. (For the PMOS devices, the sign of the supply voltage is changed internally.)
- `Vdlin` [V] defines the drain bias for the I_d–V_{gs} sweep. Here, it is set to `0.05`. (For the PMOS devices, the sign of the drain bias is changed internally.)
- `IdVg_lin = 0|1` is a logical flag. The I_d–V_{gs} sweep is performed only if the flag is `1`.

Sentaurus Visual: PlotIdVg_lin

The name of the subsequent visualization tool Sentaurus Visual is `PlotIdVg_lin`. It plots the low drain bias I_d–V_{gs} characteristics and extracts:

- `Vtgm` [V]: Threshold voltage defined as the intersection of the tangent at the maximum g_m with the V_{gs} axis.
- `VtiLin` [V]: Threshold voltage defined as V_{gs} at which $I_d = 100\ \text{nA}/L_g$ for V_{ds} = `Vdlin`.
- `IdLin` [A/µm]: I_d at V_{ds} = `Vdlin` and V_{gs} = `Vdd`.
- `gmLin` [S/µm]: Maximum transconductance at V_{ds} = `Vdlin`.
- `SSlin` [mV/decade]: Subthreshold voltage swing at V_{ds} = `Vdlin`.

Sentaurus Device: IdVg_sat

The name of the next instance of Sentaurus Device is `IdVg_sat`. It performs a high drain bias I_d–V_{gs} sweep. The drain bias V_{ds} for this simulation is set to the supply voltage `Vdd`:

- `IdVg_sat = 0|1` is a logical flag. The I_d–V_{gs} sweep is performed only if the flag is `1`.

Sentaurus Visual: PlotIdVg_sat

The name of the subsequent Sentaurus Visual instance is `PlotIdVg_sat`. It plots the high drain bias I_d–V_{gs} characteristics and extracts:

- `VtiSat` [V]: Threshold voltage defined as V_g at which $I_d = 100\ \text{nA}/L_g$ for V_{ds} = `Vdd`.
- `IdSat` [A/µm]: Saturation current I_d at $V_{ds} = V_{gs}$ = `Vdd`.
- `Ioff` [A/µm]: Off-state (drain-leakage) current I_d at V_{ds} = `Vdd` and V_{gs} = 0 V.
- `gmSat` [S/µm]: Maximum transconductance at V_{ds} = `Vdd`.
- `SSsat` [mV/decade]: Subthreshold voltage swing at V_{ds} = `Vdd`.

Sentaurus Device: IdVd

The name of the following instance of Sentaurus Device is `IdVd`. It simulates a family of I_d–V_{ds} curves. The gate biases are defined by the Sentaurus Workbench parameters:

- `Vgmin` [V] defines the gate bias for the first drain voltage sweep. It is set to `0.3`. (For the PMOS devices, the sign of the gate bias is changed internally.)
- `Vgmax` [V] defines the gate bias for the last drain voltage sweep. It is set to `1.2`. (For the PMOS devices, the sign of the gate bias is changed internally.)
- `NVg` [1] defines the number of sweeps. It is set to `4`. The gate biases are linearly distributed between the minimum and the maximum values.
- `IdVd = 0|1` is a logical flag. The I_d–V_{ds} sweep is performed only if the flag is `1`.

Sentaurus Visual: PlotIdVd

The name of the subsequent Sentaurus Visual instance is `PlotIdVd`. It plots the family of I_d–V_{ds} curves and extracts:

- `Ron` [Ωµm]: On-state output resistance extracted from the I_d–V_{ds} curve with the highest gate bias (V_{gs} = `Vdd`) and at a drain voltage of $\pm 0.8\,V_{dd}$ ($0.8\,V_{dd}$ for NMOS and $-0.8\,V_{dd}$ for PMOS).

Sentaurus Device: CV

The name of the final instance of Sentaurus Device is `CV`. It performs a small-signal analysis at a fixed frequency during a gate sweep from V_{gs} = –`Vdd` to V_{gs} = +`Vdd`:

- `CV = 0|1` is a logical flag. The CV simulation is performed only if the flag is `1`.

Sentaurus Visual: PlotCV

The name of the subsequent Sentaurus Visual instance is `PlotCV`. It plots the gate capacitance, the gate–source or gate–drain overlap capacitance, and the gate–body capacitance as a function of the gate voltage and extracts:

- `CgM` [F/µm]: Gate capacitance at V_{gs} = –`Vdd`.
- `Cg0` [F/µm]: Gate capacitance at V_{gs} = 0 V.
- `CgP` [F/µm]: Gate capacitance at V_{gs} = `Vdd`.

Sentaurus Visual: RollOff

The name of the final Sentaurus Visual instance is `RollOff`. It collects the extracted values for each device and plots `Vtgm`, `VtiLin`, `VtiSat`, drain-induced barrier lowering (DIBL), `IdLin`, `IdSat`, `Ioff`, `SSlin`, `SSsat`, `gmLin`, and `gmSat` as a function of the gate length `lgate` (roll-off curves). In addition, the I_{on}–I_{off} characteristics are shown. The Sentaurus Workbench parameter:

- `RollOff = 0|1` is a logical flag. The roll-off curves are plotted only if the flag is `1`. It is sufficient to set the flag to `1` for only a single device in each class. For example, for the NMOS and PMOS with `lgate` = 1.0.

The project has been structured to allow for flexibility. For example, new devices or experiments can be added by selecting **Experiments > Add New Experiment** from the menu bar of the Sentaurus Workbench main window.

An experiment can be edited by changing the values of the parameters. Furthermore, different experiments can be grouped into scenarios. In this project, the experiment for the NMOS and PMOS devices are grouped into scenarios named `nMOS` and `pMOS` to simplify management of the project.

Tool-specific Setups

Device Generation Using Sentaurus Structure Editor and Sentaurus Mesh

Sentaurus Structure Editor defines the NMOS and PMOS devices in a fully parameterized way.

Figure 1 shows the NMOS device with a 90-nm gate length. To adjust the details of the devices, you can modify the top section of the input file `sde_dvs.cmd` of Sentaurus Structure Editor. For example, the substrate background doping level and the peak concentration of the halo implant are defined by setting the Scheme variables:

```
(define SubDop 5e17)     ; [1/cm3]
(define HaloDop 1.5e18) ; [1/cm3]
```

The junction depth for the halo, the extension, and the source/drain implants are defined by setting the Scheme variables:

```
(define XjHalo 0.07)  ; [um] Halo depth
(define XjExt  0.026) ; [um] Extension depth
(define XjSD   0.12)  ; [um] SD Junction depth
```

The extend of the nitride spacer and the gate oxide thickness are defined by setting the Scheme variables:

```
(define Lsp 0.1)      ; [um] Spacer length
(define Tox 20e-4)    ; [um] Gate oxide thickness
```

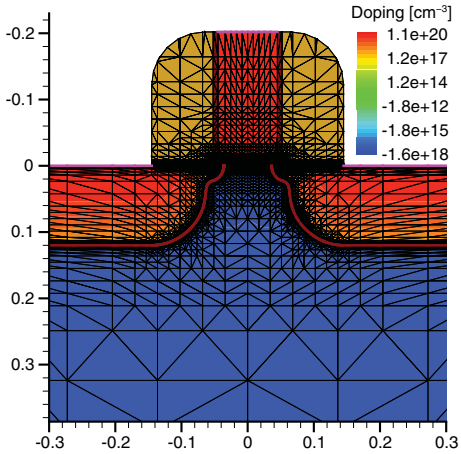

Figure 1 The 90-nm gate length NMOS device generated by Sentaurus Structure Editor and meshed with Sentaurus Mesh; concentrations of dopants in various regions are shown

Several other geometric, doping, and meshing parameters are accessible in a similar way. The meshing strategy is designed to result in a high quality without excessive node counts for a large range of geometric parameters.

Sentaurus Structure Editor calls the meshing engine Sentaurus Mesh to generate the structure files for Sentaurus Device. Sentaurus Mesh is called from within Sentaurus Structure Editor with:

```
(sde:build-mesh "snmesh" " " "n@node@_half_msh")
```

It must be noted that only half of the CMOS structure is created by Sentaurus Structure Editor and meshed with Sentaurus Mesh.

It is subsequently reflected about the vertical axis to obtain the full device. The reflection is performed in the Sentaurus Structure Editor by a system call to the utility Sentaurus Data Explorer (`tdx`):

```
(system:command "tdx -mtt -x -ren drain=source
  n@node@_half_msh n@node@_msh")
```

The option `-x` instructs Sentaurus Data Explorer to reflect the device along an axis defined by $x = x_{min}$. The given half-structure has three contacts: `drain`, `gate`, and `substrate`, which are defined in `sde_dvs.cmd`. Of these, the `gate` and `substrate` contacts touch the axis of reflection and, upon reflection, are extended and thereby preserve their names. However, the `drain` contact in the reflected half is named `drainmirrored` by default. This contact is explicitly renamed `source` with the command-line option `-ren` of Sentaurus Data Explorer.

Device Simulation Using Sentaurus Device

Sentaurus Device simulates the drain current as a function of the gate voltage at a low drain bias and a high drain bias (I_d–V_{gs}) as well as to simulate a family of drain current curves as a function of the drain voltage (I_d–V_{ds}) for different values of the gate bias. Furthermore, a small-signal AC analysis is performed to show the total gate, gate–contact, and gate–body capacitance as a function of the gate voltage (C–V).

All I–V simulations are performed using the hydrodynamic transport model, where the carrier temperature equation for both electrons and holes is solved with the electrostatic Poisson equation and the carrier continuity equations. It is not necessary to solve the carrier temperature equation for the C–V simulations. For the high drain bias I_d–V_{gs} as well as the I_d–V_{ds} simulations, lattice self-heating effects are included by also solving the lattice temperature equation.

Size quantization effects are included in all simulations by using the density gradient model. For the I–V simulations, it is sufficient to activate the density gradient model for the dominant carriers only, because it will be effective mainly in the inversion layer. For the C–V simulations, however, the model must be activated for electrons and holes, because size quantization occurs in accumulation as well.

For all simulations, the Lucent mobility model is used and high-field saturation effects are accounted for.

Sentaurus Device selects automatically the appropriate driving-force model for the high-field saturation model as follows:

```
Physics(Material="Silicon"){...
  Mobility(...
    HighFieldSaturation
  )
}
```

If the simulation is performed using the hydrodynamic transport model for a particular carrier (electrons for NMOS and holes for PMOS), Sentaurus Device automatically selects `CarrierTempDrive`. In addition, if the simulation is performed using the drift-diffusion transport model for a particular carrier (holes for NMOS and electrons for PMOS), `GradQuasiFermi` is selected automatically.

The Tcl block preprocessing mode of Sentaurus Workbench is used to distinguish the Sentaurus Device input files for the NMOS and PMOS devices. In this mode, any text enclosed in:

```
!( <Tcl-block> )!
```

is treated as a Tcl script block, which is executed at preprocessing time. The text itself is removed during preprocessing, with the exception of the output of the Tcl `puts` commands. For example, to activate the hydrodynamic transport model for the appropriate carriers for the NMOS and PMOS devices, the following Tcl block is used:

```
!(
if { "@Type@" == "nMOS" } {...
  set cTemp "eTemperature"
} else {...
  set cTemp "hTemperature"
}
)!

!(
Physics{...
  Hydrodynamic( !(puts $cTemp)! )
}
)!
```

For the NMOS devices, this segment of the input file of Sentaurus Device is translated during Sentaurus Workbench preprocessing to:

```
Physics{...
  Hydrodynamic( eTemperature )
}
```

Figure 2 shows the I_d–V_{gs} curves for the 90 nm PMOS and NMOS devices at V_{ds} = 50 mV and V_{ds} = 1.2 V.

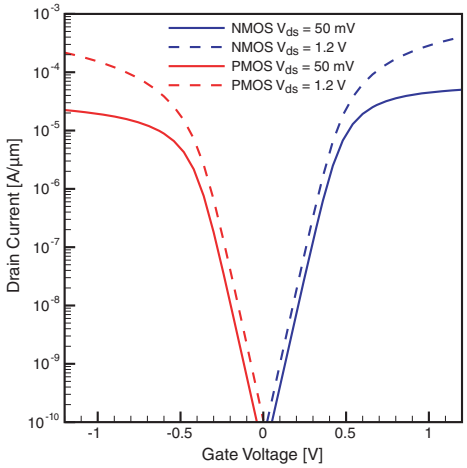

Figure 2 Drain current as a function of gate voltage for the 90 nm gate length PMOS (red) and NMOS (blue) devices simulated with Sentaurus Device; lower curves (solid lines) are for a drain bias of 50 mV and upper curves (dashes) are for 1.2 V

For the I_d–V_{ds} simulation, the input of Sentaurus Device that defines the family of curves is created with the following Tcl blocks.

The first Tcl block generates the `Time` string, which determines at which values of the gate voltage a solution is saved:

```
Quasistationary( ...
  Goal { Name="gate" Voltage=@Vgmax@ }
  ){ Coupled { Poisson ... }
    Save( FilePrefix="IdVd_n@node@" NoOverWrite
!(
  set TIMES "Time=("
  for { set i 1 } { $i < @NVg@ } { incr i } {
    set dV [expr (@Vgmax@-@Vgmin@)/(@NVg@-1.0)]
    set Time [expr (@Vgmin@ + ($i-1)*$dV)/@Vgmax@]
    append TIMES "[format %.2f ${Time}];"
  }
  set Time 1.0
  append TIMES "[format %.2f ${Time}]) )"
  puts $TIMES
)!
}
```

The Sentaurus Workbench parameters `Vgmin` and `Vgmax` define the first and last gate bias, and `NVg` defines the number of curves (gate biases) used in the family of I_d–V_{ds} curves.

The Tcl block of the input file given above is translated during Sentaurus Workbench preprocessing to:

```
Quasistationary(
    Goal { Name="gate" Voltage=1.2 }
  ){ Coupled { Poisson ... }
    Save( FilePrefix="IdVd_n162" NoOverWrite
      Time=(0.25;0.50;0.75;1.00))
  }
```

The next Tcl block generates the input file segments, which reload the previously saved solution for a given gate bias and perform the drain voltage sweep:

```
!(
  for { set i 0 } { $i < @NVg@ } { incr i } {
  set Number [format "%04d" $i]
  puts "
   NewCurrentFile=\"IdVd_${i}_\"
   Load(FilePrefix=\"IdVd_n@node@_$Number\")
   Quasistationary(
    DoZero
    InitialStep=1e-3 Increment=1.35
    MinStep=1e-5 MaxStep=0.05
    Goal \{ Name=\"drain\" Voltage=[expr $SIGN*@Vdd@]
\}
  )\{ Coupled \{ $EQNS \}
    CurrentPlot( Time=(Range=(0 1) Intervals=20) )
   \}"
  }
)!
```

After preprocessing, the Tcl block is expanded to:

```
NewCurrentFile="IdVd_0_"
  Load(FilePrefix="IdVd_n162_0000")
  Quasistationary(
   DoZero
   InitialStep=1e-3 Increment=1.35
   MinStep=1e-5 MaxStep=0.05
   Goal { Name="drain" Voltage=1.2 }
  ){ Coupled { Poisson eQuantumPotential Electron
Temperature eTemperature }
  CurrentPlot( Time=(Range=(0 1) Intervals=20) )
  }

  NewCurrentFile="IdVd_1_"
  Load(FilePrefix="IdVd_n162_0001")
  Quasistationary( ... )
  }

...
```

Figure 3 shows the I_d–V_{ds} curves for the 90 nm PMOS and NMOS devices at V_{gs} = 0.6, 0.9, and 1.2 V.

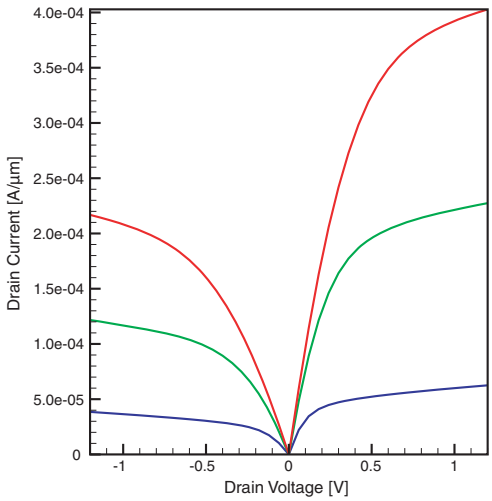

Figure 3 Drain current as a function of drain voltage for the 90 nm gate length PMOS (*left*) and NMOS (*right*) devices simulated with Sentaurus Device; gate bias for curves is 0.6 V (blue), 0.9 V (green), and 1.2 V (red)

The AC analysis is performed at 1 MHz during a gate voltage sweep from $-1.5 \cdot$ `Vdd` to $+1.5 \cdot$ `Vdd`.

Figure 4 shows the C–V curves for the 90 nm NMOS device.

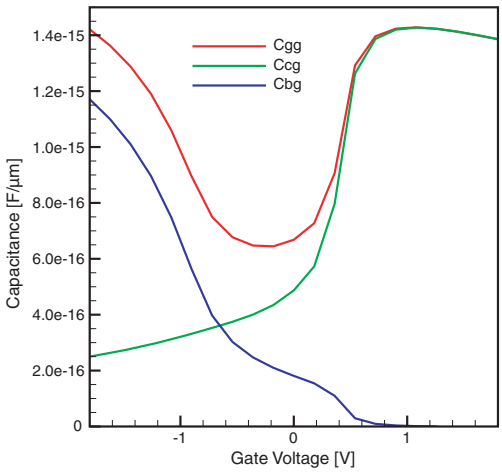

Figure 4 Total gate (Cgg, red), gate-contacts (Ccg, green), and gate-body (Cbg, blue) capacitance as a function of gate voltage for 90 nm gate length NMOS device simulated with Sentaurus Device

NOTE Sentaurus Device is changing to the following `Math` section parameters:

```
Math { ...
  eMobilityAveraging= ElementEdge
  hMobilityAveraging= ElementEdge
  GeometricDistance
  ParameterInheritance= Flatten
}
```

The `Math` section parameters are shown here with their new default values (refer to the TCAD Sentaurus release notes to see which of these changes have taken effect in the current release). Tests have shown that these parameter settings, under certain unfavorable conditions (for example, when too coarse meshes are used), result in more accurate results and more robust simulations. For the given simulation project, the changes to these parameters have no noticeable effect. Nevertheless, to facilitate a smooth transition to the new default behavior, these `Math` section parameters have been updated to their future default values in this version of the simulation project. In addition, note that with the new parameter setting `ParameterInheritance= Flatten`, a region will inherit parameters from a materialwise parameter specification (see *Sentaurus Device User Guide* for further details).

Extraction and Visualization with Sentaurus Visual

In the Sentaurus Workbench tool flow, each instance of Sentaurus Device is followed by an instance of the visualization tool Sentaurus Visual. This tool plots the

corresponding I–V or C–V characteristics, and extracts relevant electrical parameters as discussed in General Simulation Setup on page 3.

Visualizing Simulation Results Using Sentaurus Visual

Sentaurus Visual commands are used to plot the electrical characteristics of the MOSFET. For details about these commands, refer to [1]. The Sentaurus Visual commands used for plotting a curve are illustrated using an example that plots an I_d–V_{gs} curve in the linear region (Sentaurus Visual command file `PlotIdVg_lin_vis.tcl`):

```
set N @node@
set Type @Type@
set N ${Type}_${N}
# Load plt file and create dataset PLT_LIN($N).
load_file IdVg_n@node|IdVg_lin@_des.plt \
  -name PLT_Lin($N)

# Create plot named Plot_IdVg if it does not
# already exist.
if {[lsearch [list_plots] Plot_IdVg] == -1} {
  create_plot -1d -name Plot_IdVg
}
select_plots Plot_IdVg

# Create lists containing gate voltage values and
# drain current values.
set Vgs [get_variable_data "gate OuterVoltage" \
  -dataset PLT_Lin($N)]
set Ids [get_variable_data "drain TotalCurrent" \
  -dataset PLT_Lin($N)]

# Compute absolute value of drain currents using
# extraction library procedure ext::AbsList.
# Store the values in the Tcl list absIds.
ext::AbsList out= absIds x= $Ids

# Create variable absId using the Tcl list absIds.
create_variable -name absId -dataset PLT_Lin($N) \
  -values $absIds

# Create Id-Vg curve.
create_curve -name IdVgLin($N) -dataset PLT_Lin($N) \
  -axisX "gate OuterVoltage" -axisY "absId"
```

A dataset loaded from a `.plt` file contains variables corresponding to the terminal voltages, currents and so on. Curves can be created either from these variables or by computing new variables.

Here, since the PMOS drain current is negative, its I_d–V_{gs} curve is created after computing the absolute value of the drain current using the procedure `ext::AbsList`, storing these values in the Tcl variable `absIds`, and creating the variable `absId` in the dataset `PLT_Lin($N)`. The variables `"gate OuterVoltage"` and `"absId"` are used to create the I_d–V_{gs} curve:

```
create_curve -name IdVgLin($N) -dataset PLT_Lin($N) \
  -axisX "gate OuterVoltage" -axisY "absId"
```

Since the NMOS drain current is positive, its I_d–V_{gs} curve can be created directly using the variable `"drain TotalCurrent"`:

```
create_curve -name IdVgLin($N) -dataset PLT_Lin($N) \
  -axisX "gate OuterVoltage" \
  -axisY "drain TotalCurrent"
```

Curves from one or multiple Sentaurus Visual nodes can be visualized in a single Sentaurus Visual tool instance using the visualization wrapper script integrated in Sentaurus Workbench. To launch the visualization wrapper script:

1. Select the Sentaurus Visual nodes.
2. Select **Extensions > Preprocess and Run Selected Visualizer Nodes Together**.

For multiple-node curves, the wrapper script merges the selected Sentaurus Visual inputs into one command file and then evaluates the file. Therefore, to avoid any unexpected behavior while visualizing multiple-node curves using the wrapper script, the following requirements must be met:

- Sentaurus Visual nodes must be either executed or preprocessed before the wrapper script is launched since Sentaurus Workbench looks for the file `pp@node@_vis.cmd` to be available.
- In the Sentaurus Visual master input file, you must ensure that the curve name and the dataset name include a unique node identifier, for example, `$N`:

  ```
  set N @node@
  set Type @Type@
  set N ${Type}_${N}
  create_curve -name IdVgLin($N) \
    -dataset PLT_Lin($N) \
    -axisX "Vgate" -axisY "absId"
  ```

  This helps the wrapper script to distinguish the curves to be visualized. The name of the curve being visualized automatically contains the corresponding node identifier.
- Ensure that a Sentaurus Visual plot with a particular plot name is created only if it does not exist. In addition, the plot must be selected before curves are plotted using the command `select_plots`.

Figure 5 on page 9 shows the multiple I_d–V_{gs} curves for four different cases (four Sentaurus Workbench experiments).

Extraction Using Extraction Library of Sentaurus Visual

The extractions are performed using the extraction library of Sentaurus Visual. In addition, the extraction library has some utility procedures for operating on lists. For example, the procedure `ext::AbsList` computes the absolute value of all elements of a list.

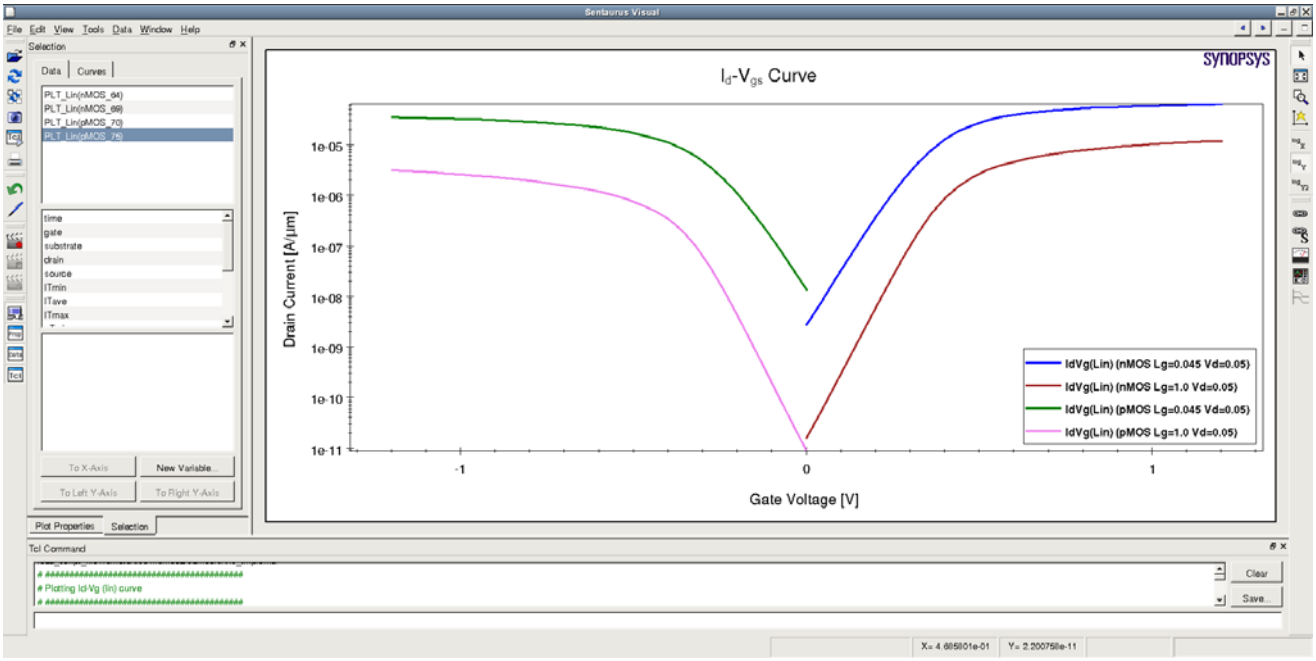

Figure 5 I_d–V_{gs} (lin) curve from four different experiments (four nodes) plotted in a single Sentaurus Visual tool instance

The extraction library is loaded automatically when Sentaurus Visual starts. However, if you have disabled the automatic loading of extension libraries, you can load the extraction library explicitly with the command:

```
load_library extract
```

(Refer to [1] for more information about this library.)

All procedures of the extraction library are called using the syntax:

```
ext::<proc_name> keyword= value
```

Here, `<proc_name>` is the name of a procedure of the extraction library, and `ext::` is a unique namespace identifier.

Each procedure has several arguments. The procedure arguments have the form:

```
keyword= value
```

These arguments can be specified in any order. For some keywords, default values are predefined. Such arguments may be omitted. The keywords and their default values, if any, are documented in [1].

For example, the procedure `ext::ExtractVtgm` extracts the threshold voltage from an I_d–V_{gs} curve using the maximum transconductance method as follows:

```
ext::ExtractVtgm out= Vt name= "Vtgm" \
  v= $Vgs i= $absIds
```

The I_d–V_{gs} curve is represented by two Tcl lists: `Vgs` and `absIds`. The `Vgs` list contains the gate voltage points, and the `absIds` list contains the corresponding absolute value of the drain current values. The keywords `v` and `i` of the procedure `ext::ExtractVtgm` are set to `Vgs` and `absIds`, respectively (`v= $Vgs i= $absIds`).

All procedures of the extraction library pass the results back to the calling program by storing the results in a Tcl variable. The name of this Tcl variable is specified as the value of the `out` keyword. Here, the extracted threshold voltage is stored in the Tcl variable `Vt` since `out= Vt`.

All procedures of the extraction library beginning with `ext::Extract` pass the extracted value to the Sentaurus Workbench Family Tree (if the `name` keyword differs from `"noprint"`). The extracted quantity is displayed as a Sentaurus Workbench variable. Here, since `name= "Vtgm"`, the extracted threshold voltage value is displayed as the Sentaurus Workbench variable `Vtgm`.

All procedures of extraction library beginning with `ext::Extract` also print the extracted value in the output file (with the extension `.out`). The amount of information to be printed in the output file depends on the information level specified using the procedure `ext::SetInfoDef`.

Here, the following command sets the information level to 1 for all procedures of the extraction library:

```
ext::SetInfoDef 1
```

As a result, the following lines are printed in the Sentaurus Visual run-time output file:

```
DOE: Vtgm 0.316
Vtgm (Max gm method): 0.316
```

If the extraction library procedure cannot extract the parameter, the parameter is set to the character 'x' and a message is printed in the output file. In the case of `ext::ExtractVtgm`, the following message is printed:

```
DOE: Vtgm x
ext::ExtractVtgm: Vtgm not found!
```

Extracting CMOS Parameters

The procedure `ext::ExtractVtgm` for extracting the threshold voltage using maximum transconductance method is discussed in Extraction Using Extraction Library of Sentaurus Visual on page 8.

The procedures `ext::ExtractVti`, `ext::ExtractSS`, `ext::ExtractGm`, and `ext::ExtractIoff` are used to extract the threshold voltage for a given subthreshold current level, subthreshold voltage swing, maximum transconductance, and drain leakage current, respectively.

`IdLin` and `IdSat` are extracted using the procedure `ext::ExtractExtremum`, which extracts the maximum of a curve if the keyword `extremum` is set to `max`:

```
ext::ExtractExtremum out= Idmax name= "IdLin" \
  x= $Vgs y= $absIds extremum= "max"
```

The on-state resistance `Ron` is extracted from the I_d–V_{ds} curve with the highest gate bias (V_{gs} = `Vdd`) and at the drain voltage of 80% of `Vdd` for NMOS using the procedure `ext::ExtractRdiff`:

```
ext::ExtractRdiff out= Ron name= "out" \
  v= $Vds i= $Ids vo= [expr 0.80*$Vdd]
```

Here, the Tcl list `Vds` contains the drain voltage points and the Tcl list `Ids` contains the corresponding drain current values.

The gate capacitances mentioned in Sentaurus Visual: PlotCV on page 4 are extracted using the procedure `ext::ExtractValue`. This procedure extracts, for a given target x-value, the (first) corresponding interpolated y-values in a curve.

For example, `Cg0` is extracted using:

```
ext::ExtractValue out= Cgg0 name= "Cg0" \
  x= $Vgs y= $Cggs xo= 0.0
```

Refer to [1] for more details about the abovementioned procedures of the extraction library.

Plotting Roll-off Curves with Sentaurus Visual

The last instance of Sentaurus Visual in the Sentaurus Workbench tool flow gathers the extracted values and plots them for each device as a function of the gate length. These roll-off curves are generated using the extraction library procedure `ext::FilterTable`.

The Sentaurus Visual script `RollOff_vis.tcl` uses the dynamic preprocessing feature of Sentaurus Workbench `@<parameter_name>:all@` to access the list of input parameters and extracted values. For example, the lists `Types`, `Lgs`, and `Vtgms` are generated automatically as a result of the following commands in the Sentaurus Visual script `RollOff_vis.tcl`:

```
set Types   [list @Type:all@]
set Lgs     [list @lgate:all@]
set Vtgms   [list @Vtgm:all@]
```

Here, the Tcl list `Types` contains, for all experiments, the values of the Sentaurus Workbench input parameter `Type`, which for example takes on the values `nMOS` or `pMOS`, depending on whether in this experiment an NMOS or a PMOS structure is created.

Similarly, the Tcl list `Lgs` contains, for all experiments, a 'parallel' list of values of another Sentaurus Workbench input parameter, which for example contains the value of the gate length of the given MOSFETs. The corresponding extracted parameter can be accessed in the same way. For example, the Tcl list `Vtgms` contains the extracted values for the threshold voltage for each respective experiment.

These lists are used to plot the roll-off curves. However, to plot the roll-off curves only for NMOS (PMOS) devices, the values for all the PMOS (NMOS) devices must be filtered out from the above lists, and new lists containing values only for NMOS (PMOS) devices must be created. This filtering is performed by the `ext::FilterTable` procedure. The syntax of this procedure is:

```
ext::FilterTable out= <var_name> x= <list_of_r> \
  y= <list_of_r> conditions= <array_name> \
  ncond= <i>
```

The lists of x- and y-values, which are to be processed (filtered) for creating the graph, are specified using the keywords `x` and `y`. For example, the V_t roll-off curves for NMOS devices are created using the command:

```
ext::FilterTable out= LgVtgm x= $Lgs y= $Vtgms \
  conditions= Conditions ncond= 2
```

Here, the lists of gate lengths (`x= $Lgs`) and threshold voltage values (`y= $Vtgms`) are processed by `ext::FilterTable`.

The keyword `conditions` controls the conditions an experiment must fulfill to be included in the graph. The total number of conditions is specified by the keyword `ncond`. All conditions are specified in a string-indexed array using the keyword `conditions`. Each condition is defined by both a target value and a corresponding list of Sentaurus Workbench parameters. The target value is the required value of the parameter to be used as a filter condition.

Each element of the string-indexed array has two indices. The first index is either `"Target"` or `"Values"`. The

second index is the condition number. For each condition number:

- The target value is specified using the `"Target"` element (element with first index named `"Target"`) of the array.
- The corresponding list of Sentaurus Workbench parameters is specified using the `"Values"` element (element with first index named `"Values"`) of the array.

For example, to filter out all gate lengths (L_g) and threshold voltage (V_{tgm}) values for NMOS devices, this condition is defined using the array named `Conditions`:

```
set Conditions(Target,1) "nMOS"
set Conditions(Values,1) $Types
```

The keyword `conditions` is set to the array named `Conditions` (`conditions= Conditions`).

Here, only one condition is defined. Therefore, `ncond= 1`. The target value is `"nMOS"` and the Sentaurus Workbench list is `$Types`. This selects all experiments for the NMOS devices.

The procedure returns an array (specified by the keyword `out`) with a one string-valued index. The index contains the elements X and Y. The values of the X element are a subset of a list of values, specified using the keyword `x`. These values are in ascending order. The values of the Y element are a subset of a list of values, specified using the keyword `y`. These lists in the array can be used to create a graph.

Here, the procedure returns the array `LgVt` (`out= LgVt`) consisting of a list of L_g values and a list of V_{tgm} values for NMOS devices. The L_g and V_{tgm} values for PMOS devices are not included. These lists can be used directly for creating the V_t roll-off curve for NMOS devices as follows:

```
create_plot -1d -name Plot_VtRollOff
create_variable -name Lg -dataset VtgmLg($N) \
  -values $LgVtgm(X)
create_variable -name Vtgm -dataset VtgmLg($N) \
  -values $LgVtgm(Y)
create_curve -name Vtgm($N) -dataset VtgmLg($N) \
  -axisX "Lg" -axisY "Vtgm"
```

Refer to [1] for an example of using `ext::FilterTable` for multiple conditions.

As an additional feature, the `ext::FilterTable` procedure uses the character ‘x’ as a marker, which identifies that a certain electrical parameter was not extracted or could not be extracted. The procedure removes these entries automatically.

In the Sentaurus Visual plotting and extraction input files (`Plot*_vis.tcl`), each extracted variable is initialized with the Sentaurus Workbench preprocessor directive `#set` to the ‘remove’ character. For example, at the beginning of `PlotIdVg_lin_vis.tcl` are the following lines:

```
#set Vtgm   x
#set VtiLin x
#set IdLin  x
#set SSlin  x
#set gmLin  x
```

As discussed in Extraction Using Extraction Library of Sentaurus Visual on page 8, if the extraction of a parameter fails, the extraction procedure sets the parameter to the character `x`.

In addition, if an I–V or a C–V simulation is suppressed by setting the corresponding Sentaurus Workbench logical flag to 0, for example `IdVd=0` (see General Simulation Setup on page 3), the values of the corresponding extracted parameters remain at their initialization value `x`.

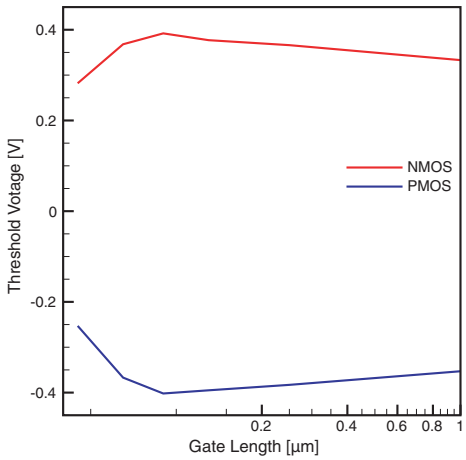

Figure 6 Threshold voltage as function of gate voltage for NMOS (red) and PMOS (blue) devices

References

[1] *Sentaurus Visual User Guide*, Version H-2013.03, Mountain View, California: Synopsys, Inc., 2013.