

Chapter 2

2D MOSFET Simulation

2.1 Complementary MOS (CMOS) Technology

Since the development of metal-oxide-semiconductor field-effect transistor (MOSFET) started in 1950s, relevant technologies have been constantly improving [1]. An N-channel MOSFET is shown in Fig. 2.1a, which is also known as nMOSFET or nFET. It is called N-channel because the conduction channel (which is the inversion layer) is filled with inversion electrons (N-type carriers) as shown in Fig. 2.1b. Figure 2.1c, d both are showing P-channel MOSFET (inversion holes), which is also known as pMOSFET or pFET. The V_g and V_d of these two kinds of transistors are both ranging from 0 V to V_{dd} . The body of nFET is connected to the lowest voltage of circuit, 0 V, as shown in Fig. 2.1b. Therefore, the PN junction is always in a reverse bias or no bias such that there will not be any forward bias current. When V_g is equaled to V_{dd} and larger than threshold voltage (V_{th}) $V_g = V_{dd} > V_{th}$, the electron inversion layer will appear, and the nFET will be conducting. On the other hand, the source and body of pFET are connected to V_{dd} as shown in Fig. 2.1d, thus the applied V_g is just the opposite of nFET. When $V_g = V_{dd}$, the nFET is conducting and the pFET is off. On the contrary, when $V_g = 0$, nFET is off and pFET is conducting.

The low power circuit designed based on the complementary characteristic of nFET and pFET is called complementary MOS (or CMOS) circuit as shown in Fig. 2.2a. There is a circle at the gate with the circuit symbol for pFET, meaning this circuit is an inverter. It will charge the load capacitor C at the output terminal to V_{dd} or discharge it to 0 V in accordance with the command of gate voltage V_g . When $V_g = V_{dd}$, nFET is conducting and pFET is off (these two transistors are regarded as a simple switch), and the output terminal is pulled down to ground point ($V_{out} = 0$). When $V_g = 0$, nFET is off and pFET is conducting, and the output terminal V_{out} is

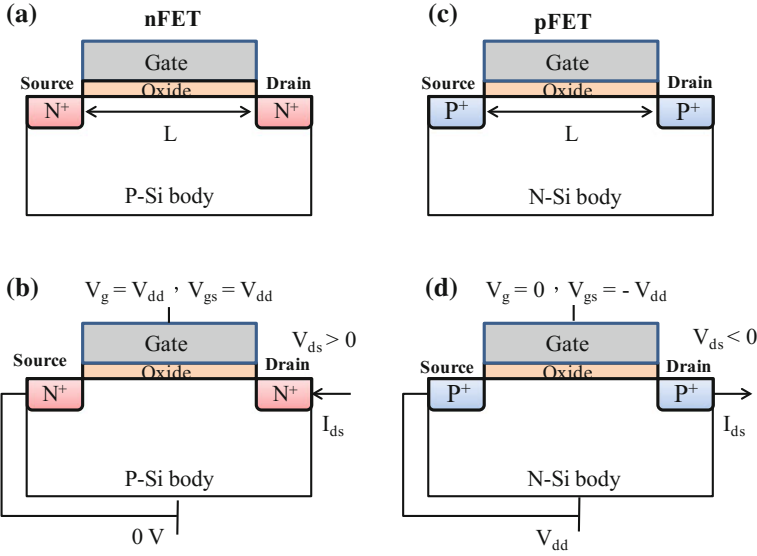


Fig. 2.1 a N-channel FET is in off-state, b N-channel FET is in on-state, c P-channel FET is in off-state, d P-channel FET is in the on-state [1]

pulled up to V_{dd} . Figure 2.2b shows how nFET and pFET are fabricated on the same chip. An N-type well is formed in a portion of area in the P-type silicon substrate by implantation or diffusion of N-type dopant. The contacts of P-type silicon substrate and N-type well are both shown in the figure. The layout of basic CMOS inverter is as shown in Fig. 2.2c. This is an image of top-down view of silicon wafer, which is also the image composed of several overlapped masks for fabrication of inverter. V_{in} , V_{out} , V_{dd} , and ground point voltage are all based on metal lines. Polysilicon gate or metal gate is the vertical line connected to V_{in} .

In this chapter, the physical and electrical properties of MOSFET will be investigated by 2D TCAD simulation. It is shown in Fig. 2.2 that the current is flowing from the high potential terminal to the low potential terminal.

The current (I_{ds}) in Fig. 2.3 can be derived from the charge concentration and width of inversion layer as shown below:

$$\begin{aligned} I_{ds} &= W \times Q_{inv}(x) \times v = W \times Q_e \times \mu_{ns} \times E \\ &= W \times C_{oxe}(V_{gs} - V_{cs} - V_t)\mu_{ns}dV_{cs}/dx \end{aligned} \quad (2.1)$$

$$\int_0^L I_{ds}dx = WC_{oxe}\mu_{ns} \int_0^{V_{ds}} (V_{gs} - V_{cs} - V_t)dV_{cs} \quad (2.2)$$

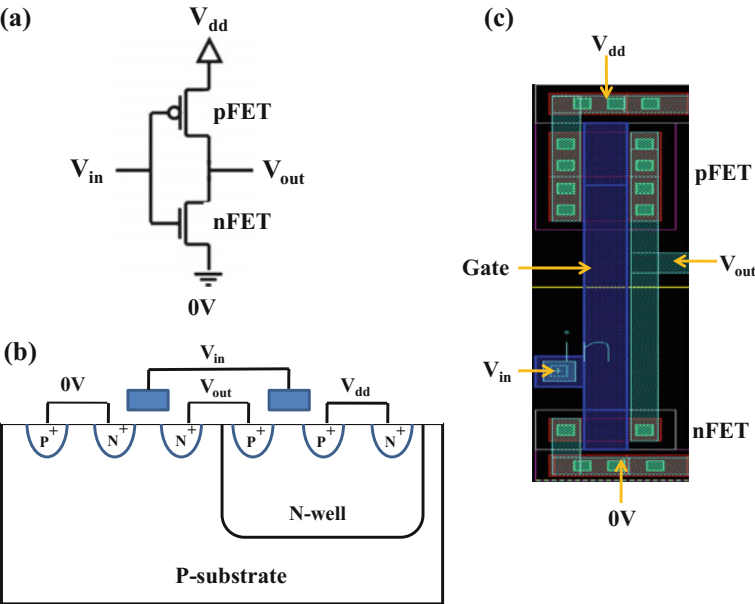
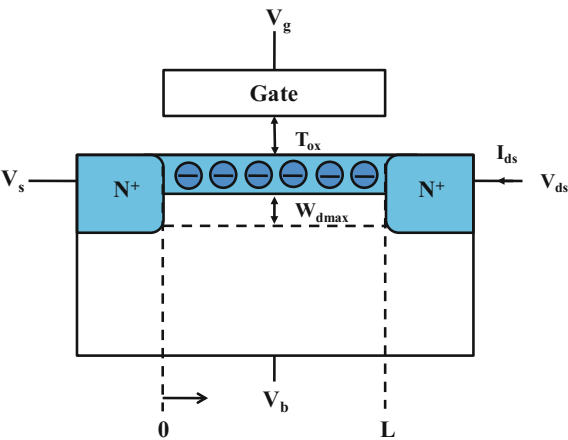


Fig. 2.2 Three figures of CMOS Inverter, **a** CMOS inverter circuit consists of a pFET pull-up device and a nFET pull-down device, **b** nFET and pFET device structures are integrated in a same chip, and **c** the layout of CMOS inverter

Fig. 2.3 Operating mechanism of nMOSFET



$$I_{ds}L = WC_{oxe}\mu_{ns}\left(V_{gs} - V_t - \frac{1}{2}V_{ds}\right)V_{ds} \quad (2.3)$$

$$I_{ds} = \frac{W}{L}C_{oxe}\mu_{ns}\left(V_{gs} - V_t - \frac{1}{2}V_{ds}\right)V_{ds} \quad (2.4)$$

Equation (2.4) reveals that I_{ds} is proportional to W , μ_{ns} , V_{ds}/L (average electric field in the channel), and $C_{ox}(V_g - V_t - \frac{1}{2}V_{ds})$. The $C_{ox}(V_g - V_t - \frac{1}{2}V_{ds})$ can reflect the inversion electron density Q_{inv} (C/cm²) in the channel. C_{oxe} is effective gate capacitance (F/cm²).

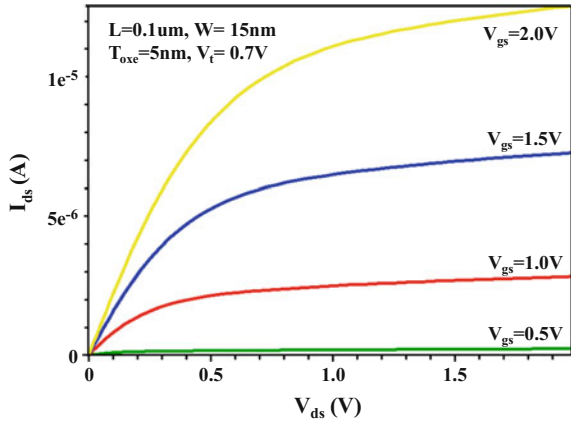
When V_{ds} is very small, the item of $1/2 V_{ds}$ of (2.4) can be neglected, so I_{ds} is proportional to V_{ds} , which means the conduction behavior of transistor under such voltage making it just like a resistor. The I - V When V_{ds} is increased, Q_{inv} , and dI_{ds}/dV_{ds} are decreased. The differentiation of Eq. (2.4) with respect to V_{ds} will result in dI_{ds}/dV_{ds} , which will be equal to 0 under a specific V_{ds} . At this moment, the V_{ds} is called V_{dsat} as shown below:

$$\frac{dI_{ds}}{dV_{ds}} = 0 = \frac{W}{L}C_{oxe}\mu_{ns}(V_{gs} - V_t - V_{ds}), \quad \text{when } V_{ds} = V_{dsat} \quad (2.5)$$

$$V_{dsat} = V_{gs} - V_t \quad (2.6)$$

V_{dsat} is called drain saturation voltage, and different V_{gs} will lead to different V_{dsat} . The region with V_{ds} far less than V_{dsat} is called linear region, and the region with V_{ds} greater than V_{dsat} is called “saturation region.” The part of I - V curve of Fig. 2.4 with $V_{ds} \ll V_{dsat}$ is the “linear region.”

Fig. 2.4 Output characteristics of nMOSFET



The current in saturation region can be derived from Eq. (2.4) as:

$$I_{dsat} = \frac{W}{2L} C_{oxe} \mu_{ns} (V_{gs} - V_t)^2 \quad (2.7)$$

In addition, the transconductance is defined as:

$$g_m \equiv dI_{ds}/dV_{gs}|_{V_{ds}} \quad (2.8)$$

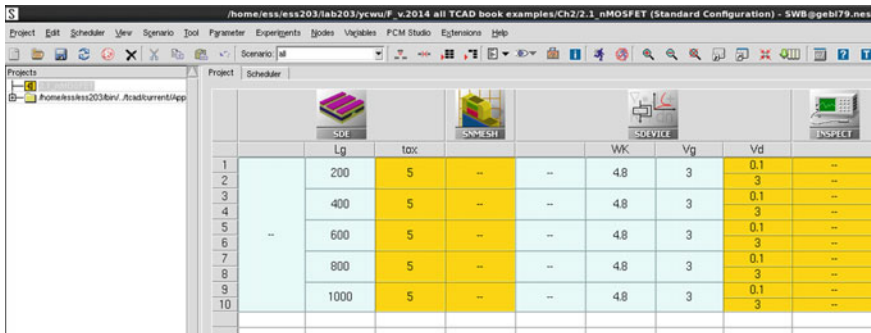
And the saturation transconductance is defined as:

$$g_{msat} = \frac{W}{L} C_{oxe} \mu_{ns} (V_{gs} - V_t) \quad (2.9)$$

The simulation and analysis of 2D MOSFET will be described in the following sections. Designers often refer to this region as the active region. In the last section, we will discuss about TCAD Simulation of 2D nMOSFET and pMOSFET.

2.2 [Example 2.1] 2D n-Type MOSFET with I_d - V_g Characteristics Simulation

The N-type 2D nMOSFET with different $L_g = 200, 400, 600, 800, 1000$ nm is used as an example for the introduction of simulation technology. First, the Synopsys Sentaurus TCAD 2014 version is used to establish the **four tools of SDE, SNMESH, SDEVICE, and INSPECT** as shown in Fig. 2.5.



	Lg	tox	WK	Vg	Vd
1	200	5	4.8	3	0.1
2					3
3	400	5	4.8	3	0.1
4					3
5	600	5	4.8	3	0.1
6					3
7	800	5	4.8	3	0.1
8					3
9	1000	5	4.8	3	0.1
10					3

Fig. 2.5 Required simulation four tools are shown in the workbench of 2D MOSFET simulation

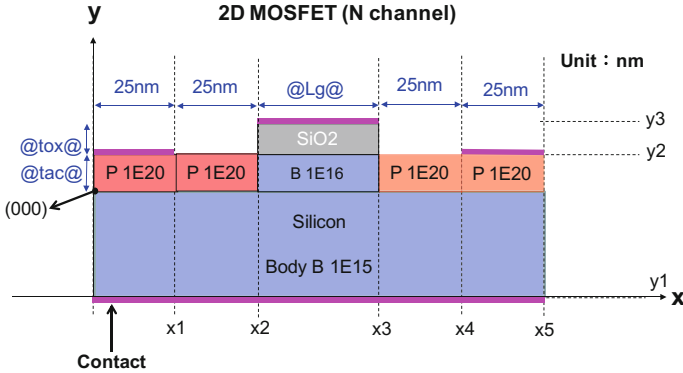


Fig. 2.6 Cartoon plot of 2D MOSFET structure for simulation

Now, we start to establish the device structure. The commands icon on **SDE** tool should be right-clicked to enter codes to establish the structure. At first, we need to draw a cartoon plot as shown in Fig. 2.6 of 2D MOSFET structure for simulation.

The structure of 2D MOSFET to be established is as shown in Fig. 2.6.

1. SDE → devise_dvs.cmd

Now, we use Fig. 2.6 to explain the SDE tool commands (code file is devise_dvs.cmd)

In principle, the flow of structure establishment is as shown below:

- (1) **Set zero point and coordinates**
- (2) **Composition of 2D Structure**
- (3) **Composition of 3D Structure and 2D Y-cut diagram (in Chap. 3)**
- (4) **Set 2D rectangles or 3D cuboids (in Chap. 3)**
- (5) **Set Electrodes**
- (6) **Set doping region**
- (7) **Set Mesh**

The codes of devise_dvs.cmd can be divided into six parts.

- (1) **Parameter**
- (2) **Structure**
- (3) **Contact**
- (4) **Doping**
- (5) **Mesh**
- (6) **Save**

The complete codes of this 2D MOSFET example are as shown below:

;---- (1). parameter ----;

(define Lg @Lg@) ;set Lg as variables Lg = 200, 400, 600, 800, 1000 nm

(define tox @tox@) ;set tox as variables

(define tac 100)

(define Body 400)

(define LSDC 25)

(define LSD 25)

(define C_Doping 1e16)

(define D_Doping 1e20)

(define S_Doping 1e20)

(define B_Doping 1e15)

(define nm 1e-3)

(define x1 LSDC)

(define x2 (+ x1 LSD))

(define x3 (+ x2 Lg))

(define x4 (+ x3 LSD))

(define x5 (+ x4 LSDC))

(define y1 (- Body))

(define y2 tac)

(define y3 (+ tac tox))

;---- (2). Structure ----;

"ABA"

;--- source ---

(sdegeo:create-rectangle

(position 0 0 0)

(position x1 y2 0) "Silicon" "SourceC")

(sdegeo:create-rectangle

(position x1 0 0)

(position x2 y2 0) "Silicon" "Source")

;--- Channel ---

(sdegeo:create-rectangle

(position x2 0 0)

(position x3 y2 0) "Silicon" "Channel")

```

;--- Drain ---
(sdegeo:create-rectangle
  (position    x3 0 0)
  (position    x4 y2 0) "Silicon" "Drain" )
(sdegeo:create-rectangle
  (position    x4 0 0)
  (position    x5 y2 0) "Silicon" "DrainC" )
;--- Body ---
(sdegeo:create-rectangle
  (position    0 0 0)
  (position    x5 y1 0) "Silicon" "Body" )
;--- Gate oxide ---
(sdegeo:create-rectangle
  (position    x2 y2 0)
  (position    x3 y3 0) "SiO2" "Gateoxide" )

```

; ---- (3). Contact ----;

```

;---- Gate ----
(sdegeo:define-contact-set "G" 4.0 (color:rgb 1.0 0.0 0.0 ) "##")
(sdegeo:define-2d-contact (find-edge-id (position (+ x2 (/ Lg 2)) y3 0)) "G")
;---- Source ----
(sdegeo:define-contact-set "S" 4.0 (color:rgb 1.0 0.0 0.0 ) "##")
(sdegeo:define-2d-contact (find-edge-id (position 10 tac 0)) "S")
;---- Drain ----
(sdegeo:define-contact-set "D" 4.0 (color:rgb 1.0 0.0 0.0 ) "##")
(sdegeo:define-2d-contact (find-edge-id (position (+ 50 Lg 35) tac 0)) "D")
;---- Substrate ----
(sdegeo:define-contact-set "substrate" 4.0 (color:rgb 1.0 0.0 0.0 ) "##")
(sdegeo:define-2d-contact (find-edge-id (position (+ x2 (/ Lg 2)) (- Body) 0))
"substrate")

```

;---- (4). Doping ----;

```

;--- Channel ---
(sdedr:define-constant-profile "dopedC" "BoronActiveConcentration" C_Doping )
(sdedr:define-constant-profile-region "RegionC" "dopedC" "Channel" )

```

```

;--- Source ---
(sdedr:define-constant-profile "dopedS" "PhosphorusActiveConcentration"
S_Doping )
(sdedr:define-constant-profile-region "RegionS" "dopedS" "Source" )
(sdedr:define-constant-profile "dopedSC" "PhosphorusActiveConcentration"
S_Doping )
(sdedr:define-constant-profile-region "RegionSC" "dopedSC" "SourceC" )
;--- Drain ---
(sdedr:define-constant-profile "dopedD" "PhosphorusActiveConcentration"
D_Doping )
(sdedr:define-constant-profile-region "RegionD" "dopedD" "Drain" )
(sdedr:define-constant-profile "dopedDC" "PhosphorusActiveConcentration"
D_Doping )
(sdedr:define-constant-profile-region "RegionDC" "dopedDC" "DrainC" )
;--- Body ---
(sdedr:define-constant-profile "dopedB" "BoronActiveConcentration" B_Doping )
(sdedr:define-constant-profile-region "RegionB" "dopedB" "Body" )
;---- (5). Mesh ----;
;--- AllMesh ---
(sdedr:define-refinement-size "Cha_Mesh" 20 20 0 10 10 0)
(sdedr:define-refinement-material "channel_RF" "Cha_Mesh" "Silicon" )
;--- ChannelMesh ---
(sdedr:define-refinement-window "multiboxChannel" "Rectangle"
(position 25 (- 50) 0)
(position (+ 50 Lg 25) (+ tac 50) 0))
(sdedr:define-multibox-size "multiboxSizeChannel" 5 5 0 1 1 0)
(sdedr:define-multibox-placement "multiboxPlacementChannel"
"multiboxSizeChannel" "multiboxChannel")
(sdedr:define-refinement-function "multiboxPlacementChannel"
"DopingConcentration" "MaxTransDiff" 1)

;---- (6). Save (BND and CMD and rescale to nm) ----;
(sde:assign-material-and-region-names (get-body-list) )
(sdeio:save-tdr-bnd (get-body-list) "n@node@_nm.tdr")
(sdedr:write-scaled-cmd-file "n@node@_msh.cmd" nm)

```

```
(define sde:scale-tdr-bnd
(lambda (tdrin sf tdrout)
  (sde:clear)
  (sdegeo:set-default-boolean "XX")
  (sdeio:read-tdr-bnd tdrin)
  (entity:scale (get-body-list) sf)
  (sdeio:save-tdr-bnd (get-body-list) tdrout)
)
)
(sde:scale-tdr-bnd "n@node@_nm.tdr" nm "n@node@_bnd.tdr")
;----- END -----;
```

Now, we explain the code file of **devise_dvs.cmd** item by item.
In Devise commands, the codes behind “;” and “#” are the two prompt characters as the notes by program designer which will not be executed by computer.

Silicon is selected as the active device material. The length of source and drain is set at 50 nm. To be able to create the contact boundary within 50 nm, source and drain are each composed of two extra 2D rectangles, and the length of each rectangle is 25 nm. **@Lg@ is the symbol of @variable@**, and the value can be set as the variable in **Sentaurus WorkBench (SWB)** in Fig. 2.5. The benefit in doing this is that multiple variables L_g can be assigned all at once to analyze the difference of 2D MOSFET under different L_g as shown in **SWB** of Fig. 2.7.

The thickness of gate oxide **@Tox@** is also the symbol of **@variable@**, but in this Example 2.1, we just set a single value of 5 nm. Users can change the tox value in Workbench.

Fig. 2.7 Assigning values of @Lg@ in workbench



 SDE	Lg	tox	 SNMESH
--	200	5	--
	400	5	--
	600	5	--
	800	5	--
	1000	5	--

In this example, the thicknesses of gate length (L_g) and gate oxide (tox) layer are set as variables @Lg@ and @tox@, each of which is composed by two rectangles. The body is composed of one rectangle, so it is composed of a total of “seven 2D rectangles.”

In the same line of code, materials, number or parameters with different definitions must be separated by blanks. The universal code for parameter definition is (**define A B**), in which A represents the name declared to the computer, and B represents the value of A. The default unit of this simulation software is μm , but the unit of nm is more suitable for current semiconductor device design. Therefore, the unit of nm must be set in the parameter setting with the value of μm multiplied by 10^{-3} , defined as “(**define nm 1e-3**).” In addition, in this book, “**tac**” is defined as the thickness of active layer, “**tox**” is defined as the thickness of gate oxide, “**Body**” is defined as the thickness of body beneath active layer, “**LSDC**” is defined as the length of source/drain contact, “**LSD**” is defined as the length of source/drain, “**C_Doping**” is defined as the doping concentration of channel, “**D_Doping**” is defined as the doping concentration of drain, “**S_Doping**” is defined as the doping concentration of source, and “**B_Doping**” is defined as the doping concentration of body.

And then the algebra of x1, x2, x3... is used for representing all values to easily facilitate the code adjustment during lots of parameter modification in the future and the device structure scaling. It is revealed in Fig. 2.6 that the algebra of every coordinate (x1, x2, ...) is the sum of the previous algebra and the previously defined parameter. For example, x2 is equaled to x1 plus LSDC.

It is noteworthy that, in this Sentaurus TCAD software, the form of universal codes of addition/subtraction/multiplication/division is (operator A B). Its mathematical meaning is the operator calculation of A with respect to B. For example, (+ 10 5) means $10 + 5$, (- 10 5) means $10 - 5$, and (- 10 5 2) means $10 - 5 - 2$. Attentions must be paid to the sequence of multiplication and division operation. For example, (/10 5 2) means $(10/5)/2$, and it is because the order of calculation in this program is from left to right.

If there is another parenthesis behind 10, the calculation order can be changed. For example, (/10 (/5 2)) means $10/(5/2)$. And then, we enter the section of device structure establishment.

In this example, the device structure is established via rectangle stacking. It is shown in Fig. 2.6 that a total of seven rectangles are used in this example. The benefit of this method is that it can easily establish the desired structure. However, there will be the problem of computer failed to identify the intersection or overlapped region of rectangles. “**ABA**” is exactly the code to deal with this issue. It means that the new rectangle will replace the old rectangle at the region with overlapping new and old rectangles, and the material property of this region will be determined by the new rectangle as shown in Fig. 2.8.

As for the device structure establishment, the establishment of rectangles will be declared by sdegeo. The rectangle refers to the 2D rectangular rectangle, and the size of rectangle is determined by the diagonal.

As shown in Fig. 2.9, the size of rectangle can be determined by the assigned coordinates of A and B. For example, (**sdegeo:create-rectangle (position 0 0 0)**)

Fig. 2.8 Illustration of “ABA” command. It is very useful in 3D FET; **a** is suitable for gate-all-around FET, and **b** is suitable for FinFET

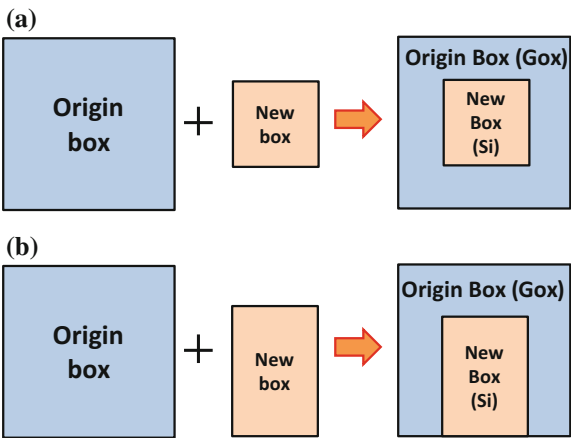
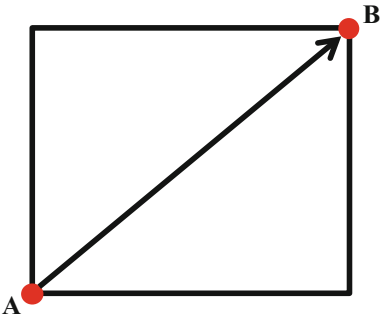


Fig. 2.9 Illustration of rectangle establishment, vector from point A to point B



(position x1 y2 0) “Silicon” “SourceC”), where the coordinate of A is (0, 0, 0), and the coordinate of B is (x1, y2, 0). After the coordinates of A and B are defined, the material of this rectangle will be declared, and in this example, the material is silicon. After material definition, the name of this rectangle will be declared, and it is defined as SourceC in this example, which represents Source Contact.

Then the six rectangles of **SourceC**, **Source**, **Channel**, **Drain**, **DrainC**, and **gate oxide** will be properly established from left to right in accordance with Fig. 2.6. After the rectangles required by **active layer** are established, the next thing is to establish the rectangles for body and gate oxide. In the end, the voltage must be applied. When a voltage is applied to gate G, drain D, source S, and body B, the entire rectangle of electrode can be regarded as a complete conductor, and all electrodes are at the same potential. The definition of gate electrode on the gate oxide will form an equipotential surface without the need for additionally defined gate material. The only thing to be noted is that the **work function of metal gate** must be properly defined in order to achieve the expected threshold voltage (V_{th}). And then, all contacts required by the device must be defined.

Contact code: **define-contact-set “G”** is for assigning the name of contact which can be self-defined. **(color:rgb 1.0 0.0 0.0) “##”** is for defining the color and form of contact, and here it can be set as default. This example is a 2D simulation, so the 2D contact needs to be established such as **define-2d-contact**. It is noteworthy that, as described in previous section, only the electrode equipotential lines need to be defined (equipotential surface in 3D structure), and **there is unnecessary for defining physical material and physical space of contact**. By assigning a point to the defined equipotential line, the program will automatically stretch to the left and to the right from that point until reaching the boundaries of this 2D rectangle, and this extended line is the closed electrode equipotential line. The code is **find-edge-id (position (+ x2 (/ Lg 2)) y3 0) “G”**. It is noteworthy that the name of electrode must be given once here, and the definition G means “gate contact electrode.” And then, the contact electrodes of source, drain, and substrate must be established in proper order.

After all required contacts are established, the next thing is to define the doping styles and doping concentrations of seven rectangles.

The rule of doping establishment is to determine the style and concentration of doping before putting into the previously established rectangles to complete the doping process. The code of **“define-constant-profile”** refers to a doping with fixed style and concentration which does not take into consideration the concentration gradient.

“dopedC” refers to the names of this doping style and concentration, which can be assigned based on personal preference. **“BoronActiveConcentration”** refers to a dopant which is already activated, and here it is boron. The **C_Doping** refers to the concentration of this doping, and the value of C_Doping has been assigned during parameter definition.

Code: **define-constant-profile-region** refers to putting the pre-defined doping style and concentration into a fixed region without considering the situation of dopant diffusion. **“RegionC”** refers to the name of this action which can be set in accordance with personal reference. The content of this action is to combine all codes showing up afterward, so if **“RegionC” is followed by “dopedC” and “Channel,” it refers to putting dopedC into the rectangle of Channel, and this action is called RegionC**. This is how the doping of seven rectangles is composed.

After the definition of doping is completed, the mesh of mathematical calculation should be determined. The simulation calculation of semiconductor device must be analyzed by various physics formula, the most fundamental of which are the Poisson equation to determine electrical potential and the continuity equation to determine carrier concentration. The electrical properties of semiconductor device must be based on the simultaneous solution, such as Poisson equation, continuity equation, and transport equation. The location of such solution is the intersection of mesh as shown in Fig. 2.10. However, there cannot be infinite number of solutions for semiconductor device, so Newton interpolation method is used for approximation between points. However, the approximation by interpolation is not a proper

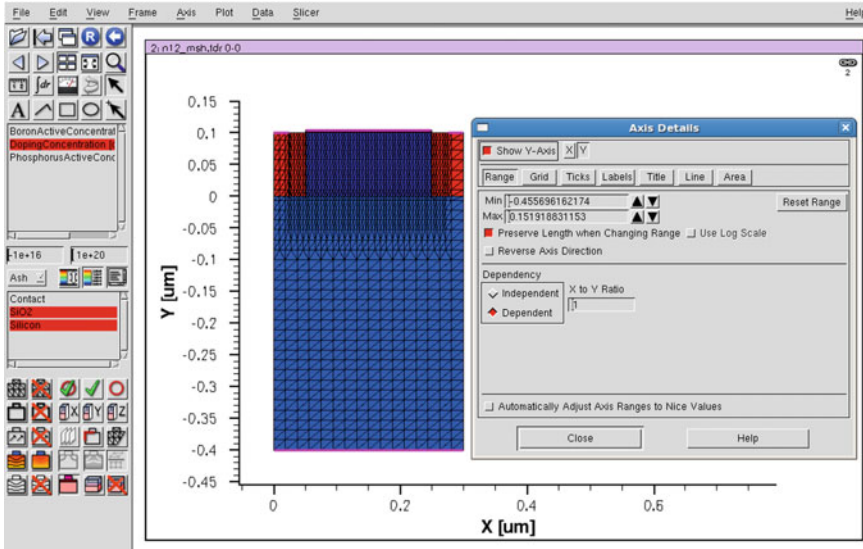


Fig. 2.10 Actual example of mesh. The active layer is designed with denser mesh, and the substrate layer is designed with less dense mesh in order to obtain optimized simulation design efficiency

option for regions with large concentration gradient or electric field variation. These regions must be analyzed by more precise solutions, so the meshes in these regions must be denser.

The logic behind the establishment of mesh is: First a comprehensive mesh is assigned. Since the locations of solutions are the intersections of mesh, a dense mesh will be assigned to the region with large concentration gradient, large electric field variation, or significant impact on electrical properties (such as the active region). Therefore, an **AllMesh** is assigned first. **define-refinement-size** refers to the definition of distance between points. With varying geometric shapes of semiconductor device, there will be different distributions of dopant concentrations such that the program will automatically adjust the distance on the boundary as long as the maximum and minimum values are declared to the computer. “**Cha_Mesh**” refers to the name of aforementioned action, which can be determined based on personal preference. Among the next six numbers, the first three of them refer to the maximum values along the three directions of X-axis, Y-axis, and Z-axis, and the next three of them refer to the minimum values along the three directions of X-axis, Y-axis, and Z-axis. In this case, these numbers are **(20 20 0 10 10 0)**, because the maximum value of X is 20 nm and the minimum value of X is 10 nm, and the same shall apply to Y, so the maximum and minimum values of Z are set to be zero, because this example is 2D device.

The mesh established earlier can be placed in specific region or specific material, and here it is placed in the specific material by the code: **define-refinement-material**. “channel_RF” is the name of this action, which can be determined based on personal preference. However, it is better to be in accordance with the syntax suggested in this book to avoid unnecessary error. This action is to combine the following code, so if “channel_RF” is followed by “Cha_Mesh” and “Silicon,” it refers to the action of placing Cha_Mesh into the material of silicon. This action is called channel_RF (i.e., channel refinement).

After the comprehensive mesh is established, a denser mesh will be established in the active layer. The method for doing this is to establish a mesh rectangle and then put it in the designated region. The code: **define-refinement-window** is for establishing the mesh rectangle. The name of this action is **multiboxChannel**, which can be determined based on personal preference, yet it is better to be in accordance with the syntax suggested in this book to avoid unnecessary error. “Rectangle” refers to the establishment of a 2D mesh rectangle based on the method of determining the size of rectangle by diagonal before assigning the maximum and minimum values along the three directions of X-axis, Y-axis, and Z-axis. And the code: **multiboxPlacementChannel** is the name of this action which can be determined based on personal preference. This action is to integrate the following code, so if “**multiboxPlacementChannel**” is followed by “**multiboxSizeChannel**” and “**multiboxChannel**,” it refers to placing **multiboxSizeChannel** into **multiboxChannel**, and this action is called **multiboxPlacementChannel**.

The denser mesh will be stretched across the region with larger variation of concentration gradient, so here another line of code will be added for the denser mesh to be stretched toward the depletion region. The code is **define-refinement-function**. This action will take place in **multiboxPlacementChannel** with extension in accordance with **DopingConcentration**. The **MaxTransDiff** refers to the degree of extension, which is one. There has more selection of **MaxTransDiff** degree; readers can refer the TCAD manual.

And next the file is saved with the code as shown in the last part of code:

The part can be used as default. Among them, **assign-material-and-region-names** is for saving the previously established materials and names, and the line of **write-scaled-cmd-file** is for saving the aforementioned scale. With the default unit of program being μm , X-axis, Y-axis, and Z-axis must be multiplied by 10^{-3} and the unit should be converted to nm. The following **scale-tdr-bnd** saves all files as boundary format. It will be used by the following tool. All codes required by SDE are hereby completed.

The second tool in the tool column is “**SNMESH**” tool in SWB, which is mainly used for establishing the mesh required by device simulation, and the mesh code has been written in the commands of SDE. So we only have to set **SNMESH** to access the commands of SDE as shown in Fig. 2.11.

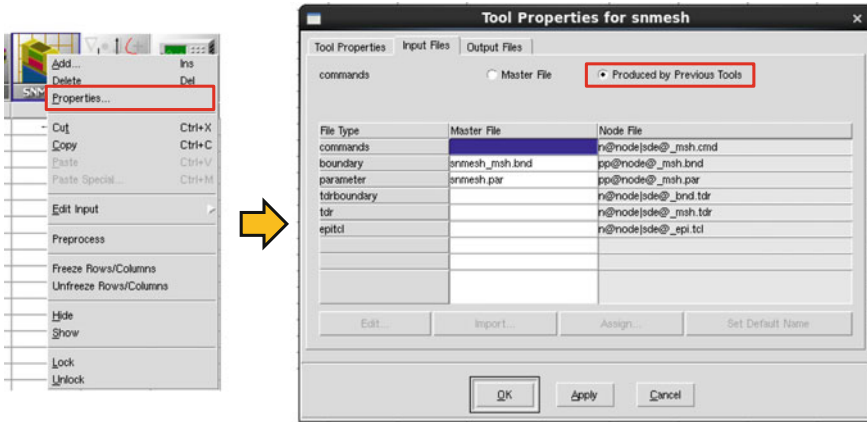


Fig. 2.11 Modification of properties of SNMESH

By right-clicking on Properties on **SNMESH tool** to enter the input files and selecting Produced by Previous Tools, **SNMESH** will automatically grab the codes of previous **SDE**, and there is no need for entering any code into **SNMESH**.

By now a complete 2D MOSFET device structure has been established, yet it has not been given any physics model. Therefore, the next tool: **SDEVICE** is used for applying the physics model and mathematic model of this semiconductor device and the conditions of threshold voltage (V).

2. SDEVICE → dessis_des.cmd

SDEVICE tool commands (code file is **dessis_des.cmd**) can be divided into six parts.

- (1) **Electrode**
- (2) **File**
- (3) **Physics**
- (4) **Math**
- (5) **Plot**
- (6) **Solve**

The complete code of **SDEVICE** is as shown below. It is noteworthy that in the commands of **SDEVICE**, the symbol * or # indicates that the following line of code is the prompt character for program designers to keep notes, and it will not be executed by the computer.

The **SDEVICE** codes of this 2D MOSFET example are as shown below:

```
#----- dessis_des.cmd -----#
Electrode{
{name="D" voltage=0.0}
{name="S" voltage=0.0}
{name="G" voltage=0.0 WorkFunction=@WK@}
}
File{
    Grid="@tdr@"
    Plot="@tdrdat@"
    Current="@plot@"
    Output="@log@"
    parameter="@parameter@"
}
Physics{
    Mobility( DopingDep HighFieldSaturation Enormal )
    EffectiveIntrinsicDensity( OldSlotboom )
    Recombination( SRH(DopingDep) )
    eQuantumPotential
}
Math{
    -CheckUndefinedModels
    Number_Of_Threads=4
    Extrapolate
    Derivatives
    * Avalderivatives
    RelErrControl
    Digits=5
    ErRef(electron)=1.e10
    ErRef(hole)=1.e10
    Notdamped=50
    Iterations=20
    Directcurrent
    Method=ParDiSo
    Parallel= 2
    -VoronoiFaceBoxMethod
```

```

    NaturalBoxMethod
}
Plot{
    eDensity hDensity
    eCurrent hCurrent
    TotalCurrent/Vector eCurrent/Vector hCurrent/Vector
    eMobility hMobility
    eVelocity hVelocity
    eEnormal hEnormal
    ElectricField/Vector Potential SpaceCharge
    eQuasiFermi hQuasiFermi
    Potential Doping SpaceCharge
    SRH Auger
    AvalancheGeneration
    DonorConcentration AcceptorConcentration
    Doping
    eGradQuasiFermi/Vector hGradQuasiFermi/Vector
    eEparallel hEparallel
    BandGap
    BandGapNarrowing
    Affinity
    ConductionBand ValenceBand
    eQuantumPotential
}
Solve {
    Coupled ( Iterations= 150){ Poisson eQuantumPotential }
    Coupled { Poisson eQuantumPotential Electron Hole }
    Quasistationary(
        InitialStep= 1e-3 Increment= 1.2
        MinStep= 1e-12 MaxStep= 0.95
        Goal { Name= "D" Voltage=@Vd@ }
    ){ Coupled { Poisson eQuantumPotential Electron Hole } }
    Quasistationary(
        InitialStep= 1e-3 Increment= 1.2
        MinStep= 1e-12 MaxStep= 0.02

```

```

Goal { Name= "G" Voltage=@Vg@ }
DoZero
){ Coupled { Poisson eQuantumPotential Electron Hole } }
}
#----- END -----#

```

At first, all the conditions of voltage (V_d , V_s , and V_g) will be defined together with the gate. The metal gate work function (WorkFunction(V)) sets as a variable @WK@.

The name of each electrode terminal must be identical to the name in SDE tool, or the program cannot be executed due to interpretation failure. The work function will be set as variables to be defined in workbench to facilitate the calibration of threshold voltage (V_{th}).

The next step is to set up the files to be read from the previous tool during the operation of **SDEVICE**. This part is based on default value such that it cannot be modified without permission, because the file name must be in compliance with the program regulation.

And then, we need to tell the computer what are the physics formula to be substituted into this simulation calculation, such as the recombination model or some quantum modification models.

Among them, the **eQuantumPotential** is the quantum modification item with respect to the density of state of electron. When the device dimension is very small, some quantum modification items must be added for the simulation results to be closer to the real condition. The default setting of this program is based on the most complete physics model, so please read the manual thoroughly before making any modification.

After assigning physics model, the next step is the assignment of mathematical model, which can be based on default setting. The default setting of this program is based on the most complete mathematic model, so please read the manual thoroughly before making any modification.

It is noteworthy that **Iterations = 20**, it indicates the number of points to be given during approximation by Newton interpolation method between two mesh points, and 20 points are given here. Usually the less dense mesh will be set up first before the mesh of important simulation step and crucial region (such as active layer) can be optimized during simulation process in order to be in compliance with the correct electrical and physical properties of the device. Therefore, this part of

setting can be determined based on personal preference. Insufficient number of points given here will lead to divergence of simulation calculation, yet excessive number of points given will lead to prolonged simulation calculation.

The next step is to tell the computer what are the diagrams to be extracted, such as energy band diagram and carrier distribution diagram, which can be deleted based on personal preference.

In the end, **SDEVICE** tool should tell the computer what are the calculations to be done by the combination of aforementioned physics model and mathematic model (such as Poisson equation).

It is revealed in the code that electron and hole must be substituted in Poisson equation for solution, and eQuantumPotential must be added for correction. The reader should find out that in this book, the drain (D) voltage is addressed before gate (G) voltage (V_g). It should be noted that the voltage mentioned earlier is regarded as a constant value, and the voltage mentioned later is for sweeping action based on the range set in accordance with workbench. Another point to be noted is that InitialStep = $1e^{-3}$ refers to the first point to be calculated will be $1e^{-3}$ unit away from threshold voltage. If it cannot be converged, the voltage pitch will be reduced to continue with the search for value that can be converged. The minimum value to be reduced is $1e^{-12}$ unit, and the maximum value is 0.02 unit, which can be modified based on personal preference.

3. **INSPECT** → **inspect_ins.cmd**

In the end, it comes to the part of **INSPECT** tool, which should be set up in accordance with the default value without the need for any modification. This is because **INSPECT** will only affect what are the parameters to be extracted. The author has suggested that this function should be for reference only. If there is any need for cautious data analysis, the data should be extracted and analyzed by professional engineering and scientific application graphic software such as SigmaPlot and Origin. The **symbol #** indicates that the following line of code is the prompt character for program designers to keep notes, and it will not be executed by the computer.

The command lines of INSPECT are usually using default file as shown below:

```

#----- inspect_ins.cmd -----#
#       Script file designed to compute   : #
#       * The threshold voltage           : VT #
#       * The transconductance            : gm#
#-----#
if { ! [catch {open n@previous@_ins.log w} log_file] } {
    set fileId stdout
}
puts $log_file " "
puts $log_file "          ----- "
puts $log_file "          Values of the extracted Parameters : "
puts $log_file "          ----- "
puts $log_file " "
puts $log_file " "
set DATE [ exec date ]
set WORK [ exec pwd ]
puts $log_file "    Date      : $DATE "
puts $log_file "    Directory : $WORK "
puts $log_file " "
puts $log_file " "
#                               idvgs=y(x);    vgsvgs=x(x);                               #
set out_file n@previous@_des
proj_load "${out_file}.plt"
# -----#
# I) VT = Xintercept(maxslope(ID[VGS])) or VT = VGS( IDS= 0.1 ua/um ) #
# -----#
cv_create idvgs "${out_file} G OuterVoltage" "${out_file} D TotalCurrent"
cv_create vdsvgs "${out_file} G OuterVoltage" "${out_file} D OuterVoltage"
#.....#
# 1) VT extracted as the intersection point with the X axis at the point #
#   where the id(vgs) slope reaches its maximum : #
#.....#
set VT1 [ f_VT1 idvgs ]
#.....#
# 2) Printing of the whole set of extracted values (std output) : #

```

```

#..... #
puts $log_file "Threshold voltage VT1 = $VT1 Volts"
puts $log_file " "
#..... #
# 3) Initialization and display of curves on the main Inspect screen : #
# ..... #
cv_display idvgs
cv_lineStyle idvgs solid
cv_lineColor idvgs red
# ----- #
# II) gm = maxslope((ID[VGS])) #
# ----- #
set gm [ f_gm idvgs ]
puts $log_file " "
puts $log_file "Transconductance gm = $gm A/V"
puts $log_file " "
set ioff [ cv_compute "vecmin(<idvgs>)" A A A A ]
puts $log_file " "
puts $log_file "Current ioff = $ioff A"
puts $log_file " "
set isat [ cv_compute "vecmax(<idvgs>)" A A A A ]
puts $log_file " "
puts $log_file "Current isat = $isat A"
puts $log_file " "
set rout [ cv_compute "Rout(<idvgs>)" A A A A ]
puts $log_file " "
puts $log_file "Resistant rout = $rout A"
puts $log_file " "
cv_createWithFormula logcurve "log10(<idvgs>)" A A A A
cv_createWithFormula difflog "diff(<logcurve>)" A A A A
set sslop [ cv_compute "1/vecmax(<difflog>)" A A A A ]
puts $log_file " "
puts $log_file "sub solp = $sslop A/V"
puts $log_file " "
#### Putting into Family Table #####

```

```
ft_scalar VT $VT1
ft_scalar gmax $gm
ft_scalar ioff $ioff
ft_scalar isat $isat
ft_scalar sslop $sslop
ft_scalar rout $rout
close $log_file
#----- END -----#
```

As for the extraction of V_{th} , the drain current is to be collected with this example based on nFET, and it is set as D. If the simulation is for pFET, it should be changed to S. As for the extraction of SS, if it is for pFET, diff(<logcurve>) must be multiplied by (-1) before the current of pFET is in opposite direction to the current of nFET.

By now all codes have been entered, and in the end, all parameters of workbench should be set, and please remember to enter all variables to be included in the workbench as shown in Fig. 2.12. The setting method is as shown in Fig. 2.13.

First, right-click to select **Add** in the lower column of Tool, and then enter the name of variable in Parameter and the value of variable in Default Value. The values can be deleted or changed by right-clicking **Edit Values** in the lower column of Tool as shown in Fig. 2.14.

After the variables are completed, the project name can be selected, and **run** can be clicked to start the simulation calculation as shown in Fig. 2.15.

It is shown in Fig. 2.16 that the name of project will turn **yellow** after the completion of simulation, indicating a successful simulation calculation; if it turns **red**, it means the simulation cannot be converged or has some syntax error. User can check error message in log file. Also the mesh calculation may need to be adjusted for convergence.

							
SDE		SMMESH		SDEVICE		INSPECT	
	Lg	tox			WK	Vg	Vd
--	200	5	--	--	4.8	3	0.1
							3
	400	5	--	--	4.8	3	0.1
							3
	600	5	--	--	4.8	3	0.1
							3
	800	5	--	--	4.8	3	0.1
							3
	1000	5	--	--	4.8	3	0.1
							3

Fig. 2.12 Important variables being set up of each tool in the SWB

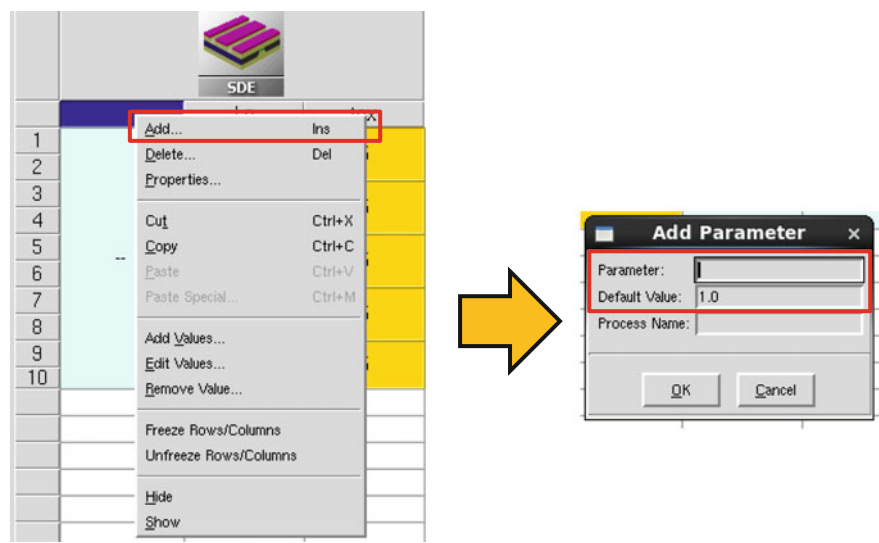


Fig. 2.13 Method to set up the variables in the workbench

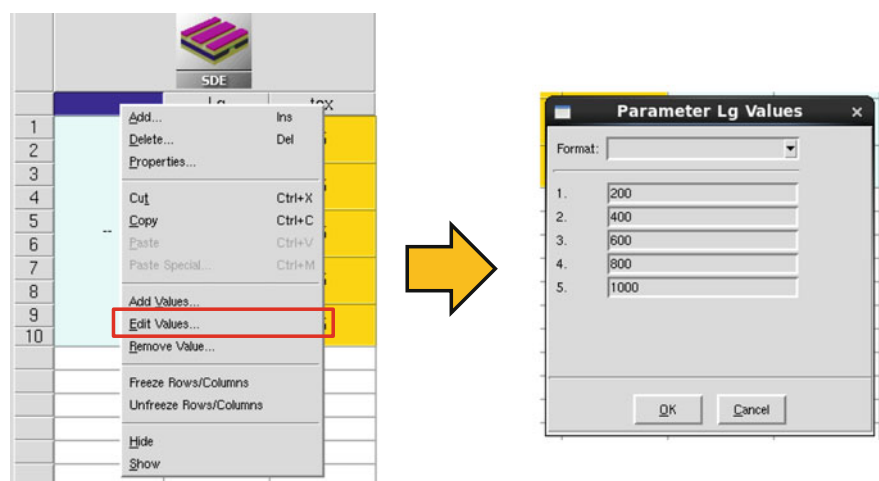


Fig. 2.14 Method for modifying variables of workbench

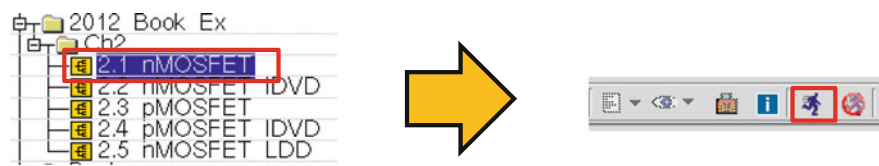


Fig. 2.15 Starting the simulation calculation, pressing RUN icon

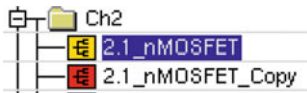


Fig. 2.16 Icons of simulation successful in yellow or failed in red

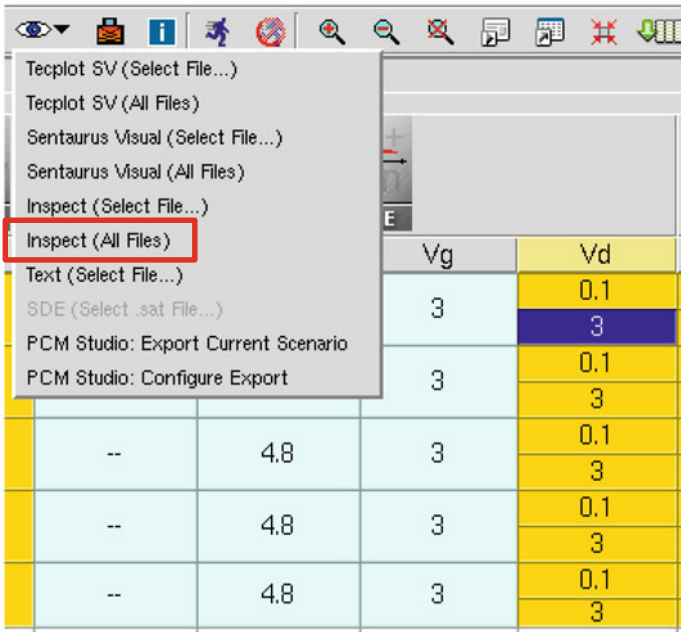


Fig. 2.17 Open Inspect tool to show electrical property

After the completion of simulation, the next step uses **INSPECT tool to analyze electrical properties** as shown in Fig. 2.17, where the node of the electrical property should be right-clicked and the eye icon on the visualize bar should be clicked to select Inspect (All Files).

The selected interface is as shown in Fig. 2.18. The node to be inspected should be selected on the datasheet of workbench, and the set electrode and the electrical property to be inspected should be selected before being put on the preset axis. For operation example: **G** → **OuterVoltage** → **To X-Axis**; **D** → **TotalCurrent** → **To Left Y-Axis**. In the end, the Y-axis on the topmost tool bar should be changed to be displayed in log scale to lead us to the frequently seen I_d - V_g curve.

In addition, the data of I_d - V_g curve can be exported in txt file format to be analyzed by other professional engineering and scientific application graphics software such as **SigmaPlot** and **Origin**. As indicated in Fig. 2.19, the data of

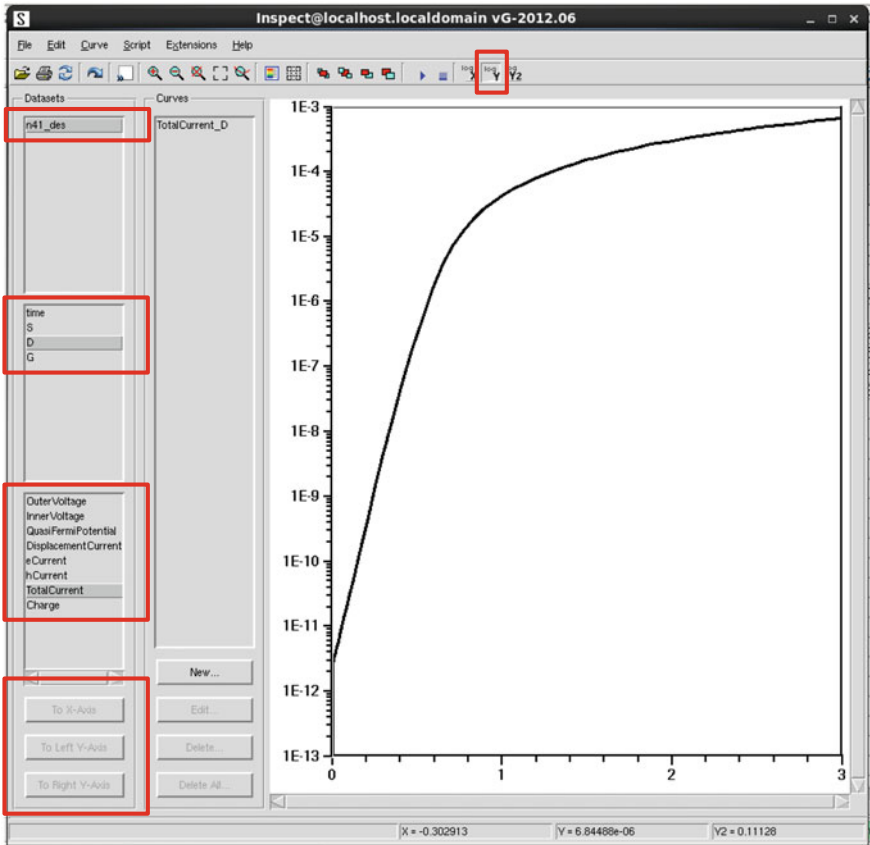


Fig. 2.18 Using Inspect tool plots electrical properties display and analysis. The plot shows a typical I_d – V_g transfer curve of nMOSFET

electrical properties can be exported by clicking the **File** → **Export** → **csv or txt format** on the upper left corner. The data of electrical properties is exported in the format of csv or txt file as shown on the right side of Fig. 2.19, where the data of x is V_g , and the data of y is I_d .

Except for the electrical property diagram, other important physical properties, such as electric field, electrostatic potential, and charge concentration, need to be examined during the analysis of semiconductor device. As shown in Fig. 2.20, the node of electrical property to be examined should be right-clicked, and the eye icon on the Visualize bar should be selected to access the drop-down menu of visualization software. And then, we should select Sentaurus Visual (All Files).

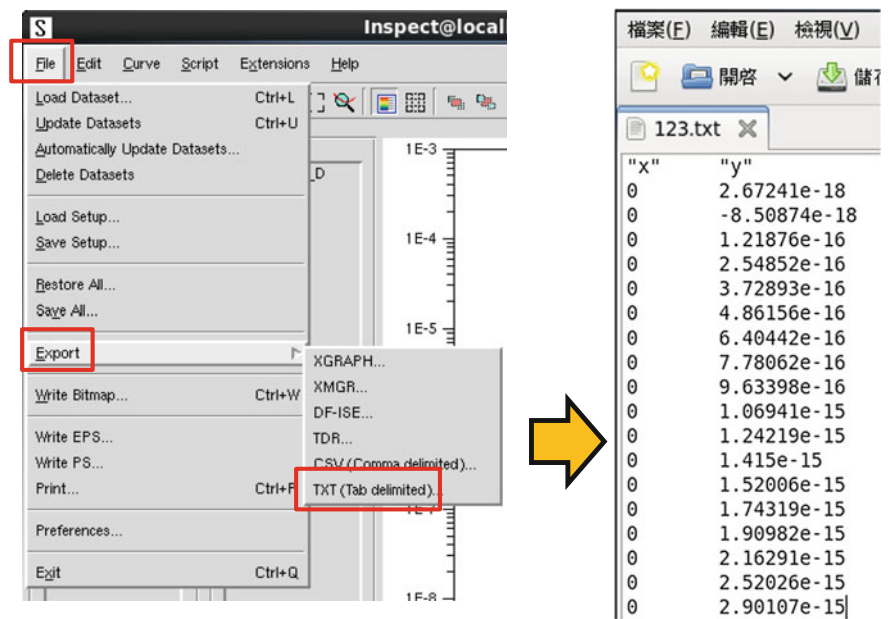
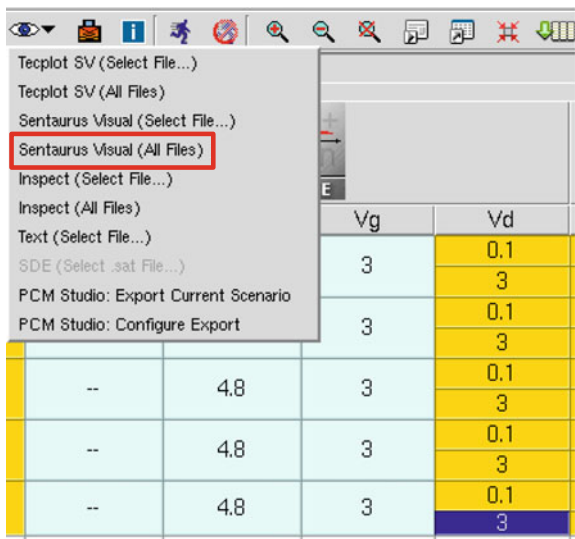


Fig. 2.19 Exported data of electrical properties is in the txt file format (ex: IdVg.txt)

Fig. 2.20 Select Sentaurus Visual to examine physical properties



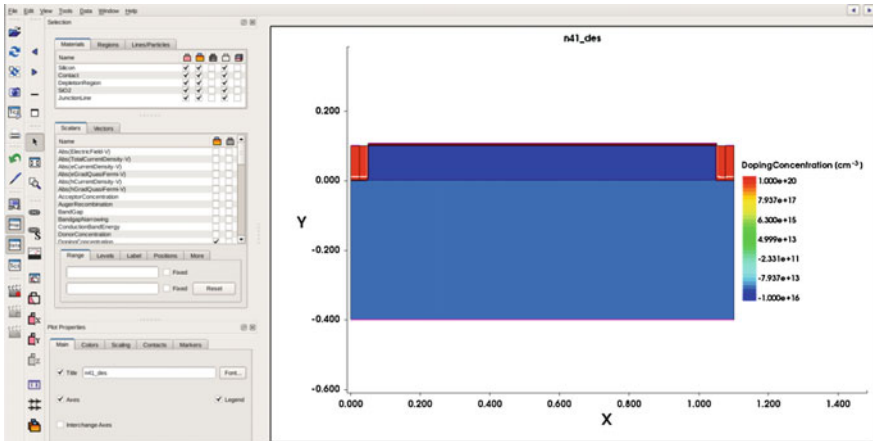


Fig. 2.21 Sentaurus Visual interface (window-based user-friendly interface)

Figure 2.21 is the Sentaurus Visual interface. The Selection on the left is for selecting the material to be displayed and its mesh. In addition, the physical property to be inspected can be selected on the lower part of screen such as: energy band diagram and carrier distribution of electrons and holes. If either X-axis or Y-axis is to be fixed to observe the variation of physical property along with the other axis, the icons in the red frame on the right can be selected. For example, fixing X-axis at the position of 0.5 will allow us to observe the variation of physical property along Y-axis with $X = 0.5$.

Special Note: FAQ and Troubleshooting

The most frequently seen problem is that the value does not converge during the simulation, and the error message is as shown in Fig. 2.22.

Most of these problems are due to the difficulty in mesh calculation which has resulted in singular points generation at the location with large concentration variation gradient thus causing diversion. The better mesh code which is less vulnerable to diversion as shown below:

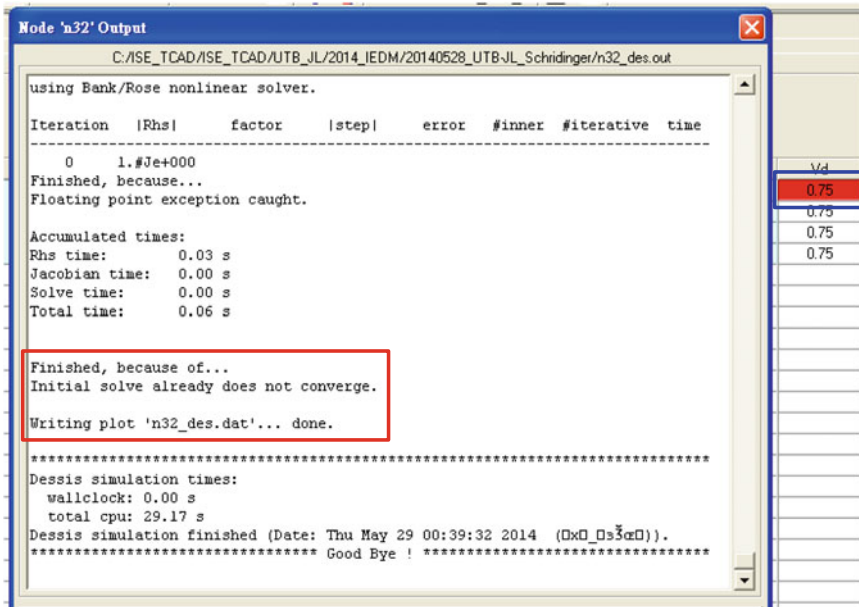


Fig. 2.22 Error message of the value of simulation does not converge

```
;----- AllMesh -----;
```

```
(isedr:define-refinement-size "Cha_Mesh" 5 5 5 1 1 1)
```

```
(isedr:define-refinement-material "channel_RF" "Cha_Mesh" "Silicon")
```

The Channel width here is 5 nm, so the initial rough cutting range is 5 nm → 1 nm for the cutting of the entire silicon.

```
;----- Channel Mesh example of 3D FinFET -----;
```

```
;--- AllMesh ---;
```

```
(sdedr:define-refinement-size "Cha_Mesh" 5 5 5 1 1 1)
```

```
(sdedr:define-refinement-material "channel_RF" "Cha_Mesh" "Silicon")
```

```
;--- ChannelMesh ---;
```

```
(sdedr:define-refinement-window "multiboxChannel" "Cuboid"
```

```
(position x1 0 0) (position x4 y1 z1))
```

```
(sdedr:define-multibox-size "multiboxSizeChannel" 2 2 2 2 2 2)
```

```
(sdedr:define-multibox-placement "multiboxPlacementChannel"
"multiboxSizeChannel" "multiboxChannel")
```

```
(sdedr:define-refinement-function "multiboxPlacementChannel"
"DopingConcentration" "MaxTransDiff" 1)
```

The following fine cutting is mainly placed inside the channel because there is large carrier variation gradient and large electric field variation. So the cutting should be $1\text{ nm} \rightarrow 0.5\text{ nm}$ or $1\text{ nm} \rightarrow 2\text{ nm}$. In this book, it is suggested that mesh variation should be kept within 100%.

The max value and min value of mesh should be determined in coordination with device dimension from high to low. The divergence will most likely to take place in the region with large concentration gradient and electric field variation. This is because the Newton interpolation method is used for approximation at the intersection of mesh, and the value is confirmed by left limit and right limit approach. If the values of left limit and right limit do not match, the simulation result will diverge. Therefore, smaller mesh should be assigned to the location with large concentration gradient or large electric field variation. The device for the first run can be assigned with a larger mesh or fewer elements to see if the electric properties are as expected and finer segmentation can be applied for observation of electric field distribution or carrier distribution. It still cannot converge after mesh adjustment, fine-tuning of **workfunction** can be considered. For example, if the original value is **4.6 eV**, we can try with **4.601 eV**. An additional **0.001 eV** will **not lead to too much impact on V_{th}** , but it can help with convergence. Voltage fine-tuning can also be applied such as changing $V_D = 1\text{ V}$ to $V_D = 1.01\text{ V}$. In this book, it is suggested that the adjustment shall not exceed **2%** (Fig. 2.23).

The analysis of physical properties of 2D nMOSFET based on **Sentaurus Visual interface** is as shown in Fig. 2.24.

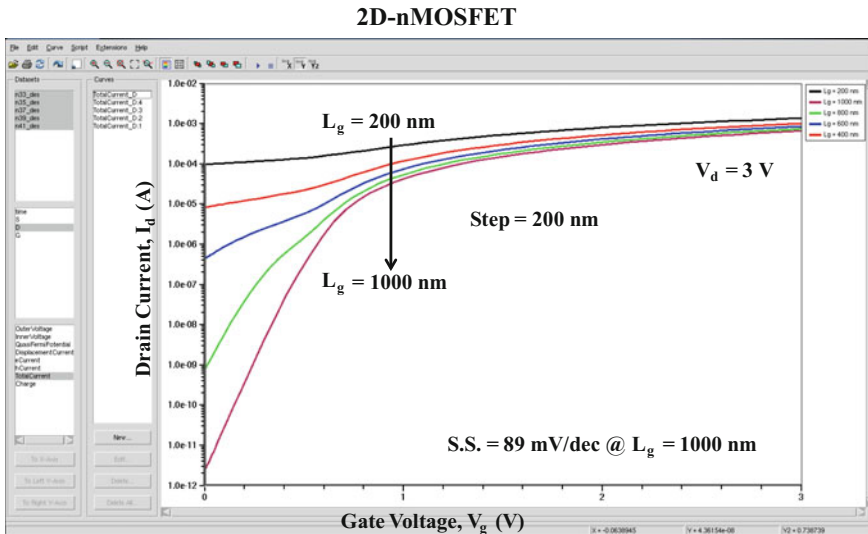


Fig. 2.23 I_d - V_g curves of simulation of 2D nMOSFET. Some descriptions are added by PowerPoint after snapshot from Inspect. It reveals that the 2D nMOSFET with $L_g < 600\text{ nm}$ suffers severe short-channel effect (SCE)

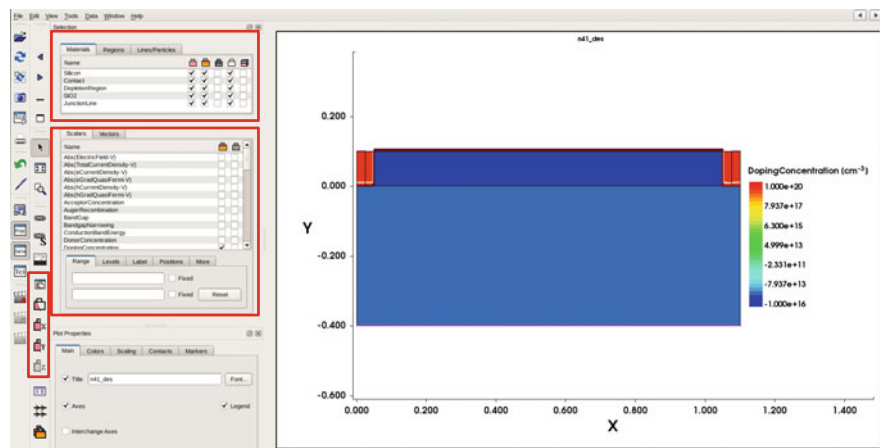


Fig. 2.24 Use Sentaurus Visual interface. Important device physical properties can be visualized and analyzed

Figure 2.24 is the Sentaurus Visual interface. The Selection on the left is for selecting the material to be displayed and its mesh. In addition, the physical property to be inspected can be selected on the lower part of screen such as: energy band diagram and carrier distribution of electrons and holes.

The electron concentration distribution, electric field distribution, electrostatic potential distribution, and energy band diagram along the channel direction are as shown in Figs. 2.25, 2.26, 2.27, and 2.28 based on the conditions of $L_g = 1000$ nm, $V_d = 3$ V, and $V_g = 3$ V.

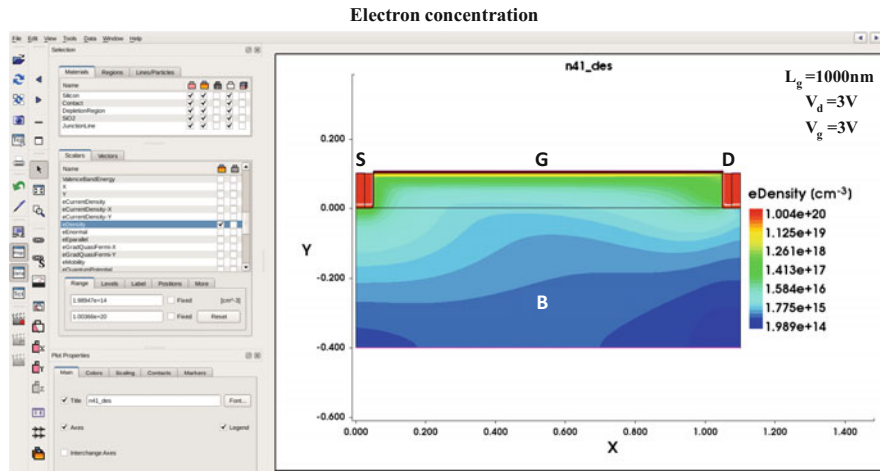


Fig. 2.25 Electron concentration distribution of the simulation of 2D n-type semiconductor device

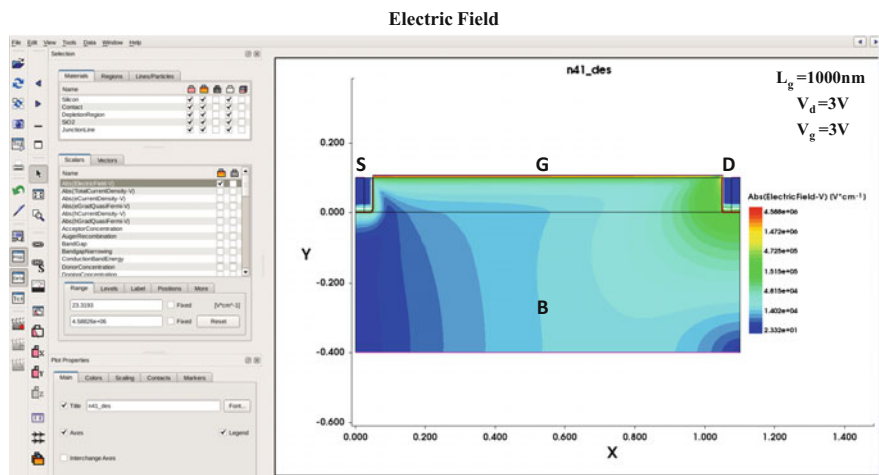


Fig. 2.26 Electric field distribution of the simulation of 2D n-type semiconductor device

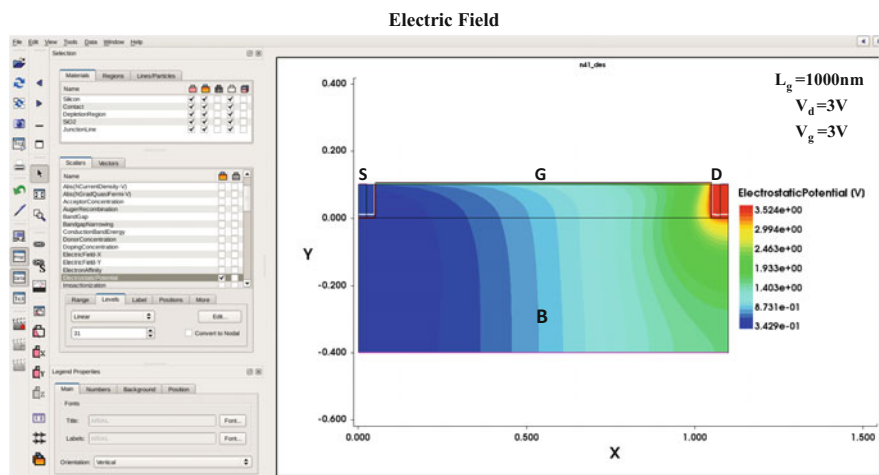


Fig. 2.27 Electrostatic potential distribution of the simulation of 2D n-type semiconductor device

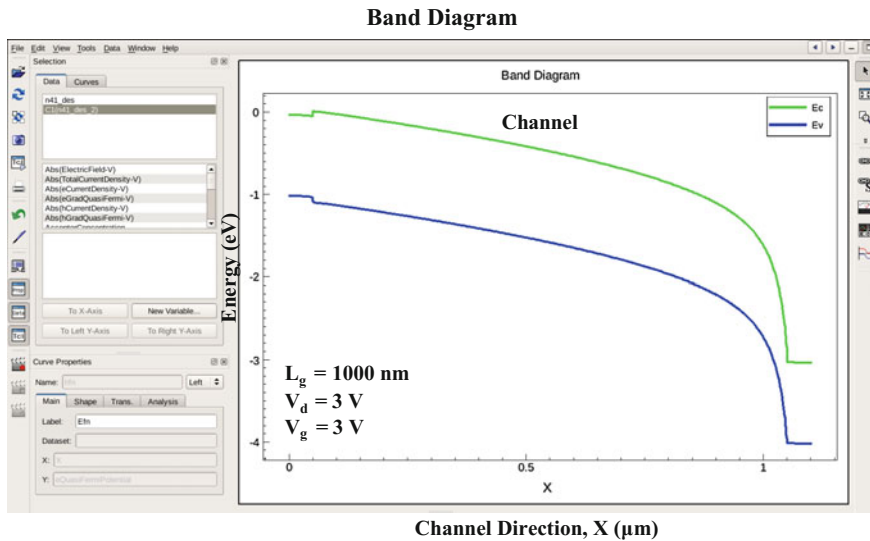


Fig. 2.28 Energy band diagram along the channel direction of the simulation of 2D n-type semiconductor device

2.3 [Example 2.2] 2D n-Type MOSFET with I_d – V_d Characteristics Simulation

This I_d – V_d example is very similar to Example 2.1, only has difference in electrodes and its bias setting.

The following three main program code files are all based on Synopsys Sentaurus TCAD 2014 version.

1. SDE—devise_dvs.cmd

This is the best example of 2D nMOSFET I_d – V_d .

The line of code following; **is the prompt character for program designer to take note such that it will not be executed by the computer.**

```

;----- parameter -----;
(define Lg @Lg@)
(define tox @tox@)
(define tac 100)
(define Body 400)
(define LSDC 25)
(define LSD 25)
(define C_Doping 1e16)
(define D_Doping 1e20)
(define S_Doping 1e20)
(define B_Doping 1e15)
(define nm 1e-3)
(define x1 LSDC)
(define x2 (+ x1 LSD))
(define x3 (+ x2 Lg))
(define x4 (+ x3 LSD))
(define x5 (+ x4 LSDC))
(define y1 (- Body))
(define y2 tac)
(define y3 (+ tac tox))
;----- Structure -----;
"ABA"
;--- source ---
(sdegeo:create-rectangle
  (position 0 0 0)
  (position x1 y2 0) "Silicon" "SourceC" )
(sdegeo:create-rectangle
  (position x1 0 0)
  (position x2 y2 0) "Silicon" "Source" )
;--- Channel ---
(sdegeo:create-rectangle
  (position x2 0 0)
  (position x3 y2 0) "Silicon" "Channel" )
;--- Drain ---
(sdegeo:create-rectangle

```

```

(position    x3 0 0)
(position    x4 y2 0)  "Silicon" "Drain" )
(sdegeo:create-rectangle
(position    x4 0 0)
(position    x5 y2 0)  "Silicon" "DrainC" )
;--- Body ---
(sdegeo:create-rectangle
(position    0 0 0)
(position    x5 y1 0)  "Silicon" "Body" )
;--- Gate oxide ---
(sdegeo:create-rectangle
(position    x2 y2 0)
(position    x3 y3 0)  "SiO2" "Gateoxide" )
;----- Contact -----;
;---- Gate ----
(sdegeo:define-contact-set "G"
  4.0 (color:rgb 1.0 0.0 0.0) "##")
(sdegeo:define-2d-contact (find-edge-id (position (+ x2 (/ Lg 2)) y3 0)) "G")
;---- Source ----
(sdegeo:define-contact-set "S"
  4.0 (color:rgb 1.0 0.0 0.0) "##")
(sdegeo:define-2d-contact (find-edge-id (position 10 tac 0)) "S")
;---- Drain ----
(sdegeo:define-contact-set "D"
  4.0 (color:rgb 1.0 0.0 0.0) "##")
(sdegeo:define-2d-contact (find-edge-id (position (+ 50 Lg 35) tac 0)) "D")
;---- Substrate ----
(sdegeo:define-contact-set "substrate"
  4.0 (color:rgb 1.0 0.0 0.0) "##")
(sdegeo:define-2d-contact (find-edge-id (position (+ x2 (/ Lg 2)) (- Body) 0))
"substrate")
;----- Doping -----;
;--- Channel ---
(sdedr:define-constant-profile "dopedC" "BoronActiveConcentration" C_Doping )
(sdedr:define-constant-profile-region "RegionC" "dopedC" "Channel" )

```

```

;--- Source ---
(sdedr:define-constant-profile "dopedS" "PhosphorusActiveConcentration"
S_Doping )
(sdedr:define-constant-profile-region "RegionS" "dopedS" "Source" )
(sdedr:define-constant-profile "dopedSC" "PhosphorusActiveConcentration"
S_Doping )
(sdedr:define-constant-profile-region "RegionSC" "dopedSC" "SourceC" )
;--- Drain ---
(sdedr:define-constant-profile "dopedD" "PhosphorusActiveConcentration"
D_Doping )
(sdedr:define-constant-profile-region "RegionD" "dopedD" "Drain" )
(sdedr:define-constant-profile "dopedDC" "PhosphorusActiveConcentration"
D_Doping )
(sdedr:define-constant-profile-region "RegionDC" "dopedDC" "DrainC" )
;--- Body ---
(sdedr:define-constant-profile "dopedB" "BoronActiveConcentration" B_Doping )
(sdedr:define-constant-profile-region "RegionB" "dopedB" "Body" )
;----- Mesh -----;
;--- AllMesh ---
(sdedr:define-refinement-size "Cha_Mesh" 20 20 0 10 10 0)
(sdedr:define-refinement-material "channel_RF" "Cha_Mesh" "Silicon" )
;--- ChannelMesh ---
(sdedr:define-refinement-window "multiboxChannel" "Rectangle"
(position 25 (- 50) 0)
(position (+ 50 Lg 25) (+ tac 50) 0))
(sdedr:define-multibox-size "multiboxSizeChannel" 5 5 0 1 1 0)
(sdedr:define-multibox-placement "multiboxPlacementChannel"
"multiboxSizeChannel" "multiboxChannel")
(sdedr:define-refinement-function "multiboxPlacementChannel"
"DopingConcentration" "MaxTransDiff" 1)
;----- Save BND and CMD and rescale to nm -----;
(sde:assign-material-and-region-names (get-body-list) )
(sdeo:save-tdr-bnd (get-body-list) "n@node@_nm.tdr")
(sdedr:write-scaled-cmd-file "n@node@_msh.cmd" nm)
(define sde:scale-tdr-bnd

```

```

(lambda (tdrin sf tdrout)
  (sde:clear)
  (sdegeo:set-default-boolean "XX")
  (sdeio:read-tdr-bnd tdrin)
  (entity:scale (get-body-list) sf)
  (sdeio:save-tdr-bnd (get-body-list) tdrout)
)
)
(sde:scale-tdr-bnd "n@node@_nm.tdr" nm "n@node@_bnd.tdr")
;----- END -----;

```

2 SDVICE—dessis_des.cmd

The line of code following # and * are the prompt character for program designer to take note such that it will not be executed by the computer.

```

Electrode{
{name="D" voltage=0.0}
{name="S" voltage=0.0}
{name="G" voltage=0.0 WorkFunction=@WK@}
}
File{
  Grid="@tdr@"
  Plot="@tdrdat@"
  Current="@plot@"
  Output="@log@"
  parameter="@parameter@"
}
Physics{
  Mobility( DopingDep HighFieldSaturation Enormal )
  EffectiveIntrinsicDensity( OldSlotboom )
  Recombination( SRH(DopingDep) )
  eQuantumPotential
}
Math{
  -CheckUndefinedModels
  Number_Of_Threads=4
  Extrapolate
  Derivatives
  * Avalderivatives
  RelErrControl
  Digits=5

```

```

    ErRef(electron)=1.e10
    ErRef(hole)=1.e10
    Notdamped=50
    Iterations=20
    Directcurrent
    Method=ParDiSo
    Parallel= 2
*-VoronoiFaceBoxMethod
    NaturalBoxMethod
}
Plot{
    eDensity hDensity
    eCurrent hCurrent
    TotalCurrent/Vector eCurrent/Vector hCurrent/Vector
    eMobility hMobility
    eVelocity hVelocity
    eEnormal hEnormal
    ElectricField/Vector Potential SpaceCharge
    eQuasiFermi hQuasiFermi
    Potential Doping SpaceCharge
    SRH Auger
    AvalancheGeneration
    DonorConcentration AcceptorConcentration
    Doping
    eGradQuasiFermi/Vector hGradQuasiFermi/Vector
    eEparallel hEparallel
    BandGap
    BandGapNarrowing
    Affinity
    ConductionBand ValenceBand
    eQuantumPotential
}
Solve {
    Coupled ( Iterations= 150){ Poisson eQuantumPotential }
    Coupled { Poisson eQuantumPotential Electron Hole }

```

```

    Quasistationary(
    InitialStep= 1e-3 Increment= 1.2
    MinStep= 1e-12 MaxStep= 0.02
    Goal { Name= "G" Voltage=@Vg@ }
){ Coupled { Poisson eQuantumPotential Electron Hole } }
    Quasistationary(
    InitialStep= 1e-3 Increment= 1.2
    MinStep= 1e-12 MaxStep= 0.95
    Goal { Name= "D" Voltage=@Vd@ }
        DoZero
        ){ Coupled { Poisson eQuantumPotential Electron Hole } }
    }
*----- END -----*

```

3. INSPECT—inspect_inc.cmd

The line of code following # is the prompt character for program designer to take note such that it will not be executed by the computer.

```

#-----#
#      Script file designed to compute      :      #
#      * The threshold voltage                :   VT   #
#      * The transconductance                 :   gm   #
#-----#
if { ![catch {open n@previous@_ins.log w} log_file] } {
    set fileId stdout
}
puts $log_file " "
puts $log_file "      ----- "
puts $log_file "      Values of the extracted Parameters : "
puts $log_file "      ----- "
puts $log_file " "

```

```

puts $log_file " "
set DATE [ exec date ]
set WORK [ exec pwd ]
puts $log_file " Date : $DATE "
puts $log_file " Directory : $WORK "
puts $log_file " "
puts $log_file " "
#
#
# idvgs=y(x); vgsvgs=x(x); #
#
set out_file n@previous@_des
proj_load "${out_file}.plt"
# ----- #
# I) VT = Xintercept(maxslope(ID[VGS])) or VT = VGS( IDS= 0.1 ua/um ) #
# ----- #
cv_create idvgs "${out_file} G OuterVoltage" "${out_file} D TotalCurrent"
cv_create vdsvgs "${out_file} G OuterVoltage" "${out_file} D OuterVoltage"
#..... #
# 1) VT extracted as the intersection point with the X axis at the point #
# where the id(vgs) slope reaches its maximum : #
#..... #
set VT1 [ f_VT1 idvgs ]
#..... #
# 2) Printing of the whole set of extracted values (std output) : #
#..... #
puts $log_file "Threshold voltage VT1 = $VT1 Volts"
puts $log_file " "
# 3) Initialization and display of curves on the main Inspect screen : #
cv_display idvgs
cv_lineStyle idvgs solid
cv_lineColor idvgs red
# ----- #
# II) gm = maxslope((ID[VGS]) #
# ----- #
set gm [ f_gm idvgs ]
puts $log_file " "

```

```

puts $log_file "Transconductance gm          = $gm  A/V"
puts $log_file " "
set ioff [ cv_compute "vecmin(<idvgs>)" A A A A ]
puts $log_file " "
puts $log_file "Current ioff          = $ioff  A"
puts $log_file " "
set isat [ cv_compute "vecmax(<idvgs>)" A A A A ]
puts $log_file " "
puts $log_file "Current isat          = $isat  A"
puts $log_file " "
set rout [ cv_compute "Rout(<idvgs>)" A A A A ]
puts $log_file " "
puts $log_file "Resistant rout          = $rout  A"
puts $log_file " "
cv_createWithFormula logcurve "log10(<idvgs>)" A A A A
cv_createWithFormula difflog "diff(<logcurve>)" A A A A
set sslop [ cv_compute "1/vecmax(<difflog>)" A A A A ]
puts $log_file " "
puts $log_file "sub solp          = $sslop  A/V"
puts $log_file " "
### Putting into Family Table #####
ft_scalar VT $VT1
ft_scalar gmax $gm
ft_scalar ioff $ioff
ft_scalar isat $isat
ft_scalar sslop $sslop
ft_scalar rout $rout
close $log_file
#----- END -----#

```

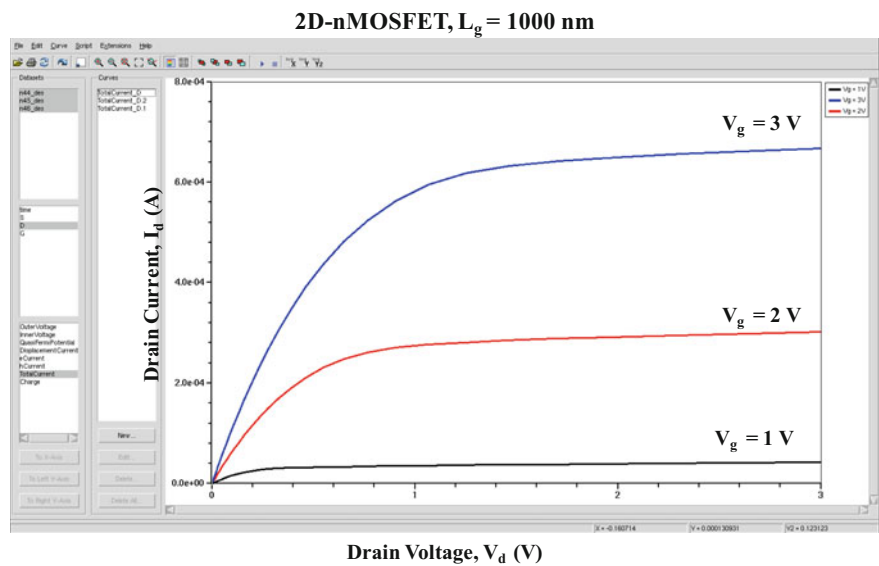


Fig. 2.29 I_d – V_d curve of simulation of 2D nMOSFET in Inspect tool

The electric properties of the simulation result of 2D nMOSFET output curve is as shown in Fig. 2.29 by **Inspect tool**.

2.4 [Example 2.3] 2D p-Type MOSFET with I_d – V_g Characteristics Simulation

The following three main program code files are all based on Synopsys Sentaurus TCAD 2014 version.

This 2D pMOSFET I_d – V_g simulation example is very similar to Example 2.1 nMOSFET I_d – V_g , only have difference in doping and electrodes bias setting. This is the standard example of 2D pMOSFET I_d – V_g example.

1. SDE – devise_dvs.cmd

This is the best example of 2D p MOSFET.

The line of code following; **is the prompt character for program designer to take note such that it will not be executed by the computer.**

```

;----- parameter -----;
(define Lg @Lg@)
(define tox @tox@)
(define tac 100)
(define Body 400)
(define LSDC 25)
(define LSD 25)
(define C_Doping 1e16)
(define D_Doping 1e20)
(define S_Doping 1e20)
(define B_Doping 1e15)
(define nm 1e-3)
(define x1 LSDC)
(define x2 (+ x1 LSD))
(define x3 (+ x2 Lg))
(define x4 (+ x3 LSD))
(define x5 (+ x4 LSDC))
(define y1 (- Body))
(define y2 tac)
(define y3 (+ tac tox))
;----- Structure -----;
"ABA"
;--- source ---
(sdegeo:create-rectangle
  (position 0 0 0)
  (position x1 y2 0) "Silicon" "SourceC" )
(sdegeo:create-rectangle
  (position x1 0 0)
  (position x2 y2 0) "Silicon" "Source" )
;--- Channel ---
(sdegeo:create-rectangle
  (position x2 0 0)
  (position x3 y2 0) "Silicon" "Channel" )
;--- Drain ---
(sdegeo:create-rectangle

```

```

(position    x3 0 0)
(position    x4 y2 0)  "Silicon" "Drain" )
(sdegeo:create-rectangle
(position    x4 0 0)
(position    x5 y2 0)  "Silicon" "DrainC" )
;--- Body ---
(sdegeo:create-rectangle
(position    0 0 0)
(position    x5 y1 0)  "Silicon" "Body" )
;--- Gate oxide ---
(sdegeo:create-rectangle
(position    x2 y2 0)
(position    x3 y3 0)  "SiO2" "Gateoxide" )
;----- Contact -----;
;---- Gate ----
(sdegeo:define-contact-set "G"
  4.0 (color:rgb 1.0 0.0 0.0) "###")
(sdegeo:define-2d-contact (find-edge-id (position (+ x2 (/ Lg 2)) y3 0)) "G")
;---- Source ----
(sdegeo:define-contact-set "S"
  4.0 (color:rgb 1.0 0.0 0.0) "###")
(sdegeo:define-2d-contact (find-edge-id (position 10 tac 0)) "S")
;---- Drain ----
(sdegeo:define-contact-set "D"
  4.0 (color:rgb 1.0 0.0 0.0) "###")
(sdegeo:define-2d-contact (find-edge-id (position (+ 50 Lg 35) tac 0)) "D")
;---- Substrate ----
(sdegeo:define-contact-set "substrate"
  4.0 (color:rgb 1.0 0.0 0.0) "###")
(sdegeo:define-2d-contact (find-edge-id (position (+ x2 (/ Lg 2)) (- Body) 0))
"substrate")
;----- Doping -----;
;--- Channel ---
(sdedr:define-constant-profile "dopedC" "PhosphorusActiveConcentration"
C_Doping )

```

```

(sdedr:define-constant-profile-region "RegionC" "dopedC" "Channel" )
;--- Source ---
(sdedr:define-constant-profile "dopedS" "BoronActiveConcentration" S_Doping )
(sdedr:define-constant-profile-region "RegionS" "dopedS" "Source" )
(sdedr:define-constant-profile "dopedSC" "BoronActiveConcentration" S_Doping )
(sdedr:define-constant-profile-region "RegionSC" "dopedSC" "SourceC" )
;--- Drain ---
(sdedr:define-constant-profile "dopedD" "BoronActiveConcentration" D_Doping )
(sdedr:define-constant-profile-region "RegionD" "dopedD" "Drain" )
(sdedr:define-constant-profile "dopedDC" "BoronActiveConcentration" D_Doping )
(sdedr:define-constant-profile-region "RegionDC" "dopedDC" "DrainC" )
;--- Body ---
(sdedr:define-constant-profile "dopedB" "PhosphorusActiveConcentration"
B_Doping )
(sdedr:define-constant-profile-region "RegionB" "dopedB" "Body" )
;----- Mesh -----;
;--- AllMesh ---
(sdedr:define-refinement-size "Cha_Mesh" 20 20 0 10 10 0)
(sdedr:define-refinement-material "channel_RF" "Cha_Mesh" "Silicon" )
;--- ChannelMesh ---
(sdedr:define-refinement-window "multiboxChannel" "Rectangle"
(position 25 (- 50) 0)
(position (+ 50 Lg 25) (+ tac 50) 0))
(sdedr:define-multibox-size "multiboxSizeChannel" 5 5 0 1 1 0)
(sdedr:define-multibox-placement "multiboxPlacementChannel"
"multiboxSizeChannel" "multiboxChannel")
(sdedr:define-refinement-function "multiboxPlacementChannel"
"DopingConcentration" "MaxTransDiff" 1)
;----- Save BND and CMD and rescale to nm -----;
(sde:assign-material-and-region-names (get-body-list) )
(sdeio:save-tdr-bnd (get-body-list) "n@node@_nm.tdr")
(sdedr:write-scaled-cmd-file "n@node@_msh.cmd" nm)
(define sde:scale-tdr-bnd
(lambda (tdrin sf tdrout)
(sde:clear)

```

```

(sdegeo:set-default-boolean "XX")
(sdeio:read-tdr-bnd tdrin)
(entity:scale (get-body-list) sf)
(sdeio:save-tdr-bnd (get-body-list) tdrout)
)
)
(sde:scale-tdr-bnd "n@node@_nm.tdr" nm "n@node@_bnd.tdr")
;----- END -----;

```

2. SDVICE – dessis_des.cmd

The line of code following ***** and **#** are the prompt characters for program designer to take note such that it will not be executed by the computer.

```

Electrode{
{name="D" voltage=0.0}
{name="S" voltage=0.0}
{name="G" voltage=0.0 WorkFunction=@WK@}
}
File{
  Grid="@tdr@"
  Plot="@tdrdat@"
  Current="@plot@"
  Output="@log@"
  parameter="@parameter@"
}
Physics{
  Mobility( DopingDep HighFieldSaturation Enormal )
  EffectiveIntrinsicDensity( OldSlotboom )

```

```

    Recombination( SRH(DopingDep) )
    hQuantumPotential
}
Math{
    -CheckUndefinedModels
    Number_Of_Threads=4
    Extrapolate
    Derivatives
    * Avalderivatives
    RelErrControl
    Digits=5
    ErRef(electron)=1.e10
    ErRef(hole)=1.e10
    Notdamped=50
    Iterations=20
    Directcurrent
    Method=ParDiSo
    Parallel= 2
*-VoronoiFaceBoxMethod
    NaturalBoxMethod
}
Plot{
    eDensity hDensity
    eCurrent hCurrent
    TotalCurrent/Vector eCurrent/Vector hCurrent/Vector
    eMobility hMobility
    eVelocity hVelocity
    eEnormal hEnormal
    ElectricField/Vector Potential SpaceCharge
    eQuasiFermi hQuasiFermi
    Potential Doping SpaceCharge
    SRH Auger
    AvalancheGeneration

```

```

DonorConcentration AcceptorConcentration
Doping
  eGradQuasiFermi/Vector hGradQuasiFermi/Vector
  eEparallel hEparallel
BandGap
BandGapNarrowing
Affinity
ConductionBand ValenceBand
hQuantumPotential
}
Solve {
  Coupled ( Iterations= 150){ Poisson hQuantumPotential }
  Coupled { Poisson hQuantumPotential Electron Hole }
  Quasistationary(
    InitialStep= 1e-3 Increment= 1.2
    MinStep= 1e-12 MaxStep= 0.95
    Goal { Name= "D" Voltage=@Vd@ }
  ){ Coupled { Poisson hQuantumPotential Electron Hole } }
  Quasistationary(
    InitialStep= 1e-3 Increment= 1.2
    MinStep= 1e-12 MaxStep= 0.02
    Goal { Name= "G" Voltage=@Vg@ }
    DoZero
  ){ Coupled { Poisson hQuantumPotential Electron Hole } }
}
*----- END -----*
```

3. INSPECT – inspect_inc.cmd

The line of code following # is the prompt character for program designer to take note such that it will not be executed by the computer.

```
#-----#
#      Script file designed to compute      :      #
#      * The threshold voltage              :   VT
#
#      * The transconductance               :   gm   #
#-----#
if { ! [catch {open n@previous@_ins.log w} log_file] } {
    set fileId stdout
}
puts $log_file " "
puts $log_file "      ----- "
puts $log_file "      Values of the extracted Parameters : "
puts $log_file "      ----- "
puts $log_file " "
puts $log_file " "
set DATE    [ exec date ]
set WORK    [ exec pwd  ]
puts $log_file "      Date      : $DATE "
puts $log_file "      Directory : $WORK "
puts $log_file " "
puts $log_file " "
#
#
#      idvgs=y(x);    vgsvgs=x(x); #
#set out_file n@previous@_des
proj_load "${out_file}.plt"
# ----- #
# I) VT = Xintercept(maxslope(ID[VGS])) or VT = VGS( IDS= 100nA/um ) #
# ----- #
cv_create    idvgs    "${out_file} G OuterVoltage" "${out_file} S TotalCurrent"
cv_create    vdsvgs   "${out_file} G OuterVoltage" "${out_file} S OuterVoltage"
#..... #
# 1) VT extracted as the intersection point with the X axis at the point #
#      where the id(vgs) slope reaches its maximum :      #
#..... #
set VT1      [ f_VT1 idvgs ]
```

```

#..... #
# 2) Printing of the whole set of extracted values (std output) : #
#..... #
puts $log_file "Threshold voltage VT1 = $VT1 Volts"
puts $log_file " "
#..... #
# 3) Initialization and display of curves on the main Inspect screen : #
cv_display idvgs
cv_lineStyle idvgs solid
cv_lineColor idvgs red
# ----- #
# II) gm = maxslope((ID[VGS])) #
# ----- #
set gm [ f_gm idvgs ]
puts $log_file " "
puts $log_file "Transconductance gm = $gm A/V"
puts $log_file " "
set ioff [ cv_compute "vecmin(<idvgs>)" A A A A ]
puts $log_file " "
puts $log_file "Current ioff = $ioff A"
puts $log_file " "
set isat [ cv_compute "vecmax(<idvgs>)" A A A A ]
puts $log_file " "
puts $log_file "Current isat = $isat A"
puts $log_file " "
set rout [ cv_compute "Rout(<idvgs>)" A A A A ]
puts $log_file " "
puts $log_file "Resistant rout = $rout A"
puts $log_file " "
cv_createWithFormula logcurve "log10(<idvgs>)" A A A A
cv_createWithFormula difflog "(-1)*diff(<logcurve>)" A A A A
set sslop [ cv_compute "1/vecmax(<difflog>)" A A A A ]
puts $log_file " "
puts $log_file "sub solp = $sslop A/V"
puts $log_file " "

```

```

#### Putting into Family Table #####
ft_scalar VT $VT1
ft_scalar gmax $gm
ft_scalar ioff $ioff
ft_scalar isat $isat
ft_scalar sslop $sslop
ft_scalar rout $rout
close $log_file
#----- END -----#

```

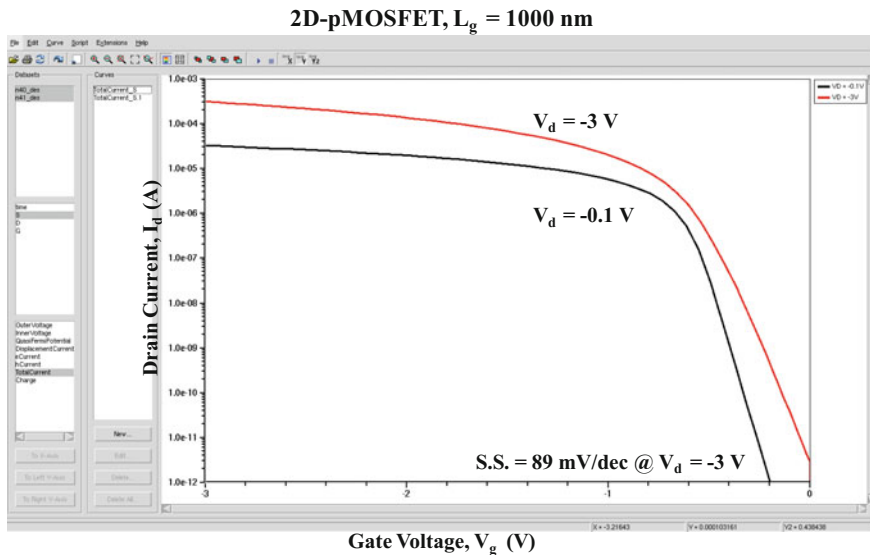


Fig. 2.30 I_d - V_g curve of simulation of 2D p-type MOSFET

The electric property of 2D pMOSFET transfer curve is as shown in Fig. 2.30.

2.5 [Example 2.4] 2D p-Type MOSFET with I_d - V_g Characteristics Simulation

This 2D pMOSFET I_d - V_d simulation example is very similar to Example 2.3 nMOSFET, only have difference in doping and electrodes bias setting. This is the standard example of 2D pMOSFET I_d - V_g .

The following three main program code files are all based on Synopsys Sentaurus TCAD 2014 version.

1. SDE – devise_dvs.cmd

The line of code following ; is the prompt character for program designer to take note such that it will not be executed by the computer.

```

;----- parameter -----;
(define Lg @Lg@)
(define tox @tox@)
(define tac 100)
(define Body 400)
(define LSDC 25)
(define LSD 25)
(define C_Doping 1e16)
(define D_Doping 1e20)
(define S_Doping 1e20)
(define B_Doping 1e15)
(define nm 1e-3)
(define x1 LSDC)
(define x2 (+ x1 LSD))
(define x3 (+ x2 Lg))
(define x4 (+ x3 LSD))
(define x5 (+ x4 LSDC))
(define y1 (- Body))
(define y2 tac)
(define y3 (+ tac tox))
;----- Structure -----;
"ABA"
;--- source ---
(sdegeo:create-rectangle
  (position 0 0 0)
  (position x1 y2 0) "Silicon" "SourceC" )
(sdegeo:create-rectangle
  (position x1 0 0)
  (position x2 y2 0) "Silicon" "Source" )
;--- Channel ---
(sdegeo:create-rectangle
  (position x2 0 0)
  (position x3 y2 0) "Silicon" "Channel" )
;--- Drain ---
(sdegeo:create-rectangle

```

```

(position    x3 0 0)
(position    x4 y2 0) "Silicon" "Drain" )
(sdegeo:create-rectangle
(position    x4 0 0)
(position    x5 y2 0) "Silicon" "DrainC" )
;--- Body ---
(sdegeo:create-rectangle
(position    0 0 0)
(position    x5 y1 0) "Silicon" "Body" )
;--- Gate oxide ---
(sdegeo:create-rectangle
(position    x2 y2 0)
(position    x3 y3 0) "SiO2" "Gateoxide" )
;----- Contact -----;
;---- Gate ----
(sdegeo:define-contact-set "G"
  4.0 (color:rgb 1.0 0.0 0.0) "##")
(sdegeo:define-2d-contact (find-edge-id (position (+ x2 (/ Lg 2)) y3 0)) "G")
;---- Source ----
(sdegeo:define-contact-set "S"
  4.0 (color:rgb 1.0 0.0 0.0) "##")
(sdegeo:define-2d-contact (find-edge-id (position 10 tac 0)) "S")
;---- Drain ----
(sdegeo:define-contact-set "D"
  4.0 (color:rgb 1.0 0.0 0.0) "##")
(sdegeo:define-2d-contact (find-edge-id (position (+ 50 Lg 35) tac 0)) "D")
;---- Substrate ----
(sdegeo:define-contact-set "substrate"
  4.0 (color:rgb 1.0 0.0 0.0) "##")
(sdegeo:define-2d-contact (find-edge-id (position (+ x2 (/ Lg 2)) (- Body) 0))
"substrate")
;----- Doping -----;
;--- Channel ---
(sdcd:define-constant-profile "dopedC" "PhosphorusActiveConcentration"
C_Doping )

```

```

(sdedr:define-constant-profile-region "RegionC" "dopedC" "Channel" )
;--- Source ---
(sdedr:define-constant-profile "dopedS" "BoronActiveConcentration" S_Doping )
(sdedr:define-constant-profile-region "RegionS" "dopedS" "Source" )
(sdedr:define-constant-profile "dopedSC" "BoronActiveConcentration" S_Doping )
(sdedr:define-constant-profile-region "RegionSC" "dopedSC" "SourceC" )
;--- Drain ---
(sdedr:define-constant-profile "dopedD" "BoronActiveConcentration" D_Doping )
(sdedr:define-constant-profile-region "RegionD" "dopedD" "Drain" )
(sdedr:define-constant-profile "dopedDC" "BoronActiveConcentration" D_Doping )
(sdedr:define-constant-profile-region "RegionDC" "dopedDC" "DrainC" )
;--- Body ---
(sdedr:define-constant-profile "dopedB" "PhosphorusActiveConcentration"
B_Doping )
(sdedr:define-constant-profile-region "RegionB" "dopedB" "Body" )
;----- Mesh -----;
;--- AllMesh ---
(sdedr:define-refinement-size "Cha_Mesh" 20 20 0 10 10 0)
(sdedr:define-refinement-material "channel_RF" "Cha_Mesh" "Silicon" )
;--- ChannelMesh ---
(sdedr:define-refinement-window "multiboxChannel" "Rectangle"
(position 25 (- 50) 0)
(position (+ 50 Lg 25) (+ tac 50) 0))
(sdedr:define-multibox-size "multiboxSizeChannel" 5 5 0 1 1 0)
(sdedr:define-multibox-placement "multiboxPlacementChannel"
"multiboxSizeChannel" "multiboxChannel")
(sdedr:define-refinement-function "multiboxPlacementChannel"
"DopingConcentration" "MaxTransDiff" 1)
;----- Save BND and CMD and rescale to nm -----;
(sde:assign-material-and-region-names (get-body-list) )
(sdeio:save-tdr-bnd (get-body-list) "n@node@_nm.tdr")
(sdedr:write-scaled-cmd-file "n@node@_msh.cmd" nm)
(define sde:scale-tdr-bnd
(lambda (tdrin sf tdrout)
(sde:clear)

```

```

(sdegeo:set-default-boolean "XX")
(sdeio:read-tdr-bnd tdrin)
(entity:scale (get-body-list) sf)
(sdeio:save-tdr-bnd (get-body-list) tdrout)
)
)
(sde:scale-tdr-bnd "n@node@_nm.tdr" nm "n@node@_bnd.tdr")
;----- END -----;

```

2. SDVICE – dessis_des.cmd

The line of code following **#** and ***** are the prompt characters for program designer to take note such that it will not be executed by the computer.

```

Electrode{
{name="D" voltage=0.0}
{name="S" voltage=0.0}
{name="G" voltage=0.0 WorkFunction=@WK@}
}
File{
    Grid="@tdr@"
    Plot="@tdrdat@"
    Current="@plot@"
    Output="@log@"
    parameter="@parameter@"
}
Physics{
    Mobility( DopingDep HighFieldSaturation Enormal )

```

```

    EffectiveIntrinsicDensity( OldSlotboom )
    Recombination( SRH(DopingDep) )
    hQuantumPotential
}
Math{
    -CheckUndefinedModels
    Number_Of_Threads=4
    Extrapolate
    Derivatives
    * Avalderivatives
    RelErrControl
    Digits=5
    ErRef(electron)=1.e10
    ErRef(hole)=1.e10
    Notdamped=50
    Iterations=20
    Directcurrent
    Method=ParDiSo
    Parallel= 2
    *-VoronoiFaceBoxMethod
    NaturalBoxMethod
}
Plot{
    eDensity hDensity
    eCurrent hCurrent
    TotalCurrent/Vector eCurrent/Vector hCurrent/Vector
    eMobility hMobility
    eVelocity hVelocity
    eEnormal hEnormal
    ElectricField/Vector Potential SpaceCharge
    eQuasiFermi hQuasiFermi
    Potential Doping SpaceCharge
    SRH Auger

```

```

AvalancheGeneration
DonorConcentration AcceptorConcentration
Doping
eGradQuasiFermi/Vector hGradQuasiFermi/Vector
eEparallel hEparallel
BandGap
BandGapNarrowing
Affinity
ConductionBand ValenceBand
hQuantumPotential
}
Solve {
  Coupled ( Iterations= 150){ Poisson hQuantumPotential }
  Coupled { Poisson hQuantumPotential Electron Hole }
    Quasistationary(
      InitialStep= 1e-3 Increment= 1.2
      MinStep= 1e-12 MaxStep= 0.02
      Goal { Name= "G" Voltage=@Vg@ }
    ){ Coupled { Poisson hQuantumPotential Electron Hole } }
    Quasistationary(
      InitialStep= 1e-3 Increment= 1.2
      MinStep= 1e-12 MaxStep= 0.95
      Goal { Name= "D" Voltage=@Vd@ }
      DoZero
    ){ Coupled { Poisson hQuantumPotential Electron Hole } }
  }
}
*----- END -----*
```

3. INSPECT – inspect_inc.cmd

The line of code following **#** is the **prompt character** for program designer to take note such that it will not be executed by the computer.

```
#-----#
#      Script file designed to compute      :      #
#      * The threshold voltage              :   VT
#      * The transconductance               :   gm   #
#-----#
if { ! [catch {open n@previous@_ins.log w} log_file] } {
    set fileId stdout
}
puts $log_file " "
puts $log_file "      ----- "
puts $log_file "      Values of the extracted Parameters : "
puts $log_file "      ----- "
puts $log_file " "
puts $log_file " "
set  DATE   [ exec  date ]
set  WORK   [ exec pwd  ]
puts $log_file "      Date       : $DATE "
puts $log_file "      Directory : $WORK "
puts $log_file " "
puts $log_file " "
#
#
#      idvgs=y(x);   vgsvgs=x(x);      #
set out_file n@previous@_des
```

```

proj_load "${out_file}.plt"
# I) VT = Xintercept(maxslope(ID[VGS])) or VT = VGS( IDS= 0.1 ua/um ) #
# ----- #
cv_create idvgs "${out_file} G OuterVoltage" "${out_file} S TotalCurrent"
cv_create vdsvgs "${out_file} G OuterVoltage" "${out_file} S OuterVoltage"
#..... #
# 1) VT extracted as the intersection point with the X axis at the point #
# where the id(vgs) slope reaches its maximum : #
#..... #
set VT1 [ f_VT1 idvgs ]
#..... #
# 2) Printing of the whole set of extracted values (std output) : #
#..... #
puts $log_file "Threshold voltage VT1 = $VT1 Volts"
puts $log_file " "
#..... #
# 3) Initialization and display of curves on the main Inspect screen : #
# ..... #
cv_display idvgs
cv_lineStyle idvgs solid
cv_lineColor idvgs red
# ----- #
# II) gm = maxslope((ID[VGS]) #
# ----- #
set gm [ f_gm idvgs ]
puts $log_file " "
puts $log_file "Transconductance gm = $gm A/V"
puts $log_file " "
set ioff [ cv_compute "vecmin(<idvgs>)" A A A A ]

```

```

puts $log_file " "
puts $log_file "Current ioff          = $ioff  A"
puts $log_file " "
set isat [ cv_compute "vecmax(<idvgs>)" A A A A ]
puts $log_file " "
puts $log_file "Current isat          = $isat  A"
puts $log_file " "
set rout [ cv_compute "Rout(<idvgs>)" A A A A ]
puts $log_file " "
puts $log_file "Resistant rout          = $rout  A"
puts $log_file " "
cv_createWithFormula logcurve "log10(<idvgs>)" A A A A
cv_createWithFormula difflog "(-1)*diff(<logcurve>)" A A A A
set sslop [ cv_compute "1/vecmax(<difflog>)" A A A A ]
puts $log_file " "
puts $log_file "sub solp          = $sslop  A/V"
puts $log_file " "
### Putting into Family Table #####
ft_scalar VT $VT1
ft_scalar gmax $gm
ft_scalar ioff $ioff
ft_scalar isat $isat
ft_scalar sslop $sslop
ft_scalar rout $rout
close $log_file
#----- END -----#

```

The electric property of 2D pMOSFET I_s-V_d is as shown in Fig. 2.31.

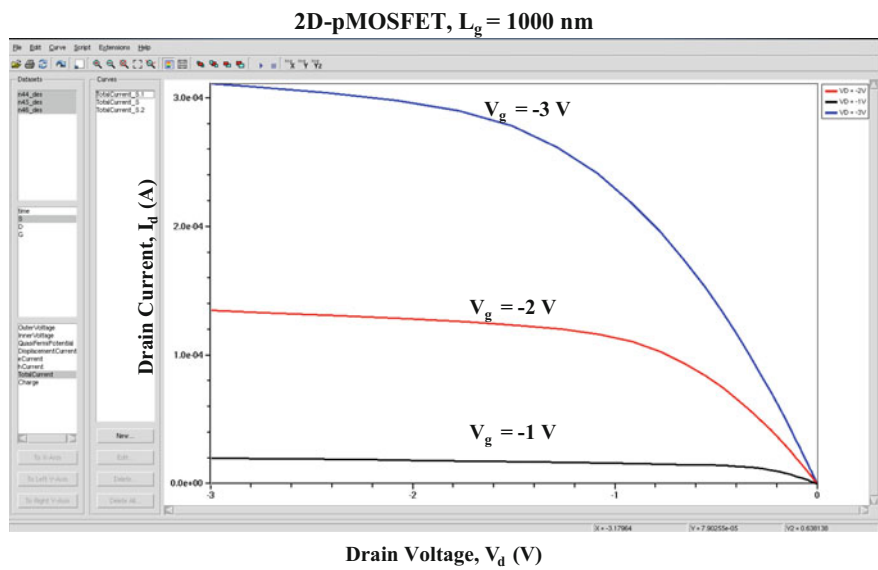


Fig. 2.31 I_s – V_d curve of the simulation of 2D pMOSFET by Inspect tool

2.6 [Example 2.5] 2D n-Type MOSFET with LDD (Lightly Doped Drain) Simulation

This is the standard example of 2D LDD nMOSFET. This 2D LDD nMOSFET I_d – V_g simulation example is very similar to Example 2.1 nMOSFET, only add and LDD doping region.

The following three main program code files are all based on Synopsys Sentaurus TCAD 2014 version.

Description: LDD (lightly doped drain) is an extremely effective method for reducing SCE of 2D MOSFET. The breakdown voltage at the junction is the function of highest electric field. When channel length is reduced, the bias voltage might not be reduced by the same ratio, such that the junction electric field will get even higher, which will make the effects of approximating accumulated breakdown and approximating penetration become more significant. In addition, when the device dimension is reduced, the parasite BJT will become more decisive, and the breakdown effect will be enhanced [1].

A method for reducing these breakdown effects is to change the dopant distribution of drain contact. By using region with light doping, the peak electric field in

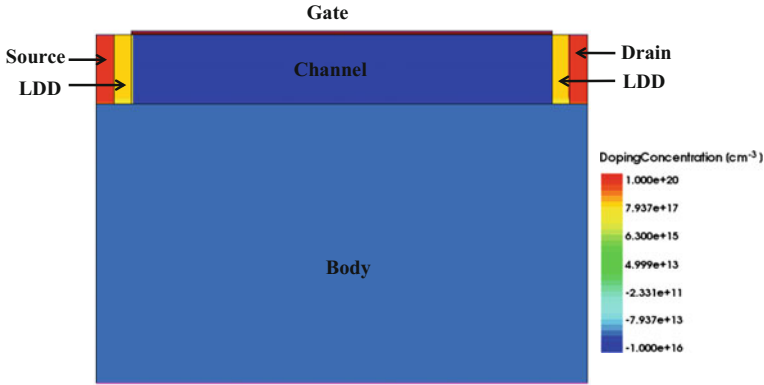


Fig. 2.32 Simulation of LDD device structure of 2D nMOSFET

the spatial charge region will be reduced, thus minimizing the breakdown effect. As for the peak value at drain junction, the electric field is the function of semiconductor doping and the function of curvature of n^+ drain region. In the LDD structure, the electric field of oxide-semiconductor junction is lower than the traditional structure. Among traditional devices, electric fields usually peak at the metallurgical junction, and it will be quickly reduced to zero at the drain. This is because the electric field cannot exist in the highly conductive n^+ region. On the other hand, the electric field in LDD device will be extended across the n region before being reduced to zero, and this effect will minimize the breakdown effect and hot carrier effect.

There are two disadvantages of LDD device. For one, the fabrication complexity is increased. For the other, the drain resistance is increased. Nonetheless, this extra process step can indeed fabricate the device with significantly improved performance. The cross section of LDD device is as shown in Fig. 2.32, in which the source terminal is changed to the lightly doped n region, which will lead to improve the device operating performance while reducing the process complexity. The series resistance will lead to increased device power consumption, so this factor must be taken into consideration for high-power device.

1. SDE – devise_dvs.cmd

The line of code following ; is the prompt character for program designer to take note such that it will not be executed by the computer;

```

;----- parameter -----;
(define Lg @Lg@)
(define tox @tox@)
(define tac 100)
(define Body 400)
(define LSDC 25)
(define LSD 25)
(define C_Doping 1e16)
(define DC_Doping 1e20)
(define D_Doping 1e18)
(define S_Doping 1e18)
(define SC_Doping 1e20)
;(define B_Doping 1e15)
(define B_Doping 1e16)
(define nm 1e-3)
(define x1 LSDC)
(define x2 (+ x1 LSD))
(define x3 (+ x2 Lg))
(define x4 (+ x3 LSD))
(define x5 (+ x4 LSDC))
(define y1 (- Body))
(define y2 tac)
(define y3 (+ tac tox))
;----- Structure -----;
"ABA"
;--- source ---
(sdegeo:create-rectangle
  (position 0 0 0)
  (position x1 y2 0) "Silicon" "SourceC" )
(sdegeo:create-rectangle
  (position x1 0 0)
  (position x2 y2 0) "Silicon" "Source" )
;--- Channel ---
(sdegeo:create-rectangle
  (position x2 0 0)

```

```

        (position      x3 y2 0)  "Silicon" "Channel" )
;--- Drain ---
(sdegeo:create-rectangle
  (position      x3 0 0)
  (position      x4 y2 0)  "Silicon" "Drain" )
(sdegeo:create-rectangle
  (position      x4 0 0)
  (position      x5 y2 0)  "Silicon" "DrainC" )
;--- Body ---
(sdegeo:create-rectangle
  (position      0 0 0)
  (position      x5 y1 0)  "Silicon" "Body" )
;--- Gate oxide ---
(sdegeo:create-rectangle
  (position      x2 y2 0)
  (position      x3 y3 0)  "SiO2" "Gateoxide" )
;----- Contact -----;
;---- Gate ----
(sdegeo:define-contact-set "G"
  4.0 (color:rgb 1.0 0.0 0.0) "###")
(sdegeo:define-2d-contact (find-edge-id (position (+ x2 (/ Lg 2)) y3 0)) "G")
;---- Source ----
(sdegeo:define-contact-set "S"
  4.0 (color:rgb 1.0 0.0 0.0) "###")
(sdegeo:define-2d-contact (find-edge-id (position 10 tac 0)) "S")
;---- Drain ----
(sdegeo:define-contact-set "D"
  4.0 (color:rgb 1.0 0.0 0.0) "###")
(sdegeo:define-2d-contact (find-edge-id (position (+ 50 Lg 35) tac 0)) "D")
;---- Substrate ----
(sdegeo:define-contact-set "substrate"
  4.0 (color:rgb 1.0 0.0 0.0) "###")
(sdegeo:define-2d-contact (find-edge-id (position (+ x2 (/ Lg 2)) (- Body) 0))
"substrate")
;----- Doping -----;

```

```

;--- Channel ---
(sdedr:define-constant-profile "dopedC" "BoronActiveConcentration" C_Doping )
(sdedr:define-constant-profile-region "RegionC" "dopedC" "Channel" )
;--- Source ---
(sdedr:define-constant-profile "dopedS" "PhosphorusActiveConcentration"
S_Doping )
(sdedr:define-constant-profile-region "RegionS" "dopedS" "Source" )
(sdedr:define-constant-profile "dopedSC" "PhosphorusActiveConcentration"
SC_Doping )
(sdedr:define-constant-profile-region "RegionSC" "dopedSC" "SourceC" )
;--- Drain ---
(sdedr:define-constant-profile "dopedD" "PhosphorusActiveConcentration"
D_Doping )
(sdedr:define-constant-profile-region "RegionD" "dopedD" "Drain" )
(sdedr:define-constant-profile "dopedDC" "PhosphorusActiveConcentration"
DC_Doping )
(sdedr:define-constant-profile-region "RegionDC" "dopedDC" "DrainC" )
;--- Body ---
(sdedr:define-constant-profile "dopedB" "BoronActiveConcentration" B_Doping )
(sdedr:define-constant-profile-region "RegionB" "dopedB" "Body" )
;----- Mesh -----;
;--- AllMesh ---
(sdedr:define-refinement-size "Cha_Mesh" 20 20 0 10 10 0)
(sdedr:define-refinement-material "channel_RF" "Cha_Mesh" "Silicon" )
;--- ChannelMesh ---
(sdedr:define-refinement-window "multiboxChannel" "Rectangle"
(position 25 (- 50) 0)
(position (+ 50 Lg 25) (+ tac 50) 0))
(sdedr:define-multibox-size "multiboxSizeChannel" 5 5 0 1 1 0)
(sdedr:define-multibox-placement "multiboxPlacementChannel"
"multiboxSizeChannel" "multiboxChannel")
(sdedr:define-refinement-function "multiboxPlacementChannel"
"DopingConcentration" "MaxTransDiff" 1)
;----- Save BND and CMD and rescale to nm -----;
(sde:assign-material-and-region-names (get-body-list) )

```

```

(sdeio:save-tdr-bnd (get-body-list) "n@node@_nm.tdr")
(sdedr:write-scaled-cmd-file "n@node@_msh.cmd" nm)
(define sde:scale-tdr-bnd
  (lambda (tdrin sf tdrout)
    (sde:clear)
    (sdegeo:set-default-boolean "XX")
    (sdeio:read-tdr-bnd tdrin)
    (entity:scale (get-body-list) sf)
    (sdeio:save-tdr-bnd (get-body-list) tdrout)
  )
)
(sde:scale-tdr-bnd "n@node@_nm.tdr" nm "n@node@_bnd.tdr")
;----- END -----;

```

2. SDVICE – dssis_des.cmd

The line of code following **#** and ***** are the prompt characters for program designer to take note such that it will not be executed by the computer.

```

Electrode{
  {name="D" voltage=0.0}
  {name="S" voltage=0.0}
  {name="G" voltage=0.0 WorkFunction=@WK@}
}
File{
  Grid="@tdr@"
  Plot="@tdrdat@"
  Current="@plot@"
  Output="@log@"
  parameter="@parameter@"
}
Physics{
  Mobility( DopingDep HighFieldSaturation Enormal )
  EffectiveIntrinsicDensity( OldSlotboom )
  Recombination( SRH(DopingDep) )
  eQuantumPotential
}
Math{
  -CheckUndefinedModels

```

```

Number_Of_Threads=4
Extrapolate
Derivatives
* Avalderivatives
RelErrControl
Digits=5
ErRef(electron)=1.e10
ErRef(hole)=1.e10
Notdamped=50
Iterations=20
Directcurrent
Method=ParDiSo
Parallel= 2
*-VoronoiFaceBoxMethod
    NaturalBoxMethod
}
Plot{
    eDensity hDensity
    eCurrent hCurrent
    TotalCurrent/Vector eCurrent/Vector hCurrent/Vector
    eMobility hMobility
    eVelocity hVelocity
    eEnormal hEnormal
    ElectricField/Vector Potential SpaceCharge
    eQuasiFermi hQuasiFermi
    Potential Doping SpaceCharge
    SRH Auger
    AvalancheGeneration
    DonorConcentration AcceptorConcentration

Doping
    eGradQuasiFermi/Vector hGradQuasiFermi/Vector
    eEparallel hEparallel
    BandGap
    BandGapNarrowing

```

```

Affinity
ConductionBand ValenceBand
eQuantumPotential
}
Solve {
  Coupled ( Iterations= 150){ Poisson eQuantumPotential }
  Coupled { Poisson eQuantumPotential Electron Hole }
  Quasistationary(
    InitialStep= 1e-3 Increment= 1.2
    MinStep= 1e-12 MaxStep= 0.95
    Goal { Name= "D" Voltage=@Vd@ }
  ){ Coupled { Poisson eQuantumPotential Electron Hole } }
  Quasistationary(
    InitialStep= 1e-3 Increment= 1.2
    MinStep= 1e-12 MaxStep= 0.02
    Goal { Name= "G" Voltage=@Vg@ }
    DoZero
  ){ Coupled { Poisson eQuantumPotential Electron Hole } }
}
*----- END -----*

```

3. INSPECT – inspect_inc.cmd

The line of code following # **is the prompt character** for program designer to take note such that it will not be executed by the computer.

```

#-----#
#      Script file designed to compute      :      #
#      * The threshold voltage      : VT#      #
#      * The transconductance      : gm      #
#-----#
if { ! [catch {open n@previous@_ins.log w} log_file] } {
    set fileId stdout
}

puts $log_file " "
puts $log_file "      ----- "
puts $log_file "      Values of the extracted Parameters : "
puts $log_file "      ----- "
puts $log_file " "
puts $log_file " "
set DATE [ exec date ]
set WORK [ exec pwd ]
puts $log_file "      Date      : $DATE "
puts $log_file "      Directory : $WORK "
puts $log_file " "
puts $log_file " "
#
#
#      idvgs=y(x) ;      vgsvgs=x(x) ; #
set out_file n@previous@_des
proj_load "${out_file}.plt"
# ----- #
# I) VT = Xintercept(maxslope(ID[VGS])) or VT = VGS( IDS= 0.1 ua/um ) #
# ----- #
cv_create idvgs "${out_file} G OuterVoltage" "${out_file} D TotalCurrent"
cv_create vdsvgs "${out_file} G OuterVoltage" "${out_file} D OuterVoltage"
#..... #
# 1) VT extracted as the intersection point with the X axis at the point #
#      where the id(vgs) slope reaches its maximum :      #
#..... #
set VT1 [ f_VT1 idvgs ]
#..... #

```



```
#### Putting into Family Table #####
ft_scalar VT $VT1
ft_scalar gmax $gm
ft_scalar ioff $ioff
ft_scalar isat $isat
ft_scalar sslop $sslop
ft_scalar rout $rout
close $log_file
#----- END -----#
```

The electric property is as shown in Fig. 2.33, which shows that when LDD (lightly doped drain) is added into the original device, the leakage current (I_{off}) is reduced and the sub-threshold slope (SS) is significantly improved as compared to Fig. 2.23. Using LDD for reducing short-channel effect, the 2D n-type MOSFET still has good performance at $L_g = 400\text{ nm}$.

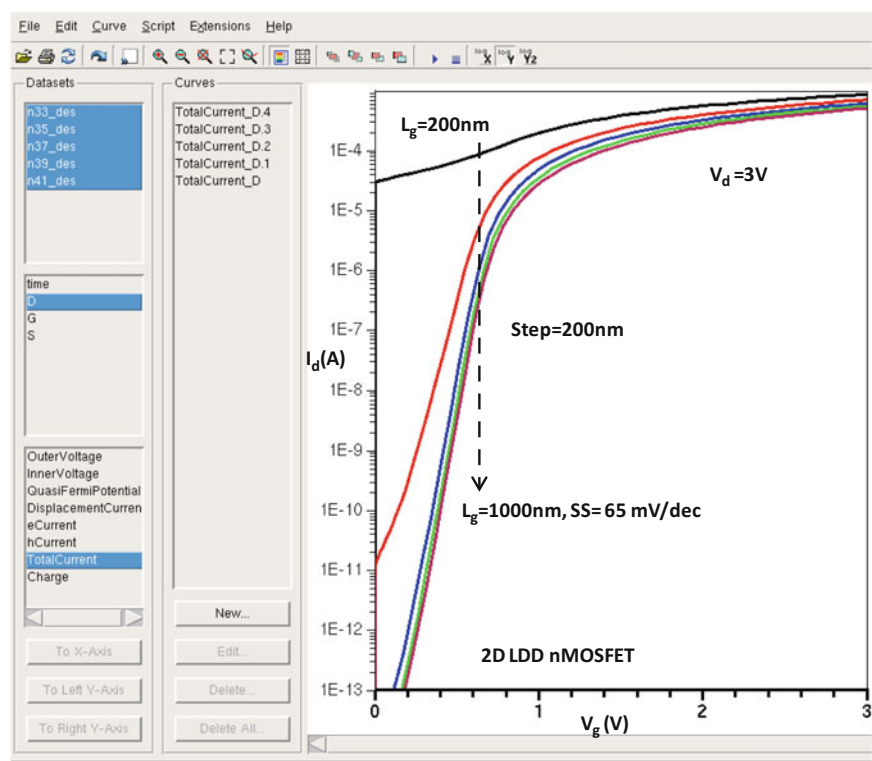


Fig. 2.33 Simulation of I_d - V_g curve of 2D n-type device with LDD

The more complicated well-doping process, anti-punch through process, and retrograde process can be simulated and developed by readers.

2.7 Summary

We introduce fundamental applications of TCAD simulation software, including electric property and physical property analysis in several 2D MOSFET simulations. The meanings of various tool codes are also explained. Readers can get a quick start for using Synopsys Sentaurus TCAD 2014 version.

References

1. S.M. Sze, K.K. Ng, Physics of Semiconductor Devices, 3rd ed (Wiley, New York, 2007)
2. TCAD Sentaurus Device, Synopsys SDevice Ver. J-2014.09 (Synopsys, Inc., Mountain View, CA, USA, 2014)
3. C.C. Hu, Modern Semiconductor Devices for Integrated Circuits (Pearson Education, Inc., 2010)

3D TCAD Simulation for CMOS Nanoelectronic Devices

Wu, Y.-C.; Jhan, Y.-R.

2018, XIII, 330 p. 243 illus., 240 illus. in color.,

Hardcover

ISBN: 978-981-10-3065-9