

MBP 2017

Scripting

Notices

© Keysight Technologies Incorporated, 2002-2017

1400 Fountaingrove Pkwy., Santa Rosa, CA 95403-1738, United States

All rights reserved.

No part of this documentation may be reproduced in any form or by any means (including electronic storage and retrieval or translation into a foreign language) without prior agreement and written consent from Keysight Technologies, Inc. as governed by United States and international copyright laws.

Restricted Rights Legend

If software is for use in the performance of a U.S. Government prime contract or subcontract, Software is delivered and licensed as "Commercial computer software" as defined in DFAR 252.227-7014 (June 1995), or as a "commercial item" as defined in FAR 2.101(a) or as "Restricted computer software" as defined in FAR 52.227-19 (June 1987) or any equivalent agency regulation or contract clause.

Use, duplication or disclosure of Software is subject to Keysight Technologies' standard commercial license terms, and non-DOD Departments and Agencies of the U.S. Government will receive no greater than Restricted Rights as defined in FAR 52.227-19(c)(1-2) (June 1987). U.S. Government users will receive no greater than Limited Rights as defined in FAR 52.227-14 (June 1987) or DFAR 252.227-7015 (b)(2) (November 1995), as applicable in any technical data.

Portions of this software are licensed by third parties including open source terms and conditions. For detail information on third party licenses, see [Notice](#).

Contents

MBP Scripting	5
MBP Scripting Overview	5
General Operations	5
Other Functions	7
Script Application	10
Application Samples	10
Commands	10
Flow	12
IMV	13
How to add a new IMV plot	14
Target definition	16
Plot definition	20
Statistical and Mismatch	21
DP	21
GUI-Based Model Extraction Flow	21
GUI-Based Model Extraction Flow	21
Menu Bar	25
Panels in MBP Main Window	25
Project Pane	25
Edit pane	26
Script Editor	27
Script Overview	28
Script Programming	29
Data Definition	29
Quick Start	29
Data Organization	30
Data as Table	30
Data as Tree	31
How to organize a table to a tree	31
Example of sparse vgs	33
Plot Definition	34
Data Transformation	34
POINT to POINT	34
TABLE to POINT	35
Optimization and other Operations	36
Example (Assuming that you have already defined a data example): ...	36
API for Model Parameter	37
Build Extraction Flow	37
Script Programming Overview	38
Script Editor	38
Function Definition	40

- MBP Scripting Overview
- Script Application
- Script Programming

MBP Scripting

- MBP Scripting Overview

MBP Scripting Overview

This section describes the features and settings of MBP scripts, which helps in defining and automating MBP operations with script programming.

General Operations

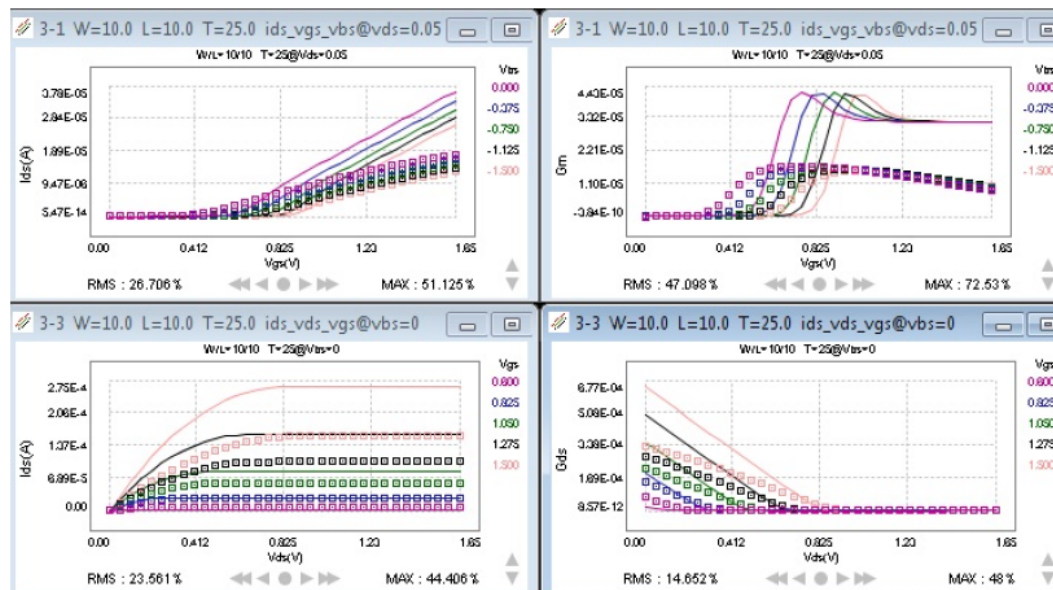
MBP adopts Jython-based script to allow customization of MBP operations. Jython is developed for Java users based on Python and it is an ideal script language for MBP. It does not require much compiling and programming knowledge. You can use this script language to define, customize, and automate all MBP operations.

Lets use an example to explain how to use script functions in MBP.

If you want to create a shortcut icon on the tool bar to access certain characteristics of the device quickly. Lets say the device is 10X10um and has four pages as shown in [Figure: Device characterization pages](#).

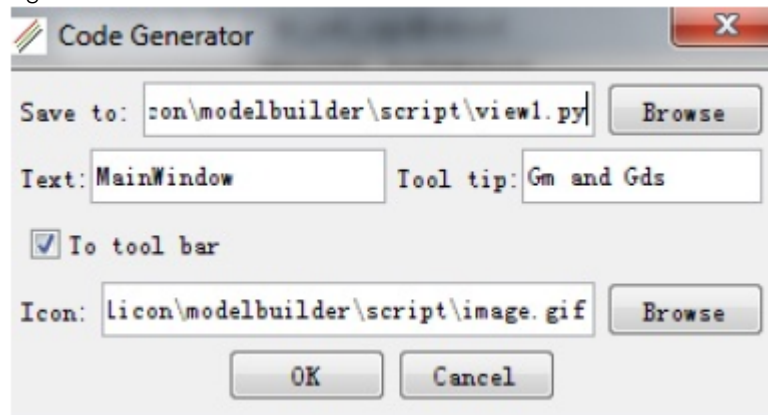
1. Ids_Vgs_Vbs@Vds=0.05
2. Gm_Vgs_Vbs@Vds=0.05
3. Ids_Vds_Vgs@Vbs=0
4. Gds_Vds_Vgs@Vbs=0

Device characterization pages

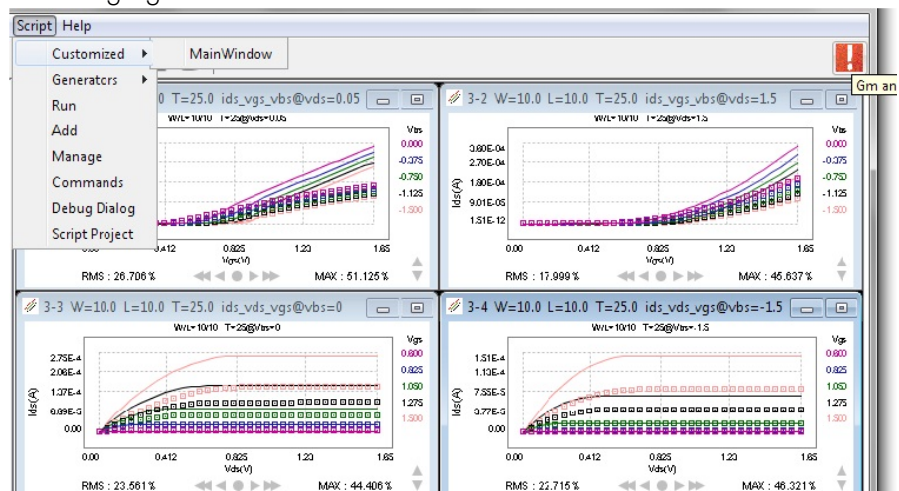



Now, create a shortcut on the tool bar by using MBP script functions, as shown below:

1. Select the device characteristics: select the 10X10 device from Device Navigator and show All Page.
2. Generate Script:
 - a. Start code generator by clicking Script > Generators > Graph from the main menu. The following window pops up as shown in following figure: Code Generator

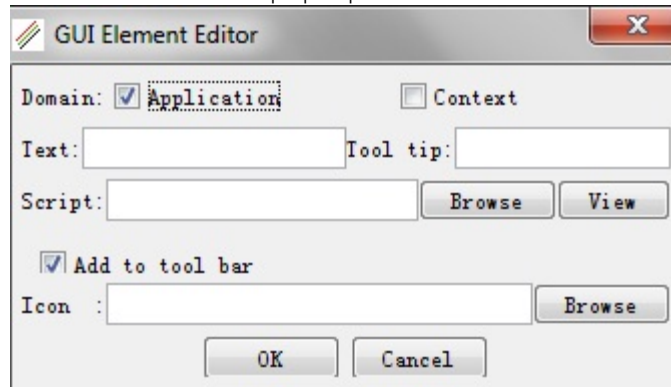


- Save to: The path and filename that you want to save the script to. You can also click Browse to make the selection.
- Text: The name for the script.
- Tool tip: You can add comments for the script here.
- To tool bar: Checking this selection enables creation of an icon on the tool bar.
- Icon: The shortcut icon for the generated script. To be mentioned here: if you choose To tool bar and does not assign an icon to it, MBP takes script name as the icon. After clicking OK, the script is generated and a shortcut is added to MBP menu as shown in following figure. Shortcut is added in the tool bar

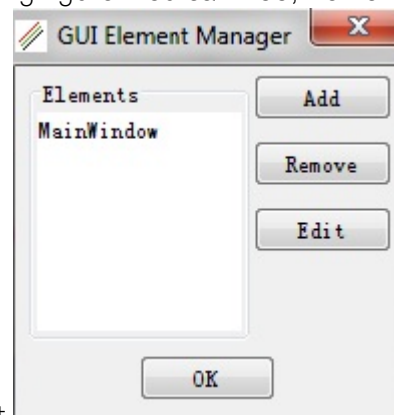


3. To run the script, you can either select from the Script > Customized list or click the shortcut icon  on the tool bar.

4. To add another shortcut by script on the tool bar, select Script > Add from the main menu. The pop-up window is shown in following figure. Add script



5. To manage the scripts, click Script > Manage from the main menu. The new window is displayed as shown in following figure. You can Add, Remove, or

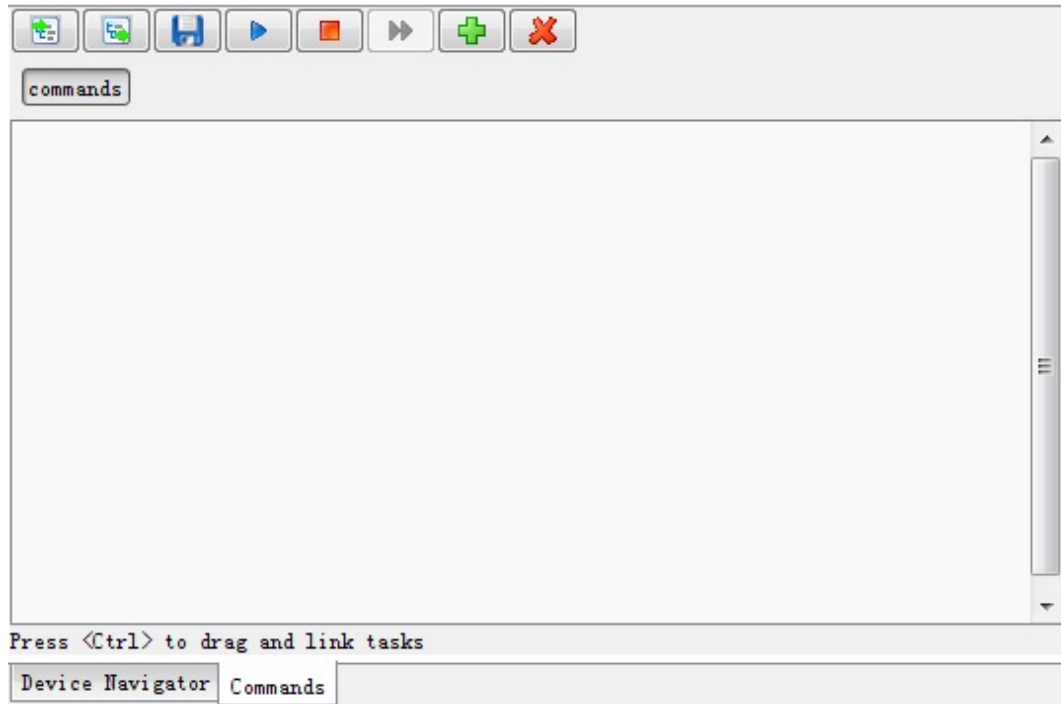


Edit the existing scripts. Manage Script

Other Functions

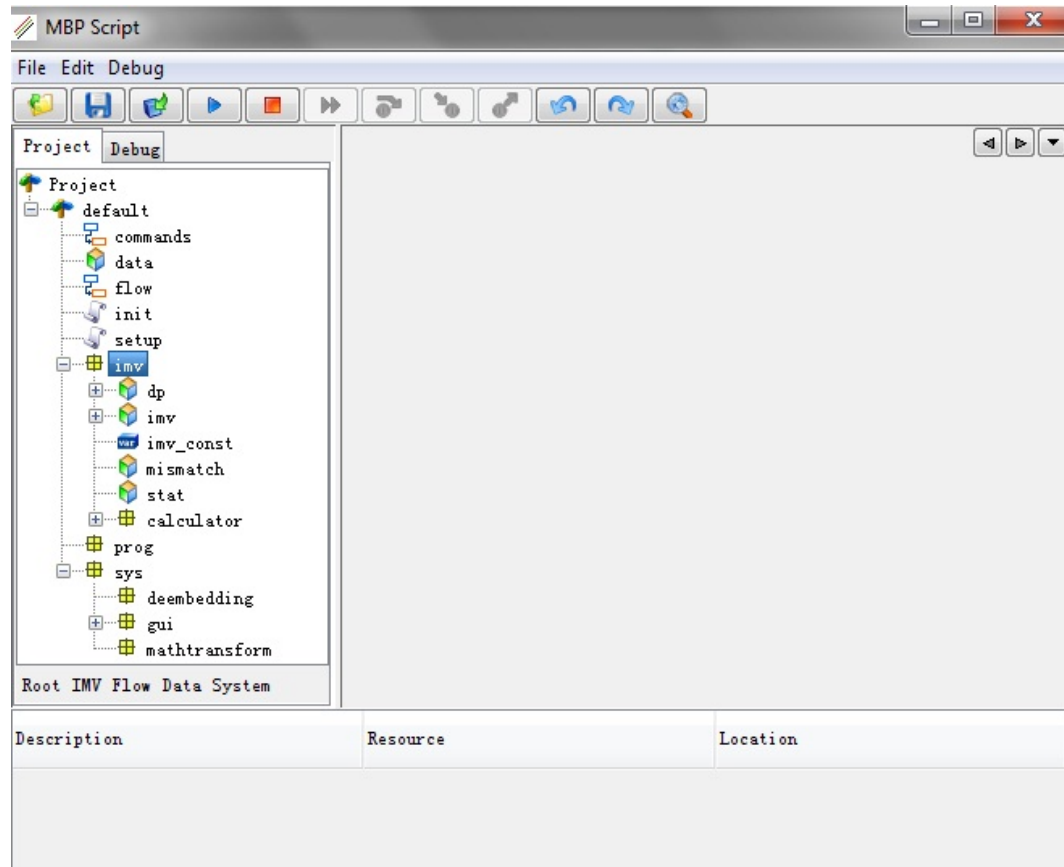
Click Script > Commands from the main menu to activate the commands window in the control pane, shown in following figure. Here you can import/export script project, run script or add/remove group, task, and branch.

Commands Window



Choose Script > Script Project from the main menu, the MBP Script window (as shown in following figure) is displayed.

Script Project



NOTE

For more details on MBP script language and the available built-in functions, you can refer to *MBP Script Application Manual* and *MBP Script Programming Manual*.

Script Application

- Application Samples
- GUI-Based Model Extraction Flow
- Menu Bar
- Panes in MBP Main Window
- Script Editor
- Script Overview

Application Samples

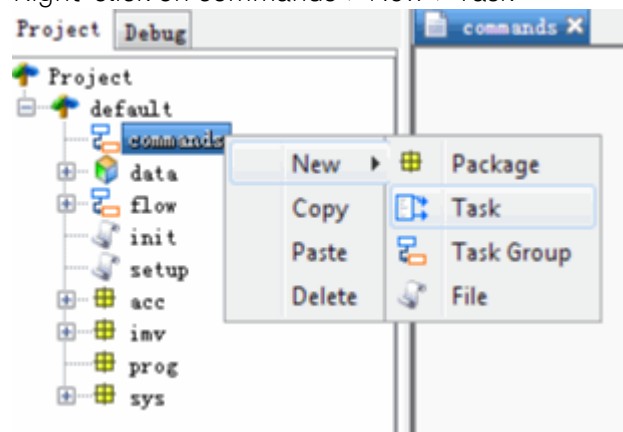
The most frequently used applications with MBP Script will be presented in this chapter.

Commands

Commands project is designed to select and organize MBP plots by scripting. It can help to standardize the extraction flow (together with parameter group function), or help to easily generate report (together with report function).

To create a new command file to show some selected plots:

1. Right-click on commands > New > Task



2. Input the task name and MBP will generate the template.

- If data is already loaded in MBP, you can define some output plots. Here is an example.

```

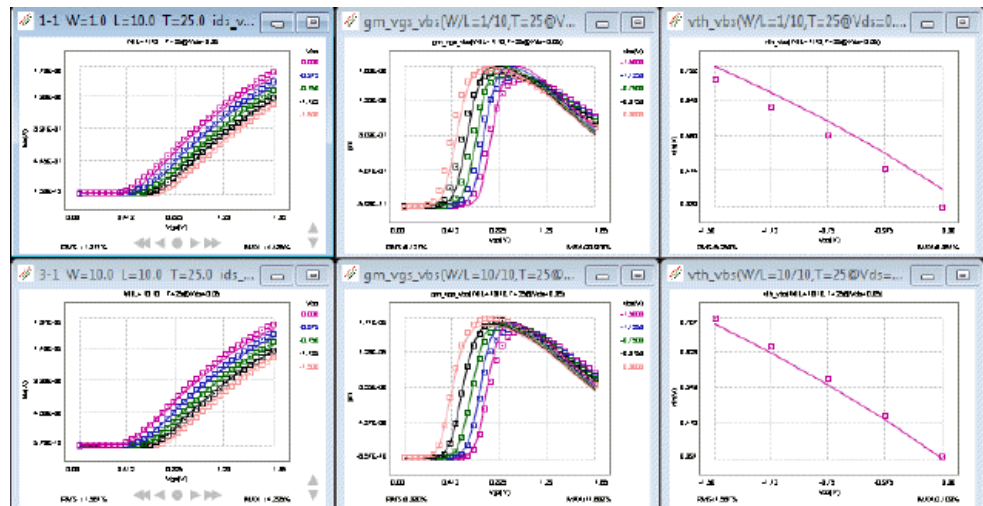
task() () {
    cmd.clearPlots();
    cmd.setPlotLayout(2,3);
    show(1.0u, 10.0u);
    show(10.0u, 10.0u);
}
TASK.start();

void show(double w, double l) {
    cmd.plot("ids_vgs_vbs@vds=0.05&&w="+w+"&&l="+l);
    cmd.plot("imv_inv.gm/gds.gm_vgs_vbs@vds=0.05&&w="+w+"&&l="+l);
    cmd.plot("imv_inv.vth.vth_vbs@vds=0.05&&w="+w+"&&l="+l);
}

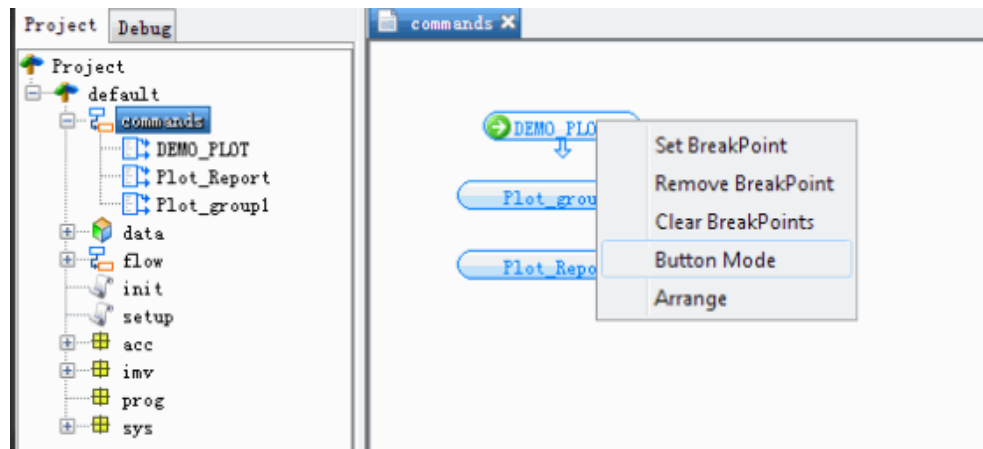
```

In this example, a function called show was defined, and plots (ids_vgs_vbs, IMV plot gm_vgs_vbs, and vth_vbs) were selected in the function. The plots were selected by placing several restrictions with logic expressions &&. The task changes the MBP layout to 2 by 3 plots, and then shows two devices with pre-defined plot selections.

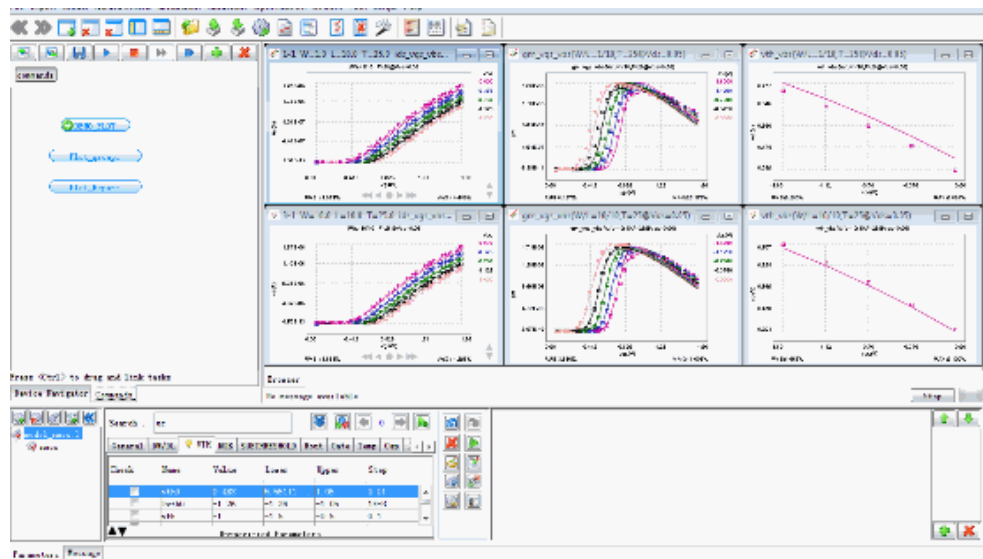
- Click Run. The output appears in the MBP main GUI.



- Double-click Commands, and drag the existing task to the Edit pane. It is possible to create multiple tasks based on different applications. You can set the task to button mode by right-clicking the task. Under the button mode, the task icon performs like a button and runs with a single click.





- After all the scripting work, open the Commands tab in the MBP main GUI by *Script > Commands. It has the same function as the one in the script project window.



Flow

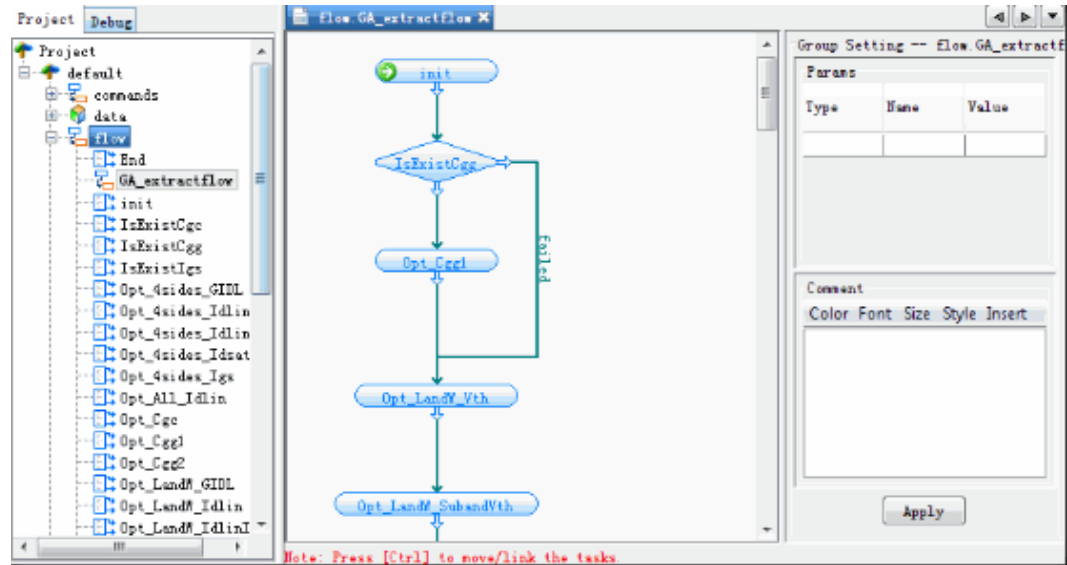
Model extraction flow can be found under the flow node.

You can create a new task  or a task group  by right-clicking flow > new.

Double-click the task flow name, and the graphic extraction flow is shown in the Edit pane. Dragging the task to the Edit pane will generate the task icon. Hold ctrl to move the task icon or link between two tasks.

To move the tasks or connections in the flow, highlight the task or connections and press Delete in keyboard.

To run the task, highlight the starting task and click Run.



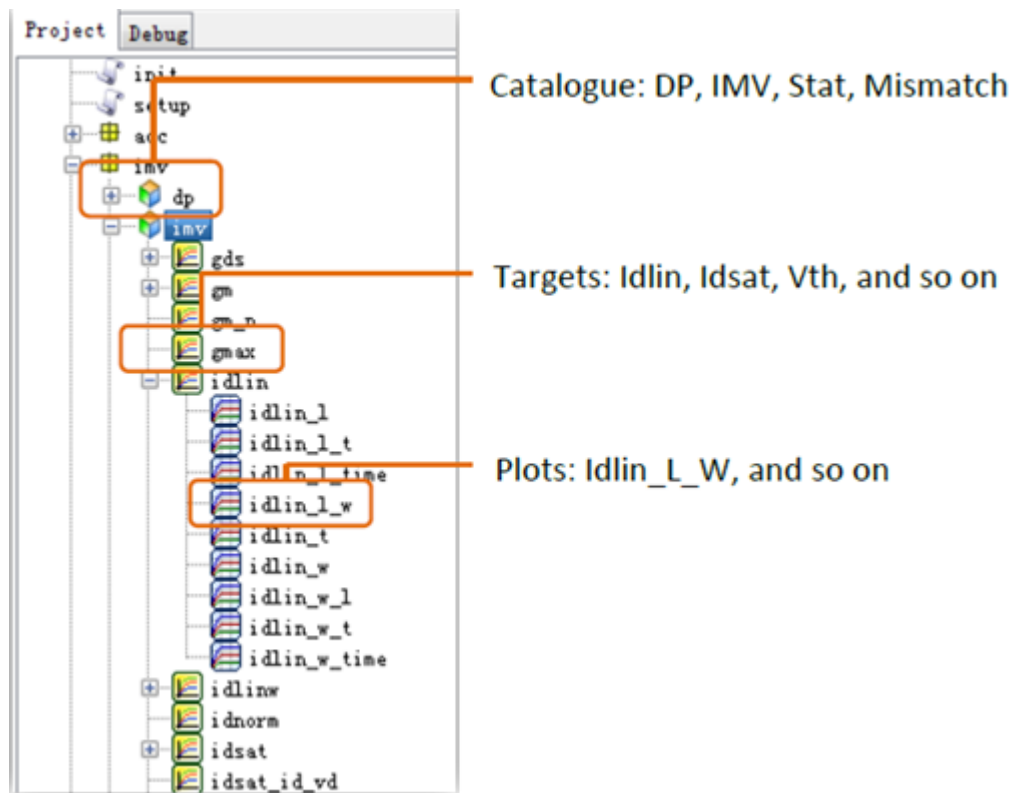
The extraction flow has hierarchical structure. You can create several task groups in one flow and make the flow clearer and more readable. The tasks need to be written by script. For more details, refer to the MBP Script Application Manual.

IMV

IMV includes four catalogs of configurations: DP, IMV, stat, and mismatch. Old MBP IMV and DP configurations are still compatible with the new versions. However, when loading them into MBP, the tool will convert them automatically to the new script format. Both the new and old versions of the IMV configuration can be saved or loaded separately with other script projects by Extraction > IMV > Save/Load IMV configurations.

NOTE

When saving the configurations, all old formats are converted to new script format.



The hierarchical structure of IMV is shown above. For each catalog, there are different targets defined by users (MBP provides default settings and the configurations are fully open to users.)

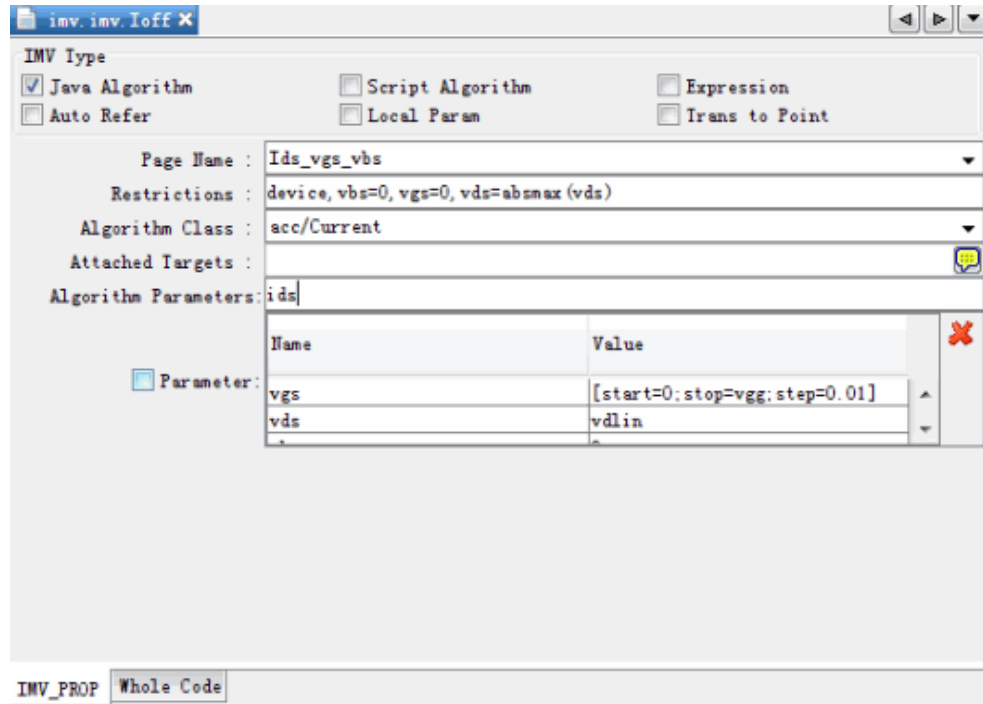
How to add a new IMV plot

In MBP, IMV stands for intermediate variable such as Vth, Idsat, etc. IMV plots show the scaling plots of IMV targets versus different instance parameters, such as Vth_L, Vth_T, and so on. In MBP script, IMV catalog is organized by different targets.

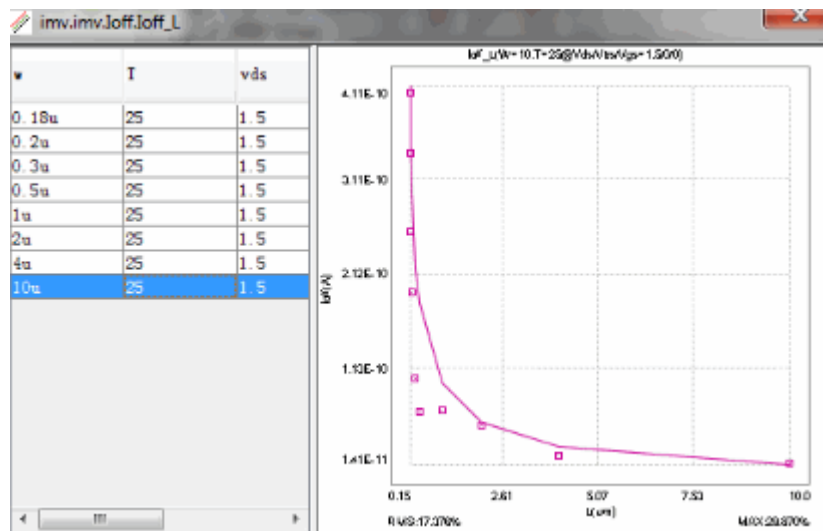
To show how to customize a new IMV plot with MBP script, here is an example of loff vs. L.

1. Right-click IMV > New > IMV.
2. Input the target name loff, and MBP generates a target definition template in Edit pane.

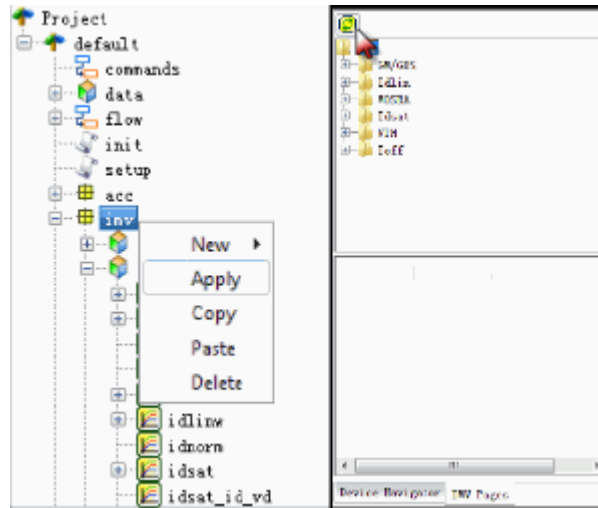
3. Modify the template to input all the related information.



4. Right-click the new created target loff and select New > Graph.
5. Enter the Graph title such as Ioff_L. MBP generates the graph setting template.
6. Select Simple Graph and enter Title, X, Y, and Hierarchy information.
7. Click View to check the graph.



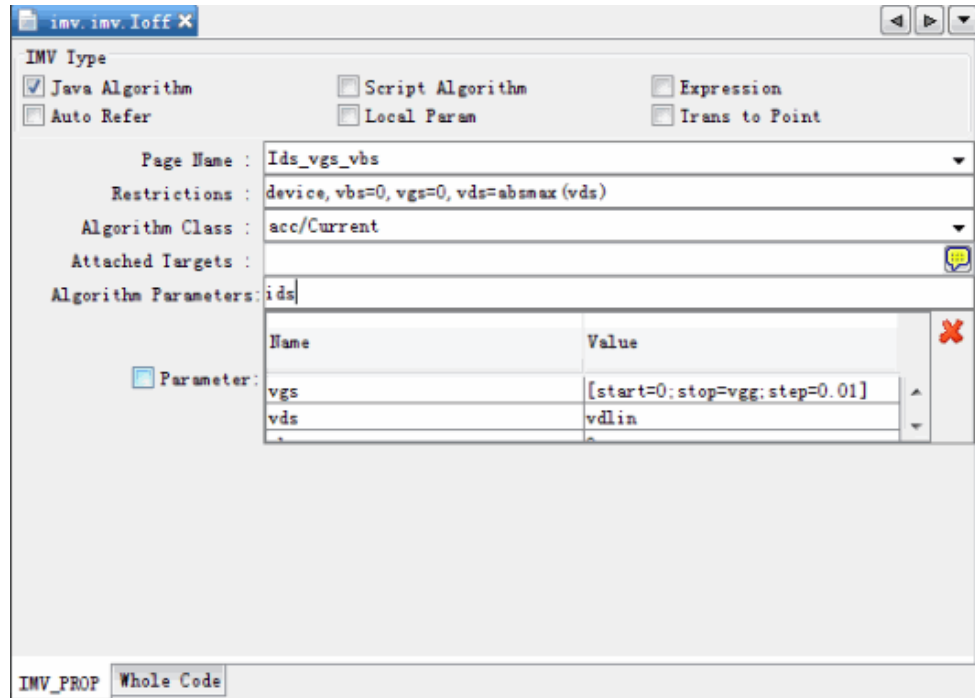
- To apply the new settings, right-click IMV and select Apply, or on the MBP main GUI > IMV tab, click on the refresh button.



Target definition

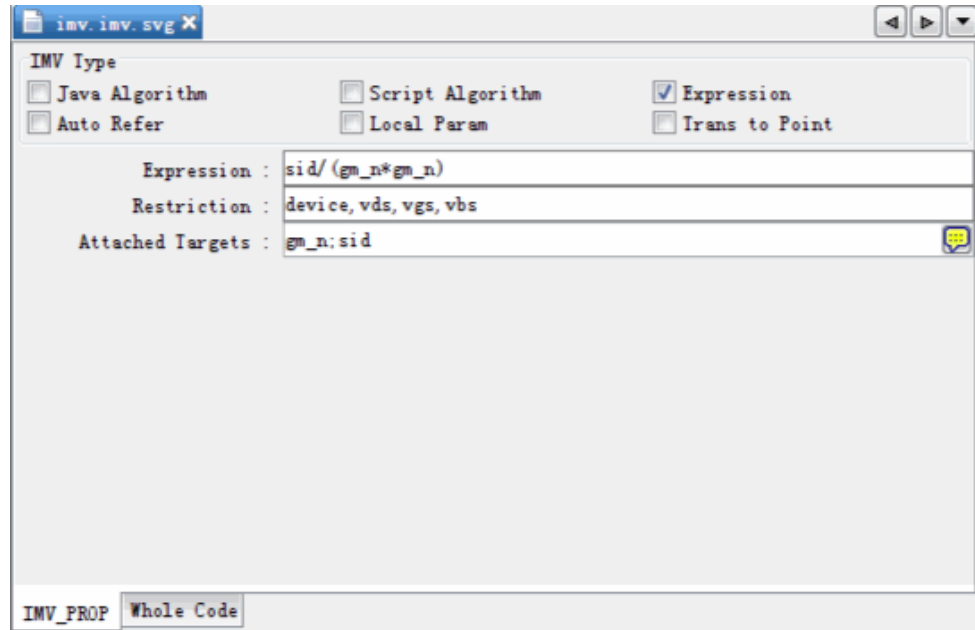
In the previous section, a new target loff was defined using the default target definition template. There are different methods to define a target in the MBP script project.

- Use built in algorithm (Check Java Algorithm)



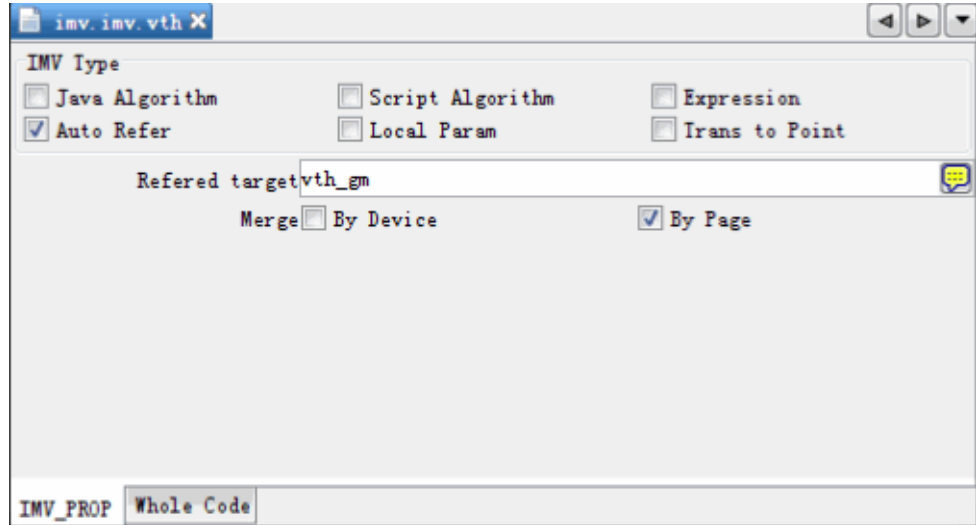
- Page Name: Plot to calculate the targets from. Page name should be matched strictly to the one shown in MBP device navigator.
- Restrictions: Restrictions of instance and bias conditions. Device stands for all the instance parameters and bias conditions can be defined here.
- Algorithm Class: MBP has built-in algorithm to use. For example, acc /Current gets the Y axis value from a specific page and bias, thus, this algorithm can be used to define all the targets like Idsat, Idlin, Ioff, Cgg, Sid, and so on.
- Attached Targets: It is possible to use the result from other targets by attaching the targets to calculate the current one.
- Algorithm parameters: It is tied to the algorithm.
- Parameter: Check to enable the parameter settings. For example, to set vgs sweep step for Vth_gm calculation (The step is the same as measured data by default).

- Use expression (Check Expression)



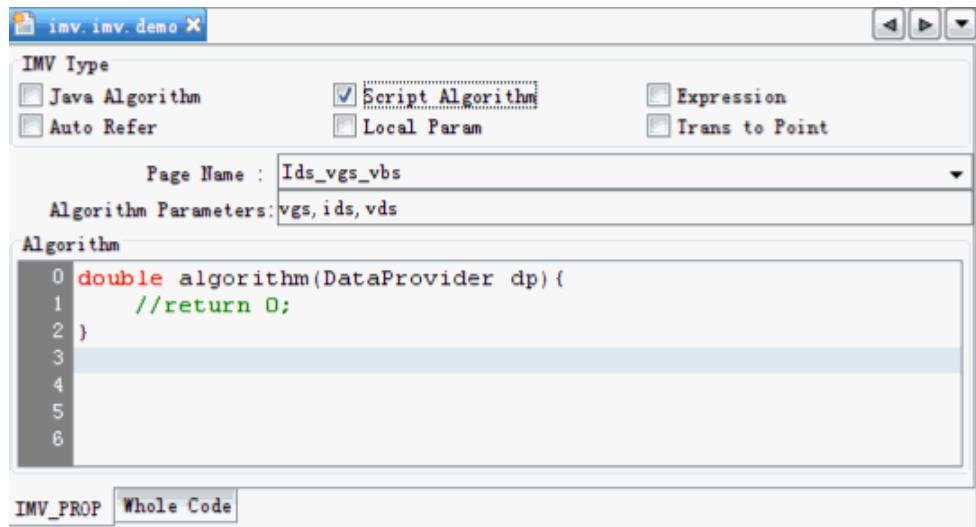
- Expression: Supports mathematic expressions between targets and constants.
- Restrictions: Restrictions of instance and bias conditions.
- Attached Targets: It is possible to use the result from other targets by attaching the targets to calculate the current one.

- Choose from existing targets (Check Auto Refer)



Some targets like Vth have different calculation methodologies such as vth_gm, vth_con, and so on. They do not need to repeat the same plot definition, but need to only define a new target Vth and refer to specific pre-defined Vth targets such as vth_gm, or vth_con. If you input more than one target in the Referred target text box, the tool will try to choose the first one. If the first target does not exist, it will auto refer to the second one, third one, and so on.

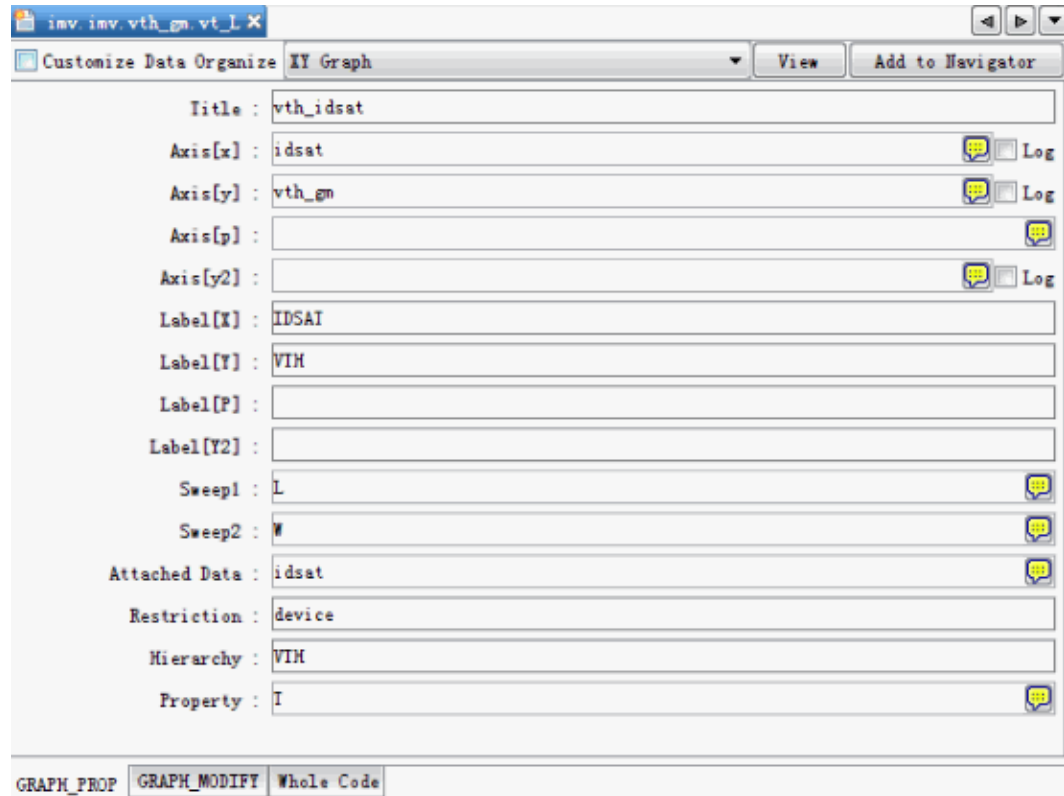
- Customize new algorithm (Check Script Algorithm)



To program your own algorithm using MBP script, select Script Algorithm. The new algorithm will be independent of MBP default Java Algorithm. For more details, refer to the MBP Script Programming Manual and contact Keysight support team (mbp_pdl-eesof@keysight.com).

Plot definition

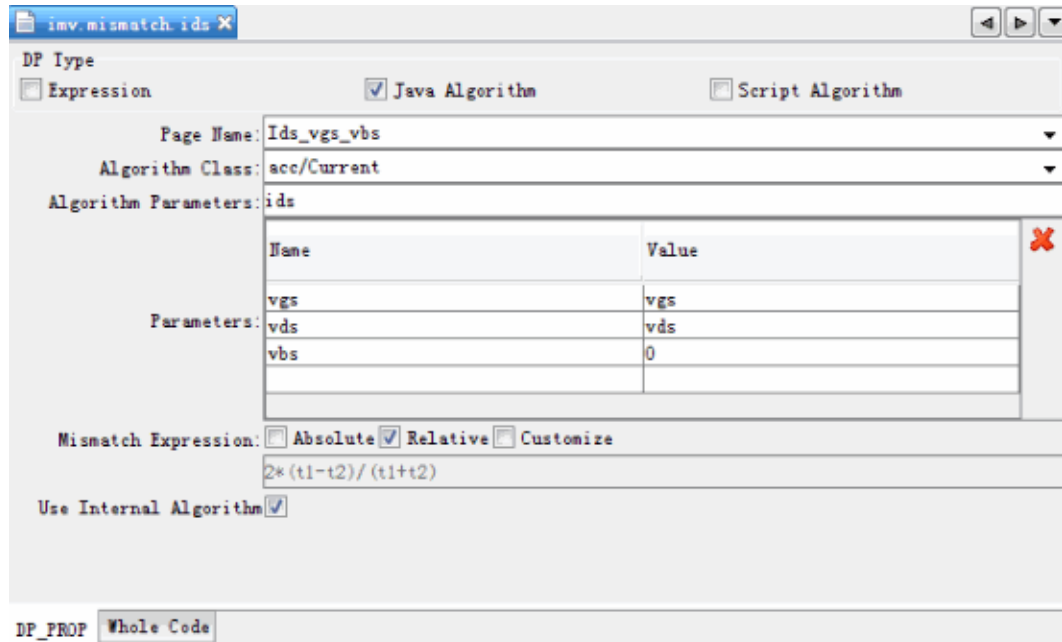
In the first section of this chapter, a new plot I_{off} vs. L was defined using a Simple Graph template. In this section, you will be able to define the plot using a more complex template named XY graph.



- Title: Plot title
- Axis[x]: X-axis variable, can be instance parameter or target
- Axis[y]: Y-axis variable, normally is target. MBP Script supports more than one target in one plot.
- Axis[p]: P axis variable, normally is instance
- Axis [y2]: Second Y axis variable, normally is target. The second Y value is put on the right side of plot.
- Label[x(y, p, y2)]: X(Y, P, Y2) axis label in the plot
- Sweep1 & Sweep2: Normally these two options are used for target vs. target plots. Different values of the instance parameters in Sweep1 and Sweep2 are put in the same plots, and the instance in Sweep1 is connected by curves.
- Attached Targets: Shows other targets in the plot by attaching the targets
- Restriction: Restriction of multiple targets
- Hierarchy: Hierarchy name shown in MBP GUI
- Property: Instance parameters shown in the title of plots

Statistical and Mismatch

MBP script supports user-defined targets. The target definition of statistical and mismatch is similar to IMV configuration. There is an additional option for both of them, Use Internal Algorithm. Uncheck this option to enable the setting in IMV, else, MBP will use internal algorithm for the targets. Compared to customized targets, the internal algorithm is faster when using MBP internal engine.



For mismatch, there is another option, Mismatch Expression. This option allows you to define the mismatch calculation methodology. By default, all the currents like Idsat, Idlin use relative method and vth uses absolute method. Check Customize to define a new mismatch method by using t1, t2, which stands for target value for a pair of devices.

DP

The DP configuration is also similar to IMV configurations. The only difference is DP data will get bias sweep conditions from DP settings instead of measurement data. Thus, the parameter sweep setting configuration is obligated and always enabled.

GUI-Based Model Extraction Flow

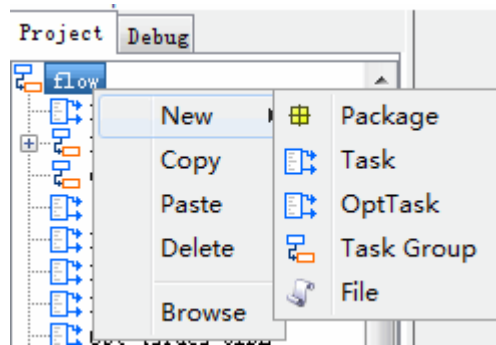
GUI-Based Model Extraction Flow

Overview

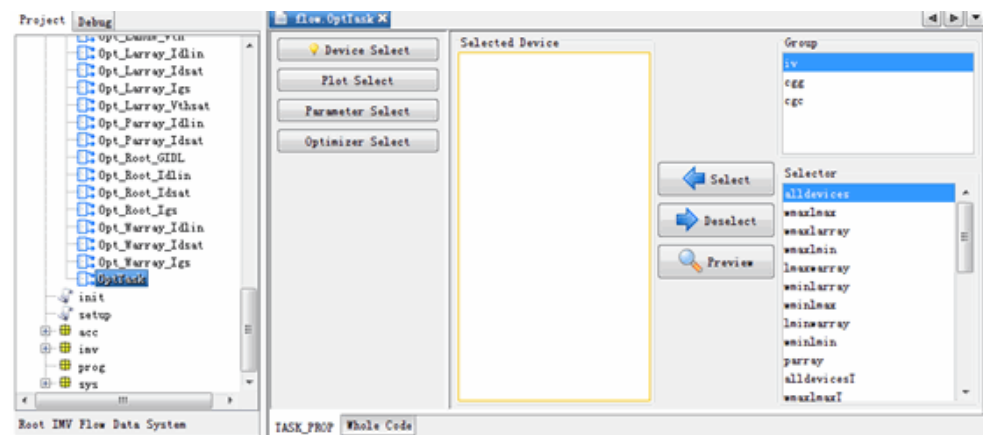
MBP script based extraction flow is a very flexible platform that allows the users to customize their own extraction flow freely. There are two types of task generation methods: OptTask which is GUI-based and Script Task which is based on pure script code. In the following chapters, we will focus on OptTask generation method.

1. Create New Task

To create a general task, please right click on the flow or Task Group and select OptTask. Then input the task name as required.



User-Interface of new task will show on the right panel.



2. OptTask Definition components

Each task is constructed by four different components:

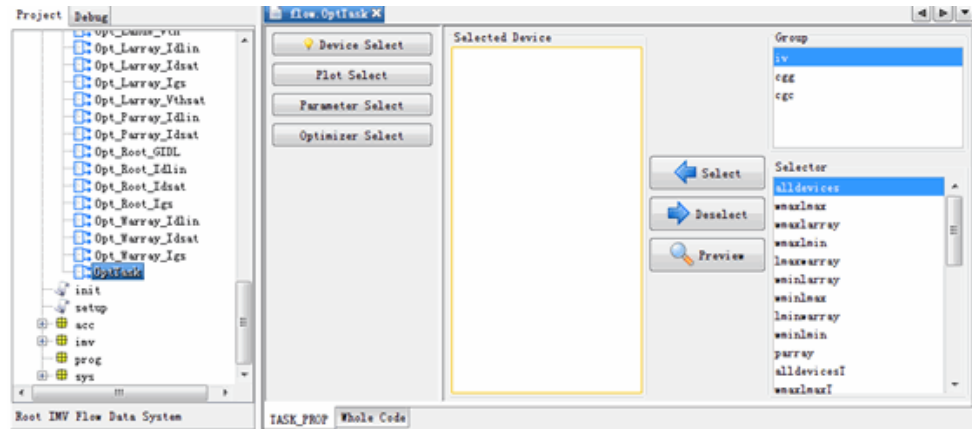
- Device Select
- Plot Select
- Parameter Select
- Optimizer Select

The purpose of these four components is to define and automate user's manual work during model parameter extraction.

- Device Select
Device Select UI will show when clicking on Device Select button. MBP uses the device selector to get device information. The selectors are firstly grouped by IV, Cgg and Cgs, and then for each group, there are pre-defined selectors, such as WmaxLmax. The setup in Device Select will be used in the Plot select.

The step to select a device selector is shown below.

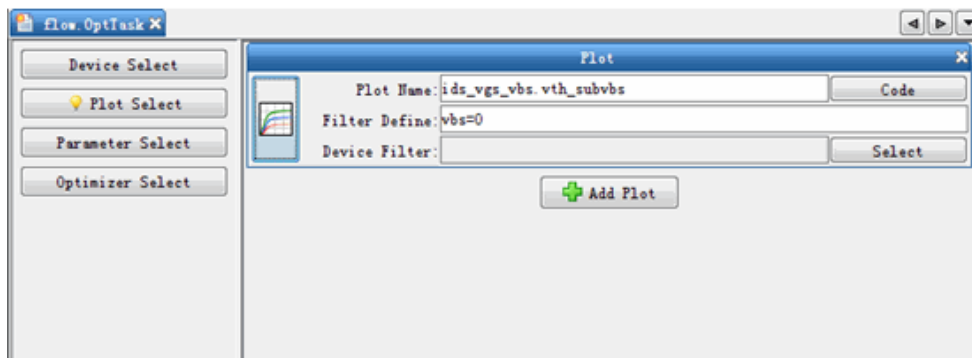
Firstly click on one of groups, and then select one of the selectors showed in the selectors list, at last click on Select Button. There is possible to select multiple selectors, and MBP will use all the devices to further filter with the region and plots.



To deselect a device selector, firstly click on the selector in the Selected Device list, and then click on Deselect Button. We could click on Preview button to view the selection in geometry plot. The selected devices will show by filled points.

- Plot Select


We could switch to Plot Select UI by clicking Plot Select button. MBP uses plot selectors to get plot and region information. This selection will be used to calculate the fitting error(RMS) during the optimization.

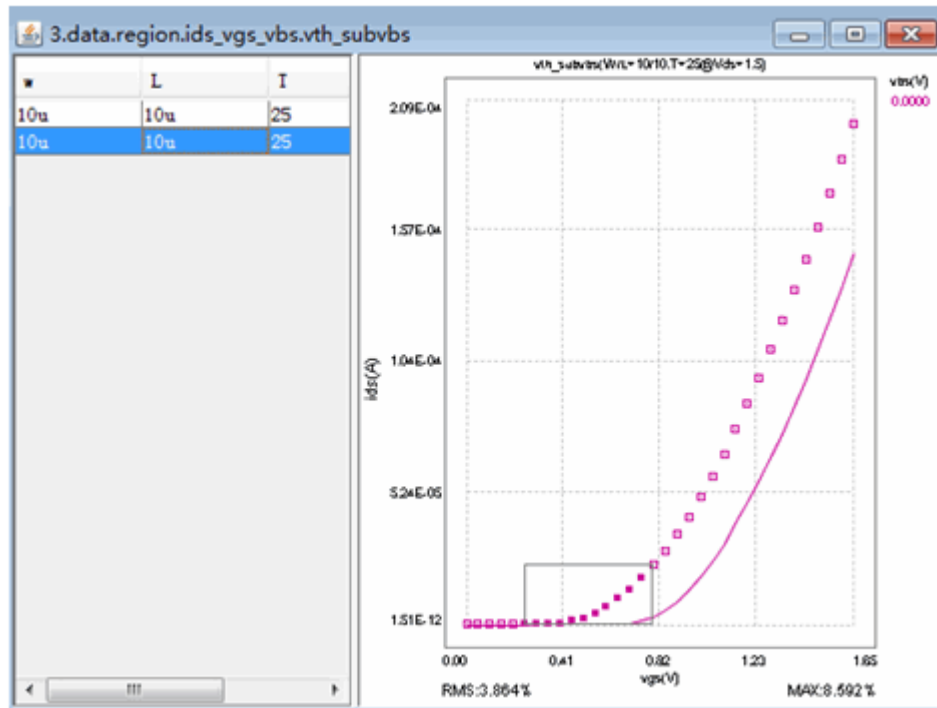


The pre-defined region is under Data > Region, and user could type in the plot name to input the selection. Click on button Code could switch to plot definition tab.

After the pre-defined plot selected, user could still add filter basing on the region, such as vbs=0. Logic expression can be used to add multiple filter, such as vbs=0&&vds=absmin(vds), vbs=0||vbs=-1.5, etc.

Additional device filter can be applied to different plot as well.

User could click on button  to preview the selection.



- Parameter Select

We could switch to Parameter Select UI by clicking Parameter Select button. User can define parameter to optimize by typing in parameter name then Enter or Click Add. Select parameter in the list and click Delete to remove the highlighted parameter.

When clicking on the parameter in the list, the corresponding parameter guide will show on the bottom. This parameter guide is built-in and also available to customize.

The screenshot shows the "Parameter Select" interface. On the left, there are buttons for "Device Select", "Plot Select", "Parameter Select", and "Optimizer Select". The main area has a search bar containing "vth0" and "Add" and "Delete" buttons. Below the search bar is a table:

Name	Value	Min	Max
vth0	0.5	0.4	1

Below the table, a detailed parameter guide for "vth0" is displayed:

```

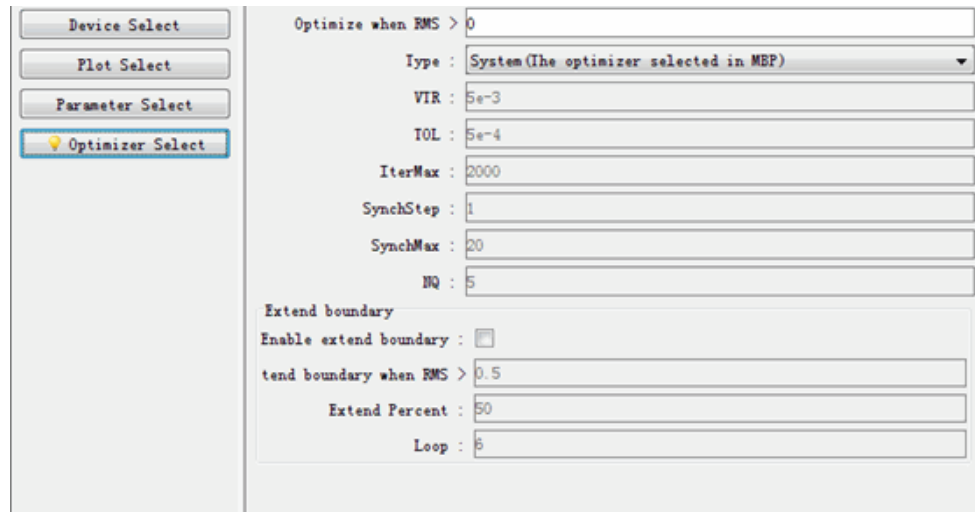
Parameter --- vth0
binable parameter.
Threshold voltage

Info
Default Value : 0.7
Hard Boundary : [-100.0,100.0]
Alias : vtho

Bin Type
  
```

User can input the initial value and boundary for each parameter by double clicking on the Value/Min/Max column.

- Optimizer Select
RMS criteria can be set here. There is also option to allow MBP extend parameter boundary when optimization result reaches the boundary. Default Optimizer is the same with the one selected in MBP. User could also select other optimizer available in MBP.



Device Select	Optimize when RMS > 0
Plot Select	Type : System (The optimizer selected in MBP)
Parameter Select	VIR : 5e-3
Optimizer Select	TOL : 5e-4
	IterMax : 2000
	SynchStep : 1
	SynchMax : 20
	NQ : 5
	Extend boundary
	Enable extend boundary : <input type="checkbox"/>
	Extend boundary when RMS > 0.5
	Extend Percent : 50
	Loop : 5

Menu Bar

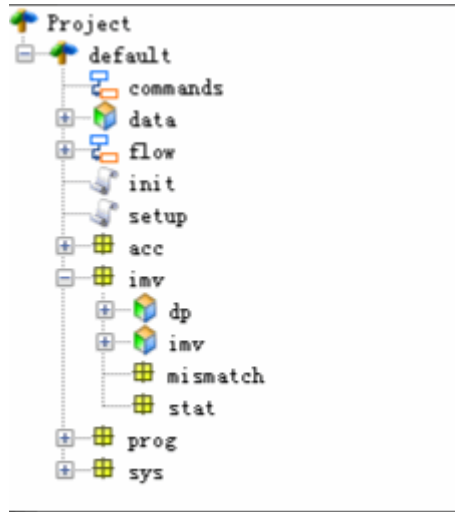
There are three main items in the Menu Bar of script project editor.

- File
 - Save: Saves current code to current project
- Edit:
 - Undo: Undoes function for text input.
 - Redo: Redoes function for text input.
- Debug
 - Run
 - Continue
 - Step Over
 - Step In
 - Step Return

Panes in MBP Main Window

Project Pane

Script in MBP is organized by project. On the left of MBP script project window, the project pane has a hierarchical tree structure. You can create script file for different kinds of usability on different nodes.



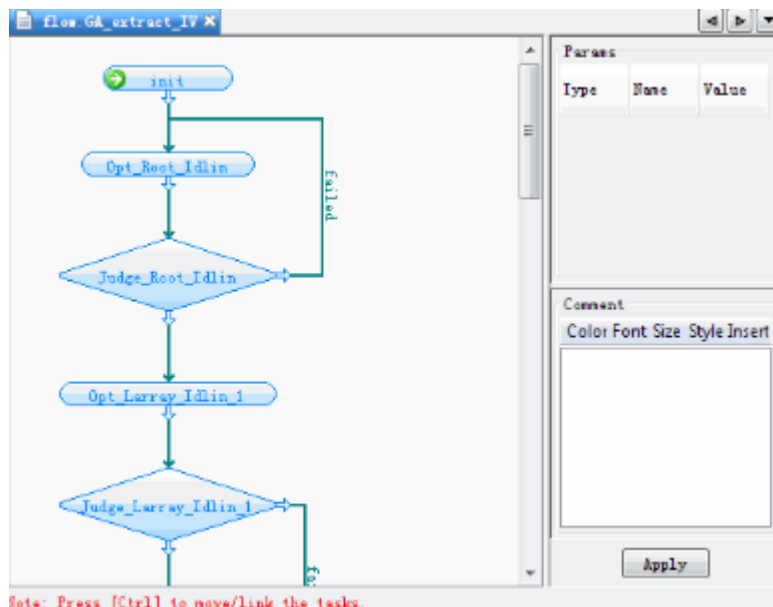
Some frequently used usability includes:

- Commands: Plots selection and organization in MBP GUI
- Data: Data manipulation, can be output to GUI or just used in extraction flow
- Flow: Model extraction flow
- IMV: Configurations for different targets: DP, IMV Mismatch, or global statistical model.
- Sys-gui-navigator: To add new navigator in the GUI.

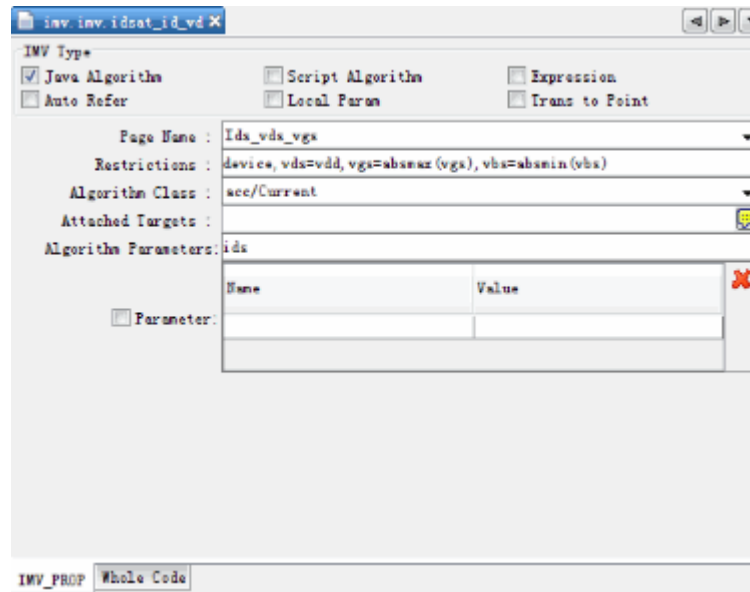
Edit pane

Edit pane is the main pane for all the modifications. It can be displayed in different styles according to different usability selected. For example, see the following figure as displayed in the Edit pane:

Edit pane of extraction flow



Edit pane of IMV configuration

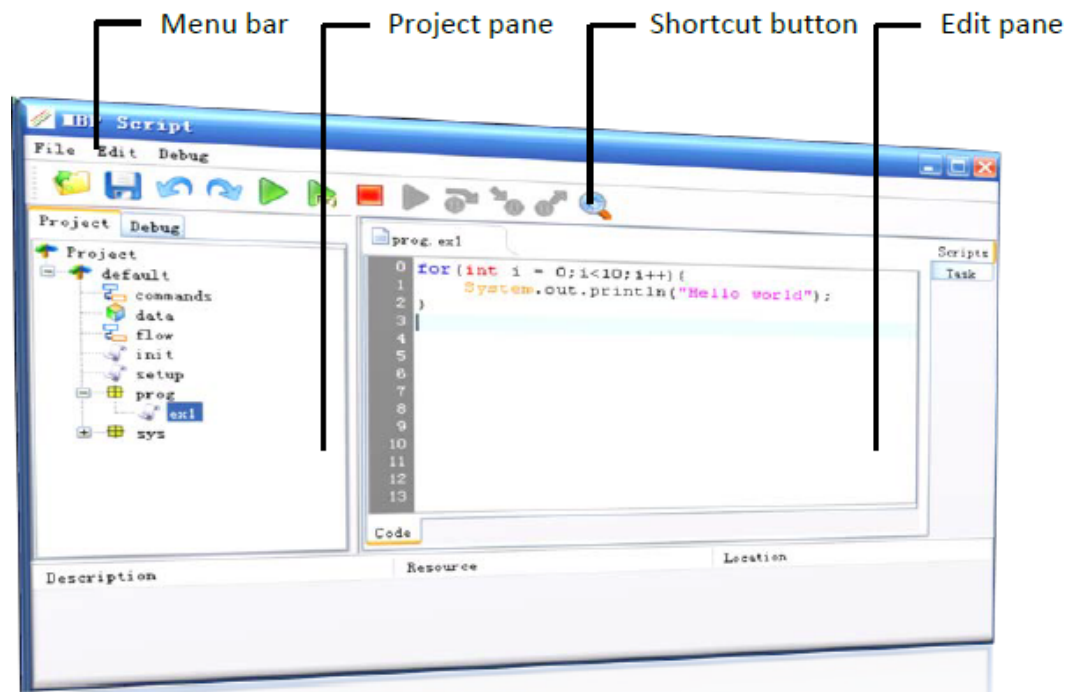


All the script code should also be created and entered in the Edit pane.

For more details, refer to the subsequent chapters and MBP Script Programming Manual.

Script Editor

MBP provides an editor for users to create or modify their own script. Click MBP main menu Script > Script Project to start the editor.



Script Overview

MBP Script is based on Java language, and supports most of the Java expressions (jdk1.4), except class definition. With this script, you can transform data, define plots, do optimization, and build extraction flow. MBP Script is tied to project. Thus, for any modification and customization, save your MBP project in the beginning. After customization, the MBP script can be reused for other project or shared with others. This section describes the MBP Script application in detail. For more advanced application, refer to the [Script Programming](#).

Script Programming

- Data Definition
- Data Organization
- Data Transformation
- Optimization and other Operations
- Script Programming Overview

Data Definition

After defining data, you can perform the following tasks:

1. Define plots.
2. Re-organize or transfer other data.
3. Define the cost function to do optimization.

Quick Start

To create a data file and a plot file, follow these steps:

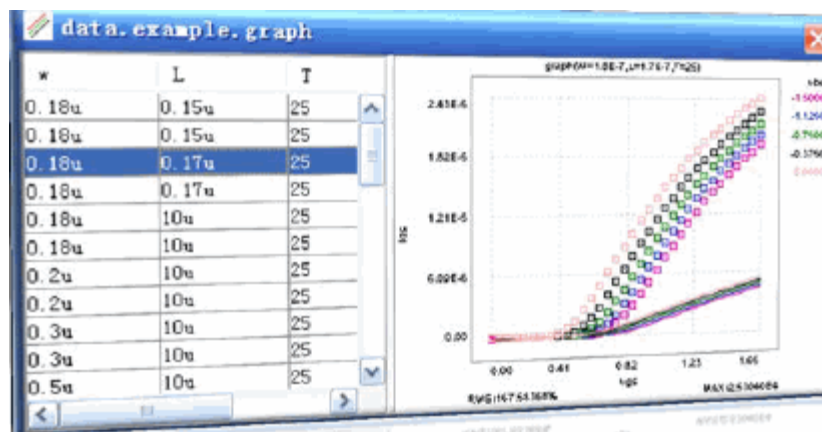
1. Select a bsim3v3 model, and load the demo measurement data.
2. Select the node data and click New > Data on the pop-up menu to create a data script file.
3. Replace the code with the following:

```
1 DATA load(sfunc select){  
2     DATA m = select("ids_vgs_vbs");  
3     return m;  
4 }
```

4. Select the data file you created and click Show Data Table on the pop-up menu. You get a table that lists the data of ids_vgs_vbs.

w	L	T	vds	vbs	vgs	ids
1u	10u	25	0.05	0	0	1.08E-12
1u	10u	25	0.05	0	0.05	4.54E-12
1u	10u	25	0.05	0	0.1	2.07E-11
1u	10u	25	0.05	0	0.15	9.19E-11
1u	10u	25	0.05	0	0.2	4.06E-10
1u	10u	25	0.05	0	0.25	1.71E-9
1u	10u	25	0.05	0	0.3	6.37E-9
1u	10u	25	0.05	0	0.35	1.97E-8
1u	10u	25	0.05	0	0.4	4.75E-8
1u	10u	25	0.05	0	0.45	8.32E-8
1u	10u	25	0.05	0	0.5	1.52E-7
1u	10u	25	0.05	0	0.55	2.70E-7
1u	10u	25	0.05	0	0.6	4.94E-7
1u	10u	25	0.05	0	0.65	9.19E-7
1u	10u	25	0.05	0	0.7	1.71E-6
1u	10u	25	0.05	0	0.75	3.21E-6
1u	10u	25	0.05	0	0.8	6.06E-6
1u	10u	25	0.05	0	0.85	1.15E-5
1u	10u	25	0.05	0	0.9	2.18E-5
1u	10u	25	0.05	0	0.95	4.11E-5
1u	10u	25	0.05	0	1.0	7.61E-5

5. Select the data file and click New > Graph on the pop-up menu to create a graph script file.
6. Click View to receive the plots of ids_vgs_vbs.



Data Organization

Data as Table

There are many kinds of data in MBP. You can describe all MBP data in a table. For example, the following table lists the data for ids_vgs_vbs.

Input						Output
W	L	T	Vds	Vbs	Vgs	Ids
10u	2u	25	0.05	-1.125	1.6	6.5965E-5
10u	2u	25	0.05	-1.125	1.65	6.8524E-5
10u	2u	25	0.05	-1.5	0	6.72E-12
10u	2u	25	0.05	-1.5	0.05	6.92E-12
10u	2u	25	0.05	-1.5	0.1	7.17E-12

POINT

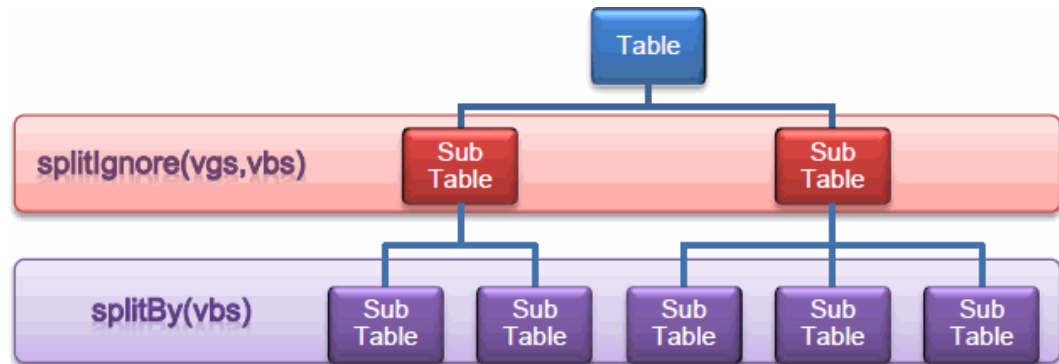
A table consists of lines. Each line has:

1. Input columns (W,L,T,Vds,Vbs,Vgs)
2. Output columns (Ids)

Each line is called a POINT, because you can display it as a point on the graph.

Data as Tree

Each table can be divided into several sub tables with some rules. Each sub table can be further divided. Thus, a table can be transformed to a tree.



How to organize a table to a tree

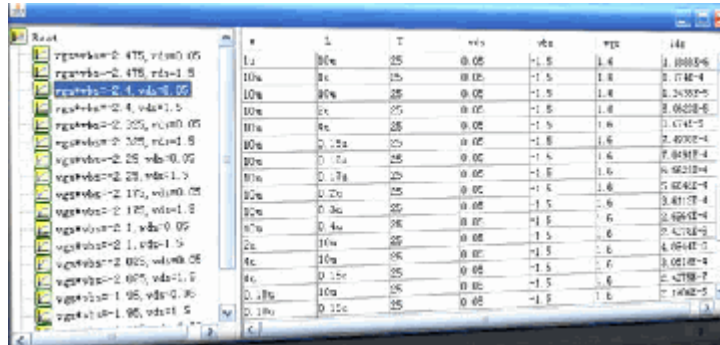
There are three kinds of functions to do the organization:

1. `splitBy(exp1,exp2,exp3...)`: Divide to several tables. Points in each table have same value of `exp1,exp2,exp3...`
2. `splitIgnore(var1,var2,var3...)`: Divide to several tables. Points in each table have same inputs, except `var1,var2,var3...`
3. `selectBy(expression)`: Select the points in each table whose expression value is not 0.
4. `sortBy(exp1,exp2,exp3...)`: Sort each sub table with `exp1,exp2,exp3`.

Here is an example:

```
1 DATA load(sfunc select){
2     DATA m = select("ids_vgs_vbs");
3     m = m.build("splitBy(vgs*vbs,vds)");
4     return m;
5 }
```

Click Show Data Table to see the results.



#	L	T	vgs	vbs	vds	id
1a	0.6e	25	0.05	-1.5	1.6	1.10005-6
10a	0.6e	25	0.05	-1.5	1.6	0.1740-4
100a	0.6e	25	0.05	-1.5	1.6	0.24302-5
1000a	0.6e	25	0.05	-1.5	1.6	0.0620-6
10000a	0.6e	25	0.05	-1.5	1.6	0.1740-4
100000a	0.25a	25	0.05	-1.5	1.6	7.4030-4
1000000a	0.7a	25	0.05	-1.5	1.6	7.0480-4
10000000a	0.17a	25	0.05	-1.5	1.6	6.9620-4
100000000a	0.25a	25	0.05	-1.5	1.6	5.8240-4
1000000000a	0.3a	25	0.05	-1.5	1.6	3.8110-4
10000000000a	0.4a	25	0.05	-1.5	1.6	2.9960-4
100000000000a	0.4a	25	0.05	-1.5	1.6	2.4760-4
1000000000000a	0.4a	25	0.05	-1.5	1.6	4.0640-5
10000000000000a	0.15a	25	0.05	-1.5	1.6	3.0210-4
100000000000000a	0.15a	25	0.05	-1.5	1.6	2.5290-4
1000000000000000a	0.15a	25	0.05	-1.5	1.6	1.9680-4
10000000000000000a	0.15a	25	0.05	-1.5	1.6	1.4680-4

Examples:

```
1 DATA load(sfunc select){
2     DATA m = select("ids_vgs_vbs");
3     m = m.build("splitIgnore(vgs,vbs).splitBy(vbs)");
4     return m;
5 }
```

```
1 DATA load(sfunc select){
2     DATA m = select("ids_vgs_vbs");
3     m = m.build("splitIgnore(vgs).splitBy(vgs>1)");
4     return m;
5 }
```

```
1 DATA load(sfunc select){
2     DATA m = select("ids_vgs_vbs");
3     m = m.build("splitIgnore(vgs).splitBy(vgs>1).selectBy(vgs==max(vgs))");
4     return m;
5 }
```


Example of sparse vgs

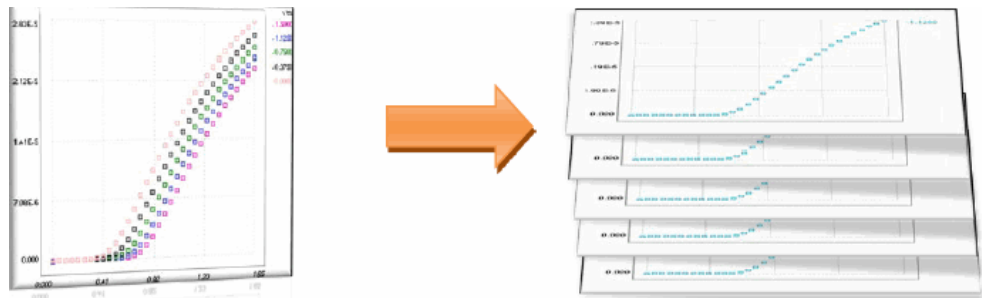
```
1 DATA load(sfunc select){
2   DATA m = select("ids_vgs_vbs");
3   m = m.build("splitignore(vgs)
4     .splitBy(int(vgs/0.5))
5     .selectBy(vgs==max(vgs))");
6   return m;
7 }
```

With this script, you get a new data with fewer vgs points, as shown in the graph.

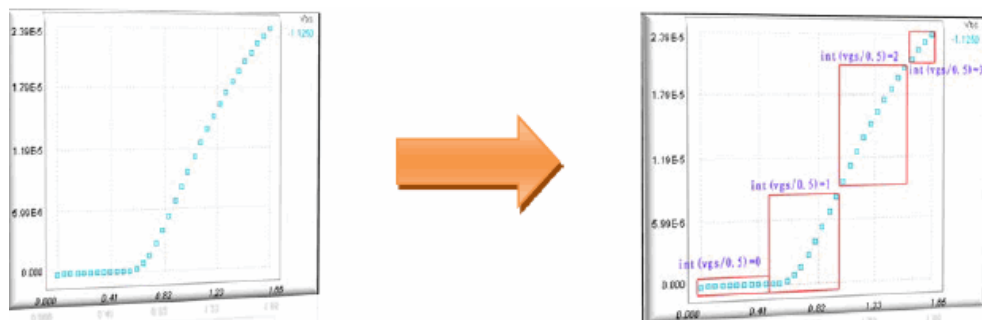


How does it work?

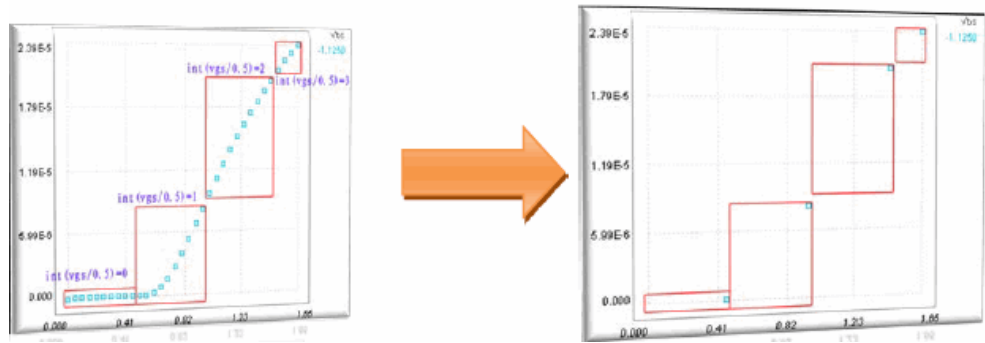
1. splitignore(vgs): Divide the data to several curves.



2. splitBy(int(vgs/0.5)): Divide each curve to several groups: int(vgs/0.5)=0, int(vgs/0.5)=1, int(vgs/0.5)=2, int(vgs/0.5)=3



3. selectBy(vgs==max(vgs)): For each group, only one point is selected.



Plot Definition

After data has been defined in the script, you can easily create a plot for it using the following steps:

1. Select the data.
2. Right-click and select New > Graph on the pop-up menu.
3. Select the graph type and set the properties.
4. Click View to get the plots.

Sweep1 and Sweep2

Sweep1 and Sweep2 are important concepts in plot definition. They are defined as follows:

1. All points in a curve differ from each other by Sweep1 only.
2. All curves in a page differ from each other by Sweep2 only.
 For XY(Cartesian) graphs, Sweep1 is X and Sweep2 is P if not defined otherwise.
 For Smith Chart graphs and Polar graphs, sweep1 must be defined, whereas Sweep2 is p if not defined otherwise.

Data Transformation



POINT to POINT

Example:

```

1 DATA load(sfunc select){
2   DATA originalTable = select("ids_vgs_vbs");
3   DATA newTable = originalTable.trans(transFunc,"ids_mul_vds");
4   return newTable;
5 }
6 void transFunc(POINT to, POINT from){
7   to->"ids_mul_vds" = from->"ids"* from->"vds";
8 }

```

It creates a new data with an output called ids_mul_vds.

w	L	T	vds	vbs	vgs	ids
1u	10u	25	0.05	0	0	1.08E-12
1u	10u	25	0.05	0	0.05	4.64E-12
1u	10u	25	0.05	0	0.1	2.07E-11
1u	10u	25	0.05	0	0.15	9.195E-11
1u	10u	25	0.05	0	0.2	4.0652E-10
1u	10u	25	0.05	0	0.25	1.7173E-9



w	L	T	vds	vbs	vgs	ids_mul_vds
1u	10u	25	0.05	0	0	5.4E-14
1u	10u	25	0.05	0	0.05	2.32E-13
1u	10u	25	0.05	0	0.1	1.035E-12
1u	10u	25	0.05	0	0.15	4.5975E-12
1u	10u	25	0.05	0	0.2	2.0326E-11
1u	10u	25	0.05	0	0.25	8.5865E-11

The output of newTable is updated by the function transFunc. In the function: transFunc(POINT to, POINT from) from is a point in originalTable, and to is a point in newTable. In this case, the function gets the ids value and vds value of from, and sets their multiple values to the ids_mul_vds in to.

TABLE to POINT


Example:

```

1 DATA load(sfunc select){
2   DATA originalTable = select("ids_vgs_vbs");
3   originalTable = originalTable.build("splitIgnore (vgs)");
4   DATA newTable = originalTable.transSubToPoint(it create
5 (transFunc,"maxids");
6   return newTable;
7 }
8 void transFunc(POINT to, DATA from){
9   double[] ids = from->"ids";
10  to->"maxids" = ids.max();

```

w	L	T	vds	vbs	vgs	ids
1u	10u	25	0.05	0	1	9.1177E-7
1u	10u	25	0.05	0	1.05	9.8483E-7
1u	10u	25	0.05	0	1.6	1.6522E-6
1u	10u	25	0.05	0	1.65	1.6988E-6
1u	10u	25	0.05	-0.375	0	1.5E-13
1u	10u	25	0.05	-0.375	0.05	4.6E-13
1u	10u	25	0.05	-0.375	0.1	1.16E-12
1u	10u	25	0.05	-0.375	0.15	4.94E-12
1u	10u	25	0.05	-0.375	0.2	2.311E-11



w	L	T	vds	vbs	maxids
1u	10u	25	0.05	-1.5	1.2414E-6
1u	10u	25	0.05	-1.125	1.3388E-6
1u	10u	25	0.05	-0.75	1.4446E-6
1u	10u	25	0.05	-0.375	1.5623E-6
1u	10u	25	0.05	0	1.6988E-6
1u	10u	25	1.5	-1.5	1.0957E-5
1u	10u	25	1.5	-1.125	1.2552E-5
1u	10u	25	1.5	-0.75	1.4368E-5
1u	10u	25	1.5	-0.375	1.6468E-5

In line 3, originalTable is divided to several groups, and in each group, only vgs is different.

Root:	v	l	..	vds	vns	vgs	lrs
v=1.8E-7, L=1.53-T, T=25, vds=0.05, vbs=-1.5	0.1E0	0.15u	25	C.05	-1.5	0	1.9E-13
v=1.8E-7, L=1.53-T, T=25, vds=0.05, vbs=-1.125	0.1E0	0.15u	25	C.05	-1.5	0.15	1.4E-13
v=1.8E-7, L=1.53-T, T=25, vds=0.05, vbs=-0.75	0.1E0	0.15u	25	C.05	-1.5	0.1	1.9E-13
v=1.8E-7, L=1.53-T, T=25, vds=0.05, vbs=0	0.1E0	0.15u	25	C.05	-1.5	0.15	2.0E-13
v=1.8E-7, L=1.53-T, T=25, vds=0.05, vbs=0	0.1E0	0.15u	25	C.05	-1.5	0.2	6.9E-13
v=1.0E-7, L=1.53-T, T=25, vds=1.5, vbs=1.5	0.1E0	0.15u	25	C.05	-1.5	0.25	2.54E-12
v=1.8E-7, L=1.53-T, T=25, vds=1.5, vbs=-1.125	0.1E0	0.15u	25	C.05	-1.5	0.3	1.15E-11
v=1.8E-7, L=1.53-T, T=25, vds=1.5, vbs=-0.75	0.1E0	0.15u	25	C.05	-1.5	0.35	5.4453-11
v=1.8E-7, L=1.53-T, T=25, vds=1.5, vbs=-0.375	0.1E0	0.15u	25	C.05	-1.5	0.4	2.5744E-10
v=1.8E-7, L=1.53-T, T=25, vds=1.5, vbs=0	0.1E0	0.15u	25	C.05	-1.5	0.45	1.1428E-9
v=1.8E-7, L=1.73-T, T=25, vds=1.05, vbs=-1.5	0.1E0	0.15u	25	C.05	-1.5	0.5	5.0130E-9
v=1.8E-7, L=1.73-T, T=25, vds=0.05, vbs=-1.125	0.1E0	0.15u	25	C.05	-1.5	0.55	2.2377E-9
v=1.0E-7, L=1.73-T, T=25, vds=0.05, vbs=0.75	0.1E0	0.15u	25	C.05	-1.5	0.7	6.8838E-9

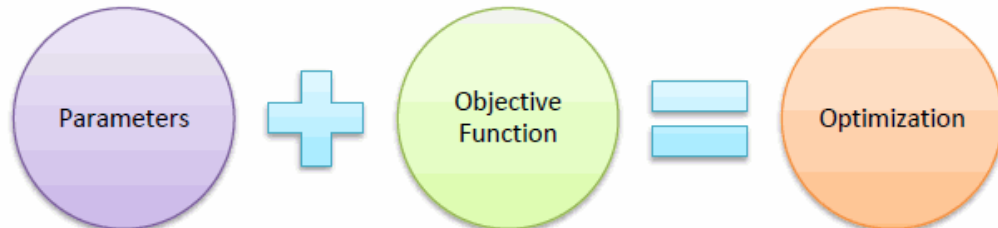
Each group of originalTable is transformed to a POINT of newTable.

The output of newTable is updated by the function transFunc.

In the function: transFunc(PPOINT to, DATA from) from is a group in originalTable, and to is a point in newTable.

In this case, the function gets the ids values of from, and sets the max value to the maxids in to.

Optimization and other Operations



Objective function returns a double value. Optimizer tries to find out the parameter values to get the minimal value of objective function.

Example (Assuming that you have already defined a data example):

```

1 | DATA dat = data.example::get();
2 | MBPOPT.optimize("vth0,u0,k1",error);
3 | MBPGUI.update();
4 | dat = null;
5 | double error(){
6 |     dat.doSimulation(MBP);
7 |     return dat.getRMS("ids");
8 | }
  
```

Here,
Line 2: Do optimization.
`vth0`, `u0`, `k1` is parameter
`error` is the objective function.
Line 6: Simulate the data.
Line 7: Return the rms value.

API for Model Parameter

```
Param param = MBP.getParam("vth0");
```

`vth0` is the name of parameter.

As you get the param, you can do the following operations:

```
double value = p.getValue();// get the value of vth0
p.setValue(0.7);//set vth0 = 0.7
p.setBoundary(0.5, 1.5);//set soft boundary = [0.5, 1.5]
p.setHardBoundary(0, 10);//set hard boundary = [0, 10]
double lower = p.getLower();//get the lower value of soft boundary
double upper = p.getUpper();//get the upper value of soft boundary
double lower = p.getHardLower();//get the lower value of hard boundary
double upper = p.getHardUpper();//get the upper value of hard boundary
```

```
double value = p.getValue();// get the value of vth0
```

Build Extraction Flow

In MBP, you can define some tasks and organize them in a flow.

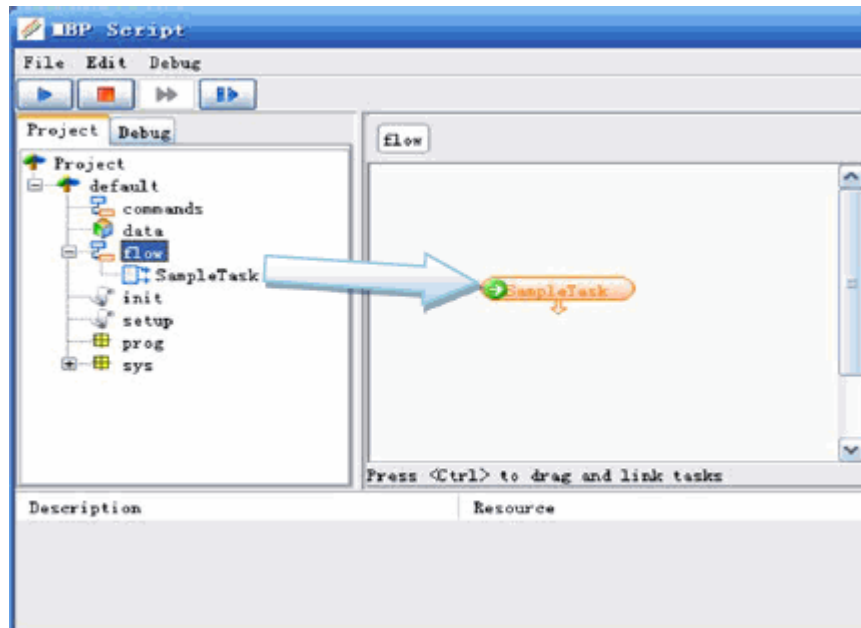
1. Select the node flow and click New > Task on the pop-up menu to create a task file. You can find the task code.

```
task(){
}
TASK.start();
```

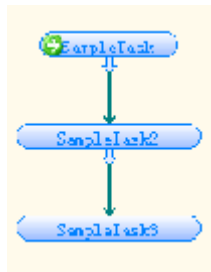
2. Write the code you want the task to do, for example, see the following:


```
task(){
    System.out.println("Hello task");
}
TASK.start();
```

3. Click the node flow and drag the task to the flow.



4. Create more tasks and add them to flow, and then press ctrl to link them.



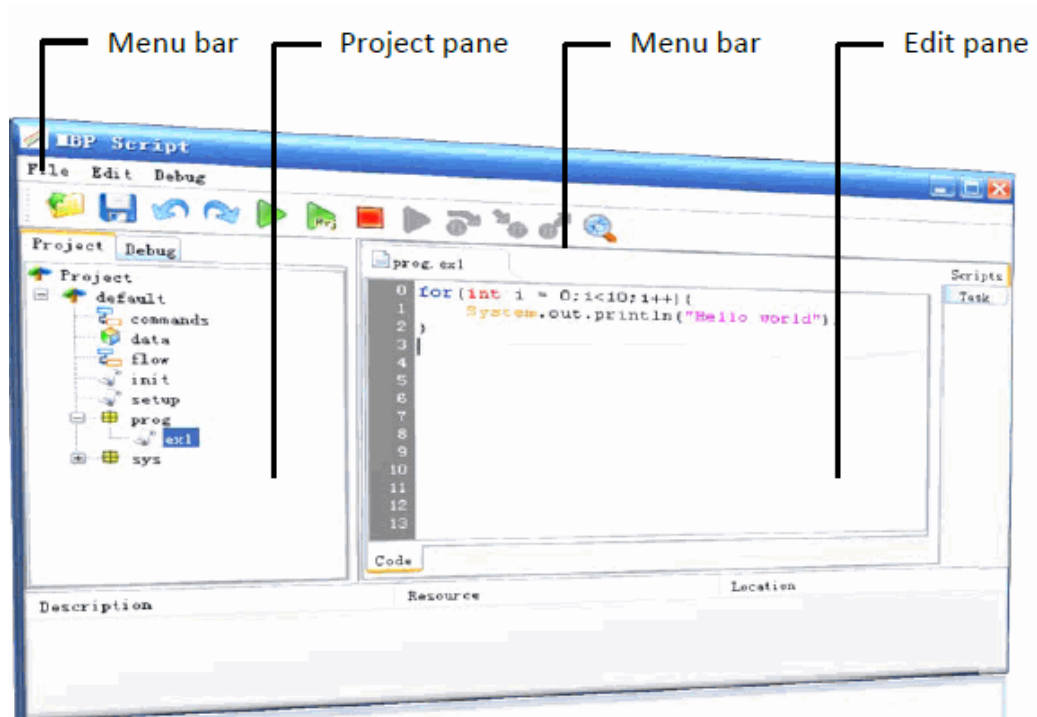
5. Click  to run the flow. You can see the task execute one by one.

Script Programming Overview

MBP Script is based on Java, and supports most of the Java expressions (jdk1.4), except class definition. With the script, you can transform data, define plots, do optimization, and build extraction flow.

Script Editor

MBP provides an editor for script. Click Script > Script Project to start the editor.



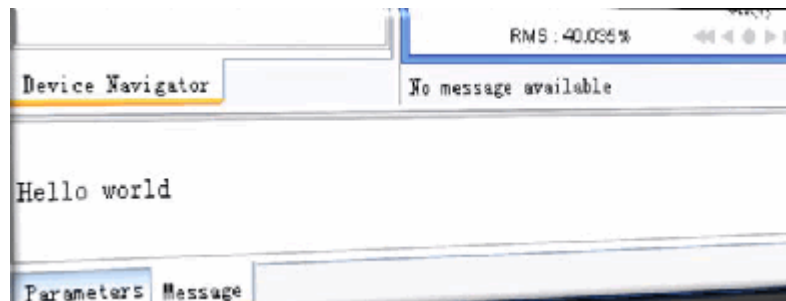
Script in MBP is organized by project. On the left of MBP script project window, the project pane has a hierarchical tree structure. On different nodes, you can create script file for different kinds of usability.

To create a new script file, these steps:

1. Select the node:prog.
2. Right-click the node and select New > File on the pop-up menu.
3. Input a name to define a script file.
4. Select the newly created file.
5. Write the following code on the edit pane.

```
System.out.println("Hello world");
```

6. Click  and view the message window in MBP. *Hello world* is printed.



Function Definition

You can define a script function. Here is an example:

```
1 double sum(double[] values){
2     double s = 0;
3     for(int i = 0;i<values.length;i++){
4         s += values[i];
5     }
6     return s;
7 }
8 double[] v = new double[]{1,2,3,4,5};
9 System.out.println(sum(v));
```

Here,

- Line 1-7: Define a function called `sum`. It returns the sum of values.
- Line 8: Create an array:1,2,3,4,5
- Line 9: Print the result of 1+2+3+4+5

You can also pass a function to another function as a variable, and use `sfunc` as the type of the variable.

```
1 double sqr(double v){
2     return v*v;
3 }
4 double sqrt(double v){
5     return Math.sqrt(v);
6 }
7 double sum(double[] values,sfunc f){
8     double s = 0;
9     for(int i = 0;i<values.length;i++){
10         double v = f(values[i]);
11         s += v;
12     }
13     return s;
14 }
15 double[] v = new double[]{1,2,3,4,5};
16 System.out.println(sum(v,sqr));
```

Here,

- Line 1-3: Define a function `sqr`: $\text{sqr}(v) = v^2$
- Line 4-6: Define a function `sqrt`: $\text{sqrt}(v) = \sqrt{v}$

- Line 7-14: Define a function `sum`: `sum(v,f) = f(v[i])=0`
- Line 15: Create an array: 1,2,3,4,5
- Line 16: Print the result of `sqr(v[i])=0`

This information is subject to change
without notice.

www.keysight.com

