i-MOS Platform

(Version 2014.01)

User Manual

Lining Zhang, Mansun Chan

Department of Electronic and Computer Engineering
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong

The *i*-MOS team

- **Principal Investigator**
  Prof. Mansun Chan

- **Developers**
  Dr. Lining Zhang
  Aixi Zhang
  Raju Salahuddin
  Dr. Muthupandian Cheralathan

- **Programmers**
  Ngai Sing Ben Ng
  Richeng Huang
  Chun Ming Kenneth Au
  Xiaodong Meng

- **Previous Version Developers**
  Dr. Hao Wang
  Dr. Xiaoxu Cheng
  Lining Zhang

- **Programmers**
  Ka Chun Cyrix Tam
  Kwok Shiu Andy Wong
  Sai Kin Cheung
  Wing Hang Seifer Tsang
  Hamza Zia

Acknowledgement

- **Collaborators**
  Prof. Yu Cao, Arizona State University
  Prof. Jin He, Peking University
  Prof. Yan Wang, Tsinghua University
  Prof. Philip Wong, Stanford University
  Dr. Stanislav Markov, The University of Hong Kong

- **Support**
  University Grant Committee, Hong Kong

***i*-MOS website**: http://i-mos.org

Table of Contents

Chapter

1

Introduction to *i*-MOS

i-MOS (*i*nteractive Modeling and Online Simulation) platform is intended to be an open platform for model developers and circuit designers to interact. It provides a simple user interface through a web-browser to allow users to gain access to a number of models before they are implemented in commercial simulators. In addition, *i*-MOS utilizes a tailored open source NGSpice [1] with additional device models as the circuit simulation engine so that users are able to perform simulation anywhere through a browser without going through complicated software installation process. Version control is also performed by the *i*-MOS group directly to ensure consistency in the simulation results.

To model developers, *i*-MOS can be a platform for them to distribute and advertise their models. Currently, many people are developing compact models with many publications every year. However, many models remain unnoticed after publication and their completeness or practicality is never fully tested. *i*-MOS provides a platform for model developers to demonstrate the performance of their models when they are implemented into a circuit simulator. With these efforts, we hope to develop a standard for the model development process and procure appropriate benchmark tests. Users and developers can also directly compare different models to select the one most suitable for their applications.

Besides the technical part, we also hope that *i*-MOS can be used as a platform to exchange idea through various discussion groups. Through the interaction of users and model developers, we hope people can work together to continuously improve the quality of existing models to benefit both the technology and design community.

1.1 System Architecture

1.1.1 Performing Simulation

From the user side, the *i*-MOS platform consists of a web-based user interface constructed using the Linux, Apache, MYSQL and PHP (LAMP) solution. The interface takes in the user input and passes it to NGSpice, an open source SPICE engine running at our server. When simulation is completed, simulation data is returned to the browser. A

client side program reads the data, formats it and displays it to the user through a graphic user interface. Simplified system architecture is shown in Fig. 1.1.

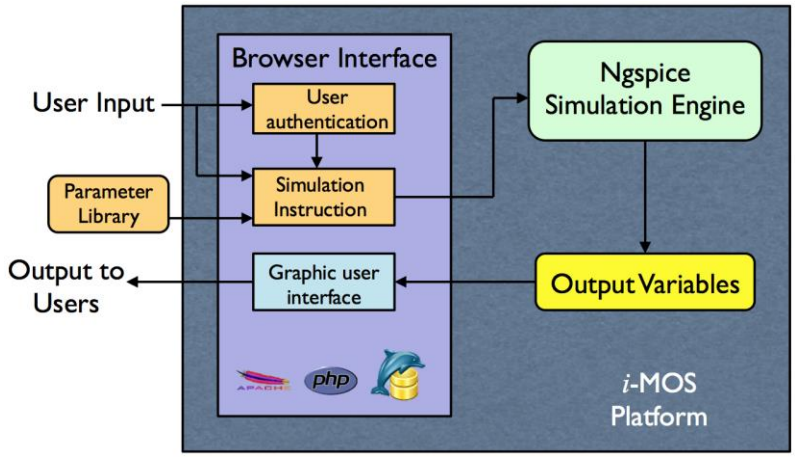

Fig. 1.1: Simplified system architecture of the *i*-MOS platform.

On the server side, the combination of Apache and PHP provides a good tradeoff between functionality, stability and flexibility. Most of the simulation is done at the server side without burdening the client side computing power besides formatting the output data.

1.1.2 Adding New Models

It is the intention of *i*-MOS to provide an authoring tool for model developer to upload their model implemented in Verilog-A [2] code directly to *i*-MOS for users to test and evaluate. However, this part of the system is not complete in this current release due to the lack of a robust open source Verilog-A compiler. Currently we are using the Automatic Device Model Synthesizer (ADMS) [3] together with experienced programmers to help the compilation of worthwhile models to the *i*-MOS platform. Model developers interested to upload their models through the *i*-MOS platform please contact the *i*-MOS team directly before the automated authoring tools are available.

1.2 Available Services

The *i*-MOS project is a large-scale project that can only be launched in phases. In the current release, the follow services are available:

- 8 models implemented in the current release, including the BSIM3 (version 3.1) for an evaluation purpose
- *I*-*V* characteristics, *Q*-*V* characteristics, *g*-*V* characteristics and *C*-*V* characteristics if the model provides the corresponding state variable
- Uploading data for comparison with the simulation

- Circuit simulation with the new models as well as other models already available in NGSpice
- Rating and commenting on a particular model
- Discussion group with various compact model related issues (lightly moderated)
- Reading and posting news and resources to various compact model related activities and organizations (lightly moderated)
- Verilog-A code of the model at the discretion of the model developers
- A collection of standard parameter sets (the Predictive Technology Model for BSIM3 [4]) for existing technology
- Friendly text editing [5] for the raw input composition of circuit simulations

Some features are going to be implemented shortly in the near future that includes:

- Benchmark circuits for model testing and comparison
- Archive of experimental data or data from first principle atomistic simulation
- More user friendly GUI for circuit simulation
- Mathematical manipulation of data in the graphic tools

Some service to expect, but need to wait a little bit longer:

- Model authoring tools to directly insert a model to the *i*-MOS platform
- Parameter extraction tools to extract parameters based on uploaded data or those available in the curated archive
- Statistical simulation data with as a result of variation of the parameters

If there are other services that you would like to see, please feel free to contact us through info@i-mos.org

References

[1] http://ngspice.sourceforge.net/
[2] Verilog-AMS Laguage Reference Manual, Open Verilog International, 1999
[3] L. Lemaitre, C. McAndrew, and S. Hamm, "ADMS-automatic device model synthesizer", IEEE CICC 2002, pp. 27-30
[4] http://ptm.asu.edu
[5] http://codemirror.net/index.html

Chapter

2

Single Device Simulation

The main purpose for single device simulation is to evaluate models or to obtain predictive device data. Only registered members can perform this function. The registration process is self-descriptive with the instructions online, and is free of charge.

2.1 The Model Page

To enter the single device simulation page, just click on the "Models" tag. It will bring you to the device model page, which displays the models currently available in *i*-MOS as shown in Fig. 2.1. You may then click on one of the models to study its characteristics.

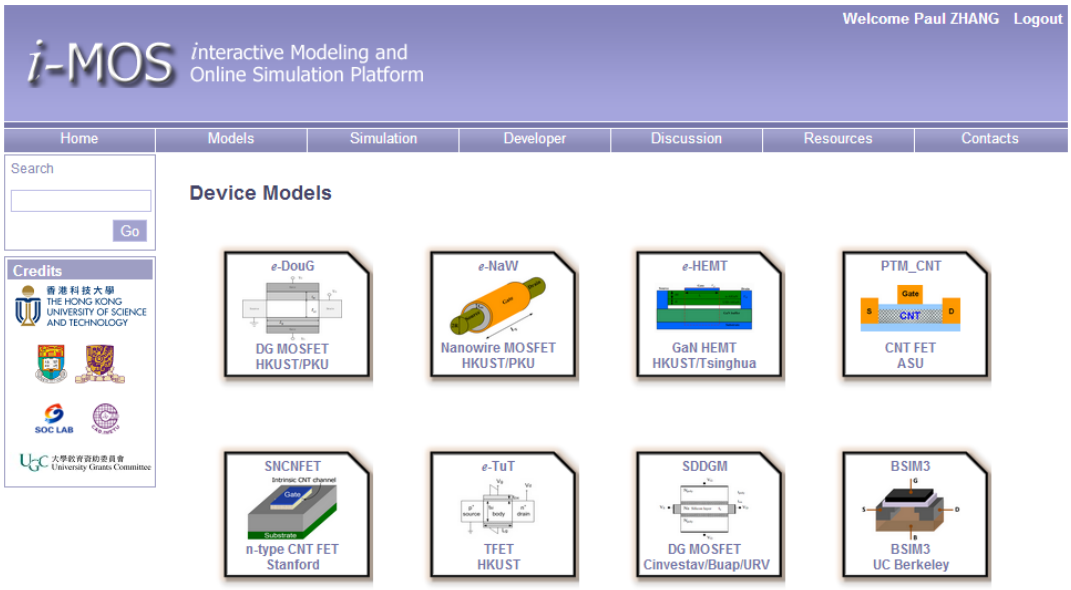

Fig. 2.1: The Model page showing available models

After you entered one specific model page, a new set of options will show up and their function will be described in the following section.

2.2 The Model Options

2.2.1 Description

The first tag in the model options brings you to the model description page as shown in Fig. 2.2, where the information on the model is given, including the authors, organization and contact information if there is any question.

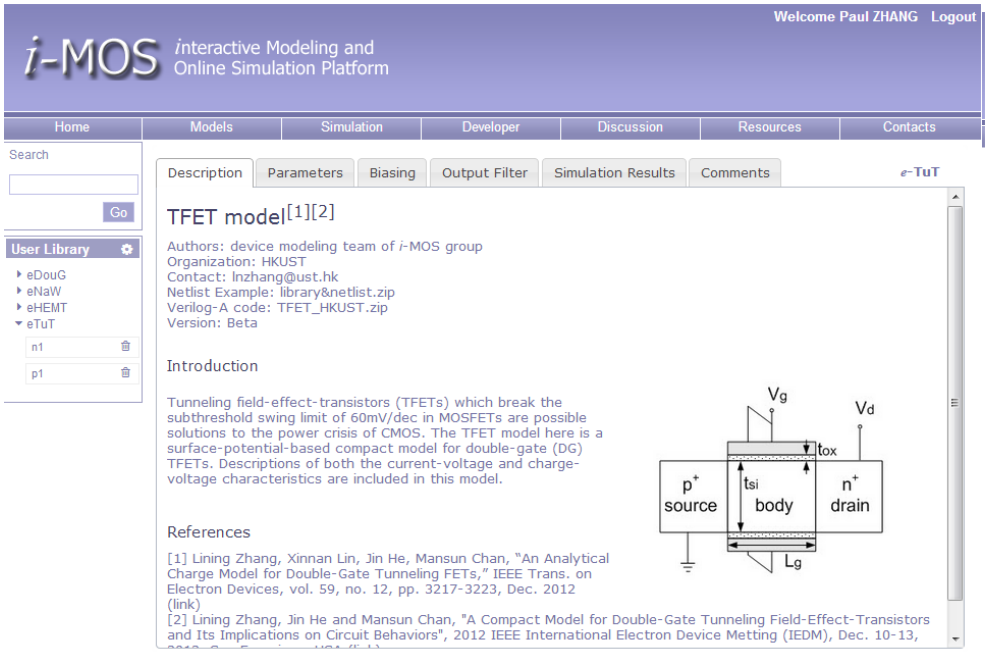

Fig. 2.2: The model description page showing all the relevant information of the model

The content in the page is provided directly by the model developer and the amount of information varies depending on the dedication of the model developer. For some models seriously formulated to be used in practical applications, information including source code, sample netlist, user guide and reference documents can be found in this page as well.

2.2.2 Parameters

The tags on the model pages indicate the sequence to display the characteristics of the device as given by the model. The first thing to define in the process is the value of

the parameters. When you click on the parameter page, you will see a list of parameters that you need to enter as shown in Fig. 2.3. The default values are already filled in in case you just want to test a generic case.

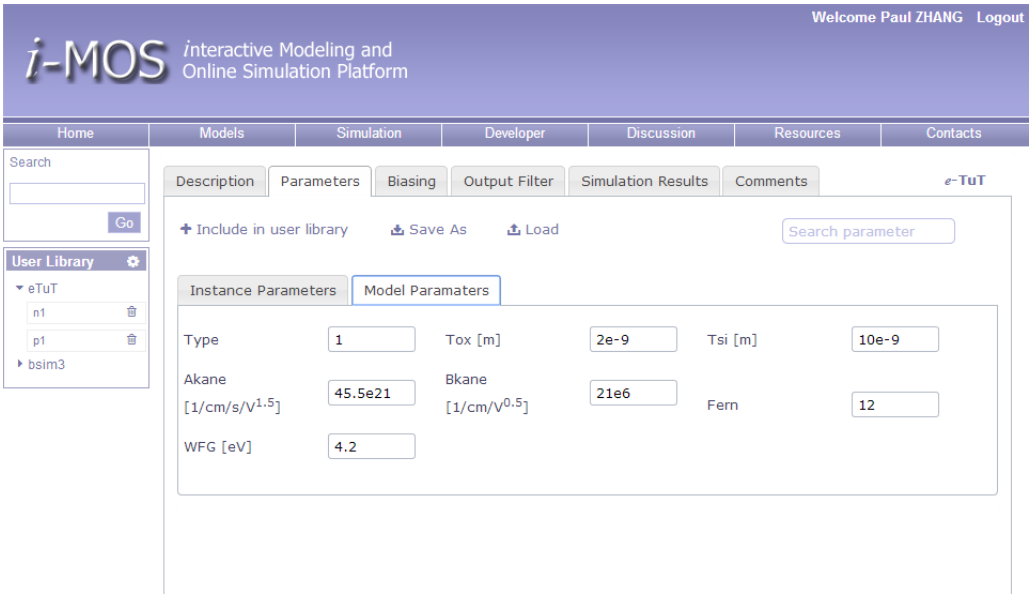

Fig. 2.3: The model parameter page showing the relevant parameters of the selected model

There are two types of parameters: instance parameters and model parameters. Instance parameters refer to those specified by circuit designers which affect the circuit layout, for example, the transistor gate length and width. Model parameters refer to those determined by the device manufacture process which in principle cannot be changed by the circuit designers, for example, the gate oxide thickness or the carrier mobility. On *i*-MOS, you have freedom to change all parameters. They can also come from the model parameter extractions according to the real device data.

You can save the parameters by clicking the 'Save as' button. It is the same as the common saving functions. Afterwards, you can import your saved parameters by clicking the 'Load' button. You can use the search box to allocate one specific parameter. This will help when the number of parameters increases.

Another 'Collection' button for the BSIM3 model as shown in Fig.2.4 is the link to a collection of standard model parameter sets. You can see the model collection window by clicking the 'Collection' button. Applying the standard parameter sets is easily done by clicking the parameter files with an extension of .ipa. The Predictive Technology Models are adopted from http://ptm.asu.edu with necessary adjustments of the syntax and simulator compatibility.

The 'Include in user library' button on the parameter page will be introduced in the next chapter.

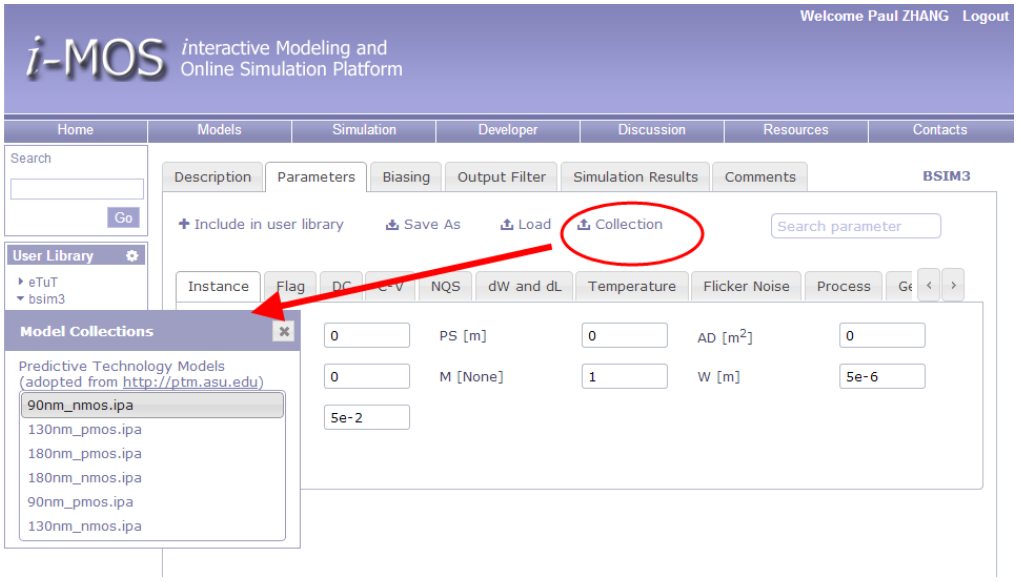

Fig. 2.4: The 'Model Collections' window appears when a user clicks the 'Collection' button. The standard model parameter set will be applied to the model parameters by left clicking.

2.2.3 Biasing

The next step is defining the terminal voltages for the modeled device in the biasing page as shown in Fig.2.5.

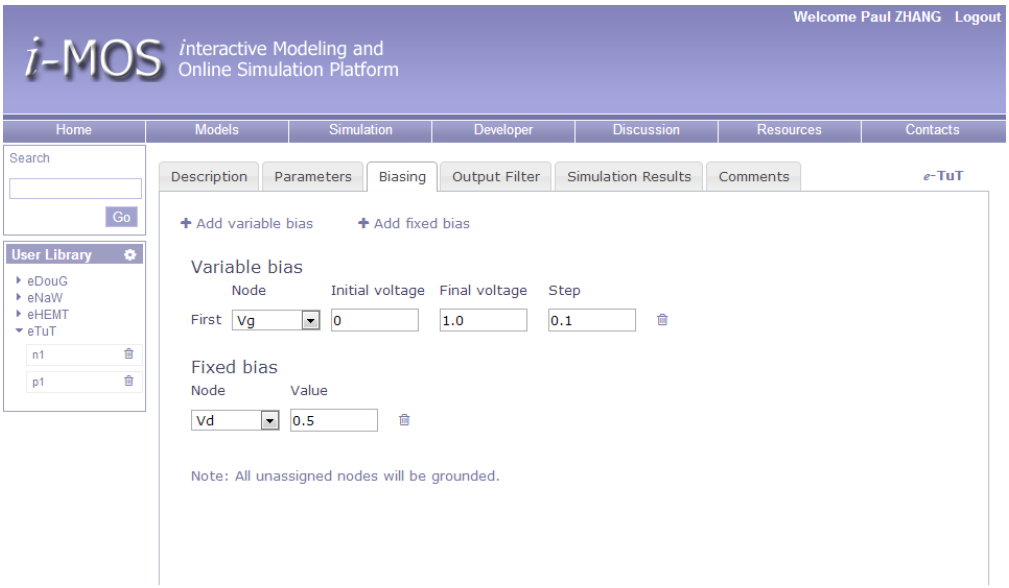

Fig. 2.5: The biasing page showing the definitions of the node voltages for the selected model

When you click the biasing tag for the first time, you will not see any node voltage definition. The voltages shown in Fig.2.5 as an example can be easily created by clicking the 'Add variable bias' or 'Add fixed bias' button. Variable bias is the one that allows you to define the *initial voltage*, *final voltage* and the *step*. At most two variable bias voltages can be added. Fixed bias is the one that has a constant value. You can specify the node voltage source by clicking the dropdown menu and choosing from the displayed nodes. In this *i*-MOS release, there is one virtual node, Vb created in the three terminal device models (*e*-DouG, *e*-NaW, *e*-TuT), which does not affect the device characteristics. This virtual node should be defined in the netlist as will be introduced in chapter 4. All unassigned nodes will be grounded.

2.2.4 Output Filter

You can check the terminal current and charges or capacitances of the selected device by ticking the items in the output filter page if those are given in analytical expressions in the model, as shown in Fig. 2.6. Both linear and logarithm scales are supported. From this page you can tell whether a model is complete by including descriptions of current-voltage and charge (capacitance)-voltage characteristics.

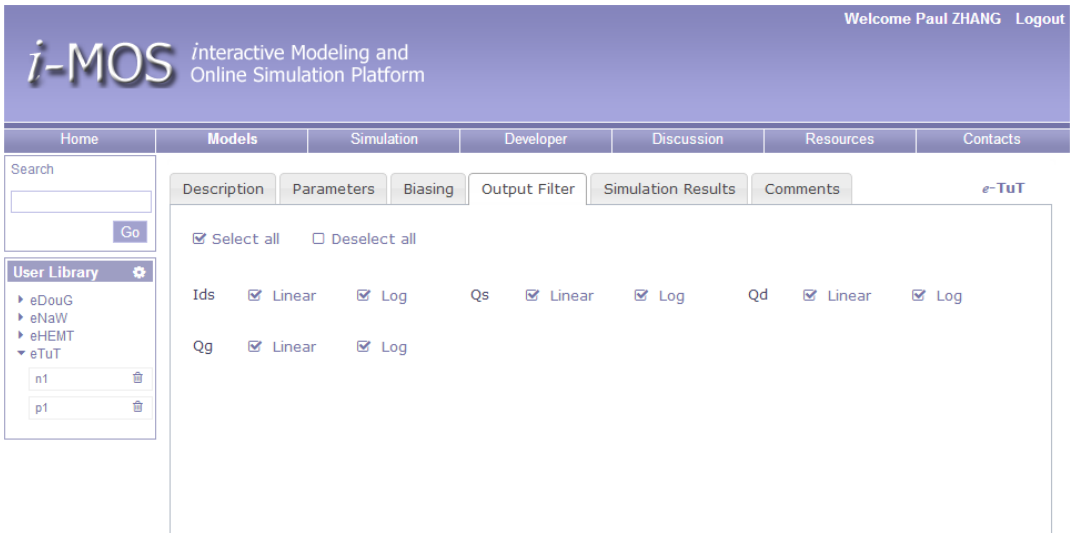

Fig. 2.6: The output filter page showing the output variables supported by the selected model

2.2.5 Simulation Results

Upon the above steps, you are ready to run simulations and see the device characteristics. After clicking the 'Run Simulation', you will be able to see the graphic output in Fig. 2.7. In case you want to abort the running, press the Abort option.

Fig. 2.7: The simulation results page showing the graphic output that you select in the output filter page

You can visualize any of the output results by selecting it from the dropdown menu. For each graphic output, you can enlarge it by left clicking. A new window pops out, where you can track the real data by pointing along the line with the mouse. The graphic output in the png format can be saved by clicking the 'Save as PNG'.

The raw data output in the csv format can be directly downloaded by clicking the 'Download raw data' button.

If you upload you real data and tick the 'Show custom data', you will append your data to the graphic output, as shown in Fig. 2.8. You can try to fit the model to your real data by tuning the model parameters and running the simulation again. This is a preliminary facility while the automated parameter extraction module is not yet available in this *i*-MOS release.

Fig. 2.8: The simulation results page showing the graphic output together with the custom data

2.2.6 Comments

You can provide your feedback about the model by leaving a comment under the comment tag, as shown in Fig. 2.9.

It is expected that the close interactions between you and the model developers will help improve the model and finally benefit both. The comment also goes to the discussion forum which will be covered in chapter 5.

A rating function is also implemented in the comment page. You can rate the models according to your experience. Multiple ratings are accepted, but only your final rating is recorded and used to derive the average score displayed as shown in Fig. 2.9.

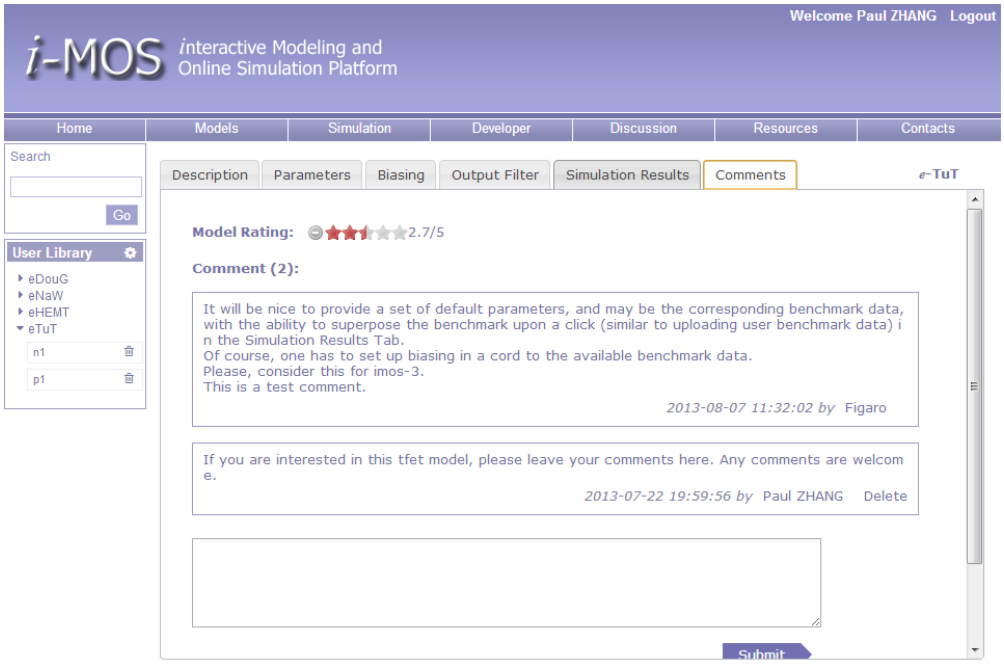

Fig. 2.9: The comments page showing the average model ratings and the users' comments

Chapter

3

User Library Construction

The user library provides the link between single device simulation and circuit simulation. After you have tuned a model to match your need, you may save it into your own user library to be used in circuit simulation described in Chapter 4.

3.1 Defining Model Parameters

The process to set model parameters has been described in Chapter 2. You may work on your model and refine the parameters following the description given in section 2.2.2. When you are happy with the parameter set, you may click the "include in user library" option and the model will be added to the corresponding user library as shown in Fig. 3.1

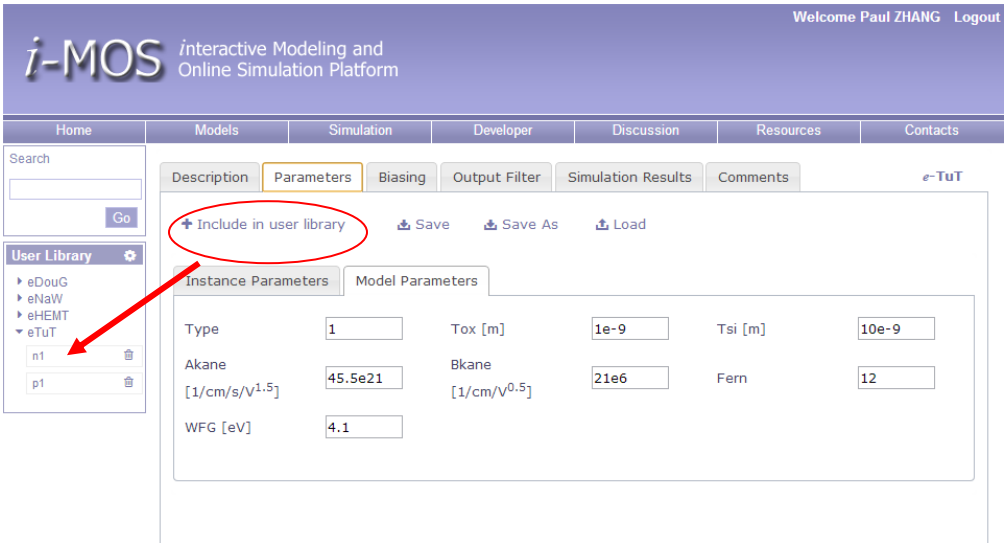

Fig. 3.1: When a user click the "include in user library" option, the model parameter set will be added to the user library with a user defined name.

You may add as many model parameter sets to the user library. This user library is shared with the circuit simulation page which means all models you added to this library are available for use in circuit simulation. If you have accidentally added a set of parameters you do not want, you may remove it by pressing the trash can symbol next to the model.

It should be noted that parameters governing the device behavior are categorized into instant parameters and model parameters. Only model parameters are added to the user library and you need to include the instance parameters in the input deck of the simulation file yourself.

3.2 Loading and Saving User Library

Some simple functions to manage the user library are provided. You may load or save an existing user library to your own computer. After you have finished constructing the user library, you may save it into your own computer by using the download function as shown in Fig. 3.2. After entering the file name, it will be saved in the default location set by your browser with an .iml extension. This file can be moved to other places and imported to the system later by using the upload function. Pressing the 'New' function clears the user library and let you start over with a blank one.

Pressing any model in the model library provide shortcut to jump to that model pages. Pressing any parameter set below a model will load that parameter set to the parameter-editing page. By doing so, the current parameter set on the parameter-editing page will be overwritten.

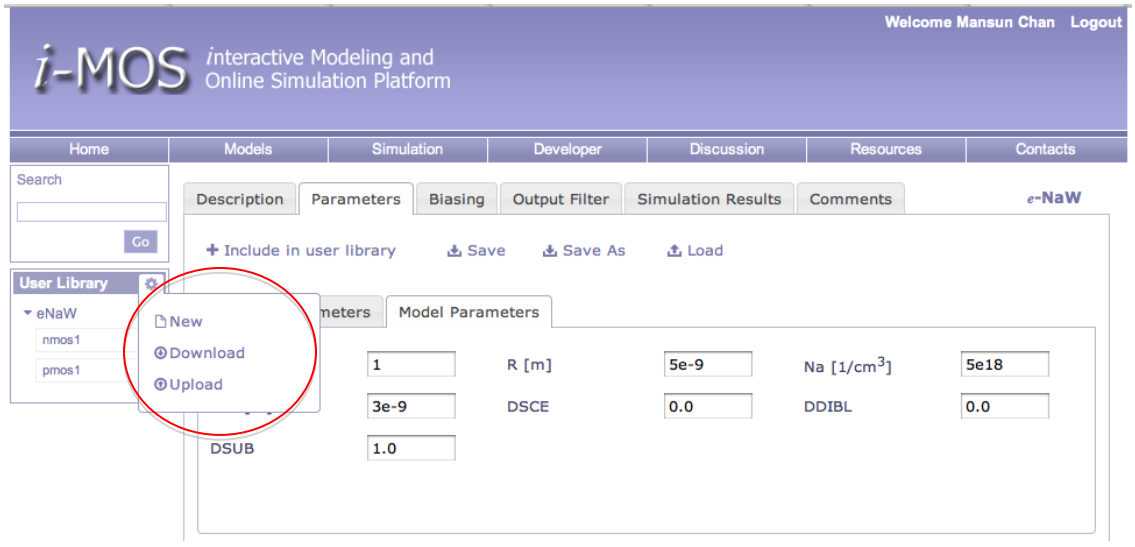

Fig. 3.2: Some simple functions available to manage the user library.

Chapter

4

Circuit Simulation

i-MOS platform has included the NGSpice [1] open source simulation engine to provide the simulation capability. New devices implemented on the *i*-MOS platform together with all other devices already available in NGSpice can be connected to form a circuit for simulation. There are three steps to perform circuit simulation: (1) user library construction; (2) submitting the netlist to the simulator; and (3) viewing the returned data

4.1 Constructing the User Library

In order to perform simulation using new devices in *i*-MOS, user can directly type in the model parameters in text mode following the format supplied by the model and the NGSpice text input interface. To simplify the work, you may fine-tune the parameters of a model and save it to the user library as described in Chapter 3. You may include as many devices and models as you like into the user library and all the devices in the model library will be automatically translated into the text input to the NGSpice when you carry out the simulation.

4.2 Entering Netlist

In order to perform circuit simulation, a netlist has to be constructed. Currently, *i*-MOS provides a simple text interface for user to enter the netlist. There are two different modes to enter the netlist, namely the Netlist mode and the Raw Input mode.

4.2.1 The Netlist Mode

The Netlist mode provides a simple user interface for user to construct the netlist while leaving the model details to be constructed from the user library. The netlist page consists of 4 different sections as shown in Fig. 4.1 that need to be completed before simulation can be executed.

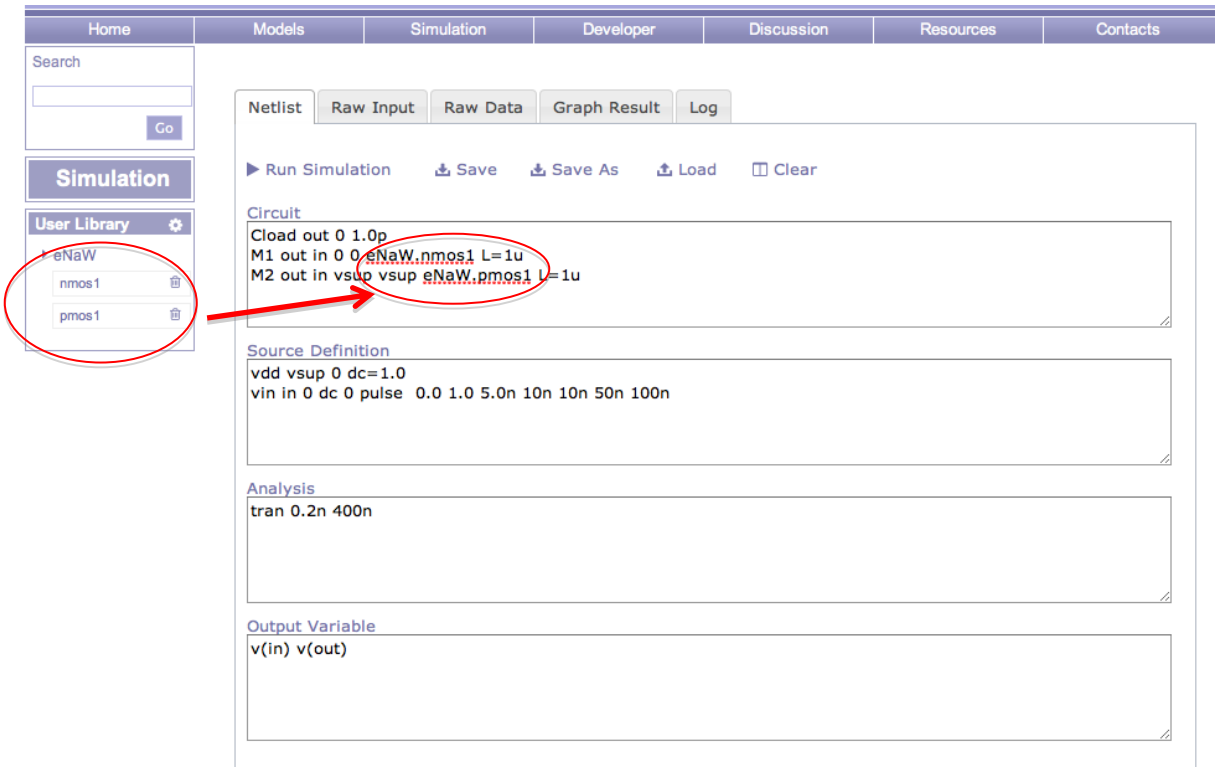

Fig. 4.1: The netlist page showing the 4 sections: (1) Circuit; (2) Source definition; (3) Analysis; and (4) Output variables

The circuit section is where you enter the netlist following the format as specified in the NGSpice manual. Each instance of different device available through *i*-MOS requires specific naming convention and format, which is described in Appendix A. The model used for a particular instance is specified using the format of <model name>.<parameter set name>. For example, in Fig. 4.1, we are using the eNaW model with two sets of parameters nmos1 and pmos1. To use the nmos1 model parameters, you can instantiate the eNaW.nmos1. In addition, all the instance parameters should be included here.

The Source Definition section allows you to enter the different power supply applied to the circuit. The sources can be fixed DC sources, a ramped DC sources, time dependent sources or AC sources.

In the Analysis section, you define the type of simulation you are going to perform. The available type of simulations depends on the model implementation. For example, if an active device model only has I-V equations implemented, it is normally only accurate for DC simulation and will not be able to produce the accurate delay behavior in TRAN simulation.

The Output Variable section defines the output data to be collected and plotted. Putting more than one variable on the same line will include them in the same table and plot on the same graph. Putting variables on different lines will place them on separate table and separate graphs.

When you are done with editing, you may submit the job to the simulator using the Run Simulation function on the top and view the result in the Raw Data page or Graph Result page to be described later.

If you are going to stop editing your input for some time and return later, you may save the netlist in the netlist mode using the save function on the page. A file with an .isp extension will be saved in the default location set by your browser.

4.2.2 The Raw Input Mode

For more experienced users or if you already have a NGSpice input deck that you want to submit to *i*-MOS directly, you may use the Raw Input mode as shown in Fig. 4.2.

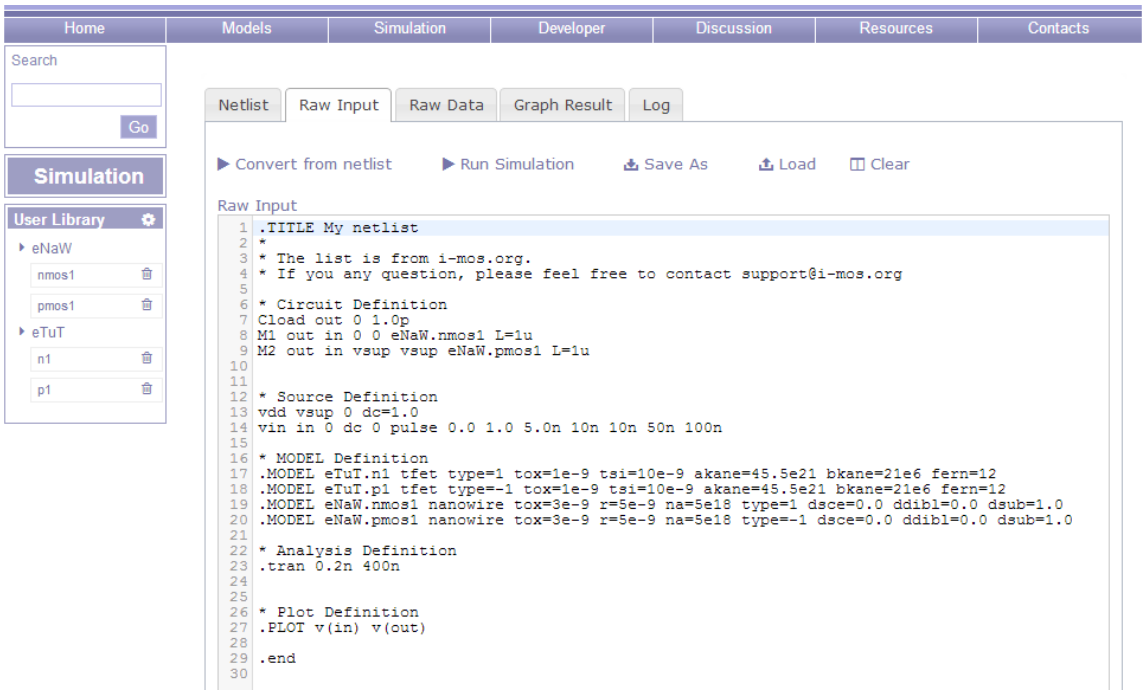

Fig. 4.2: The Raw Input mode to be submitted directly to the circuit simulator

You may type in the raw input from a blank page after clicking the clear option on top, or you may load an existing file (must have an extension of .sp) and edit it. In addition, you may convert the netlist you entered in the Netlist mode to obtain the raw input by using the 'Convert from netlist' option. Both the netlist and devices in the user library will be translated to the raw input window for you to edit. When you are working in the Raw Input mode, the user library will not be used.

When editing the raw input you may use the keyboard hotkey ALT+T to enable or disable a menu bar to assist.

When you are done with editing, you may submit the job to the simulator using the Run Simulation function on the top and view the result in the Raw Data page or Graph Result page to be described later.

If you are going to stop editing your input for some time and return later, you may save the raw input data using the 'Save As' function on the page. A file with a .sp extension will be saved in the default location set by your browser.

4.2.3 Running Simulation

After you select the Run Simulation option, the input file will be sent to the simulator. If there are errors, the error messages will be displayed in the Log page where you can check to correct your input. If you find the simulation is running for a long time without response, you may also abort the simulation by pressing the Abort option. If everything runs smoothly, the output will be ready as raw data or graphical output.

4.3 Obtaining Results

4.3.1 Viewing Raw Data

When simulation is completed, the numeric data is listed in the Raw Data page as shown in Fig. 4.3. The first column is the running variable depends on the type of simulation you are performing. Some examples are shown in table 4.1:

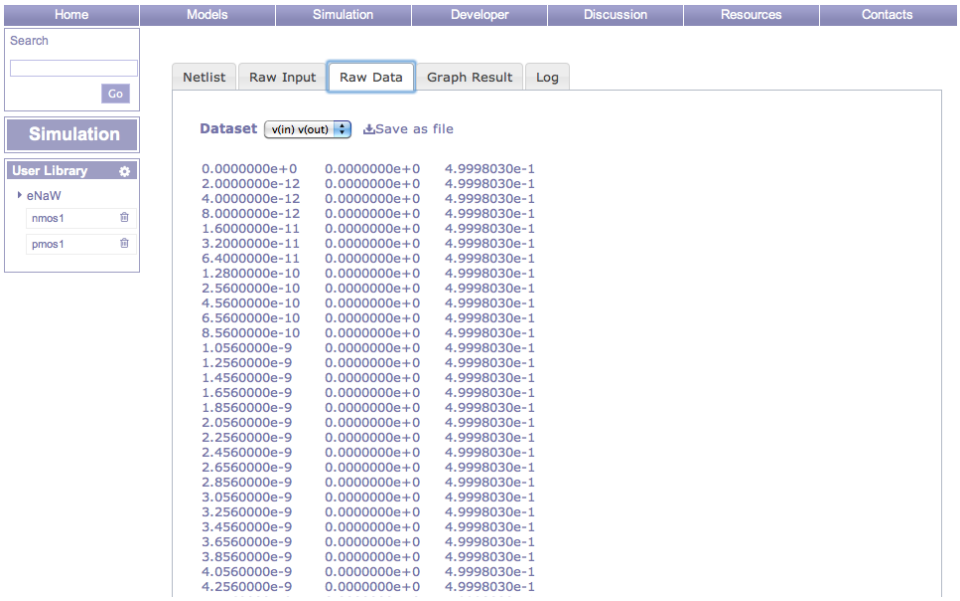

Fig. 4.3: The Raw Data output. Different variables can be selected from the Dataset drop down menu

Type of Simulation	First Column
.DC	running source voltage
.TRAN	time
.AC	frequency

Table 4.1: Type of simulation and the first column of the raw data output

Other output variables are listed on the second column onward. If you list the output variables on the same line in the netlist input, they will be displayed in the same table. If they are listed in different lines, they will be displayed in different tables to be selected with the Dataset drop down menu. You may download the numerical data from the simulation to default location of your computer as defined by the browser and plot them using other graph-plotting utilities of your preference.

4.3.3 Viewing Graphical Data

i-MOS provides a simple graphical interface for you to view the data. To see the graphical output, select the Graph Result tag on top, and the raw data will be displayed as shown in Fig. 4.4

Fig. 4.4: Graphical output of simulation results

The y-axis of the plot displays the output variables without explicit labels. As a reminder, the legends tell you what the corresponding units should be. When you mix variables of different units together (by putting them in the same line as done in Fig.4.1), you will see they are plotted in the same graph. Different data set (or tables in the Raw

Data) can be selected using the Graph drop down menu. You may also plot the y-axis in log scale if it is desired. By clicking on the graph, you may magnify the figure. Further more, you may save the graph in .png format if you want to save it for recording purpose.

4.4 The Log File

The log file tag keeps track of all messages that the simulator returns during simulation. If there is any problem with the simulation, you should check the log file to make sure all the problem listed are solved before re-submitting the next simulation.

Chapter

5

Reading and Posting Messages

In addition to providing technical service, *i*-MOS also hopes to serve as a place for the modeling community to socialize, exchange idea and share information. To achieve such goal, *i*-MOS provides two different services: discussion blog and information posting. Both pages are lightly moderated.

5.1 The Discussion Page

The discussion page is where users can post and response to articles. The appearance of the page is shown in Fig. 5.1.

Fig. 5.1: The layout of the discussion page

In order to post a message, you have to be a registered user. After you login, you may post any messages. You may respond to a post by writing the comment section.

For each model added to *i*-MOS, a discussion page for the model is created and users can discuss any issues related to that model on the corresponding page.

A discussion page called Trouble Shootings is dedicated to provide solutions for the reported problems on *i*-MOS.

5.2 The Resource Page

The resources page provides a one-stop entry point for compact model and circuit simulation related information. It includes recent news on modeling activities, upcoming events of interest to the community, newly released articles, compact modeling related organization, device models, available tools, etc. This page will be operated in a user driven mode and monitored by the system administrator of *i*-MOS. Besides that, users are encouraged to post and advertise on their related activities. To post a message, you may just click on the post option in relevant section and follow the instruction. The message will then be submitted to the page master for a review. In general, the message will be posted the next day after submission.

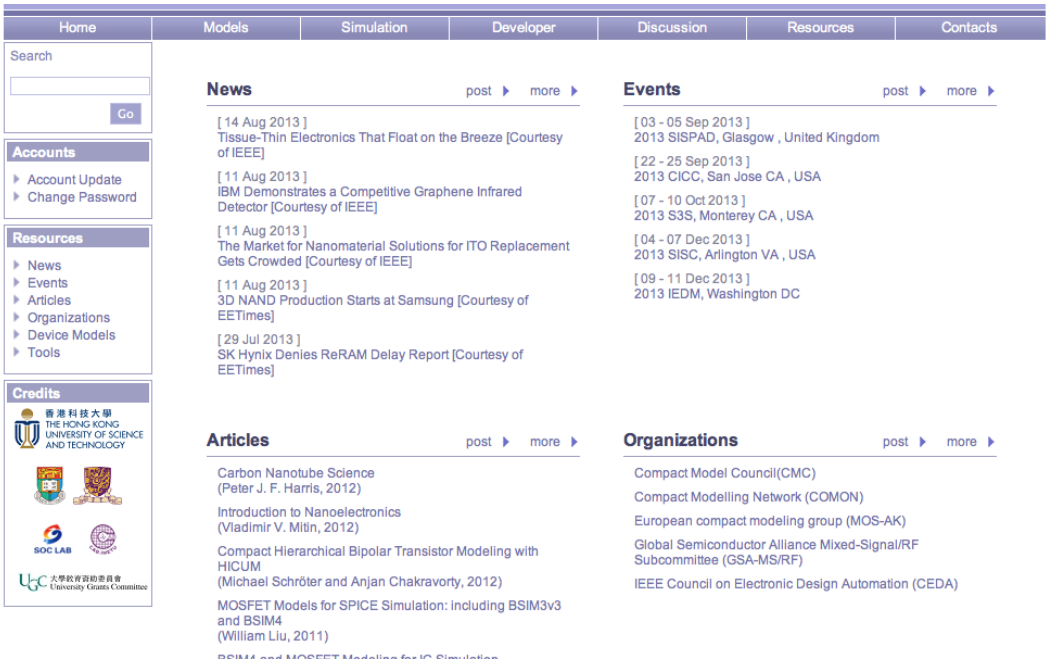

Fig. 5.2: The information provided by the resources page

5.3. Light Moderation Policy

User messages and comments are lightly moderated by the system administrator of *i*-MOS. If a message or a comment is found to be irrelevant or containing offensive material, it may be removed from the page. The account of repeated offenders may be suspended. Other than that, user exchange is as transparent as possible.

Chapter

6

Publishing Models

As a future extension, a model-authoring tool is planned to be included in the *i*-MOS platform. However, it is not available in the current release yet and the models have to be coded manually before the tool is developed. If you want to include your model to the *i*-MOS platform, please contact the *i*-MOS team directly and we shall work with you to include your model on *i*-MOS.

Appendix

A

Available Models

A.1 Double Gate MOSFET Model (*e*-DouG)

A.1.1 Instance syntax

```
Mxxx nd ng ns nb mname <l=val> <w=val>
```

It starts with the letter '`M`'. `nb` is a dummy node which can be connected to any voltage source, or ground. Formats of `mname` are described in 4.2.1.

A.1.2 Model parameters

***e*-DouG model**			
Parameter	**Meaning**	**Default**	**Unit**
L	Channel length	1.0e-7	m
W	Channel width	1.0e-6	m
Type	Device type 1 for n, -1 for p	1	unitless
Tox	Oxide thickness	1.2e-9	m
Tsi	Body thickness	1.0e-8	m
u0	Carrier mobility	400	$cm^2/V/s$

A.2 Nanowire MOSFET Model (*e*-NaW)

A.2.1 Instance syntax

```
Mxxx nd ng ns nb mname <l=val>
```

It starts with the letter '`M`'. `nb` is a dummy node which can be connected to any voltage source, or ground. Formats of `mname` are described in 4.2.1.

A.2.2 Model parameters

***e*-NaW model**			
Parameter	**Meaning**	**Default**	**Unit**
L	Channel length	4.0e-7	m
Type	Device type 1 for n, -1 for p	1	unitless
R	Channel radius	5.0e-9	m
Na	Doping density	5.0e18	$1/cm^3$
Tox	Oxide thickness	3.0e-9	m
DSCE	SCE coefficient	0 (>=0)	unitless
DDIBL	DIBL coefficient	0 (>=0)	unitless
DSUB	Exponent coefficient	1.0 (>=1.0)	unitless

A.3 HEMT Model (*e*-HEMT)

A.3.1 Instance syntax

```
Zxxx_3 nd ng ns mname <l=val> <w=val>
```

It starts with the letter '`Z`' and ends with '`_3`'. Formats of `mname` are described in 4.2.1.

A.3.2 Model parameters

***e*-HEMT model**			
Parameter	**Meaning**	**Default**	**Unit**
L	Channel length	1.0e-6	m
W	Channel width	1.0e-3	m
DD	Doped AlGaN thickness	1.5e-8	m
DI	Spacer AlGaN thickness	2.0e-9	m
Nd	AlGaN doping	4.0e24	$1/m^3$
U0	Channel mobility	0.1	$m^2/V/s$
P1	1^{st}-order mobility degradation coefficient	0	m/V
P2	2^{nd}-order mobility degradation coefficient	0	$(m/V)^2$
AX	Lateral mobility degradation coefficient	2	m/V
ESAT	Saturation field	1.8e7	V/m
PP0	Saturation length coefficient	2	unitless
AlXX	x in Al(x)Ga(1-x)N	0.15	unitless

A.4 Arizona State Carbon Nanotube Transistor Model (PTM_CNT)

A.4.1 Instance syntax

```
Zxxx_4 nd ng ns nb mname <types=val> <angle=val>
<tins=val> <eins=val> <tback=val> <eback=val>
<diameter=val> <L=val>
```

It starts with the letter '`Z`' and ends with '`_4`'. Formats of `mname` are described in 4.2.1.

A.4.2 Model parameters

PTM_CNT model			
Parameter	**Meaning**	**Default**	**Unit**
Types	Tube type 1 for n, -1 for p	1	unitless
Angle	Chiral angle	7.5	deg
Tins	Insulator thickness	1.0e-8	m
Eins	Dielectric of insulator	16	unitless
Tback	Back gate insulator thickness	1.3e-7	m
Eback	Back gate dielectric of insulator	3.9	unitless
Diameter	Tube diameter	5.0e-10	m
L	Gate length	1.15e-7	m
phisb	Barrier height	0.0	eV
rs	Source resistance	0	Ohm
rd	Drain resistance	0	Ohm
beta	Drain coupling	20	unitless
cc	Coupling capacitance	7e-12	
mob	Mobility	1.0	$cm^2/V/s$
Csubfit	Vfb adjustment	1.0	

A.5 Stanford Carbon Nanotube Model (SNCNFET)

A.5.1 Instance syntax

```
Mxxx nd ng ns nb mname <Lg=val> <Lss=val> <Ldd=val>
```

It starts with the letter '`M`'. Formats of `mname` are described in 4.2.1.

A.5.2 Model parameters

SNCNFET model			
Parameter	**Meaning**	**Default**	**Unit**
Lg	Gate length	32e-9	m
Lss	Source length	32e-9	m
Ldd	Drain length	32e-9	m
Efi	Fermi level of source/drain	0.6	eV
Kgate	Gate dielectric constant	16	unitless
Tox	Gate oxide thickness	4e-9	m
Vfbn	Flat band voltage	0	eV
n1	Chirality vector 1	19	unitless
N2	Chirality vector 2	0	unitless

A.6 Silicon Tunneling FET Model (*e*-TuT)

A.6.1 Instance syntax

```
Mxxx nd ng ns nb mname <l=val> <w=val>
```

It starts with the letter '`M`'. `nb` is a dummy node which can be connected to any voltage source, or ground. Formats of `mname` are described in 4.2.1.

A.6.2 Model parameters

e-TuT model			
Parameter	**Meaning**	**Default**	**Unit**
W	Gate width	1.0e-6	m
L	Gate length	50e-9	m
Type	Device type 1 for n, -1 for p	1	unitless
Tox	Oxide thickness	1e-9	m
Tsi	Film thickness	10e-9	m
Akane	A in Kane's model	45.5e21	$1/cm/s/V^{1.5}$
Bkane	B in Kane's model	21e6	$1/cm/V^{0.5}$
Fern	Kane's model correction factor	12	unitless
WFG	Gate work function	4.2	eV

A.7 Symmetric Doped Double-Gate MOSFET model (SDDGM)

A.7.1 Instance syntax

```
Mxxx nd ng ns nb mname <l=val> <w=val>
```

It starts with the letter '`M`'. `nb` is a dummy node which can be connected to any voltage source, or ground. Formats of `mname` are described in 4.2.1.

A.7.2 Model parameters

SDDGM model			
Parameter	**Meaning**	**Default**	**Unit**
W	Chennl width	1.0e-4	cm
L	Channel length	22e-7	cm
Type	Type	1	unitless
Tox	Oxide thickness	1.1e-7	cm
Ts	Channel thickness	10e-7	cm
Na	Channel doping	1e15	$1/cm^3$
m0	Mobility	1300	$cm^2/V/s$
E1	Critical electric field for phonon scattering	50e6	V/cm
E2	Critical electric field for surface roughness	50e6	V/cm
E2V	Parameter	1.02	unitless
P1	Exponent1	0.25	unitless
P2	Exponent2	1.7	unitless
Vsat	Velocity saturation	1.0e7	cm/s
LAMBDA	Channel shortening parameter	0.01	unitless

A.8 Bulk MOSFET model (BSIM3, Version 3.1)

A.8.1 Instance syntax

```
Mxxx nd ng ns nb mname <l=val> <w=val>
```

It starts with the letter '`M`'. Formats of `mname` are described in 4.2.1.

A.8.2 Model parameters

Please refer to the BSIM3 manual for detailed descriptions of the model parameters.

References

[1] L. Zhang, J. Zhang, Y. Song, X. Lin, J. He, and M. Chan, "Charge-based model for symmetric double-gate MOSFETs with inclusion of channel doping effect," Microelectronics Reliability, vol. 50, pp. 1062-1070, 2010.

[2] F. Liu, J. He, L. Zhang, J. Zhang, J. Hu, C. Ma, and M. Chan, "A Charge-Based Model for Long-Channel Cylindrical Surrounding-Gate MOSFETs From Intrinsic Channel to Heavily Doped Body," IEEE Transactions on Electron Devices, vol. 55, pp. 2187-2194, 2008.

[3] X. Cheng, Y. Wang, "A Surface-Potential-Based Compact Model for AlGaN/GaN MODFETs," IEEE Transactions on Electron Devices, vol. 58, pp. 448-454, 2011.

[4] S. Sinha, A. Balijepalli, and C. Yu, "Compact Model of Carbon Nanotube Transistor and Interconnect," IEEE Transactions on Electron Devices, vol. 56, pp. 2232-2242, 2009.

[5] J. Deng and H.-S. P. Wong, "A Compact SPICE Model for Carbon-Nanotube Field-Effect Transistors Including Nonidealities and Its Application Part I: Model of the Intrinsic Channel Region," IEEE Transactions on Electron Devices, vol. 54, pp. 3186-3194, 2007.

[6] J. Deng, H.-S. P. Wong, "A Compact SPICE Model for Carbon-Nanotube Field-Effect Transistors Including Nonidealities and Its Application - Part II: Full Device Model and Circuit Performance Benchmarking," IEEE Trans. Electron Devices, vol. 54, pp. 3195-3205, 2007.

[7] J. Deng and H.-S. P. Wong, "Modeling and Analysis of Planar Gate Capacitance for 1-D FET with Multiple Cylindrical Conducting Channels", IEEE Transactions on Electron Devices, vol. 54, pp. 2377-2385, 2007.

[8] L. Zhang, X. Lin, J. He, M. Chan, "An Analytical Charge Model for Double-Gate Tunneling FETs," IEEE Trans. on Electron Devices, vol. 59, no. 12, pp. 3217-3223, Dec. 2012

[9] L. Zhang, J. He and M. Chan, "A Compact Model for Double-Gate Tunneling Field-Effect-Transistors and Its Implications on Circuit Behaviors", 2012 IEEE International Electron Device Metting (IEDM), Dec. 10-13, 2012, San Francisco, USA

[10] A. Cerdeira, O. Moldovan, B. Iñiguez and M. Estrada, "Modeling of potentials and threshold voltage for symmetric doped double-gate MOSFETs", Solid-State Electronics, 52 (2008) 830-837.

[11] Cerdeira A, Iñiguez B, Estrada M. "Compact model for short channel symmetric doped double-gate MOSFETs" Solid-State Electronics 2008; 52: 1064-1070

[12] J. Alvarado, B. Iñiguez, M. Estrada, D. Flandre and A. Cerdeira, "Implementation of the symmetric doped double-gate MOSFET model in Verilog-A for circuit simulation", Int. J. Numer. Model. 2010: 23: 88-106.

[13] A. Cerdeira, I. Garduño, J. Tinoco, R. Ritzenthaler, J. Franco, M. Togo, T. Chiarella, C. Claeys, "Charge based DC compact modeling of bulk FinFET transistor", Solid-State Electronics 87 (2013) 11-16.

[14] Ghader Darbandy, Thomas Gneiting, Heidrum Alius, Joaquín Alvarado, Antonio Cerdeira and Benjamín Iñiguez, "Automatic parameter extraction techniques in IC-CAP for a compact double gate MOSFET model", Semicond. Sci. Technol. 28 (2013) 055014 (1-8).

[15] http://www-device.eecs.berkeley.edu/bsim/?page=BSIM3