tek.com/keithley

Automated Characterization Suite (ACS) Basic Edition

Reference Manual

ACSBASIC-901-01 Rev. J November 2023

ACSBASIC-901-01J

KEITHLEY
A Tektronix Company

ACS Basic Edition

Automated Characterization Suite Software

Reference Manual

These are the original instructions in English.

Document number: ACSBASIC-901-01 Rev. J November 2023

KEITHLEY
A Tektronix Company

Safety precautions

The following safety precautions should be observed before using this product and any associated instrumentation. Although some instruments and accessories would normally be used with nonhazardous voltages, there are situations where hazardous conditions may be present.

This product is intended for use by personnel who recognize shock hazards and are familiar with the safety precautions required to avoid possible injury. Read and follow all installation, operation, and maintenance information carefully before using the product. Refer to the user documentation for complete product specifications.

If the product is used in a manner not specified, the protection provided by the product warranty may be impaired.

The types of product users are:

Responsible body is the individual or group responsible for the use and maintenance of equipment, for ensuring that the equipment is operated within its specifications and operating limits, and for ensuring that operators are adequately trained.

Operators use the product for its intended function. They must be trained in electrical safety procedures and proper use of the instrument. They must be protected from electric shock and contact with hazardous live circuits.

Maintenance personnel perform routine procedures on the product to keep it operating properly, for example, setting the line voltage or replacing consumable materials. Maintenance procedures are described in the user documentation. The procedures explicitly state if the operator may perform them. Otherwise, they should be performed only by service personnel.

Service personnel are trained to work on live circuits, perform safe installations, and repair products. Only properly trained service personnel may perform installation and service procedures.

Keithley products are designed for use with electrical signals that are measurement, control, and data I/O connections, with low transient overvoltages, and must not be directly connected to mains voltage or to voltage sources with high transient overvoltages. Measurement Category II (as referenced in IEC 60664) connections require protection for high transient overvoltages often associated with local AC mains connections. Certain Keithley measuring instruments may be connected to mains. These instruments will be marked as category II or higher.

Unless explicitly allowed in the specifications, operating manual, and instrument labels, do not connect any instrument to mains.

Exercise extreme caution when a shock hazard is present. Lethal voltage may be present on cable connector jacks or test fixtures. The American National Standards Institute (ANSI) states that a shock hazard exists when voltage levels greater than 30 V RMS, 42.4 V peak, or 60 VDC are present. A good safety practice is to expect that hazardous voltage is present in any unknown circuit before measuring.

Operators of this product must be protected from electric shock at all times. The responsible body must ensure that operators are prevented access and/or insulated from every connection point. In some cases, connections must be exposed to potential human contact. Product operators in these circumstances must be trained to protect themselves from the risk of electric shock. If the circuit is capable of operating at or above 1000 V, no conductive part of the circuit may be exposed.

Do not connect switching cards directly to unlimited power circuits. They are intended to be used with impedance-limited sources. NEVER connect switching cards directly to AC mains. When connecting sources to switching cards, install protective devices to limit fault current and voltage to the card.

Before operating an instrument, ensure that the line cord is connected to a properly-grounded power receptacle. Inspect the connecting cables, test leads, and jumpers for possible wear, cracks, or breaks before each use.

When installing equipment where access to the main power cord is restricted, such as rack mounting, a separate main input power disconnect device must be provided in close proximity to the equipment and within easy reach of the operator.

For maximum safety, do not touch the product, test cables, or any other instruments while power is applied to the circuit under test. ALWAYS remove power from the entire test system and discharge any capacitors before connecting or disconnecting cables or jumpers, installing or removing switching cards, or making internal changes, such as installing or removing jumpers.

Do not touch any object that could provide a current path to the common side of the circuit under test or power line (earth) ground. Always make measurements with dry hands while standing on a dry, insulated surface capable of withstanding the voltage being measured.

For safety, instruments and accessories must be used in accordance with the operating instructions. If the instruments or accessories are used in a manner not specified in the operating instructions, the protection provided by the equipment may be impaired.

Do not exceed the maximum signal levels of the instruments and accessories. Maximum signal levels are defined in the specifications and operating information and shown on the instrument panels, test fixture panels, and switching cards.

When fuses are used in a product, replace with the same type and rating for continued protection against fire hazard.

Chassis connections must only be used as shield connections for measuring circuits, NOT as protective earth (safety ground) connections.

If you are using a test fixture, keep the lid closed while power is applied to the device under test. Safe operation requires the use of a lid interlock.

If a  screw is present, connect it to protective earth (safety ground) using the wire recommended in the user documentation.

The  symbol on an instrument means caution, risk of hazard. The user must refer to the operating instructions located in the user documentation in all cases where the symbol is marked on the instrument.

The  symbol on an instrument means warning, risk of electric shock. Use standard safety precautions to avoid personal contact with these voltages.

The  symbol on an instrument shows that the surface may be hot. Avoid personal contact to prevent burns.

The  symbol indicates a connection terminal to the equipment frame.

If this  symbol is on a product, it indicates that mercury is present in the display lamp. Please note that the lamp must be properly disposed of according to federal, state, and local laws.

The **WARNING** heading in the user documentation explains hazards that might result in personal injury or death. Always read the associated information very carefully before performing the indicated procedure.

The **CAUTION** heading in the user documentation explains hazards that could damage the instrument. Such damage may invalidate the warranty.

The **CAUTION** heading with the  symbol in the user documentation explains hazards that could result in moderate or minor injury or damage the instrument. Always read the associated information very carefully before performing the indicated procedure. Damage to the instrument may invalidate the warranty.

Instrumentation and accessories shall not be connected to humans.

Before performing any maintenance, disconnect the line cord and all test cables.

To maintain protection from electric shock and fire, replacement components in mains circuits — including the power transformer, test leads, and input jacks — must be purchased from Keithley. Standard fuses with applicable national safety approvals may be used if the rating and type are the same. The detachable mains power cord provided with the instrument may only be replaced with a similarly rated power cord. Other components that are not safety-related may be purchased from other suppliers as long as they are equivalent to the original component (note that selected parts should be purchased only through Keithley to maintain accuracy and functionality of the product). If you are unsure about the applicability of a replacement component, call a Keithley office for information.

Unless otherwise noted in product-specific literature, Keithley instruments are designed to operate indoors only, in the following environment: Altitude at or below 2,000 m (6,562 ft); temperature 0 °C to 50 °C (32 °F to 122 °F); and pollution degree 1 or 2.

To clean an instrument, use a cloth dampened with deionized water or mild, water-based cleaner. Clean the exterior of the instrument only. Do not apply cleaner directly to the instrument or allow liquids to enter or spill on the instrument. Products that consist of a circuit board with no case or chassis (e.g., a data acquisition board for installation into a computer) should never require cleaning if handled according to instructions. If the board becomes contaminated and operation is affected, the board should be returned to the factory for proper cleaning/servicing.

Safety precaution revision as of June 2018.

Table of contents

Section 1

ACS Basic introduction

In this section:

ACS Basic software overview

If you have any questions after reviewing this information, please contact your local Keithley Instruments office, sales partner, or distributor. You can also call the Tektronix corporate headquarters (toll-free inside the U.S. and Canada only) at 1-800-833-9200. For worldwide contact numbers, visit tek.com/contact-tek.

NOTE

When the Series 2600B System SourceMeter® instruments are referenced, it also includes the Series 2600A System SourceMeter instruments. When the Model 3706A and Series 3700A System Switch/Multimeter instruments are referenced, it also includes the Model 3706 and Series 3700 System Switch/Multimeter instruments.

ACS Basic Edition is software that is optimized for parametric testing of component and discrete (packaged) semiconductor devices. ACS Basic maximizes the productivity of technicians and engineers in research and development. The versatile architecture of this software allows it to meet the wide-ranging and changing requirements of semiconductor device testing. It supports the Keithley Instruments source and measure instrument products, including Series 2600B, Series 2400, and Series 2400 Graphical Series SourceMeter® Source Measure Unit (SMU) instruments, 2651A and 2657A SourceMeter SMU instruments, and 4200A-SCS SMUs and capacitance-voltage units (CVUs). For a complete list of supported instruments, refer to tek.com/keithley.

ACS Basic includes the rich set of Keithley Instrument proven parametric libraries. Choose a test and begin running it to immediately start gathering data and analyzing it. You can also customize any test with the embedded script editor.

The built-in data analysis tools allow you to quickly analyze the parametric data. For example, place device curves developed from newly collected data over example curves for fast comparisons. To perform specialized calculations on raw data, use the mathematical Formulator tool to create customized parameter calculations. Data can be easily saved in graphical and tabular formats.

ACS Basic software can control external instruments through GPIB or LXI communications. It can also control hardware on the Keithley Instruments Model 4200A-SCS Parametric Analyzer. ACS Basic can perform multisite parallel I-V testing using a Series 2600B as the primary instrument.

ACS Basic has three modes:

- **Single-test mode**: Allows you to perform a single test on a single device. You can select a test from one of three libraries based on the device and instruments available. You can also create your own test or modify available tests.
- **Multitest mode**: Allows you to perform multiple tests on a single device (target application), or multiple tests on multiple devices. It is easier to use external instruments (such as DMMs and switches) in multitest mode.
- **Trace mode**: Allows you to perform interactive operations with a SourceMeter® and is only used with the Series 2600B, Model 2651A, and Series 2400 instruments. You can adjust the maximum sweep value and maximum power to the device. You can easily transition between dc and pulse forcing modes.

ACS Basic provides graphical interactive test modules (ITMs), script test modules (STMs), and Python language test modules (PTMs). An ITM allows you to define a test interactively using a graphical user interface (GUI). An STM supports Test Script Processor (TSP™) files. A TSP file is written to customize tests for TSP products. A PTM can control any GPIB instrument. A script editor is included in ACS Basic for editing STM and PTM modules.

Reference manual content

This manual contains information about the ACS Basic software. It includes information about how to install, configure, and operate the ACS Basic software and the full details of software features such as building, loading, and executing test projects.

For more information on instruments that can be used with ACS Basic software, refer to the documentation for the specific Keithley Instrument model:

- Models 707/707A/707B and 708/708A/708B Switching Matrix
- Series 2400 SourceMeter®
- Series 2600A System SourceMeter®
- Series 2600B System SourceMeter®
- Model 2600-PCT-xB
- Model 2651A High Current System SourceMeter®
- Model 2657A High Power System SourceMeter®
- Series 3700A System Switch/Multimeter
- Model 4200A-SCS Semiconductor Characterization Suite
- Model 8010 High Power Device Test Fixture
- Model 8020 High Power Interface Panel

You can find several documents in the ACS Basic Help menu:

- This reference manual.
- The *ACS Basic Quick Start Guide*, which you can use to quickly learn about ACS Basic, including how to connect, install, and test a single instrument.
- The *ACS Basic Libraries Reference Manual*, which contains detailed information about the Series 2600B LPT library.

System requirements

ACS Basic can be installed on any computer that meets the minimum configurations provided in the ACS Basic datasheet, available at tek.com/keithley.

For detailed configuration information, refer to the *ACS Basic Software Quick Start Guide* (part number ASCBASIC-903-01), available from the ACS Basic Help menu.

NOTE

It is recommended that you back up your system before installing new software.

NOTE

If you have a PCT-CVU, the ACS Basic software is already installed on its computer.

Section 2

ACS Basic software introduction

In this section:

Start ACS Basic introduction

NOTE

After you have installed ACS Basic software and before you set up any projects or run any tests, it is recommended that you review the Release Notes. The Release Notes contain valuable information about known issues and provide guidance on how to handle issues that you may encounter. The Release Notes are available from the Windows Start menu and from the Help menu in ACS Basic.

NOTE

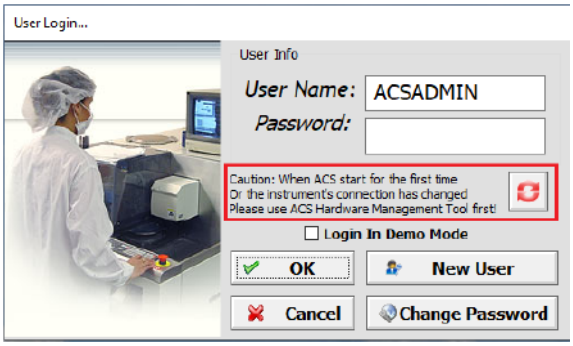
Before you start using ACS software, make sure you have reviewed the Hardware Management Tool information in the Quick Start Guide. If you encounter a fatal error message when opening ACS Basic, the hardware configuration may have changed. You need to run the Hardware Management Tool to establish the hardware configuration.

Before opening the software, make sure all the instruments are turned on and all self-tests have completed. You must establish a new user account when you open the software for the first time.

To start the software:

1. Select the ACS Basic icon (either a desktop icon or a quick launch icon). The User Login dialog box opens (see the following figure).

Figure 1: User Login



2. Type a user name and password in the User Name and Password fields. The default user name is ACSADMIN. There is no default password.

NOTE

More than one new user account can be established at this time. You can establish new accounts by selecting New User. To change the password, select Change Password.

3. Select **OK**. The ACS software start window opens.

Figure 2: Start window

Note that your version of ACS Basic may not match the previous graphic.

You can select **OK** to open ACS Basic or select **Demo mode**. Depending on the license you have, you may not have access to all the software features. However, with the 60-day trial version, you have full access to the ACS Basic edition (ACSBASCFL) software.

NOTE

When the ACS Basic software is started in online mode, the startup time depends on how many instruments are in the hardware configuration group. For example, if there are several instruments, the startup time is longer.

Software modes

Depending on your license status and the hardware configuration, the software starts in one of the following modes:

- Demo
- View
- Online

Demo mode

If you log into the software in demo mode, you cannot communicate with the test and measurement hardware. However, you can develop test projects and review results from testing.

View mode

If you log in and your hardware configuration has changed, you are in view mode. You can select the verification dialog box and choose online mode to update the hardware configuration.

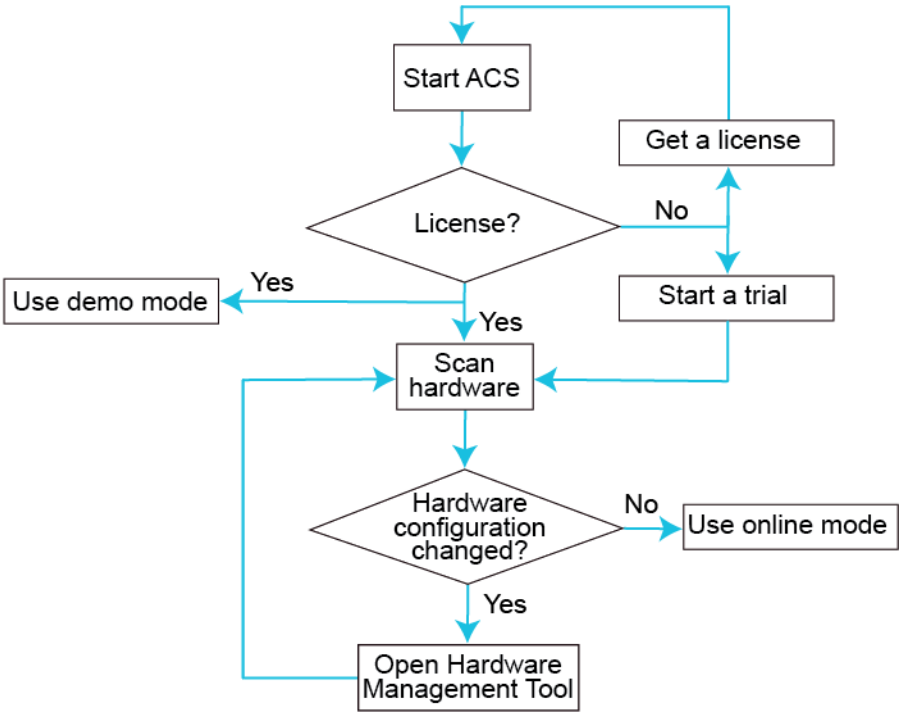
You can stay in view mode and continue to test your previous configuration; however, it will not test the updated configuration.

In view mode, you can set up a test and control a prober station, but the test modules will be executed in demo mode. This means that you cannot communicate with the test and measurement hardware, only the prober station.

Online mode

In online mode you can communicate with the test and measurement hardware and set up tests to control a prober station. Depending on your hardware configuration, you may receive an indication that you cannot open a default project. The next figure shows you how to navigate the software and available options based on your license.

Figure 3: ACS flow chart



Account management

The following topics describe how to create user accounts, define who can create new users, and describe how to initialize multiple user accounts, change a password, and delete a user account.

User accounts

ACS Basic software provides three types of user accounts and each has different rights and security features.

- **Administrator** (or engineer administrator): This account has unrestricted rights to create, edit, and run test projects, including test parameters within ACS Basic. The administrator can also edit or delete operator accounts. The default administrator account user name is ACSADMIN (case sensitive).

NOTE

There can be other engineer user accounts, however, there is only one engineer administrator.

- **Engineer**: This account has unrestricted rights to create, edit, and run test projects, including test parameters within ACS Basic. However, an engineer account cannot edit or delete operator accounts.
- **Operator**: This account can open and run available test projects that were created by either an Engineer or Administrator.

Creating new users

NOTE

Only the administrator account (ACSADMIN) has the rights to add a new Operator or Engineer account.

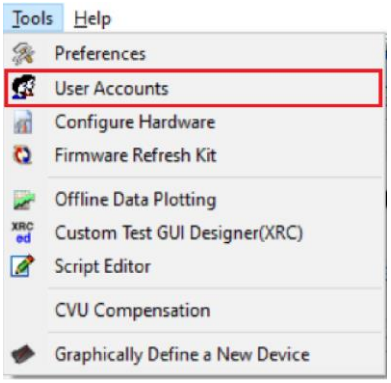
NOTE

Only the administrator account (ACSADMIN) sees the complete list of user accounts. Engineer accounts only see their own account. Operators do not have access to this feature.

To create a new user:

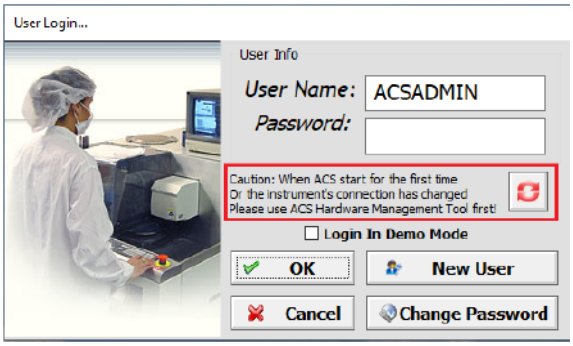
1. If you are presently logged in to ACS Basic, you can select **Tools** and select **User Accounts** from the list (see the following figure).

Figure 4: User Accounts location



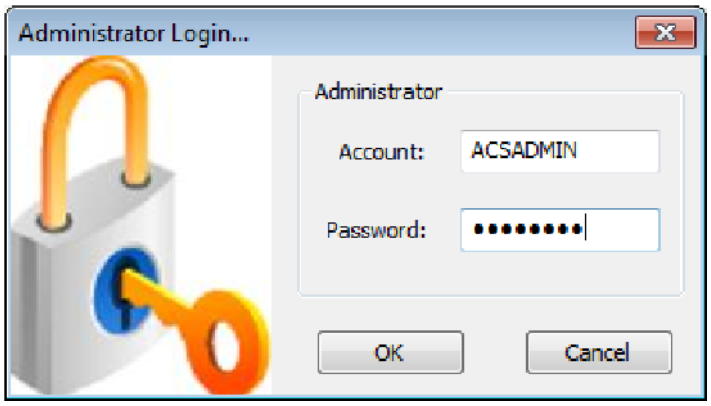
2. If you are not logged in to ACS Basic, you can select **New User** in the User Login GUI (see the following figure).

Figure 5: User Login



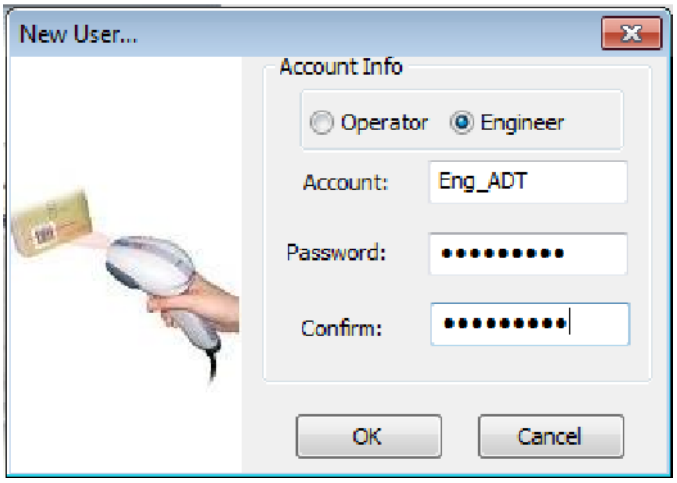
3. The Administrator Login dialog opens (see the following figure).

Figure 6: Administrator login dialog



4. Enter the password for the Administrator (the default Administrator account name is ACSADMIN).
5. Select **OK**. A New User dialog opens (see the following figure).

Figure 7: New User dialog



6. Select the type of account you to create: **Operator** or **Engineer**
   - An Operator account allows you to load and run test projects.
   - An Engineer account allows you access to all ACS Basic functions.
7. Enter your **Account** name and **Password**.
8. Enter your password again in the **Confirm** field.
9. Select **OK**. The new user account is complete.

Initialize multiple user accounts

This information is for those who need to initialize multiple user accounts in a Windows® environment.

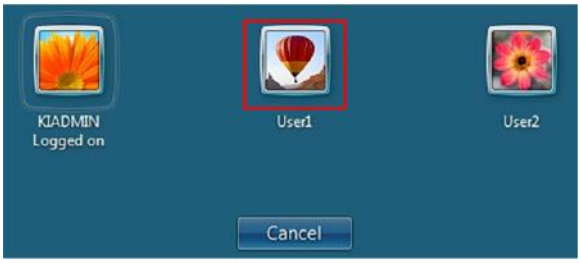
CAUTION

If you try to run ACS Basic on a computer that has more than one Windows user account and you have not initialized all the accounts to run ACS Basic, you will receive an error. Refer to the following information to establish multiple user accounts in a Windows environment.

NOTE

This information is for those who have more than one user account when logging in to a Windows environment (see the following figure). If you have more than one user account on your computer, make sure that you are logged into the user account where ACS Basic was installed.

Figure 8: Windows User Accounts



NOTE

If the administrator (for example, Admin1) has already installed ACS Basic software, none of the other users (User1 or User2) need to install the software. However, you do need to initialize the user accounts (User1 and User2) so that they have access to ACS Basic software.

To initialize multiple user accounts:

1. Log into a user account (for example, User1).
2. Select **Windows Start**.
3. In All Programs, find **ACS Basic**.
4. Select **Initialize New User**, as shown in the following figure.

Figure 9: Windows Start Function

5. Restart your computer.

When your computer reboots, you can log in to Windows with your initialized account and run ACS Basic software.

Change password

NOTE

Only the administrator account (ACSADMIN) has the necessary rights to change a password for all user accounts. An engineer account can change its own password, but not any other accounts. An operator does not have access to this function.

To change a password:

1. While in the User Accounts window, select the account to change the password.
2. Select **Edit** to open the Change password dialog. Enter the present password of the account, enter a new password, and then confirm in the appropriate fields.
3. Select **OK**.

Delete user account

NOTE

Only the administrator account (ACSADMIN) has the necessary rights to delete user accounts. The engineer and operator account do not have access to this function.

To delete a user account:

1. While in the User Accounts window, select the account you want to delete.
2. Select **Delete**.
3. To complete the process, select **Yes** when asked if you want to delete the account.

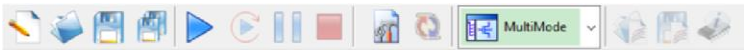
ACS Basic GUI

When you first open ACS Basic, the default test mode that you see is Single-test mode. ACS Basic has three test modes available:

- Single-test mode
- Multitest mode
- Trace mode

Operation toolbar

The Operation toolbar is at the top of the ACS Basic interface and contains a set of functions that allow you to perform the following tasks: New project, Open project, Save project, Save All, Run, Repeat, Append, Pause, Stop, Clear Append, Hardware Management Tool, Scan Hardware, and choose a Mode option. This list of tasks corresponds with the icons in the Operation toolbar from left to right. The Operation toolbar is shown in the following figure.



Project file management

The file management options allow you to create and manage project files in Multitest mode. The following options are available:

- **New:** Creates a new project
- **Open:** Opens an existing project
- **Save:** Saves all project settings of the configuration navigator
- **Save All**: Saves all project settings and test result data

Test control icons

Run, Repeat, Append, Pause, and **Stop** icons.

- **Run:** Initiates the test procedure
- **Repeat:** Continues test execution until Stop is selected
- **Append:** Initiates a test procedure and appends data in a new Data tab
- **Pause:** Allows you to temporarily stop the test flow; the test pauses right after the present operation is completed; the next test does not execute in MultiMode
- **Stop:** Immediately ends the test

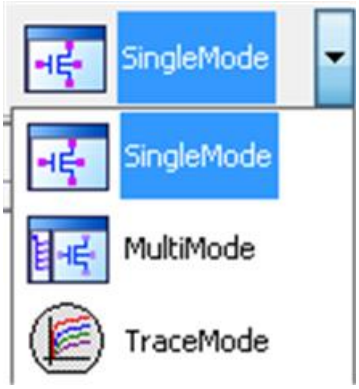
These icons are shown in the following figure.

Figure 10: Run, Repeat, Append, Pause, and Stop functions

MultiMode, SingleMode, and TraceMode

These functions switch between the test modes. Select the test mode from the list, as shown in the following figure.

Figure 11: MultiMode, SingleMode, and TraceMode



- **SingleMode:** Allows you to perform a single test on a single device.
- **MultiMode:** Allows you to perform multiple tests on a single device (target application) or multiple tests on multiple devices. It is easier to use external instruments, such as DMMs and switches, in MultiMode.
- **TraceMode:** Allows you to perform interactive operations with a SourceMeter. You can adjust the maximum sweep value and maximum power into the device. You can also easily transition between DC and pulse forcing modes.

CAUTION

If you switch between SingleMode and MultiMode, your test settings and data may be lost. Save your test data before you switch modes.

SingleMode icons

Select the Hardware Management Tool icon to configure the hardware.

The Scan hardware icon is enabled when the Hardware Management Tool GUI is open. This function scans the hardware and updates the test hardware information.

The following functions are active only in the Single-test mode:

- **Return to Library view:** Opens a test module in the device library
- **Export Test:** Saves a test module to the device library
- **Device View:** Opens a GUI that contains the device setup and connection information
- **Test View:** Opens a GUI that contains the test configuration information

NOTE

The Device view and Test view functions do not appear simultaneously. For example, when in Device view, the Test view function opens; when in Test View the Device View function opens.

ACS Basic GUI icons

The test setup icons are:

- **Return to Library View:** Returns to the GUI to add a test module from the library
- **Export Test:** Exports the present test module to the library
- **Add new Device/Test Module:** Adds a device or an ITM, STM, or PTM test module
- **Launch Script Editor:** Accesses the editing tool so you can edit the library

Log message window

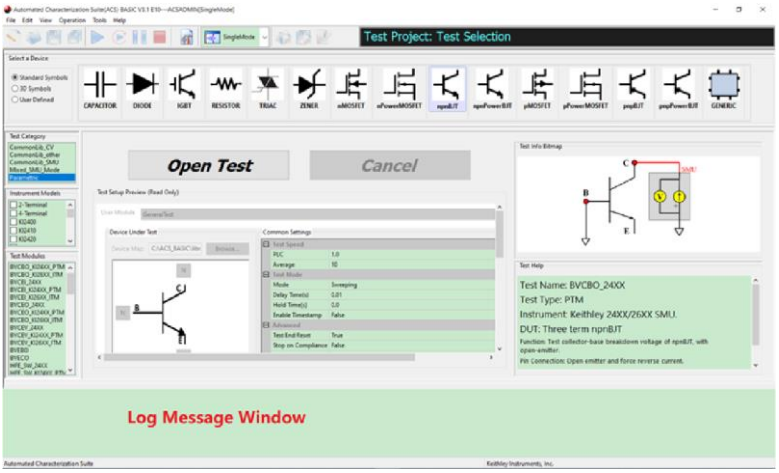
When you run a Python test module (PTM) or script test module (STM), ACS Basic logs the commands, errors, warning messages, and some test results in the log message window.

NOTE

The log window is not used for interactive test modules (ITMs). That is, the ITM test does not display in the log message window.

The log message window is at the bottom of the ACS Basic GUI (see the following figure). The log message displays in the log window in real time.

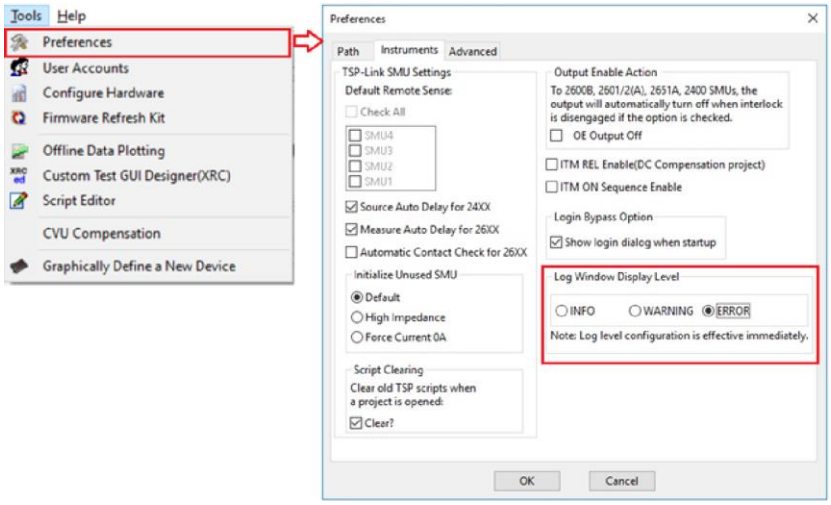
Figure 12: Log message window



In the View menu, you can choose to show or hide the log window. By default, it is hidden.

You can also select the log level to display in the log message window: **Info**, **Warning**, or **Error** in Tools > Preferences settings (see the following figure).

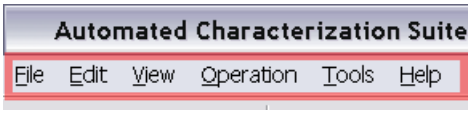
Figure 13: Preference setting for Log Level



Menu toolbar elements

The menu toolbar is at the top of the ACS Basic GUI (see the following figure).

Figure 14: Menu toolbar location



File menu

The **New**, **Open**, **Save**, **Save as Template**, and **Save All** functions are the same as previously described. Refer to Project file management icons (on page 2-10). These functions are only enabled in multimode. Only the EXIT button can be used in single-mode.

Edit menu

The **Copy**, **Cut**, and **Delete** functions are the same as previously described. Refer to Project file management icons (on page 2-10) for more information. Also, the Edit menu is disabled in Single-mode.

View menu

The **Show Log Window** option allows you to display or hide the log.

The other options cannot be accessed in ACS Basic.

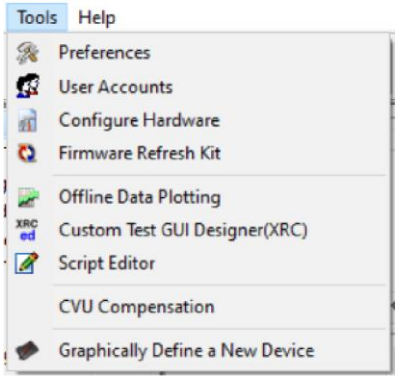
Operation menu

The **Run**, **Repeat**, **Append**, **Pause**, **Stop**, and **Clear Append** functions are the same as the toolbar options described in Test control icons (on page 2-10). Repeat, Pause, and Stop are disabled when SingleMode is selected.

Tools menu

There are several functions available in the Tools menu (see the following figure).

Figure 15: Tools menu



Preferences: Select Preferences to open the advanced Preferences dialog. Refer to Setting Preferences (on page 4-53) for details.

User Accounts: Select to open to see user accounts information (see User accounts (on page 2-4) for details).

Configure Hardware: Access the Hardware Management Tool utility. Refer to Hardware Management Tool for details.

Firmware Refresh Kit: Refresh the Series 2600 or Series 2600B firmware. This utility verifies or performs updates on the firmware on any Series 2600 and 2600B instruments connected to ACS Basic. Any updates will be made using firmware files stored locally on your computer or network. See Firmware Refresh Kit option (on page 3-34) for more detail.

NOTE

The latest firmware files are available for download on tek.com/keithley.

Offline Data Plotting: Access the Data Plot Tool. Refer to Data plot a graph sheet (on page 4-23) for details.

Custom Test GUI Designer (XRC): Access the GUI builder. Refer to Create a XRC GUI file (on page 5-40) for details.

Script Editor: Access the Script Editor Tool. Refer to the Script Editor GUI **(on page 6-2)** for details.

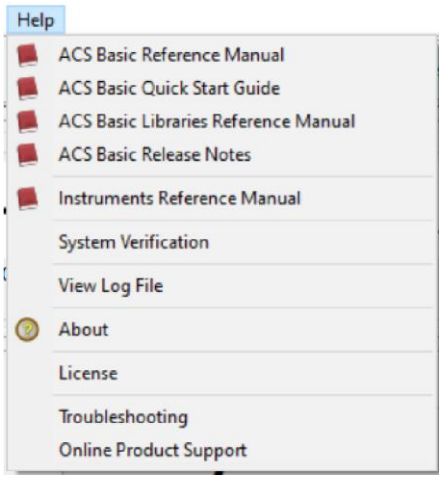
CVU Compensation: Access the CVU Compensation Tool. For more information, refer to CVU Connection Compensation (on page 4-57).

Graphically Define a New Device: Access the Device Editor tool. For more information on the Device Editor, refer to Graphically define a new device (on page 4-63).

Help menu

The options in the Help menu are shown in the following figure and described in the list following the figure.

Figure 16: Help menu



ACS Basic Reference Manual: Provides detailed comprehensive information about the software features.

ACS Basic Quick Start Guide: Contains information regarding how to install the software, install a license, and connect hardware.

ACS Basic Libraries Reference Manual: Contains LPT library programming commands and device library commands.

ACS Basic Release Notes: Contains information about known issues and provides guidance on how to handle issues that you may encounter.

Instruments Reference Manual: Select to find a list of manuals that you can use with ACS Basic supported instruments, including Series 2400, 2600B, 3700A, Model 4200-SCS, and Models 707A/707B and 708A/708B.

System Verification: Select to open the verification project in multiple-test mode.

View Log File: Select to open the saved log file.

About: Select to learn more about ACS Basic, such as the build date and type, plus other general information.

License: Select to receive a dialog box that gives detailed information on how to request a license.

Troubleshooting: Shows system information, such as hardware and software information, including information about your computer and your version of ACS Basic. You can save this information if you need to request a license, or send it to a Keithley Instruments applications engineer for troubleshooting.

Online Product Support: Select to open the online product support website at tek.com/support, where you can find information on Keithley products.

Section 3

ACS Basic hardware configuration

In this section:

Hardware configuration introduction

The Hardware Management Tool is used to manage the hardware configuration of Keithley instruments . Refer to tek.com/keithley for the list of supported instruments.

All instruments are supported with standard GPIB, LAN, USB, and serial port communications connections. They are also available in mixed applications. Additionally, GPIB, LAN, and USB are available for primary communications connections with ITM, PTM, and STM. The default communications for all testing is LAN. RS-232 is also an option.

Using the hardware management tool, these instruments are easily added, configured, and removed from the system configuration.

If you make changes to the hardware (for example, remove one instrument, change the 707B work mode or change node information, you need to use the Hardware Management Tool to scan the hardware.

PTM-only instruments

The categories for PTM-only instruments are:

- Switch Matrix
- Capacitance Meter
- General-Purpose Test Instrument

All supported PTM-only instrumentation and equipment is controlled by Python Test Modules (PTM); however, they are not compatible with STM or ITM.

The software provides Python LPT libraries for each supported external instrument. You can use these libraries to create your own test modules.

The software also provides several standard Python user libraries to control external equipment that is typically used in semiconductor characterization applications. Standard user-module libraries are provided for the equipment (see the following table).

Supported PTM-only table

Category	Instrument	LPT Library[1]	User Library[2]
Switch matrix[3]	Model 707A or 707B Switching Matrix (DDC)	ki70xlpt	switchctrl[4]
	Model 708A or 708B Switching Matrix (DDC)		
Capacitance meter[5]	Model 590 CV Analyzer	ki4200cvulpt	CVITM[6]
	Model 595 CV Analyzer		
General-purpose test instrument[7]	Any IEEE-488 controlled instrument or equipment	created by user	TEKSCOPE[8]
			HiPower_24[9]

[1] The LPT library is saved in the directory `\\ACS BASIC\library\pyLibrary\ptmlpt`. For more information about the ACS LPT library functions, refer to "Python LPT Library" in the *Automated Characterization Suite* (*ACS) Basic Edition Libraries Reference Manual* (document number ACSBASIC-908-01).

[2] The User library saved in the directory `\\ACS BASIC\library\pyLibrary\PTMLib`. For more about the PTMLib library, you can refer to the Python User Library. To learn more about how to import a PTM, refer to "PTM with the .XRC GUI File" in the *Automated Characterization Suite (ACS) Fundamental Reference Manual* (document number ACS-914-01).

[3] ACS supports the Keithley Instruments Model 707A, 707B, 708A, and 708B Switch Matrices. The Models 708A and 708B accept a single matrix card. The Models 70A7 and 707B accept up to six matrix cards. Only one switch matrix can be present in the system configuration at a time.

[4] switchctrl is saved in `\\ACS BASIC\library\pyLibrary\PTMLib\switchctrl`. To learn more about how to use the switchctrl library, refer to "Switch_Control" in the *Automated Characterization Suite* (*ACS) Basic Edition Libraries Reference Manual* (document number ACSBASIC-908-01)

[5] Up to four supported capacitance meters may be added to system configuration.

[6] The CVITM is saved in `\\ACS BASIC\library\pyLibrary\PTMLib\CVITM`. To learn more about this library, refer in the *Automated Characterization Suite* (*ACS) Basic Edition Libraries Reference Manual* (document number ACSBASIC-908-01).

[7] ACS supports up to sixteen general purpose test instruments (GPI). Two-terminal and four-terminal types may be present in the system configuration simultaneously, but the total number of GPI cannot exceed sixteen.

[8] The TEKSCOPE is saved in `\\ACS BASIC\library\pyLibrary\PTMLib\TEKSCOPE`. To learn more about how to use this library, refer to "Common other library" in the *Automated Characterization Suite* (*ACS) Basic Edition Libraries Reference Manual* (document number ACSBASIC-908-01).

[9] The HiPower_24 is saved in `\\ACS BASIC\library\pyLibrary\PTMLib\HiPower_24`.

Hardware Management Tool

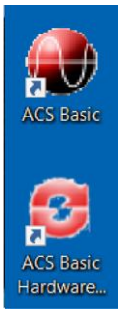
When the software is installed on your computer, the Hardware Management Tool (HMT) is installed at the same time. You see the HMT icon on your desktop (see the following figure) and the software icon. The HMT manages the hardware of your test configuration.

Select the HMT icon to scan your hardware and generate a `Hardware_Config_Online.kcf` (Keithley configuration file).

NOTE

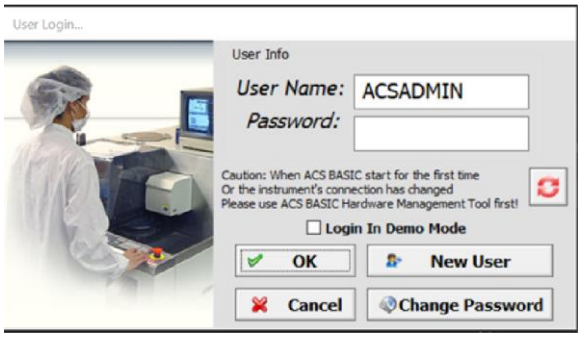
Before you use the software, you must scan the hardware of your test configuration. If you do not scan the hardware you will see a warning indicating the hardware configuration file does not exist.

Figure 17: Hardware Management Tool icon



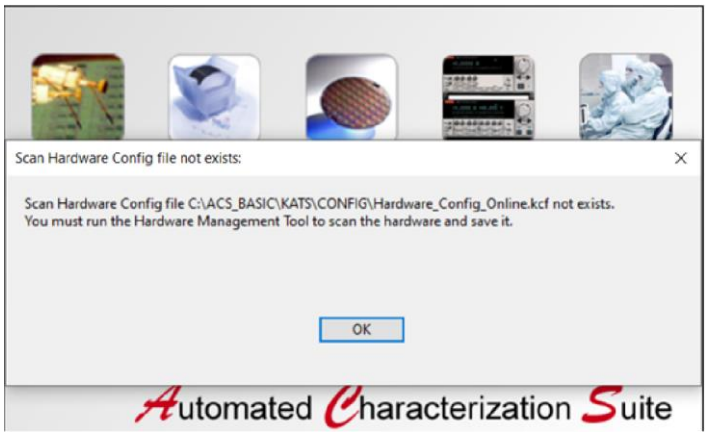
On the User Login interface, you may see a caution message. This may indicate that instrument connections have changed or that you are logging in for the first time to ACS Basic software (see the following figure).

Figure 18: Login



If the hardware has not been scanned with Hardware Management Tool before starting ACS Basic, a fatal error message will display, as shown in the following figure.

Figure 19: Fatal error rescan



Scan for instruments

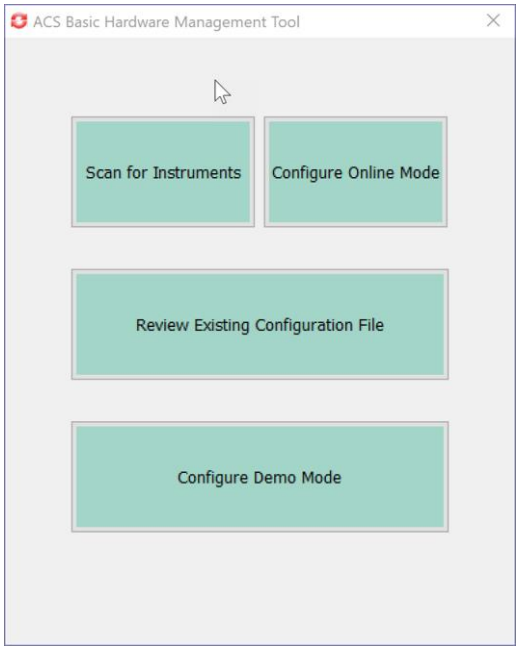
The following describes how to scan the hardware configuration.

When you open the Hardware Management Tool, the interface contains four options. You can scan the connected instruments, configure online mode, review the existing configuration file, or configure a demo mode test.

To scan for instruments:

1. Select **Scan for instruments**.

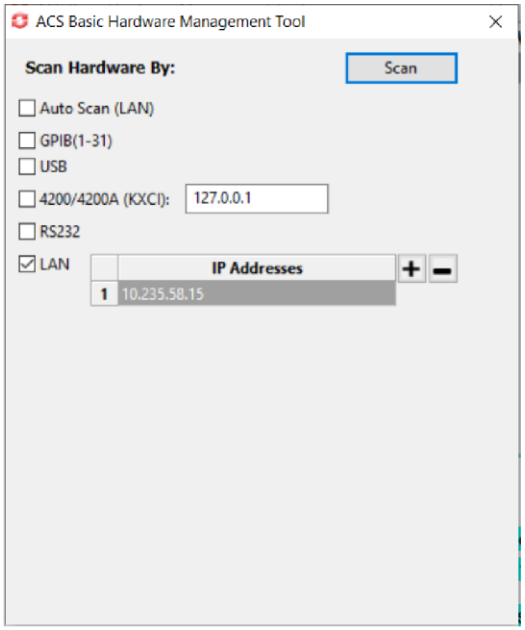
Figure 20: Hardware Management Tool options



2. Select the options for your test based on the instrument connection, as shown in the following figure.
   - Auto Scan (LAN): instruments need to be on the same subnet as the system computer (this means the same first three fields of the IP address)
   - GPIB: instrument address from 1 to 30 is supported
   - Prober GPIB Address: default value is 31
   - USB
   - 4200/4200A (KXCI)
   - IP Address: add or delete an IP address by selecting the plus or minus buttons, and configure the IP address of 4200A-SCS connected LAN
   - RS-232
   - LAN

When you have selected your options, select **Scan**, as shown in the following figure. For a list of supported instruments, refer to ACS Basic hardware configuration introduction (on page 3-1).

Figure 21: Hardware Management Tool



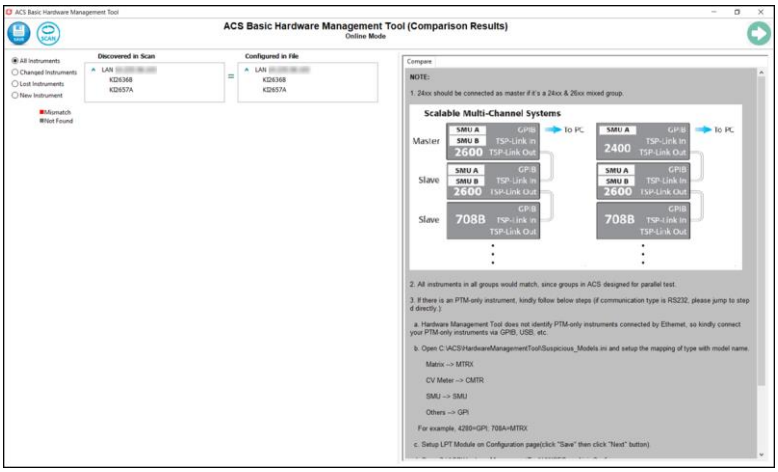
The system queries the instruments and the updated configuration information is shown in the comparison results. See the Comparison Results (Online Mode) (on page 3-6) for detail.

Comparison Results (Online Mode)

Before you can compare results, you must scan the instruments connected to your system for testing. Refer to Scan for instruments (on page 3-4) for more detail.

The comparison results, as shown in the following figure, shows the instruments connected to your system from the scan. See "Discovered in Scan" and "Configured in File" in the following figure. The scan compares the present online instrument configuration to the instruments that were previously configured. You can choose to apply the newly scanned instruments and add to the online configuration file by selecting **Save**.

Figure 22: Comparison results



You can delete instruments that are found during the scan, as shown in the following figure. Right-click to delete. Select **Save** to keep your configuration.

Figure 23: Delete in online mode

When you save your configuration, the Configuration Page (Online Mode) opens. See the Configuration Page (Online Mode) (on page 3-7) for detail.

Configuration page (Online Mode)

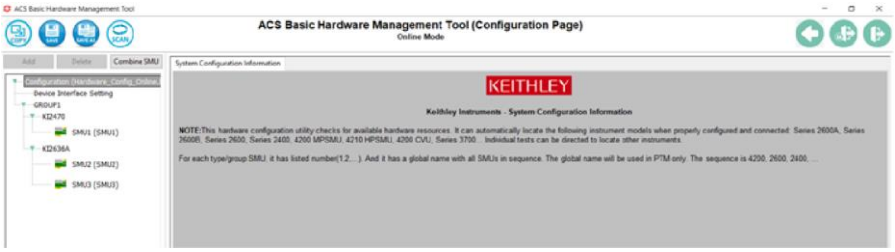
The configuration page shows the updated instrument configuration, as shown in the following figure. You can add an instrument in the configuration page by selecting **Add**.

Make sure you have the instrument connected to the test system to add it. Select an instrument, then select **Delete** to remove an instrument.

Also, to combine multiple SMU instruments to your test configuration, select the SMU instruments and select the **Combine SMU** option.

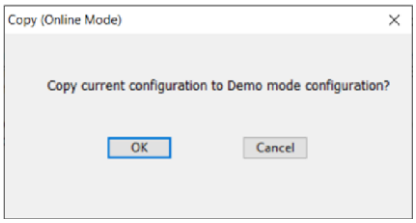
You also have the option to go back to the comparison results window, save and exit, or close the window using the arrows on the toolbar.

Figure 24: Configuration Page Online Mode



Select copy in the toolbar to save your online configuration to the online mode configuration. Select **OK** to save the configuration, as shown in the following figure.

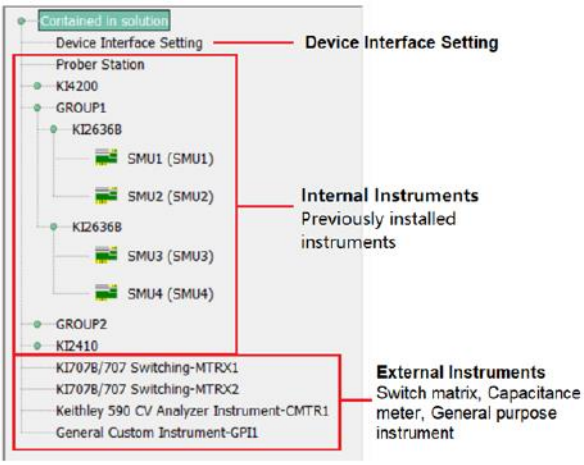
Figure 25: ACS Copy (Online Mode)



Configuration navigator

The configuration navigator displays a tree view that shows each component present in the system configuration. Selecting a component, or node, in the configuration navigator displays the properties associated with the selected component in the work area. The following figure shows a typical system configuration with the configuration navigator partially expanded.

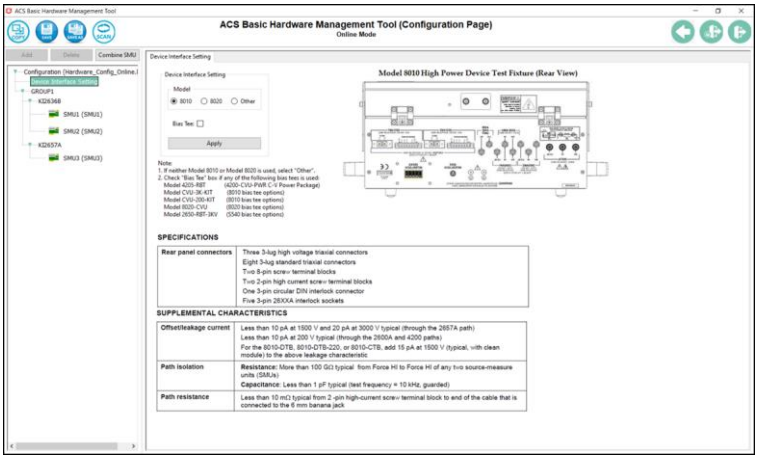
Figure 26: Configuration navigator view of typical system



Device interface setting

The Hardware Management Tool uses the device interface setting to manage the hardware configuration to optimize the default instrument settings (see the following figure). Select the Device Interface Setting, as shown in the following graphic, to see the device settings.

Figure 27: Device Interface Setting



In the Device Interface Setting GUI, you must choose the Model 8010 High Power Test Fixture, Model 8020 High Power Interface Panel, or Other device with or without a bias tee. See the table below for configurations that also include default settings.

To apply the device interface setting:

1. Select the model in your device interface. If you have a Model 8010 or Model 8020, select accordingly, or select Other.
2. Select the bias tee if you are using any of the following bias tees: Model CVU-3K-Kit, Model CVU-200-KIT, Model 8020-CVU, or Model 4205-RBT.

NOTE

To use the bias tees with the CV instrument in the Model 4200A-SCS and access the high voltage capacitance measurement modules, ACS Basic software must be installed on the Model 4200A (there is no remote support). If you do not need a bias tee and want to use the Model 4210-CVU alone, you should use the CVITM test module. The CVITM test module requires the Model 4200A-SCS KXCI software which can be used remotely if ACS software is installed on the remote computer.

3. Select **Apply**.

NOTE

If you have a bias tee in your configuration, you must select the bias tee option in the device interface setting to access the appropriate CVU connection compensation. For more information, refer to CVU connection compensation (on page 4-57).

The table below details how the device interface setting changes the default instrument settings:

Model	With bias tee option?	Delay factor	Contact check	Measurement speed
8010	No	1	On	Normal
8010	Yes (applies to CVU-3K-KIT, CVU-200-KIT, or 4205-RBT)	2	On	Slow
8020	No	1	Off	Normal
8020	Yes (applies to 8020-CVU bias tee)	2	Off	Slow
Other	No	1	On	Normal
Other	Yes (applies to CVU-3K-KIT, CVU-200-KIT, or 4205-RBT)	2	On	Slow

The delay factor is a multiplier to the Auto Delay and is applied when Auto Delay is enabled. Auto Delay is an instrument-specific delay associated with the SMU measurement range. Auto Delay and Contact Check can be enabled in the Instrument tab settings, which are in Tools > Preferences.

If you make changes in this tab after configuring device interface setting, you will override values presented in the above table. Measurement speed is controlled in the Timing window of ITMs. Refer to Model 42xx SMU ITM (on page 5-26) for more details.

If you make changes to timing after configuring device interface setting, you will override the measurement speed settings in the above table.

Open ACS Basic and confirm hardware

The Hardware Management Tool scans information for the instruments using ACS Basic and modifies configurable properties.

NOTE

Before you use ACS Basic software, you must scan the hardware of your test configuration. If you do not scan the hardware, you see a warning indicating the hardware configuration file does not exist. Refer to Hardware Management Tool for more detail.

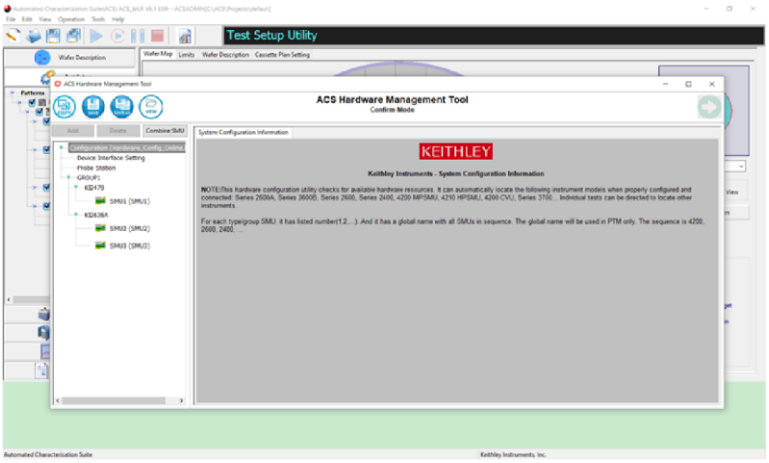
Select the Configure Hardware icon on the toolbar.

Figure 28: Start hardware configuration

The hardware management dialog opens, shown in the following figure. The interface of the hardware configuration consists of:

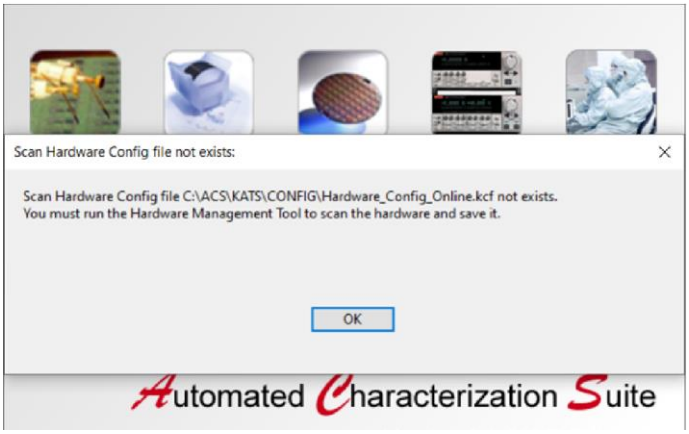
- **The configuration navigator:** The left panel, which displays the instruments and equipment that are included in the ACS Basic system configuration.
- **The work area:** The right panel, where the instrument and equipment properties are displayed.

Figure 29: Hardware Management Tool interface



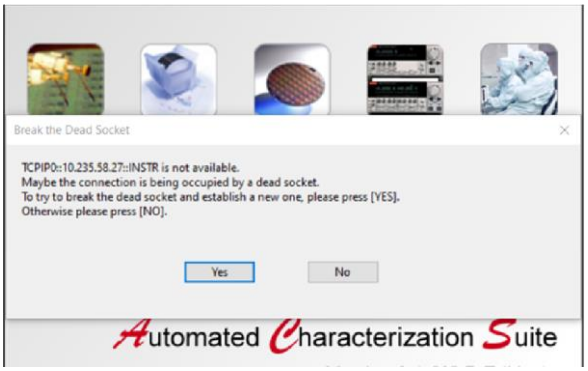
If you encounter an error message when opening ACS Basic, the hardware configuration may have changed. See the following figure for an example. You need to run the Hardware Management Tool to establish the hardware configuration.

Figure 30: Hardware Configuration File does not exist



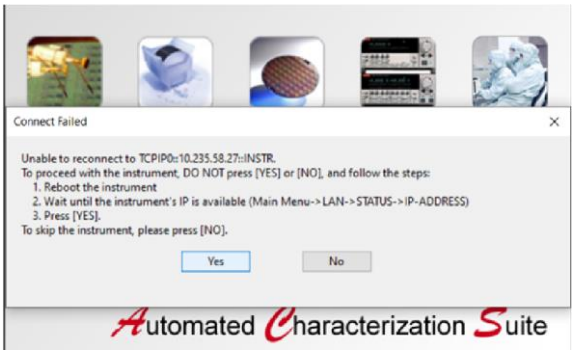
It is possible you may encounter a TCPIP error, as shown in the following figure. You can select **Yes** to continue to confirm your hardware, or select **No** to stop.

Figure 31: Break the dead socket



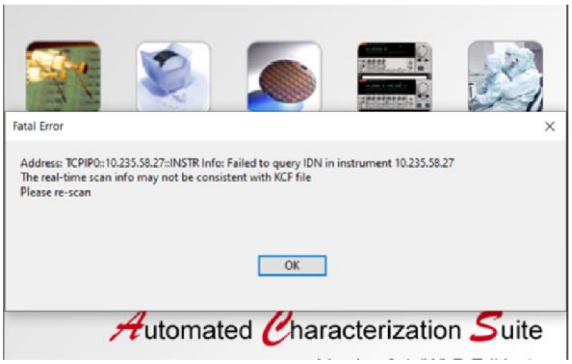
If you select **Yes** and the software cannot connect to the instrument, you are notified, as shown in the following figure.

Figure 32: Connection failed



If the TCPIP fatal error continues, refer to the following figure that indicates you should scan the instruments using the Hardware Management Tool.

Figure 33: Fatal error continues

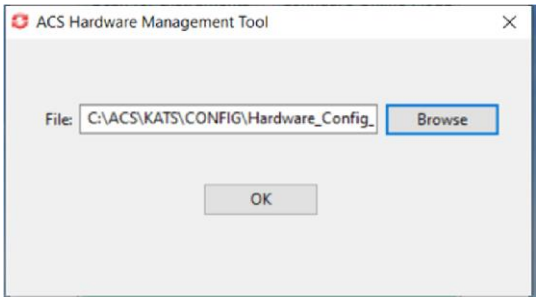


Configuration Page (Confirm Mode)

Open the Hardware Management Tool interface to access the existing configuration file. Choose **View** to select **Confirm Mode**. In confirm mode, you can select a `.kcf` file and view the instruments configuration saved in the file, as shown in the next figure.

- Select **Browse** to select the `.kcf` file.
- Select **OK** to enter the confirm mode page.

Figure 34: Review existing configuration file



The Configuration Page shows the updated instruments configuration.

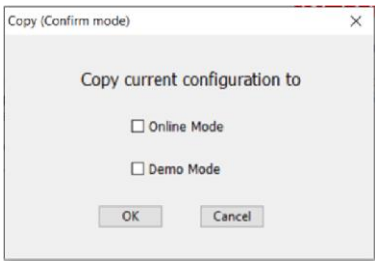
The left panel is the configuration navigator where all the instruments and equipment that are included in the ACS system configuration are displayed. The right panel is the Work Area where all the instruments and equipment properties are displayed.

- Select **Add** to open the external instrument window.
- Select **Delete** to delete an external instrument.

In the external instrument window, you can select and configure external instruments to add to the configuration, including Switch Matrix, general-purpose instrument, and a capacitance meter, (for details refer to Add and delete a PTM-only instrument (on page 3-13)).

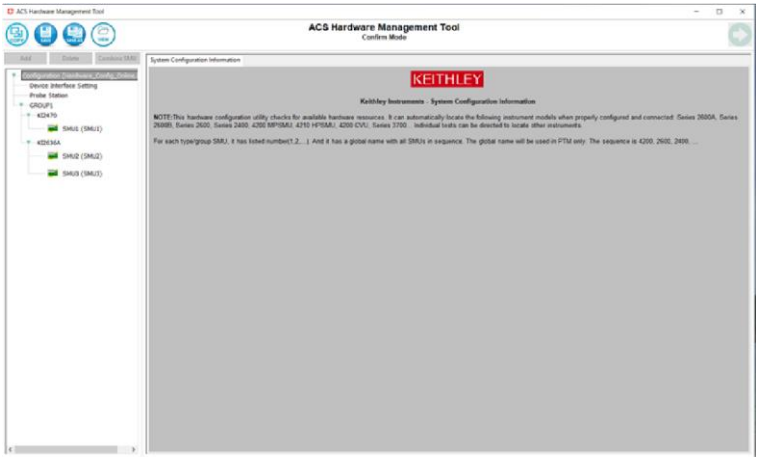
- Select **COPY** to copy the current configuration, either online or demo mode.

Figure 35: Configuration Page (Confirm Mode)



- Select **SAVE** to save the instrument configuration to the demo configuration file.
- Select **SAVE AS** to save the instruments configuration to a selected `.kcf` file.
- Select **VIEW** to review the existing configuration file.

Figure 36: Confirm mode



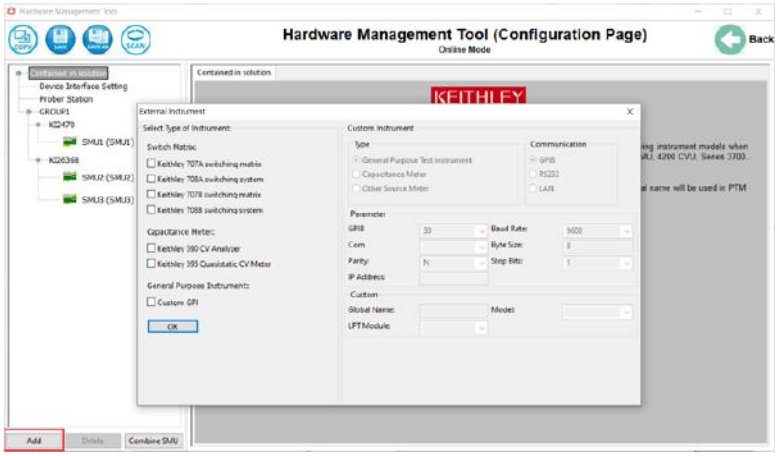
Add and delete a PTM-only instrument

A PTM-only instrument can be manually added to your hardware configuration through the Hardware Management Tool demo mode. Refer to PTM-only instruments (on page 3-2) for more detail.

To add a PTM-only instrument:

1. In the configuration navigator, select **Add** to open the external instrument dialog, as shown in the following figure.

Figure 37: Add or delete external instruments



2. In the PTM-only Instrument dialog, select the type of instrument to add.
3. Configure the parameters for a general-purpose instrument in the Custom section.

To delete a PTM-only instrument:

1. Select the PTM-only instrument.
2. Select **Delete**.
3. Select **Yes** to confirm this action.

The following table lists the remaining types of instruments you can add.

Switch matrix	
Model 707A/707B switching matrix (DDC)	Model 708A/708B switching system (DDC)
Capacitance meter	
Model 590 CV Analyzer	Model 595 Quasistatic CV Meter
General-purpose instrument	
Custom GPI	

To add a switch matrix, capacitance meter, or general-purpose instrument:

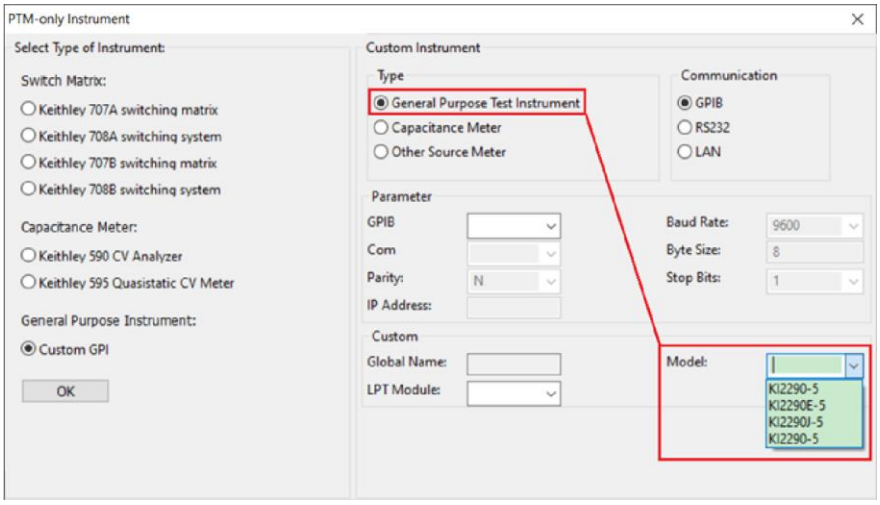
1. Select **Add** in the configuration navigator to open the external instrument dialog, as shown in the following figure.
2. Select the PTM-only instrument that you want to add.

When you choose the custom GPI, you need to set the type of General Purpose Test Instrument and the parameters, as shown in the following figure. For more information about the Keithley Instruments Model 2290 Power Supply, refer to Power supply library.

NOTE

The LPT Model name must be the same as the Python file name saved in the `ptmlpt` folder (`\\ACS_BASIC\library\pyLibrary\ptmlpt`). ACS Basic supplies these LPT library packages: `ki23xlpt`, `ki24xxlpt`, `ki26xxlpt`, `ki37xxlpt`, `ki42cvulpt`, `ki42xxlpt`, `ki70xlpt`, `kiag4284lpt`, and `ki20xxlpt`.

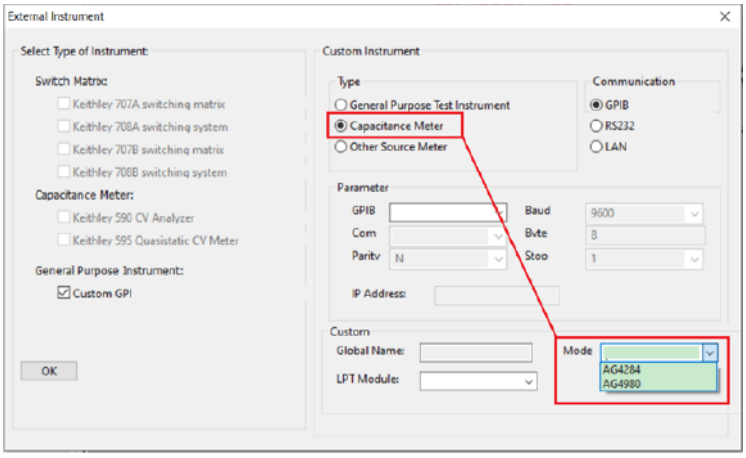
Figure 38: General Purpose Test Instruments



When you choose capacitance meter type, you have two options:

- AG4284 (Keysight)
- AG4980 (Keysight)

Figure 39: Capacitance Meter Instruments



When you choose Other Source Meter, you have the options:

- KI236
- KI237
- KI238

Configure Demo mode

Demo mode allows you to manage and simulate a virtual hardware system without having actual instruments connected to the system computer.

You can configure the instruments for the demo system in the Hardware Config (Demo) panel by selecting the Parametric Curve Tracer Demo System or the Customized Demo System, as shown in the following figure.

Select the menu to choose an instrument:

- Parametric Curve Tracer (with or without PCT-CVU)
- Series 2400 SourceMeter®
- Series 2600B SourceMeter®
- Series 3700A System Switch/Multimeter
- Models 707B, 708B Switching Matrix
- Model 4200A-SCS Parameter Analyzer
- Model 4200-SCS Semiconductor Characterization Suite
- Model DMM7510 7½-Digit Graphical Sampling Multimeter

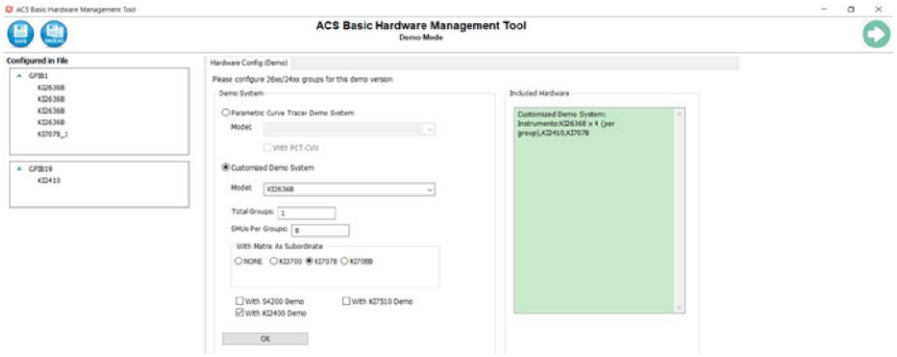
NOTE

The Series 3400 cannot be simulated in the scan in demo mode.

For the Customized Demo System, you can configure customized instruments for a demo system. Select the model, total group, and SMUs per group. The Configured in File indicates the instruments in the demo system, as shown in the following figure.

Make sure you save the demo configuration file. You can also create a `.kcf` file by selecting **Save As**.

Figure 40: Hardware Management Tool Demo mode



To simulate a scan for the Series 2600B

1. Select the **Customized Demo System** option and select your specific 2600B in the model list.
2. Input the total groups and SMUs per Group.
3. Select **OK**.

The 2600B information is displayed in the configuration navigator with corresponding properties information displayed in the work area.

To simulate a scan of the Series 3700A and Model 707B, 708B instruments:

1. Select the **Customized Demo System** option and select your specific **With Matrix as Subordinate** instrument. For more information about the 3700A, 707B, and 708B used as a subordinate or child, refer to Using TSP-Link (on page 3-23) for more detail.
2. Select **OK**.

Depending on your selection, the 3700, 707B, or 708B information is displayed in the configuration navigator with corresponding properties information displayed in the work area.

To simulate a scan of the Model 4200A-SCS Parametric Analyzer or 4200A-SCS:

1. Select the **Customized Demo System** option and select **With S4200 Demo**.
2. Select **OK**.

Configuration Page (Demo Mode)

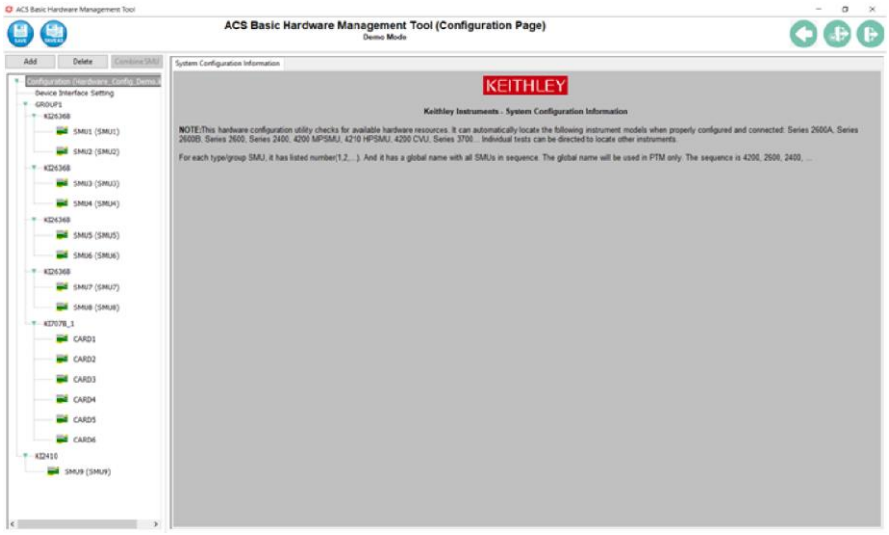
The Configuration Page shows the updated instruments configuration. The left panel is the configuration navigator. All the instruments and equipment that are included in the system configuration are displayed here. The right panel is the work area. The instruments and equipment properties are viewed and modified here.

- Select **Add** to open the external instrument window.
- Select **Delete** to delete an external instrument.

In the external instrument window, you can select and configure external instruments to add to the configuration, including Switch Matrix, General Purpose Instrument, Capacitance Meter.

- Select **SAVE** to save the instrument configuration to the demo configuration file.
- Select **SAVE AS** to save the instruments configuration to a selected `.kcf` file.
- Select **SCAN** to rescan the hardware.
- Select **Back** to go back to Hardware Config (Demo) view.

Figure 41: Configuration Page Demo Mode



Choose **View** to select **Confirm Mode**. In Confirm Mode, you can select a `.kcf` file and view the instruments configuration saved in the file. You can also add external instruments into the configuration and save the configuration to a `.kcf` file.

- Select **File** to browse to the `.kcf` file.
- Select **OK** to enter the Confirm Mode page.

The configuration page shows the updated instruments configuration.

Connect to a single instrument

NOTE

Information about connecting to the Model 4200A-SCS is not in this section. Refer to Connect to Model 4200A-SCS (on page 3-28) for more information.

When using ACS Basic to control hardware, you must select and configure a communications interface. If the hardware is configured with a communications interface, when ACS Basic starts it automatically scans and finds the hardware.

Select an interface

Make sure you choose an interface that is consistent with your instrument's connection: GPIB or ethernet. Some instruments allow multiple communications interfaces to be enabled; however, you can only communicate with one interface at a time using ACS Basic. For more information about communications interfaces, refer to "Supported communications interfaces" in the *ACS Basic Quick Start Guide* (part number ACSBASIC-903-01).

NOTE

See the instrument's manual for more information on the GPIB and ethernet communications interfaces.

GPIB interface

ACS Basic automatically scans the GPIB instruments during start.

ACS Basic software supports these GPIB interfaces:

- KUSB-488
- KUSB-488A
- KUSB-488B
- KPCI-488
- KPCI-488A
- KPCI-488LP
- KPCI-488LPA
- NI USB-488
- NI PCI-488

GPIB driver compatibility

On a system on which ACS Basic is already installed, ACS Basic must be reinstalled if the following GPIB interface drivers are uninstalled or reinstalled:

- KITE or Keithley GPIB interface drivers
- NI GPIB interface drivers or NI-VISA drivers
- CEC488 GPIB drivers 8.2 or above

If ACS Basic is not reinstalled, ACS Basic may function incorrectly due to `.dll` errors.

The functionality of the software is not affected by the order of installation of almost all other GPIB interface drivers.

NOTE

ACS Basic supports CEC488 GPIB drivers 8.2 or above. Please upgrade older drivers to ensure compatibility.

Check the GPIB communications

Prior to starting ACS Basic, it is recommended that you verify that the GPIB card and driver are installed correctly and are properly functioning in your operating system. For the Keithley/CEC GPIB interfaces, you can use the following diagnostic utilities:

- KI-488 Configuration Utility
- KI-488 Diagnostic Tool

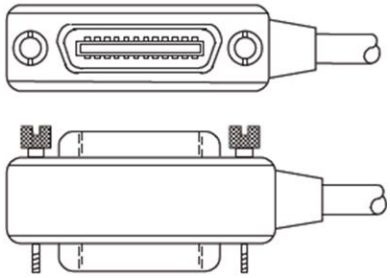
NOTE

For more information on verification instructions on the GPIB communications interface card, refer to the GPIB manufacturer’s documentation.

Connect instruments with GPIB

To connect the instruments to the GPIB bus, use an IEEE-488 GPIB cable (see the following figure).

Figure 42: Standard IEEE-488 connectors



Establish the instrument's GPIB address and make sure that the instruments are configured for a GPIB connection and has a valid, unique address. For example, the Series 2600B ships from the factory with a GPIB primary address of 26. If the GPIB interface is enabled, it momentarily displays the primary address on power-up. You can set the address to a value from 0 to 30, but do not assign the same address to another instrument or to a controller that is on the same GPIB bus.

NOTE

Controller GPIB addresses are usually 0 or 21.

The next figure shows a single instrument connected to a computer over GPIB.

Figure 43: Single instrument GPIB configuration

Connect instruments with ethernet

A computer or Model 4200A-SCS can connect to a variety of devices that supports TCP/IP and complies with IEEE standard 802.3 (ethernet), such as Series 2600B SourceMeter, Series 2400 SourceMeter, or a switching matrix. These instruments can be connected to a computer or Model 4200A-SCS through ethernet or directly using a LAN cable (as shown in the following figure). ACS Basic can access the instruments through the IP address.

The Series 2600B SourceMeter® SMU Instruments and Keithley 2400 SourceMeter® SMU Instruments connect using two separate cables. Use one cable for TSP-Link and use the other cable for the ethernet connection (for a direct instrument to computer connection).

Insert the ethernet cable to the ethernet port on the rear panel of the instrument and the other end into the ethernet port on the back of the host computer.

You need to assign an IP address to the instrument. The IP address of these instruments need to be on the same subnet with the system computer or Model 4200A-SCS.

To manually configure ACS Basic to scan for ethernet instruments, open the Hardware Management Tool (on page 3-3) and set the IP addresses.

Figure 44: Ethernet connection

NOTE

For more information on how to configure an ethernet connection, refer to the documentation for the instrument.

Connect to multiple instruments

ACS Basic supports the following instruments in a daisy chain configuration through a GPIB or ethernet cable in one group using TSP-Link or both:

- Multiple Keithley 2400 Graphical Series SourceMeter® SMU Instruments
- Multiple TSP instruments, such as Series 2600B SourceMeter® SMU Instruments, Series 2650A High Power SourceMeter® Instruments, which includes 2651A and 2657A, Series 3700A System Switch/Multimeter
- Mixed group of Keithley 2400 SourceMeter® SMU Instruments and Series 2600B SourceMeter® SMU Instruments
- Multiple external GPIB instruments

Multiple TSP instruments

Multiple TSP instruments configured in one group or several groups are supported by ACS Basic.

Using TSP-Link

You can use TSP-Link to connect subordinate instruments to the master, while the master instrument is connected to the computer through a GPIB or ethernet cable. ACS Basic treats all instruments connected with TSP-Link to a master as a single group.

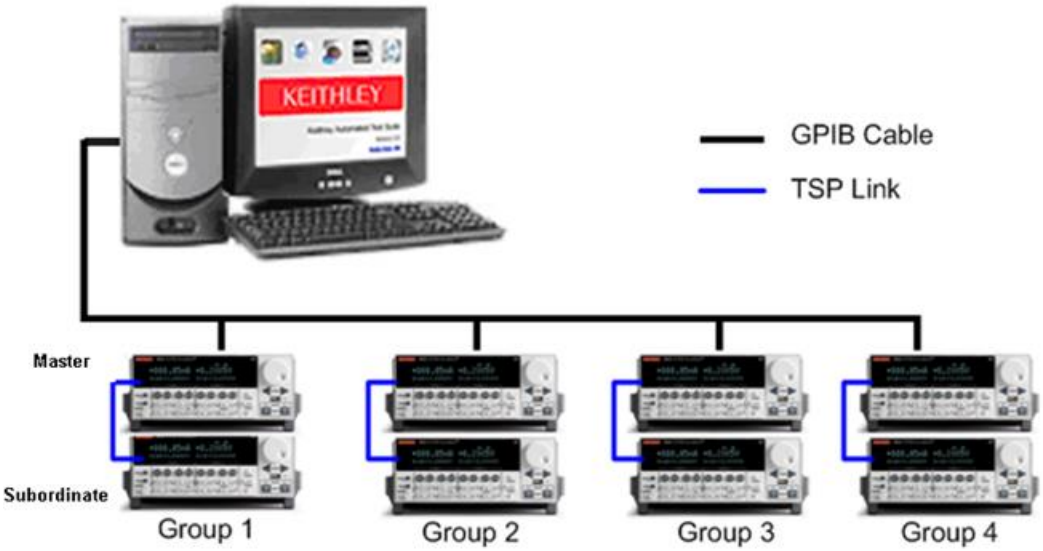
To set up a group of instruments with TSP-Link:

1. Connect one instrument to a GPIB or ethernet interface.
2. Connect another TSP-enabled instrument using TSP-Link and daisy chain the TSP-Link cables between the instruments.
3. Assign a unique node number to each instrument. If using GPIB, the master instrument should be node 1; if using ethernet, the instrument that is connected directly to the computer should be node 1.
4. Open the Hardware Management Tool (on page 3-3) and scan the hardware.

GPIB multiple interfaces

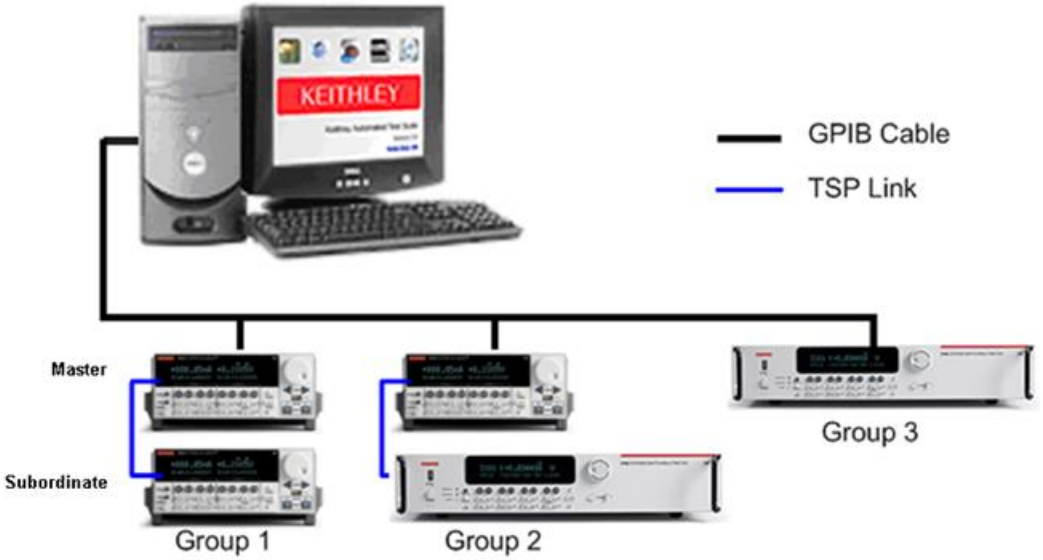
A group is connected through a GPIB cable (see the following figure) from the controller to a Series 2600/2600B instrument which serves as the master node. The other instruments connected with TSP-Link are subordinates of the master.

Figure 45: Multiple Model 2600 SMU connections



ACS Basic supports TSP groups that contain different instruments, as shown the next figure, which depicts the Series 2600B and Series 3700A instruments in multiple groups.

Figure 46: Multiple TSP instrument connections



Ethernet multiple interfaces

If you need to use multiple TSP-Link instrument groups connected by ethernet, you can use an ethernet hub or a switch. Refer to the next two figures for ethernet configuration connections.

- Connect the computer to the ethernet hub or switch.
- Establish a unique IP address for the group master instrument.
- Connect all the instruments to the ethernet hub or switch using ethernet cables.
- Open the Hardware Management Tool and add each IP address to the hardware configuration.

NOTE

For details on an ethernet connection between a computer and master instrument, and the TSP-Link connection between instruments, refer to the manuals for the instruments.

Figure 47: Ethernet configuration #1

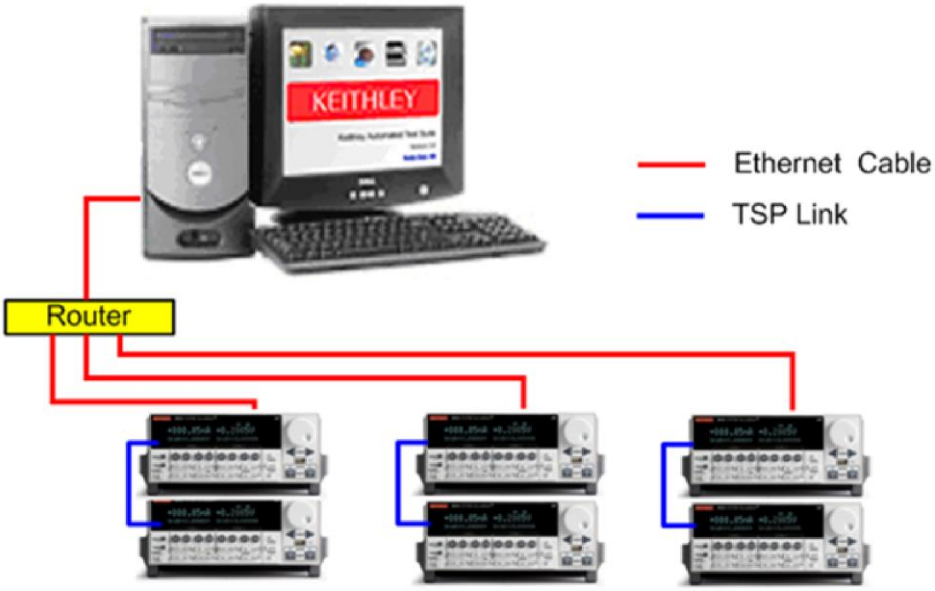
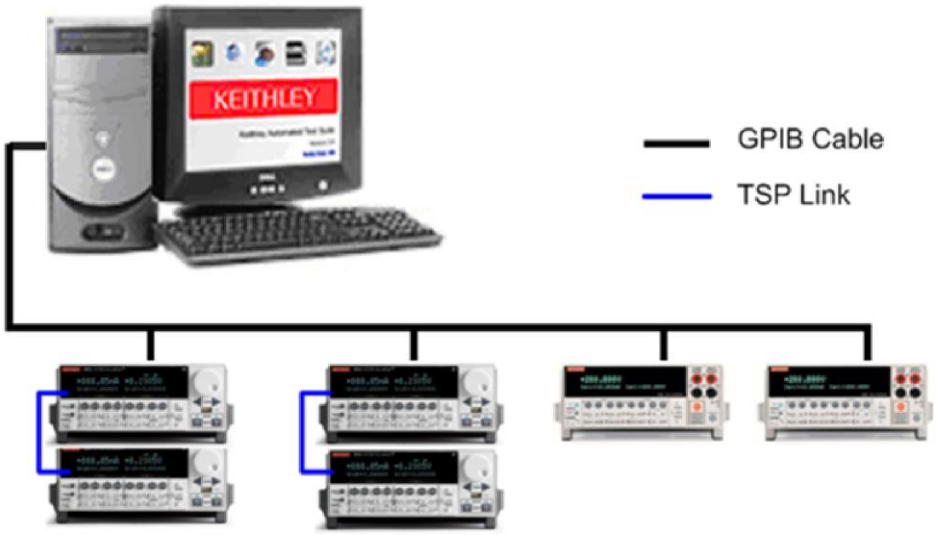


Figure 48: Ethernet configuration #2

The next figure shows ACS Basic software installed on a computer and connected by GPIB cables to Series 2600B and Series 2400 instruments.

Figure 49: Instrument connections configuration



Connect multiple 2400 Graphical Series SMU Instruments

You can connect multiple Keithley 2400 Graphical Series SourceMeter® SMU Instruments using a GPIB or ethernet cable.

To connect multiple SMUs:

1. Set a unique GPIB or IP address for each instrument.
2. Connect the GPIB or ethernet cables to each instrument and daisy chain the connectors.
3. Open the Hardware Management Tool to configure the instruments and set the instrument properties Rear Jack and Beeper.

NOTE

For details on how to configure the GPIB address of the Series 2400 Graphical Series SourceMeter instruments, refer to the Series 2400 SourceMeter manual documentation.

Figure 50: Multiple Model 2400 SMU connections

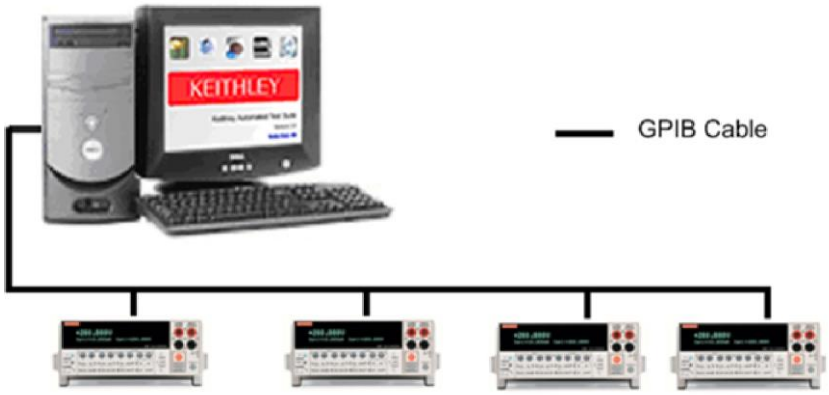
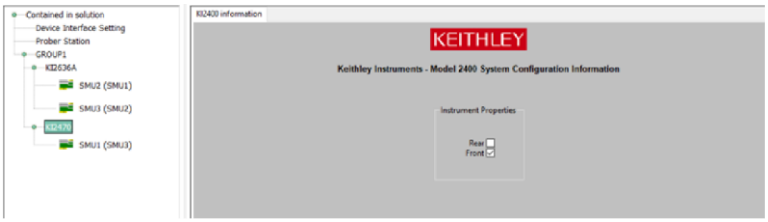


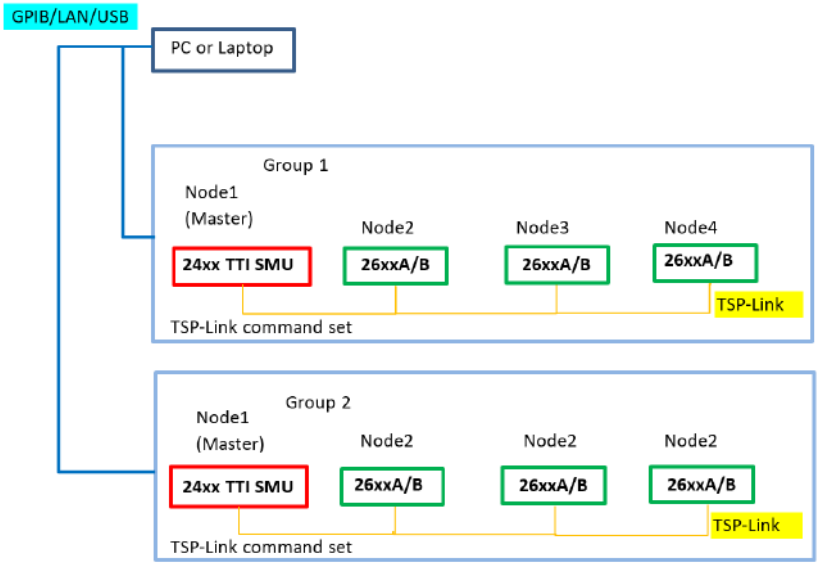
Figure 51: Model 2400 configuration GUI



Connect a mix of 2400 Graphical Series and 2600B SMUs

You can connect a mixed group of Series 2400 Graphical Series and 2600B SMU instruments with a GPIB or ethernet cable and TSP-Link interface, as shown in the following figure.

Figure 52: Test system with a mix of 26xxB and 24xx Graphical Series SMUs



Connect to a Model 4200A-SCS

ACS Basic automatically scans the Model 4200A-SCS hardware if ACS Basic is installed on the Model 4200A and the mode of communications is configured for ethernet (see the following figure). ACS Basic can control all the hardware that is supported by the Model 4200A-SCS.

Model 4200A-SCS and KCon

Connect the 4200A-SCS to the ACS Basic computer through a network connection. Record the IP address of the Model 4200A-SCS for reference.

NOTE

Before configuring a Model 4200A ITM, make sure that the ethernet link is working. If you are using ACS Basic on a local Model 4200A-SCS using the default 4200A IP address, your hardware configuration is complete.

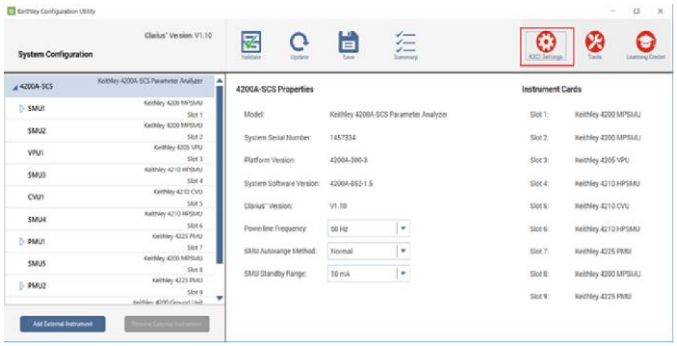
To set the Model 4200A-SCS communications mode using the Keithley Configuration Utility (KCon) software:

1. Open the KCon application on the Model 4200A-SCS.
2. Set the communications mode to ethernet.
3. Verify the communications mode by opening KCon and open the KXCI Settings tab (see the following figure). Also, the communications mode is visible in the KXCI message area.

NOTE

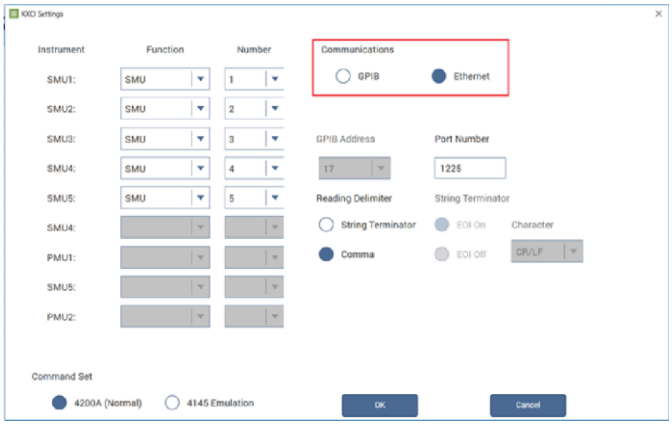
The Model 4200 can only be used in ethernet mode with ACS Basic.

Figure 53: Set the Model 4200A-SCS IP address



4. Set up the IP address of the Model 4200A-SCS in the Hardware Management Tool.

Figure 54: Set the Model 4200A-SCS communications mode



Model 4200A System properties

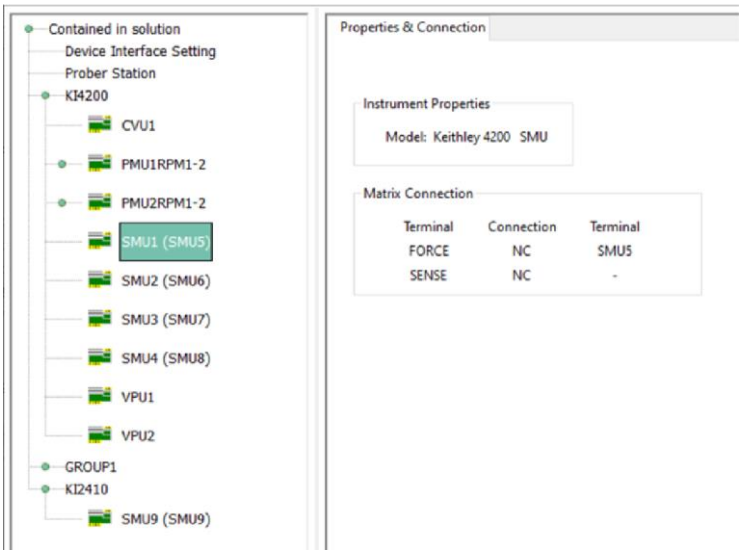
Select the KI 4200A-SCS icon in the configuration navigator and the properties are displayed in the work area (see the following figure).

Figure 55: Model 4200 group properties

Select the KI 4200 SMU node in the configuration navigator and the Model 4200-SCS SMU system are displayed in the work area (see the following figure).

The work area shows the Model 4200 SMU where you can see the Instrument Properties (the model of the SMU). Additionally, you can see the Matrix Connection of the SMU that shows the terminal and connection status.

Figure 56: Model 4200 SMU information



Series 2600B instrument properties

NOTE

This topics refers to both 2600A and 2600B SMU instruments.

Single configuration: When the node of a Series 2600B instrument on the configuration navigator is selected, the work area displays an overview of the instrument(s).

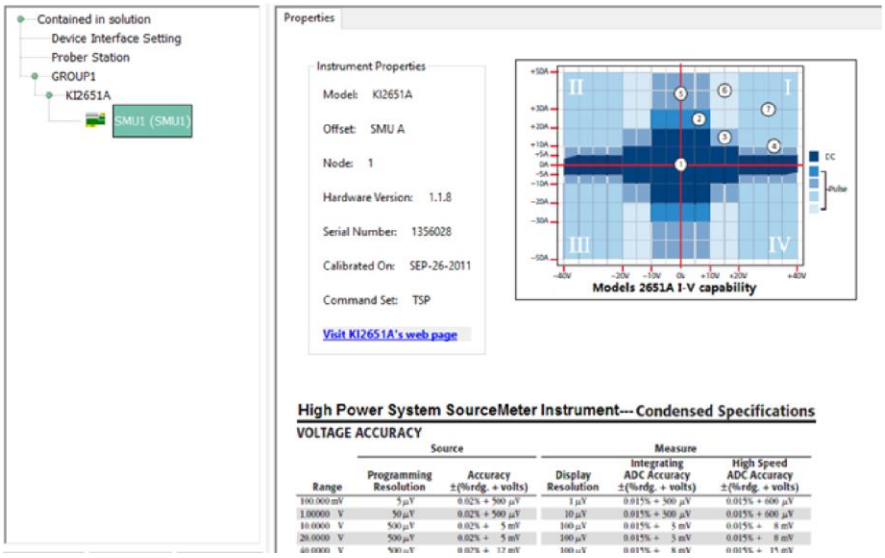
Group configuration: When the node of GROUPx on the configuration navigator is selected, the work area displays the groups' GPIB address and configuration.

SMU configuration: When the node of a Series 2600B SMUx on the configuration navigator is selected, the work area shows the SMU properties, specifications, and power capacity (see the following figure).

Example: Model 2651A

The following figure shows the instrument properties. You see the model number, node number, version of the hardware, serial number, the calibration date, and specifications for the SourceMeter. You can use this information to configure your testing and create a schedule to make sure calibration is completed as needed.

Figure 57: Model 2651A information



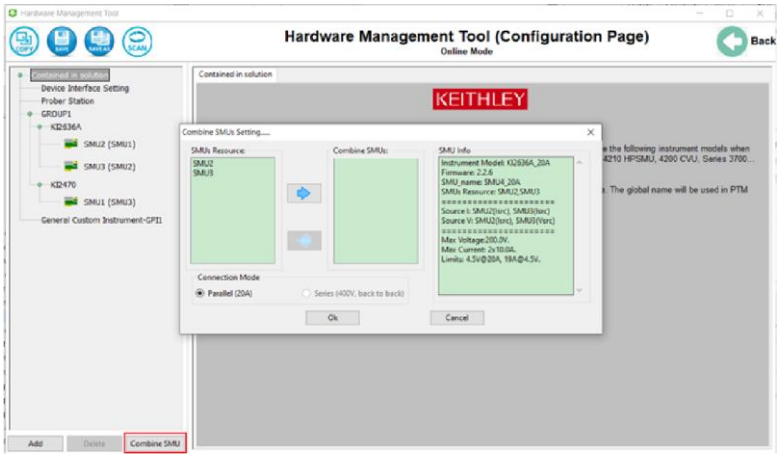
Composite SMU

ACS Basic supports combining two 2651A, 2636A/B, 2612A/B, or 2602A/B SMUs as a single SMU. Two 2651A SMUs can be combined in parallel as a single SMU. This will give you the capability to output 100 A, or use in series for an 80 V output. Combining two 2636A/B or 2612A/B or 2602A/B SMUs in parallel can extend the operating range to 20 A pulse, or use in series to extend the operating range to 400 V pulse. In the configuration navigator, the combined 2651A SMUs are considered a composite SMU. The work area will show the properties for the composite SMU.

Select **Combine SMUs**. A dialog opens where you choose the SMUs to combine (only two 2651A, 2636A/B, 2612A/B, or 2602A/B SMUs can be combined) and choose the connection mode (see the following figure).

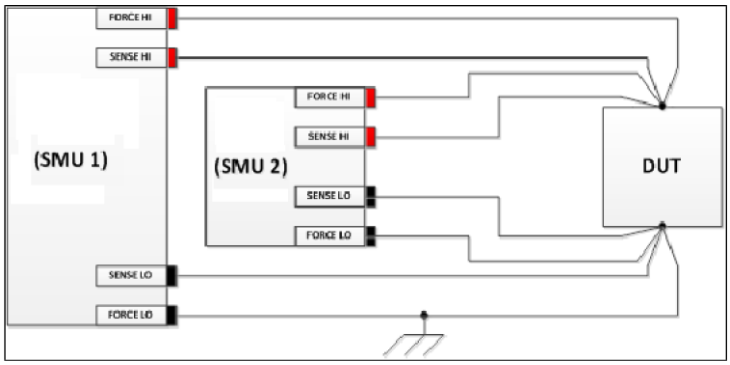
- **Parallel (100 A or 20 A):** Two SMUs are in parallel for higher current output. The composite SMU for the 2651A is named SMUx_100A (x represents the SMU number). The composite SMU for the 2636A/B, 2612A/B, or 2602A/B is named SMUx_20A.

Figure 58: Combine two SMUs



- **Series (80 V or 400 V):** Two SMUs in series back-to-back for a higher voltage output. The composite SMU for the 2651A has two terminals, SMUx_80V_HI and SMUx_80V_LO (x represents the SMU number). The composite SMU for the 2636A/B, 2612A/B, or 2602A/B has two terminals, SMUx_400V_HI and SMUx_400V_LO.

Figure 59: Parallel composite SMU connections

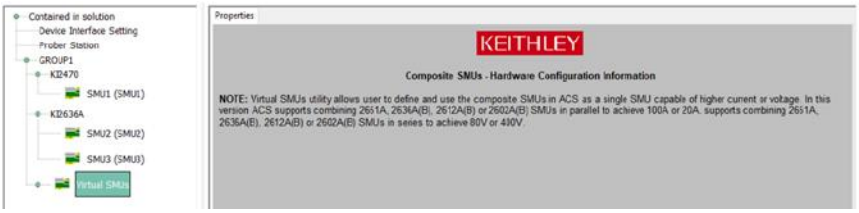


NOTE

The parallel SMU can be used in an ITM and in trace mode for certain modules. The series SMU can only be used in a 2-terminal device in trace mode.

In the configuration navigator, the composite SMU is labeled "Virtual SMUs" (see the following figure). The properties of the composite SMU are displayed in the work area.

Figure 60: Virtual SMUs



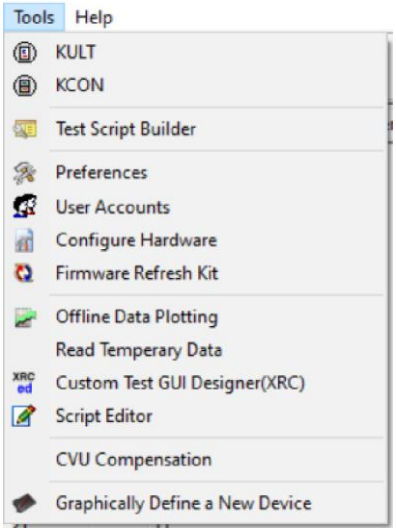
Firmware Refresh Kit option

In the Tools menu, the Firmware Refresh Kit option allows you to update the firmware for Series 2600 and 2600A instruments.

NOTE

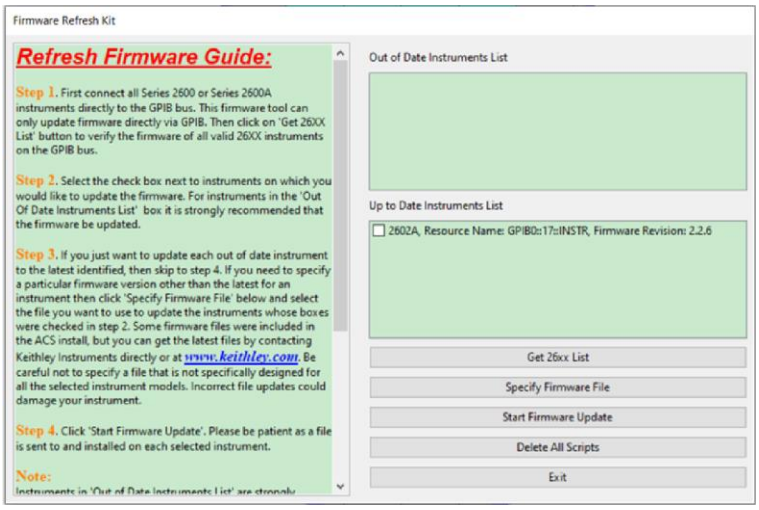
This firmware tool updates instruments connected to GPIB only. Do not to use a firmware file that is not specifically designed for the selected instrument.

Figure 61: Firmware Refresh Kit



The Firmware Refresh Kit dialog is shown in the following figure. You can use Get 26xx List, Specify Firmware File, Start Firmware Update, Delete All Scripts, and Exit to perform the firmware options.

Figure 62: Firmware Refresh Kit dialog



If you want to update or change your firmware on instruments you have, see the following list of options:

- Connect all Series 2600 and 2600A instruments directly to a GPIB cable. Select the **Get 26xx List** option to verify the firmware for the 2600 and 2600A instruments.

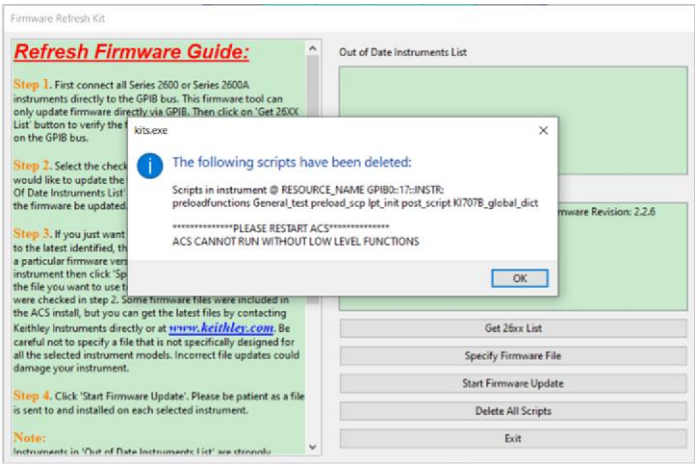
  For instruments in the Out-Of-Date Instruments List area, it is strongly recommended that firmware is updated. The software automatically applies the firmware update with this option.

CAUTION

For the next option, make sure you select the correct firmware file when updating firmware. Incorrect firmware files may damage your instrument. For example, only install 2600 firmware files on 2600 instruments, and 2600A files on 2600A instruments. ACS Basic software includes firmware files, however, you can find the latest firmware files on tek.com/keithley (it is recommended you search for 2600 firmware options or 2600A firmware options, depending on your needs).

- If you want to update an instrument to a specific firmware version, other than the latest for an instrument, select **Specify Firmware File** and select the file you want to use.
- Select **Start Firmware Update**. The firmware update time will vary depending on several factors, such as the number of instruments connected.
- If you want to delete all scripts for the instruments in the out-of-date instruments list and the up-to-date instruments list, select **Delete All Scripts** (see the following figure).

Figure 63: Delete scripts



Select **Exit** to close the Firmware Refresh Kit dialog.

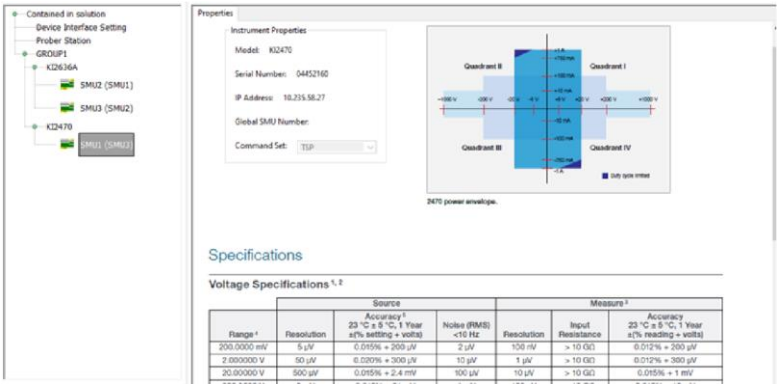
Series 2400 system properties

When the node of a Series 2400 instrument on the configuration navigator is selected, the work area shows the SMU properties, specifications, and power capacity.

The instrument Properties area displays the following information, as shown in the following figure:

- SMU Model
- SMU Serial Number
- GPIB Address
- Instrument name

Figure 64: SMU configuration information



Note that you have the option to delete scripts in a Series 2400 instrument.

To remove scripts:

1. Press the menu button on the 2400 instrument.
2. Go to **Scripts > Manage**.
3. Select the scripts to delete.

For more information, refer to the Series 2400 documentation on tek.com/keithley.

Matrix instruments

Models 707A/707B and 708A/708B switch matrices

Models 707A, 707B, 708A, and 708B switch matrices can be added to the ACS Basic hardware configurations as external instruments with GPIB connections. Models 707B and 708B can also be automatically scanned and added to the configuration through the GPIB or TSP-Link communications as a master or subordinate instrument.

NOTE

ACS Basic supports multiple matrices, such as Models 707A, 707B, 708A, 708B, and Series 3700A instruments in the hardware configuration. Additionally, ACS Basic supports multiple matrices as a group or as individual instruments.

ACS Basic supports both device dependent command (DDC) mode and Test Script Processor (TSP) mode for the Models 707B and 708B:

- DDC mode instruments must be manually added to the hardware configuration
- TSP instruments are automatically scanned and added to the hardware configuration

Model 707B and 708B configuration

When a Model 707B or 708B node is selected in the configuration navigator, the instrument Properties, Instrument Connection Scheme, and specifications can be viewed in the work area.

The Model 707B or 708B Properties tab contains:

- Instrument Properties:
  - Model of instrument
  - Associated group (TSP)
  - Node (TSP)
  - Work mode (TSP or DDC)
  - Card list (TSP)
  - GPIB address (DDC mode)
- Instrument Connection Scheme:
  - Row-Column or Instrument-Card
  - Local Sense or Remote Sense
  - Common LO or Independent LO
- Switch Card: select matrix cards for the system (see the following figure):
  - Model 7071 Matrix Card
  - Model 7072 Matrix Card
  - Model 7072 HV Matrix Card
  - Model 7136 Low Current Multiplexer Card
  - Model 7174 Low Current Matrix Card
  - Model 9174 Semiconductor Matrix Card

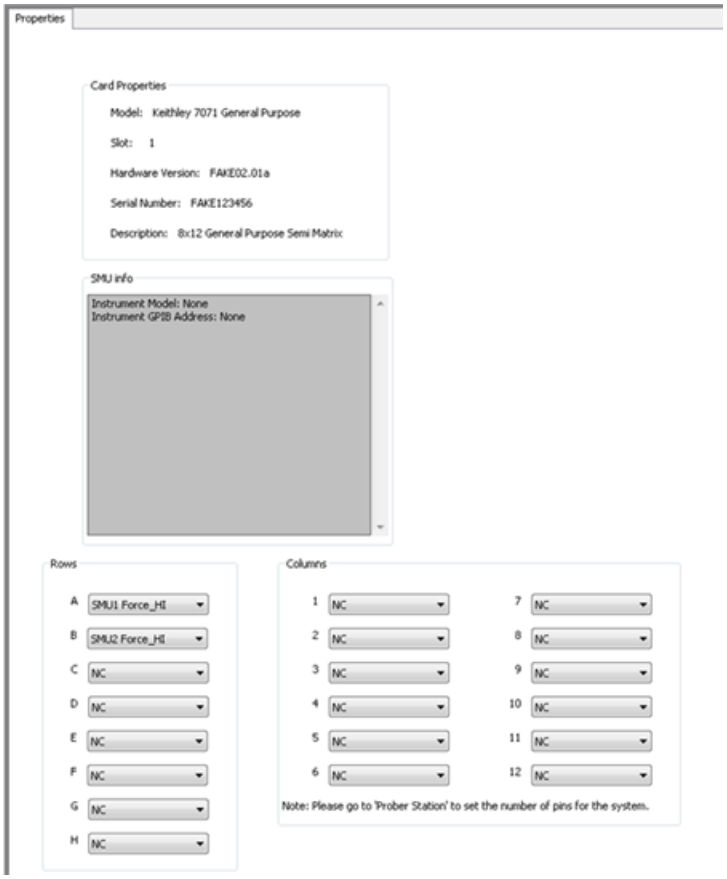
NOTE

If any of the matrices in the system are set to Remote Sense, all the Series 2600B and Series 2400 instruments in the system are set to four-wire (4-W) mode when executing a test and the SMU Remote Sense option in the Preference settings Instruments tab is disabled. When all the matrices are set to Local Sense, all the Series 2600B and Series 2400 instruments in the system are set to two-wire (2-W) mode when executing a test. It is highly recommended that all the matrices in the system are in the same mode.

Matrix Card information

When you select one of the matrix cards on the hardware configuration navigator, a GUI dialog opens. The GUI is where you configure the hardware (see the following figure).

Figure 65: Model 7071 matrix card properties



The Matrix Card Properties tab contains:

- **Card Properties:** Instrument model, slot number, hardware version, serial number, and the description of the card.
- **SMU info:** Instrument model, GPIB address, maximum voltage and current, voltage and current range.
- **Rows:** A through H correspond to the eight (8) rows of all Model 707B/708B compatible matrix cards. Use the lists to connect the rows to the instrument terminals.
- **Columns:** One through 12 correspond to columns for the Model 707B/708B compatible matrix cards. Use the lists to connect the columns to the instrument terminals or prober test-fixture pins, or any combination of both terminals and prober pins as needed for your application.

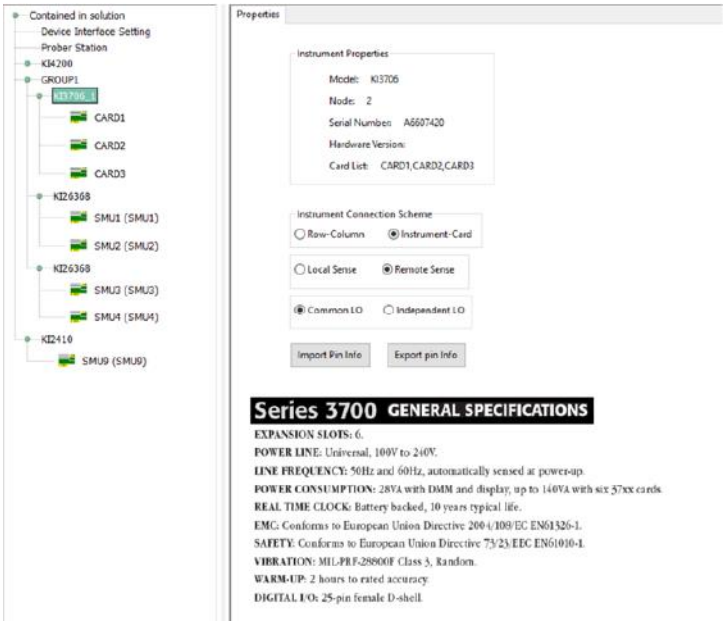
NOTE

Due to the Model 2651A high current SMU (up to 50 A), and the Model 2657A high power SMU (up to 3000 V), you cannot add a Model 2651A SMU and a Model 2657A SMU to the matrix card row or column.

Model 3706A configuration

When a Model 3706A node is selected in the configuration navigator, the work area shows the Properties, Instrument Connection Scheme, and specifications (see the following figure).

Figure 66: Model 3706 information



A Model 3706A Properties tab contains the following:

- **Instrument Properties:** Displays the node and the two types of cards: Multiplexer or matrix.

The following Instrument Connection Scheme selections define the scheme for interconnections between the instruments, the switch matrix rows and columns, and the test system (prober or test fixture).

- Row-Column (instruments to rows and prober/test fixture to columns) or Instrument-Card (both instrument and prober/test fixture to columns).
- Local Sense (connections only to the instrument FORCE terminals) or Remote Sense (connections to both the instrument FORCE and SENSE terminals).
- Series 3700A General Specifications: Displays the information about the specifications.

Series 3700A: multiplexer card

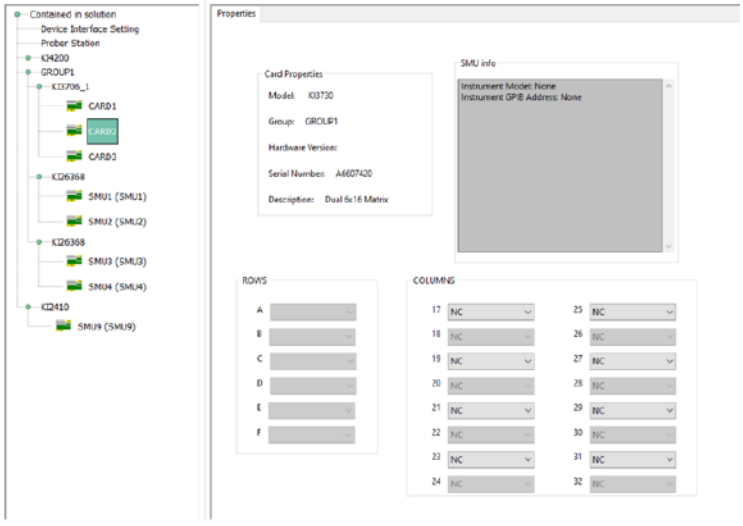
The next figure shows two cards are inserted in a Series 3700A instrument. They are the multiplexer card (CARD1) and the switch card (CARD2). When the multiplexer (CARD1) node is selected, the work area shows the properties and specifications for the multiplexer (CARD1).

The Card Properties group of a multiplexer card contains information about the model, group, hardware version, serial number, and description. Specifications for the card are also listed.

Series 3700A: matrix card

When the matrix card (CARD2) node is selected, the work area shows the properties and connection options that are defined for the device (see the following figure).

Figure 67: Matrix card information



The Card Properties area contains information about the model, group, hardware version, serial number, and description. There is also a SMU info area, and two additional areas that provide information about the connection method:

- **Rows:** In the Rows area, the edit boxes labeled A through F correspond to the 6 rows for the Model 3706A compatible matrix cards. Use the drop-down arrows to select the rows for your instrument terminals.
- **Columns:** In the Columns area, the edit boxes labeled 1 through 16 correspond to the 16 columns for the Model 3706A compatible matrix cards. Use the drop-down arrows to select the columns for your instrument terminals and/or prober/test-fixture pins.

NOTE

You can only connect instrument terminals to the matrix card columns when the Instrument Connection Scheme setting is active (see the following figure).

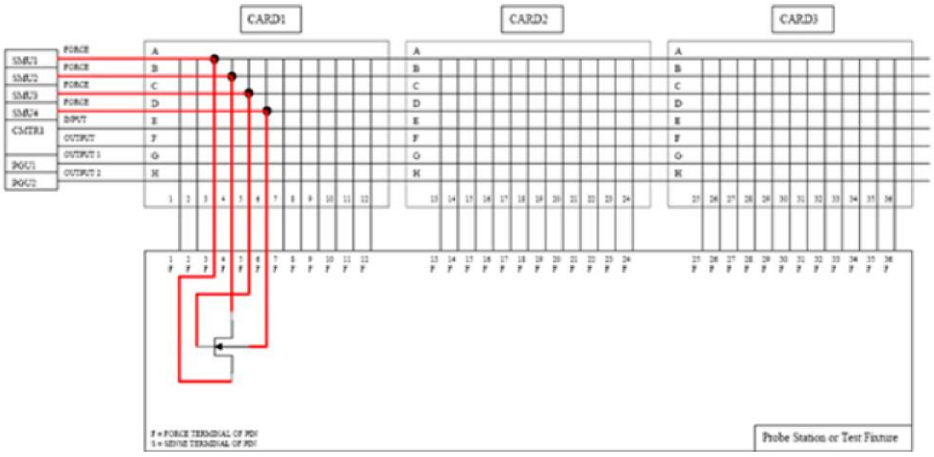
Matrix instruments

Row-Column: instruments are connected to switch matrix rows, and prober/test fixture pins are connected to switch matrix columns (see the following figure). This is the simplest connection scheme. Instrument signals can route to the prober/test-fixture pins through one matrix card. However, the Row-Column scheme limits the number of external instruments.

NOTE

Prober or test-fixture pins are always connected to matrix card columns.

Figure 68: Row-column, local sense connection scheme example



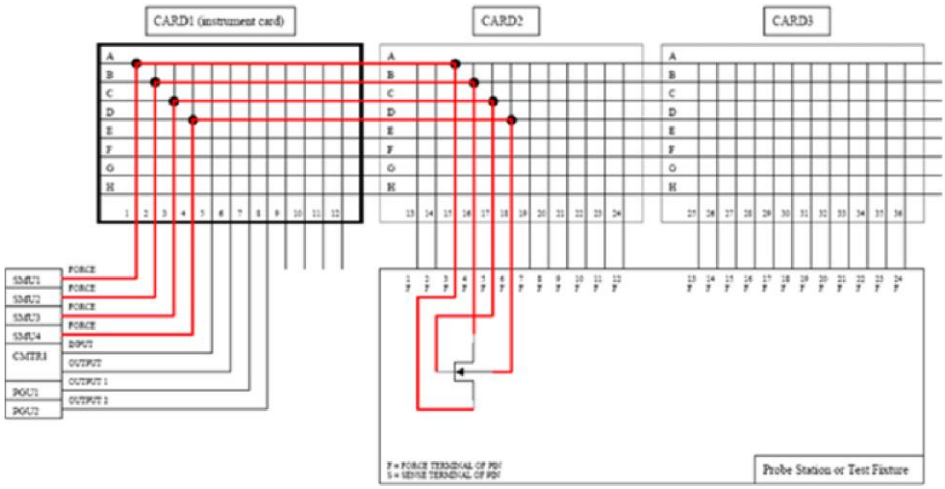
If you need to connect multiple external instruments to the prober/test-fixture, use the Instrument Card scheme.

Instrument Card: both instruments and prober/test-fixture pins are connected to the switch matrix columns. Instrument signals can route to the prober/test-fixture pins through two or more matrix cards. This connection scheme can support more instruments compared to the Row-Column scheme (see the following figure).

NOTE

Due to the Model 2651A high current SMU (up to 50A), you will not be able to add a Model 2651A SMU to the matrix card row or column.

Figure 69: Instrument card, local sense connection scheme example



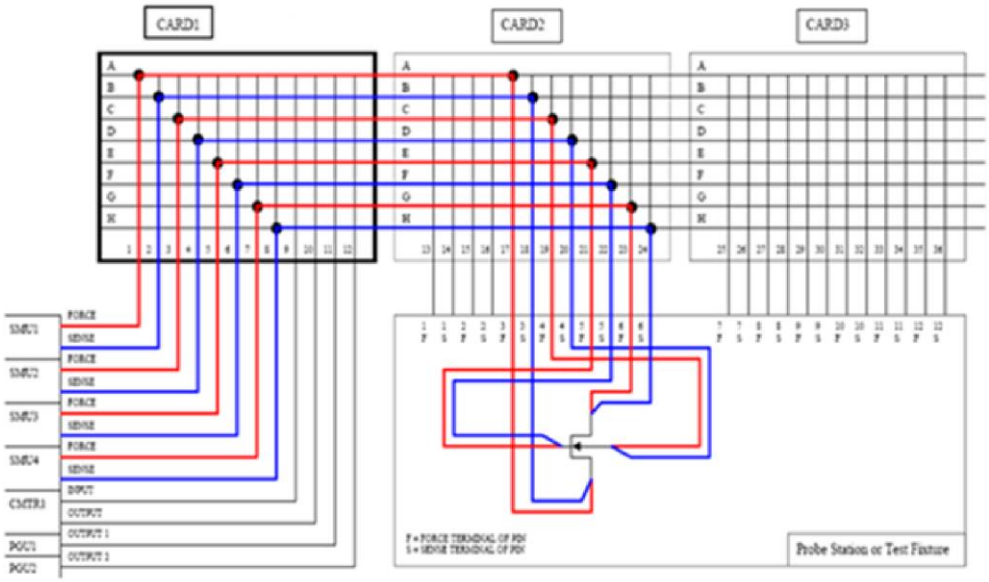
- **Two-wire local sense:** Use this when the measurement-pathway resistance is small and the associated voltage errors are negligible. The measurement pathway is comprised of the following conductors, connected in series:
  - Cables used to connect the instruments to the matrix
  - Internal matrix-card signal path
  - Cables used to connect the matrix to the prober or test fixture

Current flowing through the measurement pathway creates a voltage drop (an error voltage) that is directly proportional to the pathway resistance. This error voltage is present in all local sense voltage measurements.

Four-wire remote sense can be used when sourcing or measuring voltage in a low-impedance test circuit, and there can be errors associated with IR drops in the test leads. Voltage source and measure accuracy are optimized by using four-wire remote sense connections. When sourcing voltage, four-wire remote sensing ensures that the programmed voltage is delivered to the DUT. When measuring voltage, only the voltage drop across the DUT is measured.

- Use four-wire remote sense for the following source-measure conditions:
  - Sourcing and/or measuring voltage in low impedance (<1 kΩ) test circuits.
  - Enforce voltage compliance limit directly at the DUT.

Figure 70: Instrument card, remote sense connection scheme example



Model 590 C-V analyzer

When you select the Model 590 C-V Analyzer in the configuration navigator, its Properties & Connections tab opens in the work area.

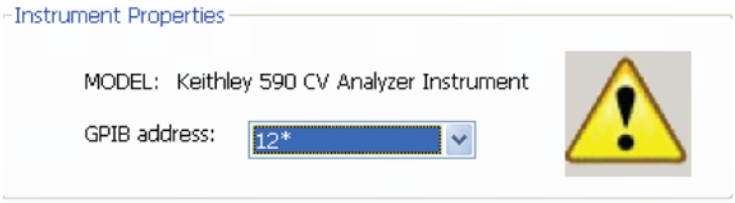
The Properties & Connections tab provides access to the Model 590 instrument properties and useful switch-matrix connection information. The Instrument Properties area of this Properties & Connections tab provides access to the following Keithley Instruments Model 590 instrument properties:

- **Model:** Full vendor name, model number, and instrument description.
- **GPIB Address:** Primary GPIB address list. Addresses that are in use are displayed with asterisks (*) next to them. Range is 0 to 30 (GPIB address 31 is reserved as the controller address). If the selected GPIB address conflicts with the GPIB address of another system component, a red exclamation-point symbol (!) is displayed next to the selected address.

NOTE

You can programmatically read the GPIB address and other instrument properties on the system configuration using the LPTLib `getinstattr` function. Proper usage of `getinstattr` allows you to develop user libraries in a configuration independent manner.

Figure 71: GPIB address properties

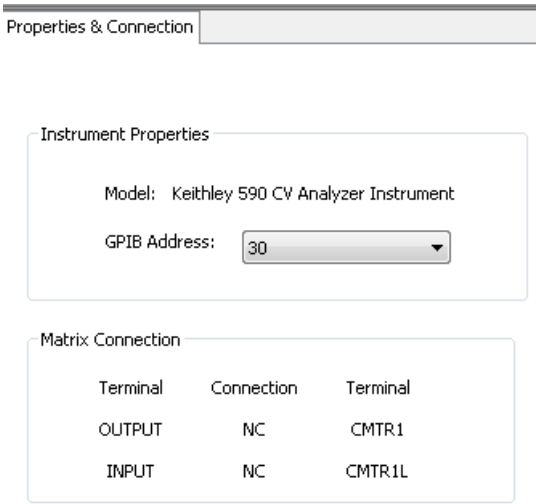


When a matrix is included in the system configuration, the Matrix Connections area of this Properties & Connections tab displays the matrix connections that are associated with the Model 590 measurement terminals.

Capacitance meter configuration

When you configure a capacitance meter, the only item that you can change is the GPIB address. You will also see the matrix connection displayed (see the following figure).

Figure 72: Configuration of capacitance meter



Section 4

ACS Basic test setup

In this section:

Test setup introduction

The ACS Basic interface consists of a variety of graphical user interfaces (GUI) that allow you to do the following:

- Create tests for one or more devices
- Build and edit test sequences
- View test results in tabular and graphical forms
- Analyze test results using built-in parameter extraction tools

Test modes

ACS Basic has three test modes:

1. Single-test mode
2. Multitest mode
3. Trace mode

Use the Single-test mode to open and run a single test on a single device at a time.

Use Multitest mode to create multiple test modules on one or more devices and organize them into a single project. For example, Multitest mode can perform multiple tests on a single device (target application) or multiple tests on multiple devices (secondary use case). You can work with external instruments (DMMs, switches, and so on) in Multitest mode more easily than in any other mode.

Trace mode is for interactive operation with a SourceMeter unit. In Trace mode, you can adjust the maximum sweep value and maximum power to the device. You can also transition between DC and pulse forcing modes.

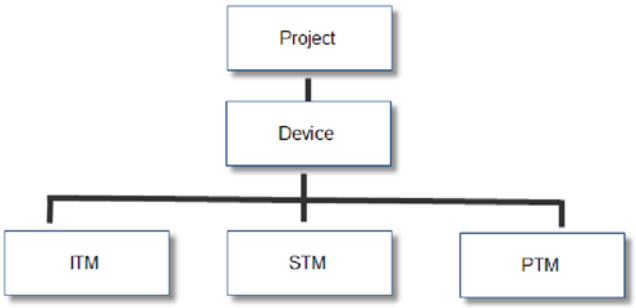
Test project

The test project defines and sequences all the devices and specifies the tests to be performed on each device. Projects are only valid for Multitest mode.

Devices and their respective tests are organized in the configuration navigator, which appears in the left panel of the ACS Basic interface. Use the configuration navigator to insert new test modules, copy and paste existing test modules, and sequence the order in which tests are performed.

In Multitest mode, the ACS Basic configuration navigator follows the logical hierarchy shown in the following figure.

Figure 73: Multitest mode hierarchy



Devices

Each project contains one or more devices to be characterized including transistors, diodes, resistors, TRIACs, and capacitors.

Test modules

There are three types of test modules in ACS Basic:

1. Graphical interactive test module (ITM)
2. Script test module (STM)
3. Python language test module (PTM)

All three types of test modules have the same datasheet and data plot analysis functions. The differences between ITMs, STMs, and PTMs are described in the following topics.

Graphical interactive test module (ITM)

An ITM allows you to define a test interactively using a graphical user interface (GUI). ITMs are available for the Model 4200-SCS SMUs, Series 2600B System SourceMeter® or Series 2400 System SourceMeter® instruments.

Script test module (STM)

An STM supports test script processor (TSP®) files. A TSP file is written for testing a device with a Series 2600B System SourceMeter, Model 3706A System Switch/Multimeter, or Model 707B/708B switch matrix in TSP mode. STM examples are installed with ACS Basic. These scripts can be imported and modified in the STM workspace. STMs can be used with a user-created GUI in an XRC format to create custom tests with a graphical interface.

Python test module (PTM)

A PTM is a user-generated Python script that can control instruments through a GPIB cable. You can create an optional GUI to interface with the PTM in XRC format or in a customized GUI format. PTM script examples are installed when ACS Basic is installed. These scripts can be imported and modified in the PTM test module.

The following table indicates the instruments and communications interfaces supported with the appropriate test modules (table legend: Y = supported, Y+ = recommended, -- = not supported).

Instrument Series	ACS Basic installed on	Communications Interface	ITM	STM	PTM
2400	PC/laptop	GPIB	--	--	Y
2400 Graphical Series	PC/laptop or 4200A	GPIB, ethernet, USB, TSP subordinate	Y	Y	Y
2600	PC/laptop or 4200A	GPIB, TSP subordinate	Y+	Y	Y
2600B	PC/laptop or 4200A	GPIB, ethernet, USB, TSP subordinate	Y+	Y	Y
2651A & 2657A	PC/laptop or 4200A	GPIB, ethernet, USB, TSP subordinate	Y+	Y	Y
4200A	4200A only	GPIB	Y+	--	Y
3700A	PC/laptop or 4200A	GPIB, ethernet, USB, TSP subordinate	--	Y+	Y
Other external instruments	PC/laptop or 4200A	GPIB, USB, ethernet	--	--	Y

NOTE

If you copy and paste a test module, the device settings will not carry over. It is necessary to reassign the device settings if the test module is pasted in a different device. For example, if you copy an ITM from a nMOSFET device and paste it in a npnBJT device, you will need to manually reassign the device settings.

See ACS Basic test setup (on page 4-1) for more information on the types of test modules.

Single-test mode introduction

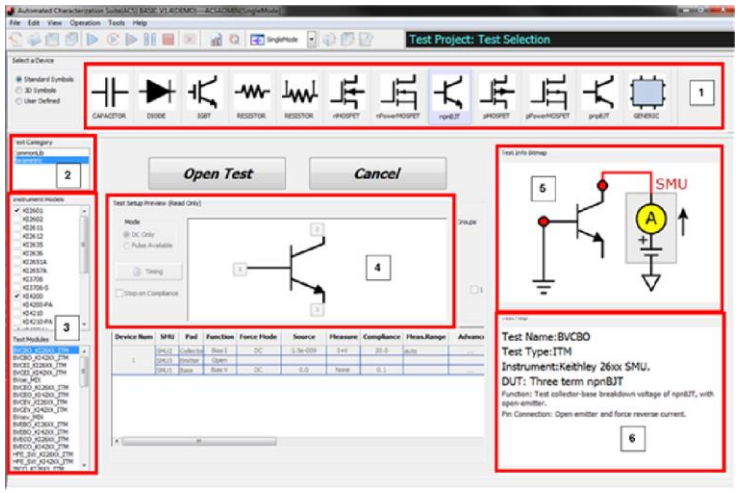
Single-test mode is used for importing and running one test module on one device at a time. The single test can be an ITM, STM, or PTM.

Introduction

The test selection GUI of the Single-test mode is shown in the next figure and includes the following sections:

1. **Select a Device:** Select a device view (Standard, 3D Symbols, or User Defined).
2. **Test Category:** Select the device. The selected icon is highlighted.
3. **Instrument and Test Modules:** Select the test instrument model. The test modules applicable for this instrument are listed in the Test Modules area. If two or more instruments are selected, the Test Modules display all the modules available to any of the selected instruments.
4. **Test Setup Preview:** A visual presentation of the type of test.
5. **Test Info Bitmap:** An image of the type of test selected.
6. **Test Help:** Specific information about the test.

Figure 74: Single-test mode test selection GUI



Select a test module in single-test mode

Select a test module in single-test mode:

1. Select a device view (Standard, 3D Symbols, or User Defined).
   - Standard Symbols uses electrical symbol images.
   - 3D Symbols uses three-dimensional images of electrical symbols.
   - User Defined items are created using the Graphically Define a New Device dialog. For more information on how to create a user-defined device, refer to Graphically define a new device.
2. Select the device. The selected icon is highlighted.
3. All test categories belonging to this device are listed in the Test Category area. Select the category:
   - **CommonLib_CV:** Includes some tests for CV testing, such as Cv4282 and Generic_HVCV_Test, KI42xxCVU.
   - **CommonLib_other:** Includes some tests for specific instruments, such as KI37xx_DMM_Switch, switchctrl_6cards_70x, switchctrl_6cards_3706, and TEKSCOPE_ReadWave.
   - **CommonLib_SMU:** Includes some tests for 3706, such as FourWireResistor_3706 and Gpibresistor_3706
   - **Mixed_SMU_Mode:** Includes some tests for specific instruments, such as BiasIMeasV_Pluse_2430, BiasVolt_SampleCurr_23x, BVcei_any_SMU, and SweepV_MeasI_24xx.
   - **Parametric:** Test libraries for device parametric tests. When you select a device and select the instruments in the Instrument Models area, the Test Modules area lists all the parametric tests available for the device. These test modules use specific SMUs for device parametric tests, such as IdVd, IdVg test for MOSFET, among others.
   - **WLR_script:** The wafer level reliability tests for selected devices, such as the HCI (Hot Carrier Injection) nMOSFET device.

NOTE

Not all devices have a test library in all six test categories.

4. Select the test instrument model. The test modules applicable for this instrument are listed in the Test Modules area. If two or more instruments are selected, the Test Modules area displays all the modules available to any of the selected instruments.
5. Select a test module.
6. The test information preview is displayed on the right panel. This information includes Test Setup Preview, Test Information Bitmap, and Test Help.

Open and edit the test module

NOTE

Select the **Return to library** icon at any time to terminate the open test process and return to the previous test.

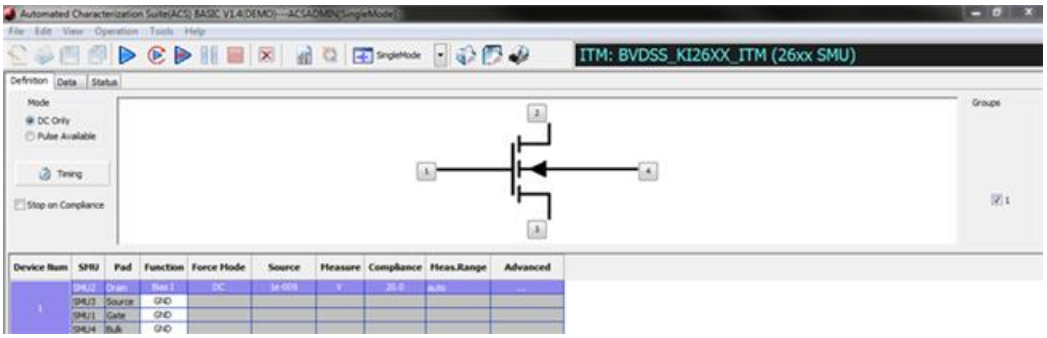
1. Select the **Open Test** button to open the test view GUI. In this GUI, you can modify the test module settings and run the test module.
2. In the test view GUI, select the **Device view** icon to open the device view GUI (see next figure). Use Device View to designate the test and measurement instruments that are connected to each device terminal.

   To configure an ITM, refer to Configure a graphical interactive test module (ITM) (on page 5-1).

   To configure an STM, refer to Configure a script test module (STM (on page 5-33)).

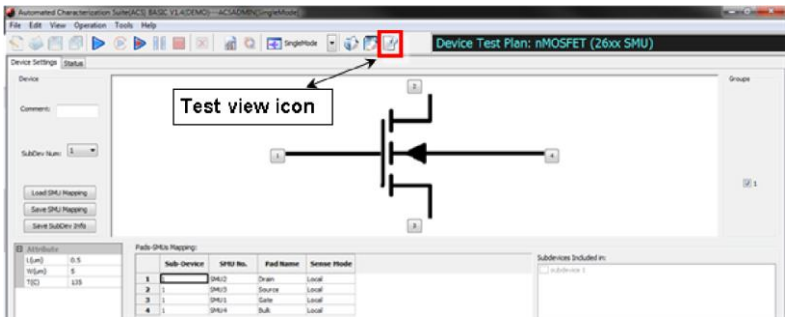
   To configure a PTM, refer to Configure a standard PTM (on page 5-58).

Figure 75: Test view GUI



3. In the device view GUI, select the **Test view** icon to return to the test view GUI (see next figure).

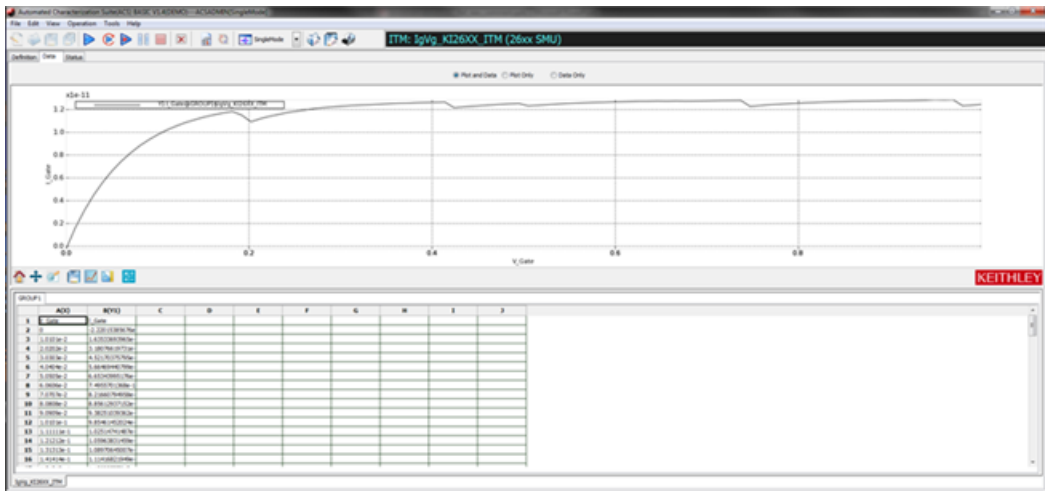
Figure 76: Device view GUI



Run the test module

Once the test setup is completed, select the **Run** icon. The test results appear in the Data tab (see next figure). Refer to Data plot a graph sheet (on page 4-23) for details on the Data tab.

Figure 77: Test Data tab



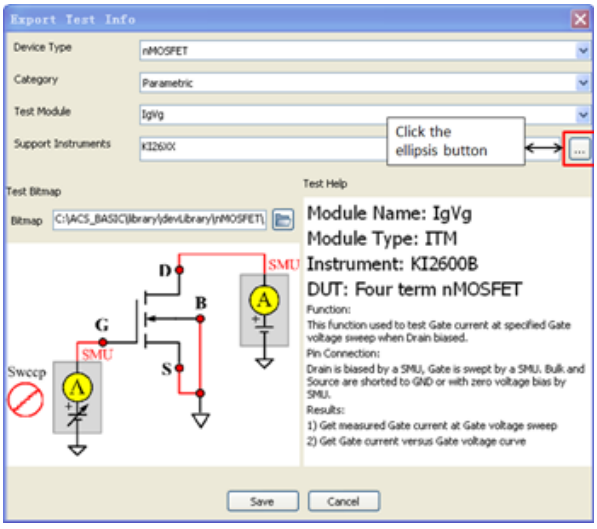
Save the test module to the library

The test module can be saved to the device library for future use. The saved information includes both test settings and test results.

To save the test module:

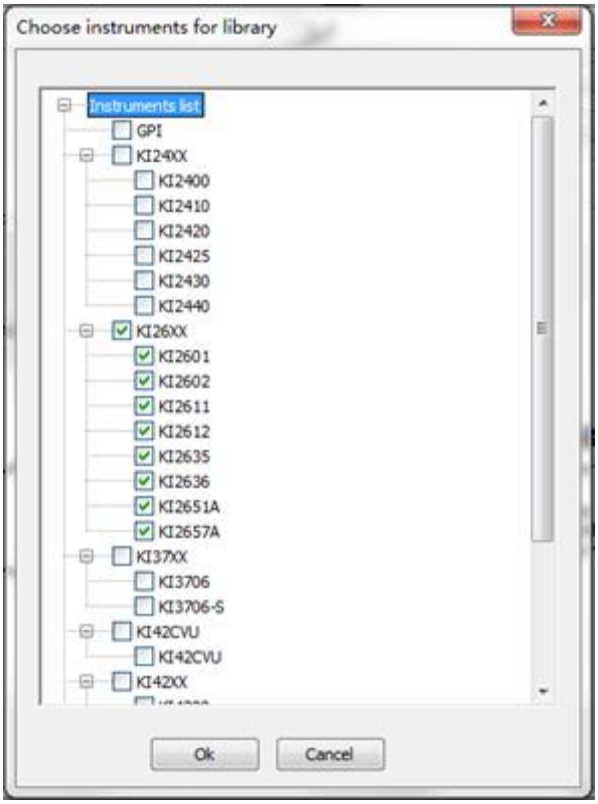
1. Select the **Export Test** icon on the toolbar; the save test information GUI opens (see next figure).

Figure 78: Save Test Info dialog



2. Select a device type for the module in the Device Type list.
3. Select a test category in the Category list.
4. Input a test name in the Test Module edit box. Note that the new Test Module name should be different from the existing name to avoid overwriting the original module.
5. Select the ellipsis button (see previous figure) of the Support Instruments edit box. In the dialog box choose the instruments for the library by selecting the box next to the instrument (see next figure). Select **OK**.

Figure 79: Save selected instruments for testing



6. If needed, change the test bitmap by selecting on the folder icon to find a new bitmap.
7. If needed, change the description by entering the text in the right editing panel.
8. Select **OK** to save the test module to the test library. After the test module is saved, ACS Basic returns to the Open Test GUI, where the previous test can be opened and used again.

MultiMode introduction

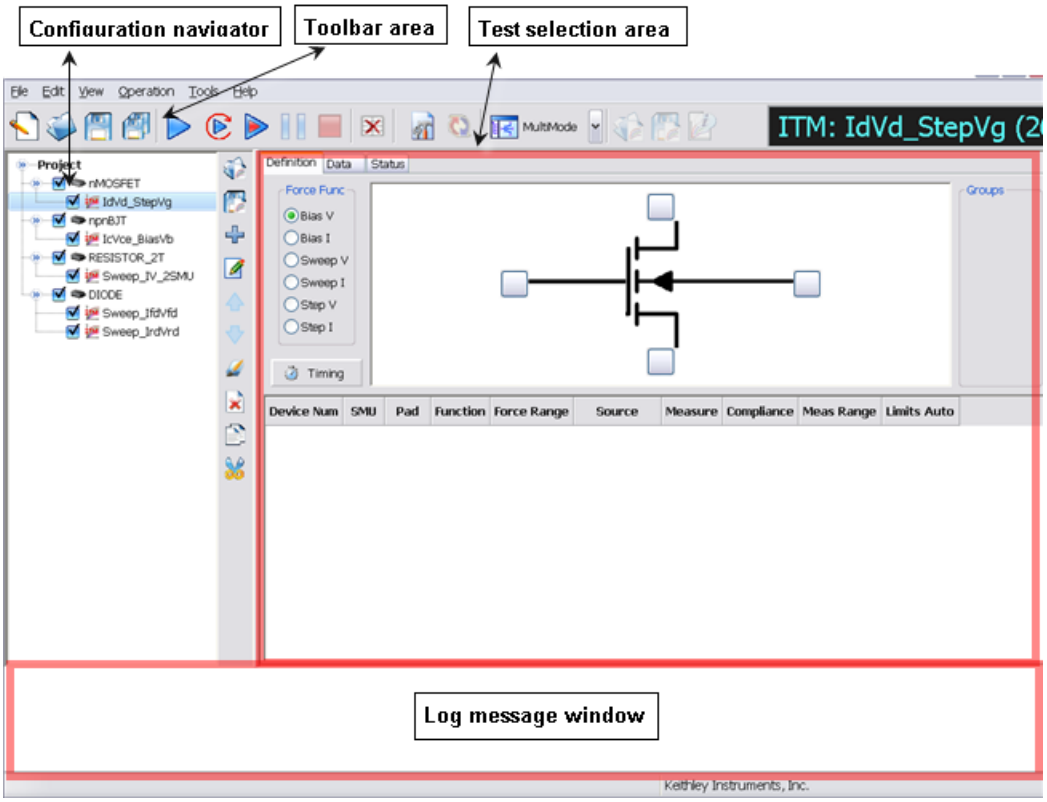
MultiMode uses a project to organize devices and tests.

Multitest mode GUI

The configuration navigator (see next figure) has the following logical hierarchy:

- Project
  - Device 1
    - Test module 1 (ITM, STM, or PTM)
    - Test module 2 (ITM, STM, or PTM)
  - Device 2
    - Test module 1 (ITM, STM, or PTM)
    - Test module 2 (ITM, STM, or PTM)

Figure 80: MultiMode user interface



The configuration navigator is the primary interface for building, editing, and viewing a project plan, and for specifying and accessing each project plan component.

When you select a navigator component (test or device) you can do the following:

- Add a new component
- Delete an existing component
- Run the tests associated with this component

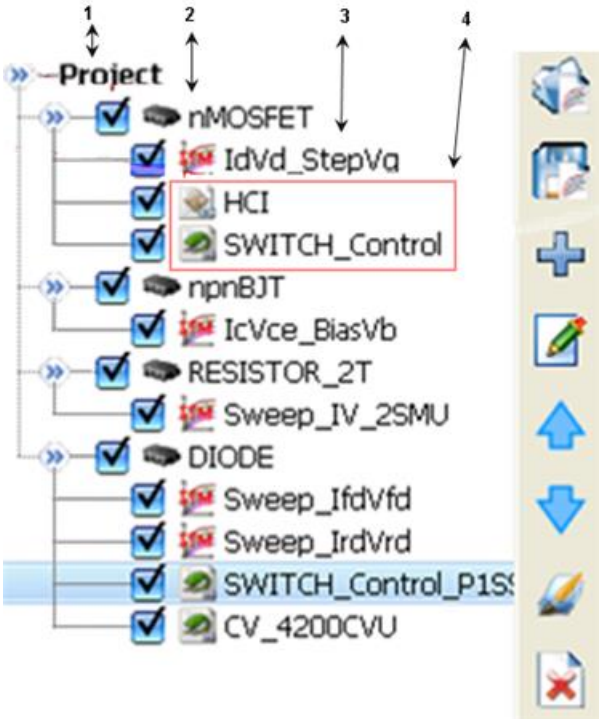
Selecting a navigator component opens the configuration interface on the right panel that includes settings, test results, and status information.

The next figure displays the typical project plan components that are displayed in the configuration navigator. The following explanation of the numbered items correspond to the numbers in the next figure:

1. **Project** Plan: Defines and sequences all devices tested, and all tests/operations to be performed at each. It typically corresponds to one die on a wafer.
2. **Device** Plan: Defines and sequences all tests for a specific device, including transistors, diodes, and resistors.
3. **ITM**: Completely defines a parametric test without programming, using a series of easily configured graphical user interfaces which show data numerically and graphically in real time. Provides for the display of both raw data and analyzed data.
4. **STM/PTM**: Defines an operation; an external instrument operation, using a TSP or PTM user module. These test modules are connected to the STM and are configured with user-supplied parameter values (several STMs may be associated with the with the same user module). Configuration is done using a supplied library or may be created by the user with Script Editor's simple graphical user interface.

For details about building a project plan using the configuration navigator, refer to Build a project plan (on page 4-12).

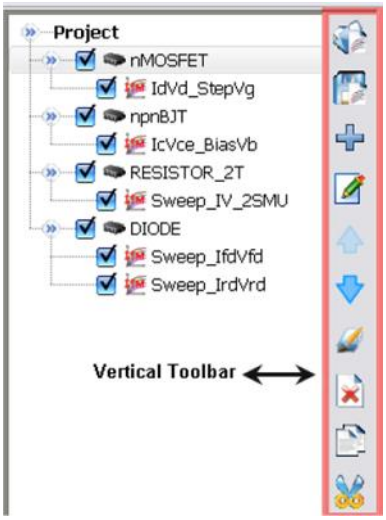
Figure 81: Configuration navigator



Vertical toolbar

The vertical toolbar is only accessible in Multitest mode. See the next figure for the location in the Multitest mode GUI:

Figure 82: Vertical toolbar location



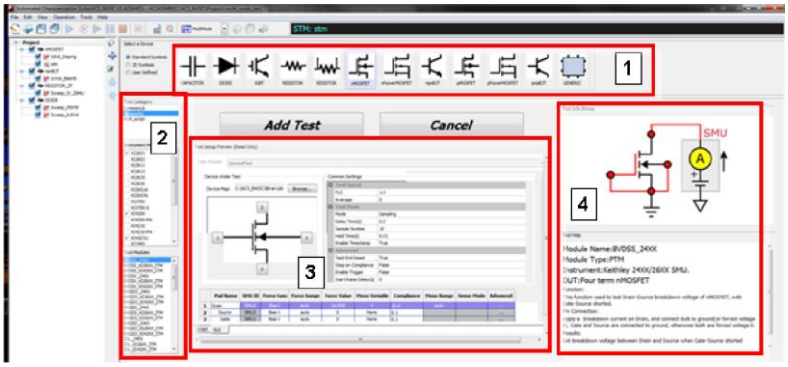
Multitest mode work area

When ACS Basic is opened in Multitest mode, you see the configuration navigator and the work area that shows the settings, test results, and status information.

The test selection GUI of Multitest mode is shown in the next figure. There are four main sections:

- Select a Device
- Test Category - select a test module
- Test Setup Preview
- Test Information

Figure 83: Multitest mode work area



Build a project plan

Create a new project:

1. Select the **New** icon on the toolbar (or you can select **New** in the File menu). The New Test Project dialog opens.
2. Enter the following information:
   - **Template**: Use to load an existing project template. Enter the storage folder name or use the Browse function to find the location. This field may be left blank.
   - **Project Name**: Enter the new project name (this is required).
   - **Project Directory**: Select a directory where you want to save the project, or use the default directory. Use the Browse function to find a location.
   - **Technology**: Optional device information.
   - **Product**: Optional device information.
3. Select **OK** to complete the new test project.

When you open ACS Basic and a test project, the project directory path displays in the title bar of the ACS Basic GUI. All project information is saved to this directory.

Add a test module to the project (see Select a test module in single-test mode (on page 4-5) for more details):

1. Select a device view (Standard, 3D Symbols, or User Defined).
2. Select the device. The selected icon is highlighted.
3. All test categories belonging to this device are listed in the Test Category area. Select the category.
4. Select the test instrument model. The test modules applicable to this instrument are listed in the Test Modules area. If two or more instruments are selected, the Test Modules area displays all modules available to any of the selected instruments.
5. Select a test module.
6. The test information preview is displayed on the right panel. This information includes Test Setup Preview, Test Information Bitmap, and Test Help.
7. Select the Add Test function to add the test module to the configuration navigator.

If the test module imported belongs to the same kind of device, as an existing device in the configuration navigator, the module will be added to the existing device automatically.

If the imported test module does not belong to the same kind of device, a new device node is added to the configuration navigator.

Configure the test module:

After the test module is added, select the device in the configuration navigator to view and edit test details.

Execute individual device plans

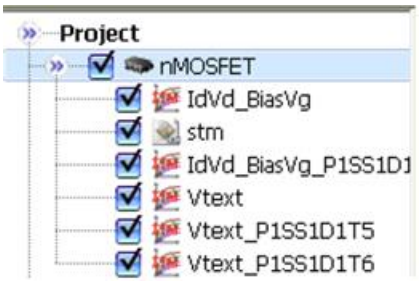
After a test plan has been built, you can execute individual parts of a project plan at the test module level, the device level, or the project level.

NOTE

Each time you execute a test or test sequence using the Run icon, the data from each test is inserted to its own Data worksheet. Each new run updates this worksheet (for more information on worksheets, refer to Data plot a graph sheet (on page 4-23). You can also generate appended worksheets for tests and test sequences.

You can execute an individual Device Plan with the selected components, ITMs, PTMs, and STMs, assigned to it in the order that they appear in the configuration navigator. In the configuration navigator, select the box corresponding to the device to run. For example, in the next figure, the nMOSFET has been selected.

Figure 84: Select the device to run



Enable the tests assigned to this device by selecting the box next to the test name. Disable a test by clearing the box.

To save the plan, select the Save icon in the ACS Basic toolbar, or select the Save function in the ACS Basic File menu.

Start execution by selecting the Run icon in the toolbar. The Run toolbar icon turns gray while the test executes, and the Abort Test/Plan icon illuminates for the duration of the test. The test sequences run one at a time.

NOTE

Selecting only an individual ITM, STM, or PTM on the test tree runs only that test.

Trace mode introduction

Trace mode provides interactive control of SMU instruments to achieve real-time performance. Only Series 2600B instruments (Models 2601B/2602B, 2611B/2612B, 2635B/2636B), Series 2650A (Models 2651A, 2657A) and Series 2400 instruments (Models 2400, 2401, 2410, 2420, 2425, 2440) are supported. ACS Basic can perform DC and pulse tests on Series 2600B, and 2650A instruments in Trace-mode, but it can only perform DC measurements on Series 2400 instruments.

Trace-mode allows you to quickly and easily setup the instrument for fast, visual results and save the data for later comparison or analysis

NOTE

The Series 2600B only supports the GPIB communications mode to the master and does not support ethernet (LXI) communications to the master. Trace-mode only supports the Series 2400, Series 2600B, and the Models 2651A/2657A. However, you cannot connect a Series 2400 and a Series 2600B instrument to the same device. If you are using a Series 2400 on one device terminal, you must use a Series 2400 instrument on the other device terminals. However, you can use a Series 2600B and a Model 2651A instrument in any combination. Also, Trace-mode only supports the SCPI protocol for Series 2400 SMUs.

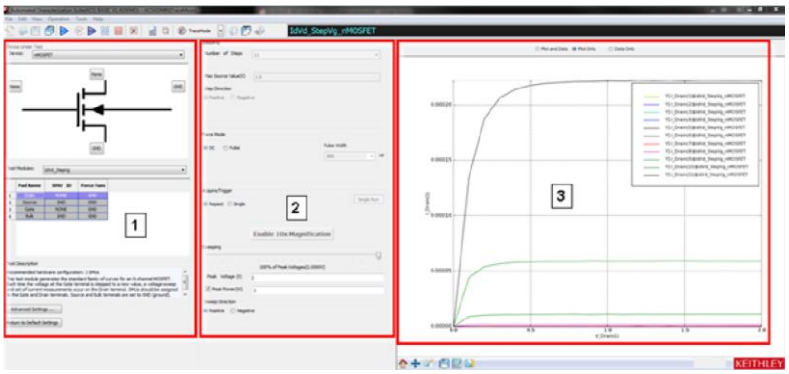
Test GUI

Select **TraceMode** after selecting the drop-down arrow from the operation toolbar at the top of the ACS Basic GUI. The Trace-mode GUI is shown in the next figure.

The GUI includes:

1. Device setting: used to map the SMU, select the test module, and set the advanced settings.
2. Test setting: used to set the test information when the test module is enabled.
3. Trace Test Data: used to show the test result.

Figure 85: TraceMode GUI

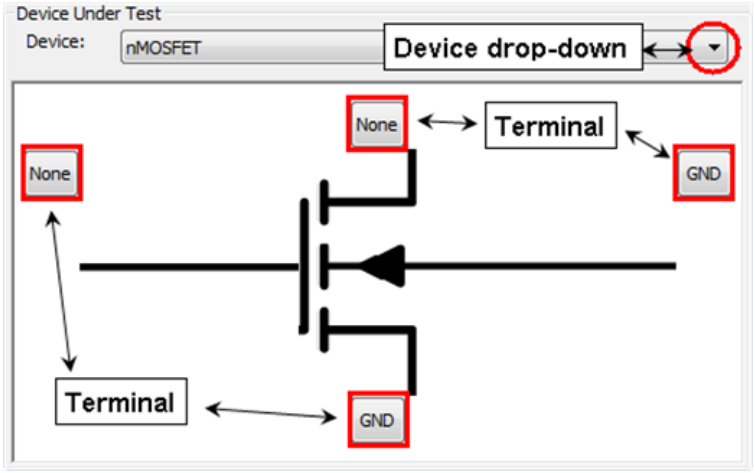


Open a test module in Trace mode

To open a test module in Trace mode:

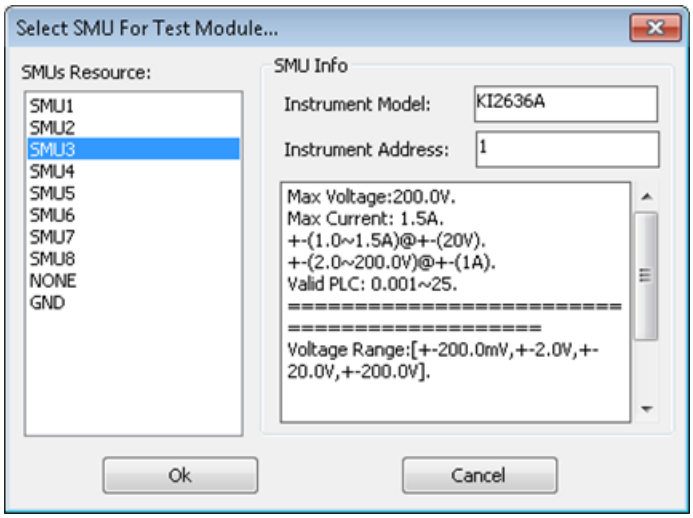
1. Select a device in the Device list in the Device Under Test area (see next figure).

Figure 86: Device Under Test area



2. Select a test module for the device. All test modules belonging to this device are listed in the Test Modules list.
3. Connect a SMU to the device terminals.
   a. Select the button next to the device terminal and the Select SMU For Test Module dialog box opens (see next figure).
   b. From the SMUs Resource list, select a SMU for the terminal. The SMU Info box indicates the Instrument Model and Instrument Address.
   c. Select **OK** to complete the SMU selection.

Figure 87: Select SMU For Test Module



NOTE

Series 2400 instruments can only be used with other Series 2400 instruments connected to the same device. Also, Series 2600B instruments can only be used with other Series 2600B or Model 2651A and 2657A instruments connected to the same device. Additionally, when a Model 2651A or 2657A is connected to the drain of a MOSFET or collector of a BJT or IGBT, the Series 2600B cannot be connected to the Source terminal or Emitter terminal. These terminals must be connected to GND. This is to prevent damaging the Series 2600B SMUs with high current from the Model 2651A or damaging the 2657A with high current from the Series 2600B.

If there is more than one group of Series 2600B SMUs in the system, only the SMUs in group one are available in the SMUs Resource box.

Configure Trace mode

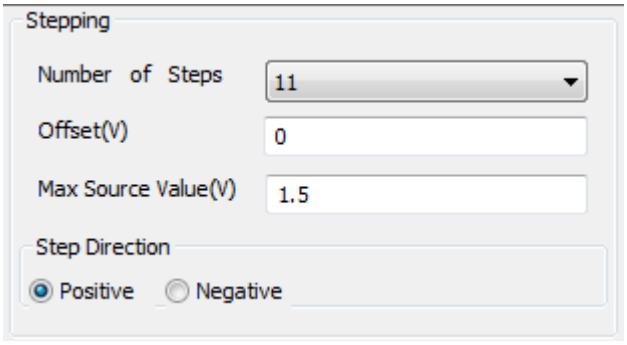
NOTE

ACS Basic has default settings for Trace mode. You can change the settings as needed. If you decide to use the default settings, you can select the **Return to Default Settings** function, which is under the Test Description.

Step setting

See the next figure for an example of the Stepping variables.

Figure 88: Stepping variables



NOTE

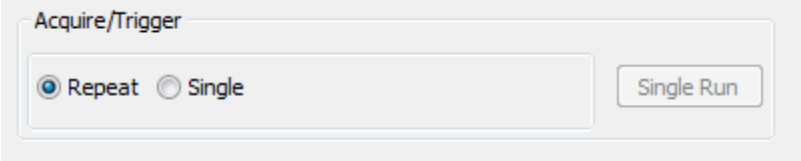
The Step panel is only available in tests that include Step V or Step I as a force function. The default setting is determined if the present test type is sweep on the Collector or Drain terminal and step on the Base or Gate terminal.

- **Number of steps**: Valid from 1 to 11; if the input is 1, a constant bias is applied to the step terminal at the Max Source Value.
- **Offset**: Starting amplitude for the step generator.
- **Max Source Value**: The stop amplitude value of the step; it is always positive. If the test is step V, the units are in volts; if it is step I, the units are in amps.
- **Step Direction**: Indicates the direction in which the source value of the Step SMU changes. When set to positive, the step test executes from the offset value to maximum; when negative, from the offset value to negative maximum.

Acquire/trigger

Select Single to perform the test once. Select Repeat to perform the test until Stop is selected. When Single is selected, the Single Start button is enabled.

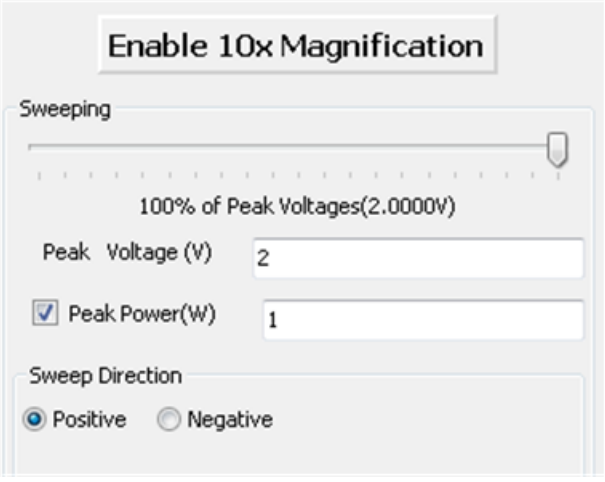
Figure 89: Acquire/Trigger



Sweep control setting

The next figure shows the Sweeping control settings.

Figure 90: Sweep variables



Enable 10X Magnification: Allows you to toggle between enable or disable and is only available during testing. When enabled, the step value will decrease by 10 times, and the function will change to "Disable 10x Magnification." You can disable by selecting the function again.

Sweeping setting

- **Sweep slider**: Controls the actual final value of the sweeping SMU. The slider position is a percentage of the Peak Value. There are always 21 points in the sweep. For example, sliding the sweep slider is similar to turning the knob on a curve tracer. Sliding the slider to the right increases the maximum value while sliding to the left decreases the maximum value. The sweep slider can be controlled in the following ways:
  - Use your mouse to drag the slider. It has a resolution of 1% of the Peak Value.
  - Use your mouse wheel to control the slider. Move the wheel down moves the slider to the left, and move the wheel up moves the slider to the right. It has a resolution of 0.5% of the Peak Value.
  - Use the left and right arrow keys on your keyboard to control the slider. It has a resolution of 0.1% of the Peak Value.
  - Use the up and down arrow keys on your keyboard to control the slider. It has a resolution of 1% of the Peak Value.
- **Peak Voltage:** The absolute maximum stop value for the sweeping SMU. The actual stop value depends on the current position of slider. The units are volts.
- **Peak Power**: Sets the maximum power output at any point of the sweep. Current compliance is calculated at each point of the sweep by peak power/programmed voltage level.

NOTE

Power compliance only affects the SMU designated as the sweeping SMU. For instance, in the `IdVg_StepVd` test, the SMU assigned to the gate is the sweeping SMU. Therefore, power compliance limits the maximum power to the gate terminal of the FET, not the drain terminal.

Stop on Compliance

There are two choices:

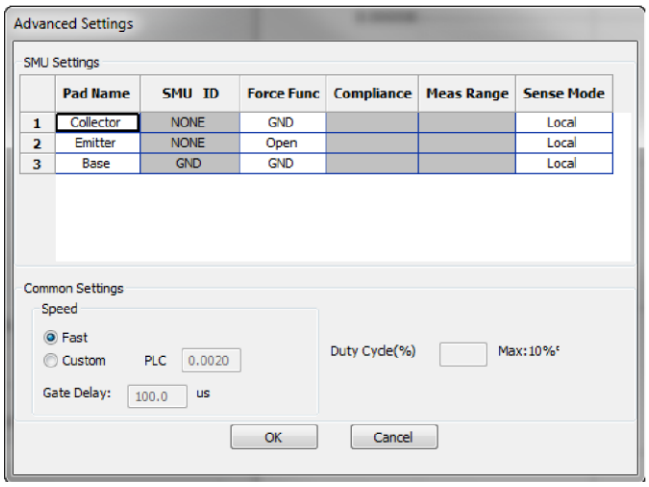
- **Uncheck:** The test will run to completion, even if a SMU enters compliance.
- **Check:** The test will terminate immediately if a SMU enters compliance.

Advanced Settings

The Advanced Settings can be used to change the measure range, number of power line cycles (NPLC), and other settings

Select the **Advanced Settings** function. The dialog shown in the following figure opens.

Figure 91: Advanced Settings dialog



There are several settings that you can adjust:

- **Pad Name:** Can be edited here, but is disabled in the test setup GUI.

NOTE

You see a Warning message if you change a pad name. The warning states that changing the pad name will cause a Pad/SMU mismatch of all related test modules in the library.

- **SMU ID:** Disabled here. It can only be assigned in the device map field in the test setup GUI.
- **Force Func:** Allows you to select SweepV/StepV/StepI/GND/Open from a list. This column is disabled in the test setup GUI.
- **Compliance:** Sets the compliance value for a given SMU. The value is limited by the specifications of the SMU. Note that this specification is not the same in Pulse and DC mode. If the compliance value is only valid in Pulse mode, you cannot select DC as the Force Mode.

- **Meas Range:** Matches the force function if the measure function matches the force function. The measurement range also affects the Force Mode. If the measurement range selected is only valid in Pulse mode, then DC is disabled in the Force Mode field. If a measurement is not assigned to the corresponding pad, the Meas Range cell is disabled.

NOTE

Current measurement ranges in Trace mode may be restricted based upon the SMU to ensure a fast and interactive response in Trace mode. If more sensitive measurement ranges are required, use Single-test Mode or Multitest Mode. For example, for the Model 2636A, the minimum settable range is 100 nA. Therefore, the 10 nA, 1 nA and 100 pA ranges are not accessible in Trace mode.

- **Speed:** Sets the measurement NPLC to Fast or Custom. For Series 2600B SMUs, fast speed is defined as 0.002 PLC, for Series 2400 SMUs, it is defined as 0.01. Refer to "Speed" in the *Automated Characterization Suite (ACS) Fundamental Reference Manual* (document number ACS-914-01) to learn more about the speed area in the common settings.

A custom PLC should be limited according to the specifications of the SMU. When you set up a custom PLC, the pulse width in the test setup GUI should be larger than 2*PLC/local frequency.

To achieve fast speeds, a PLC value less than 0.1 is recommended.

- **Gate Delay:** Sets the delay time after gate bias has been applied. This setting ensures that the gate bias is applied before the drain voltage or current bias. When the Force Mode is Pulse and the test device is a MOSFET or IGBT, the pulse period and the gate pulse width increase by the amount specified in the input value. If the value is set to a value larger than 50% of the pulse width, then the Gate delay is set to 50% of the pulse width.
- **Sense Mode:** Sets the sense mode of the SMU to either Local or Remote. The test modules for IGBTs are all set to remote sense to guarantee measurement accuracy, which is required in high current tests.
- **Duty Cycle:** Displays the maximum allowable duty cycle and the currently selected duty cycle. By default, the duty cycle is set to the maximum possible. Adjust this value to reduce the duty cycle.

	Maximum current	Maximum voltage	Maximum duty cycle
Model 2651A SMU	20 A	40 V	10%
	50 A	20 V	10%
	50 A	40 V	1%
Model 2651As combined in parallel	100 A	18 V	10%
		36 V	1%
Model 2651As combined in series	45 A	20 V	10%
		80 V	1%

For Series 2600B SMUs (non-2651A), the maximum duty cycle is adjusted by the current compliance or maximum source value. If the current value applied for tests requires more than 0.1 A, the maximum value is limited at 1%.

- **DC/Pulse:** Is applied to both the stepping and sweeping SMUs. If pulse mode is selected, the Pulse Width input box is enabled.

NOTE

If the sweeping SMU is a Series 2400, only DC mode can be applied. For a combined 2651A SMU, only Pulse mode can be applied.

- **Pulse Width:** The width of the pulse in pulse mode. The units are microseconds.

  Pulse period defaults to the maximum allowable value. You can adjust the duty cycle in the Advanced Settings GUI. For the Series 2600B SMU (non-2651A), the maximum pulse width is 1000 µs. For the 2651A SMU, if current is in the 10 A to 50 A range, and voltage is in the 10 V to 40 V range, the maximum pulse width is 300 µs. Otherwise, the maximum pulse width is 1000 µs. For the 2651A SMU 100 A/80 V combined SMU, if voltage is higher than 10 V, the maximum pulse width is 300 µs. Otherwise, the maximum pulse width is 1000 µs. Pulse mode cannot be used on Series 2400 SMUs in Trace mode.

⚠ WARNING

In Single Trigger Mode, the SMU output remains on after each test at zero volts. If you want to change connections or insert a new device, first select Stop. When the test has stopped, the SMU output is disabled.

After configuring the Advanced Settings, select **OK**.

The Advanced Settings items are the same as in ITMs in Single-test or Multitest mode.

Run the test module

After the test is configured, select **Run**.

The trace mode test will run repeatedly or once by selecting Single Run.

The test results (plot and data) are displayed in real time in the Test Data area. If testing is set to Repeat, the test data is refreshed repeatedly to show the latest test result. If testing is set to Single, the sweeping test will run when you select the Run icon and the test result is displayed in the Test Data area. Each time you select the Run icon, the test will run and the results will display in the Test Data area.

When the trace mode test is running, you can use the sweep slider to adjust the stop value of the sweep in real time. When you adjust the slider to a certain percentage, such as 60% of the peak value (for example 10 V), the next time sweep will run from start to 60% of 10 V through 6 V which will become the stop value. And the Test Data area will immediately show the results of new sweeping test. Select the Stop icon to end the trace mode test.

Save a test module to the library

The test modules for Trace mode can be saved to the device library. The saved information includes both test settings and test results.

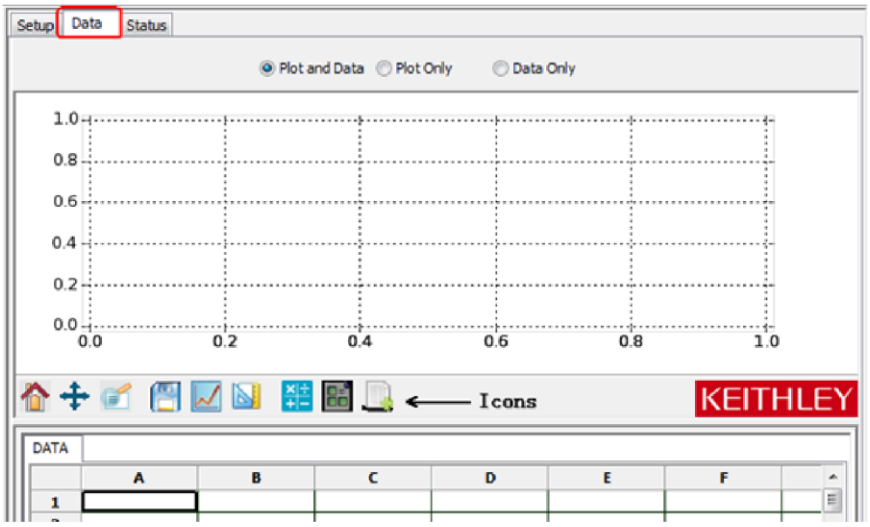
1. Select the Save Test icon on the toolbar; the save test information dialog box opens.
   - Note that the new Test Module name should be different from the existing name to avoid overwriting the original module.
2. Select **OK**. All your changes will be saved to the library.
3. After the test module is saved, ACS Basic returns to the Open Test GUI.

Data plot a graph sheet

The Graph sheet is integrated with the Data sheet. There are two ways to open a Graph sheet:

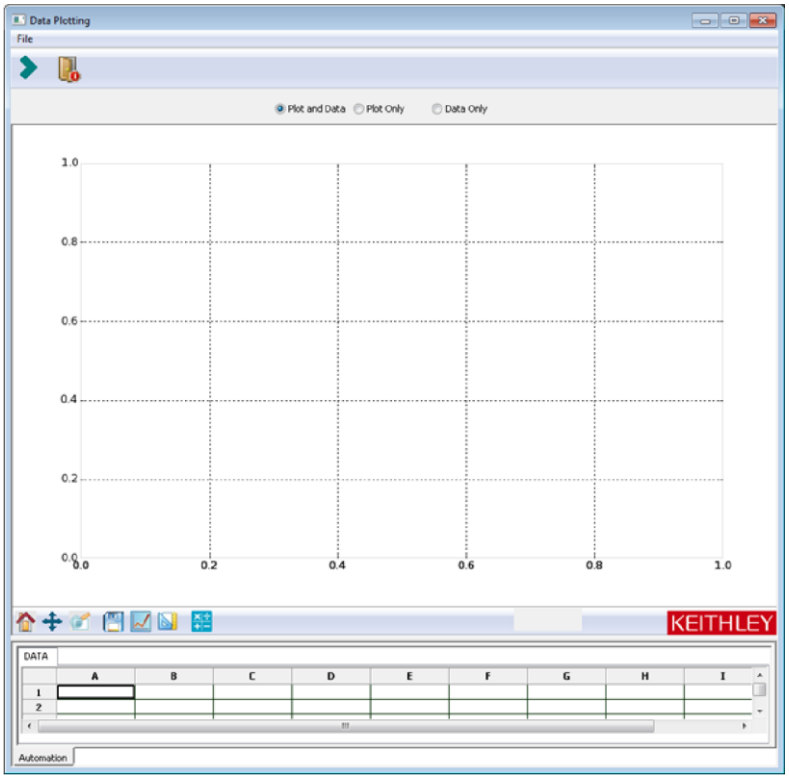
Option 1: When setting up an ITM, STM, or PTM, you can switch to the Graph sheet of the present test by selecting the **Data** tab. Results of the selected test are displayed, and graphs can be drawn according to the results. The following figure shows a graph sheet of a test with a Data sheet at the bottom.

Figure 92: Unconfigured graph sheet of a test



Option 2: Select **Tools** on the menu bar, then choose Offline Data Plotting. The Data plotting tool dialog opens (see the following figure). Select a blank Data Plotting sheet similar to the one shown in the previous figure. Data and graphs on this sheet are not limited to the current test or the current project. Select the import icon to import data. Select the exit icon to exit the page.

Figure 93: Data plotting tool



Control the properties of the graph

Items in the Graph Settings menu and the Graph toolbar control the properties of the graph.

Toolbar icons

The toolbar is between the graph section and the data section and contains the following icons:

- **Home:** Returns the graph to its original size (after zoom).
- **Pan:** Permits panning of the graph.
- **Zoom:** Enlarges a small, selected part of the graph.
- **Save:** Saves the plot as a separate graphics file. Select the Save icon to display the Save to file dialog. You can save the plot as a `.png`, `.ps`, `.eps`, `.csv`, or `.svg` format.
- **Plot:** Creates a graph out of the data selected (see Define a graph and plotting (on page 4-31) for more information).
- **Plot Auto Scale:** Autoscales all the graphs on the graph sheet.
- **Formular**: Performs data calculations on test data and on the results of other Formulator calculations (see Formulator (on page 4-43) for more information).
- **Save Reference Data**: Saves reference data in the selected format and location. You can specify where the data is saved by going to Tools>Preference>Reference Data Path.
- **Load Reference Data**: Loads reference data to compare to new data.

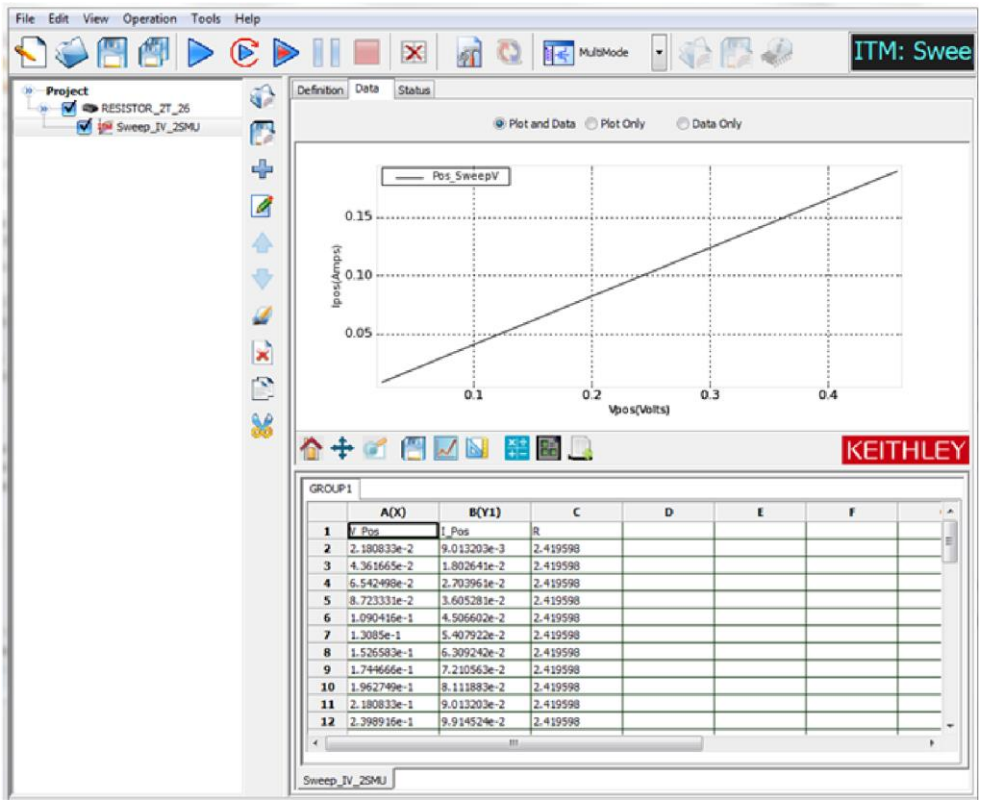
NOTE

New test data must be run before Reference Data can be loaded and plotted. The following specifies how to save and load reference data.

Save and load reference data:

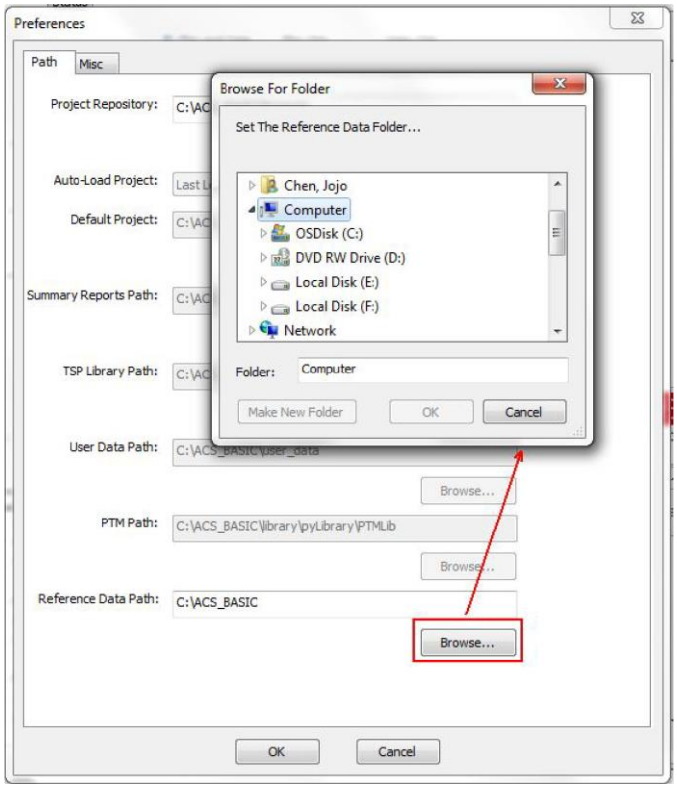
1. Run a test and collect test data (see next figure).

Figure 94: Run and collect test data



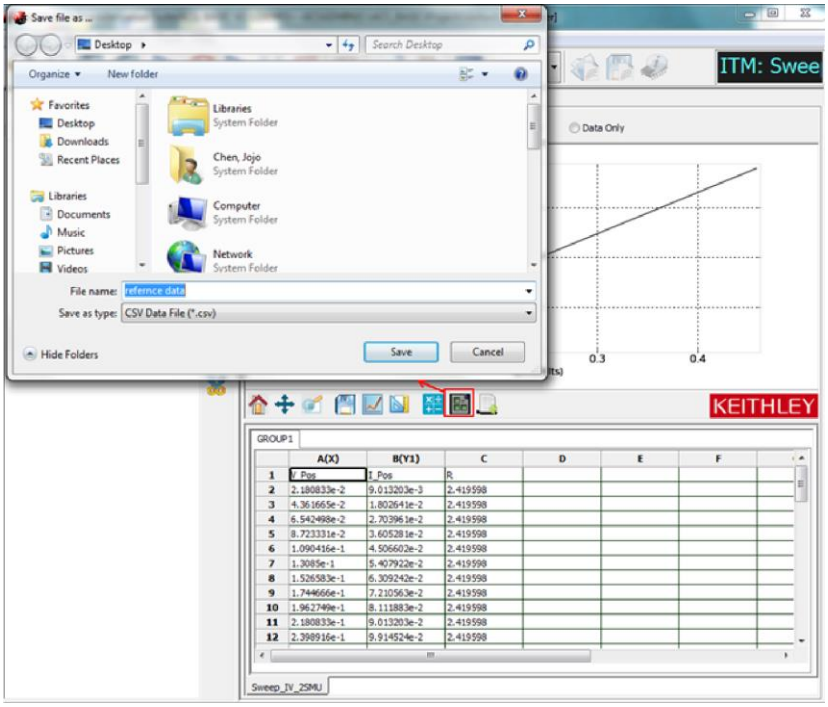
2. Once the test has completed, go to Tools>Preference>Reference Data Path and choose where you want to save the data (see next figure).

Figure 95: Choose Reference Data Path



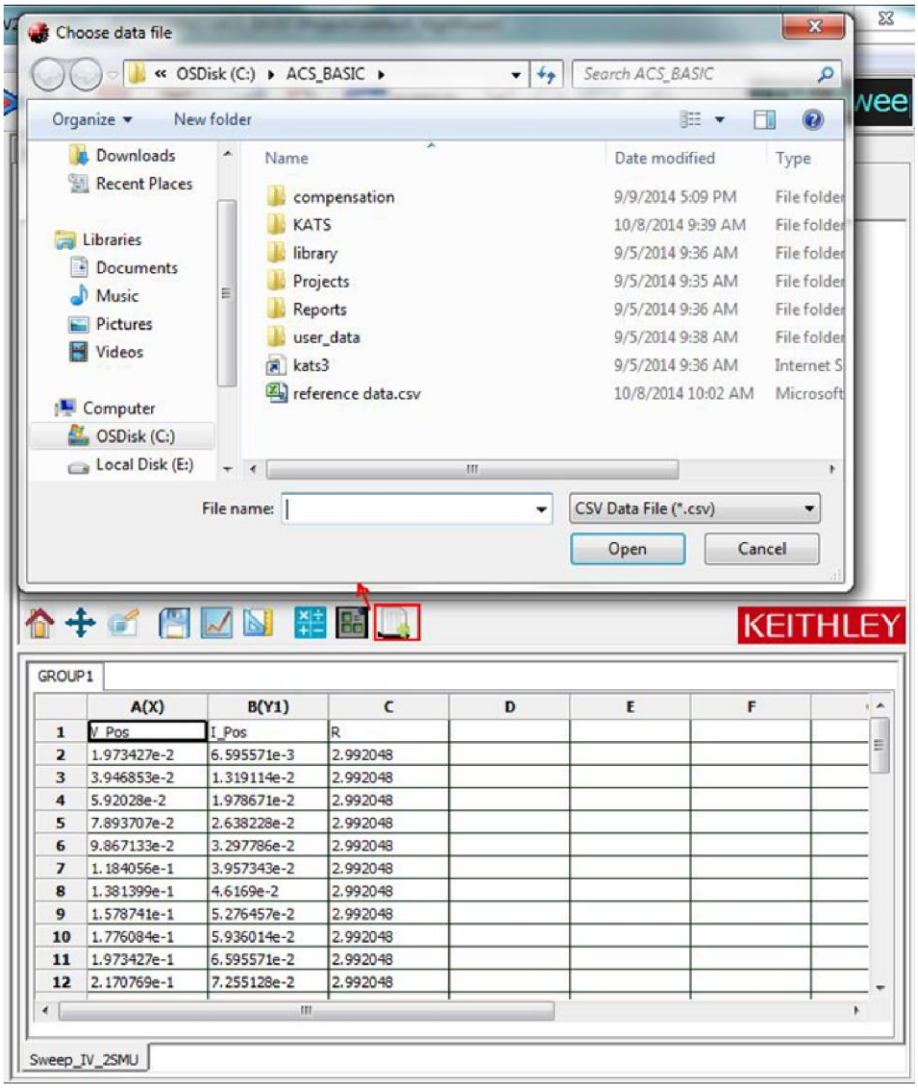
3. Select the **Save Reference Data** icon.

Figure 96: Click the Save Reference Data icon



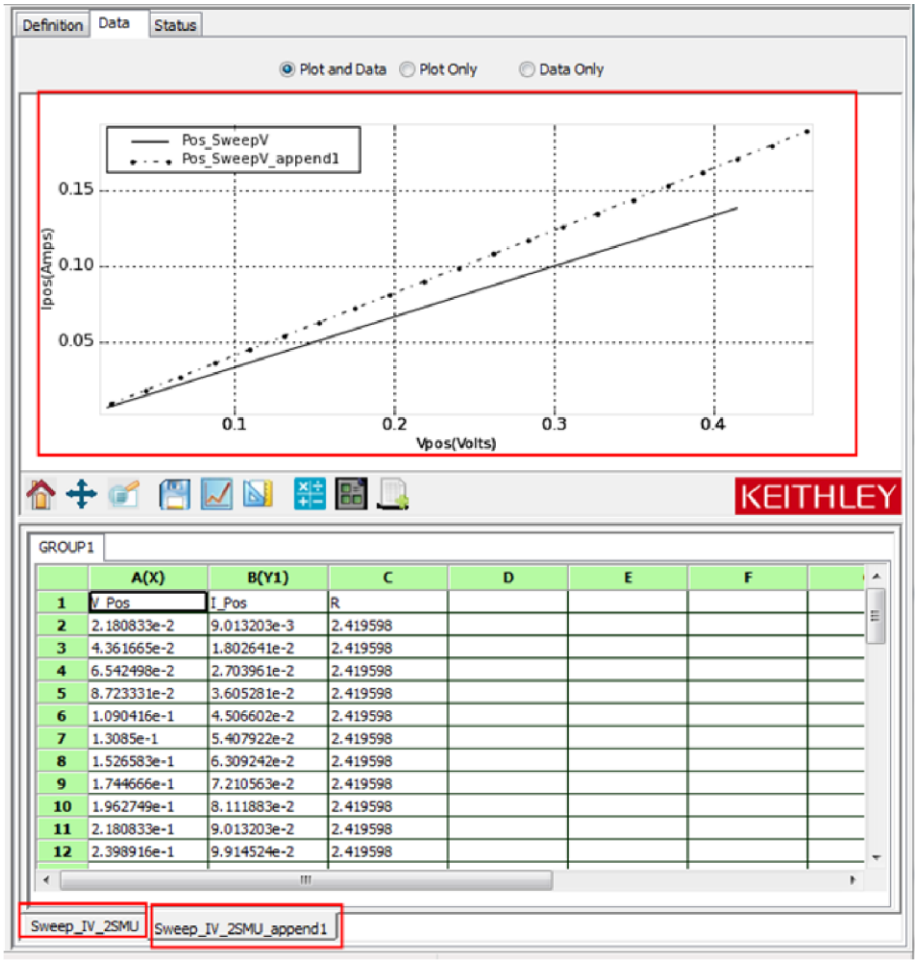
4. To load and run a test that you previously saved, select the **Load Reference Data** icon (see next figure).

Figure 97: Click the Load Reference Data icon



Any new data is appended to your test results, as shown in the following figure.

Figure 98: ACSBE Append Reference Data

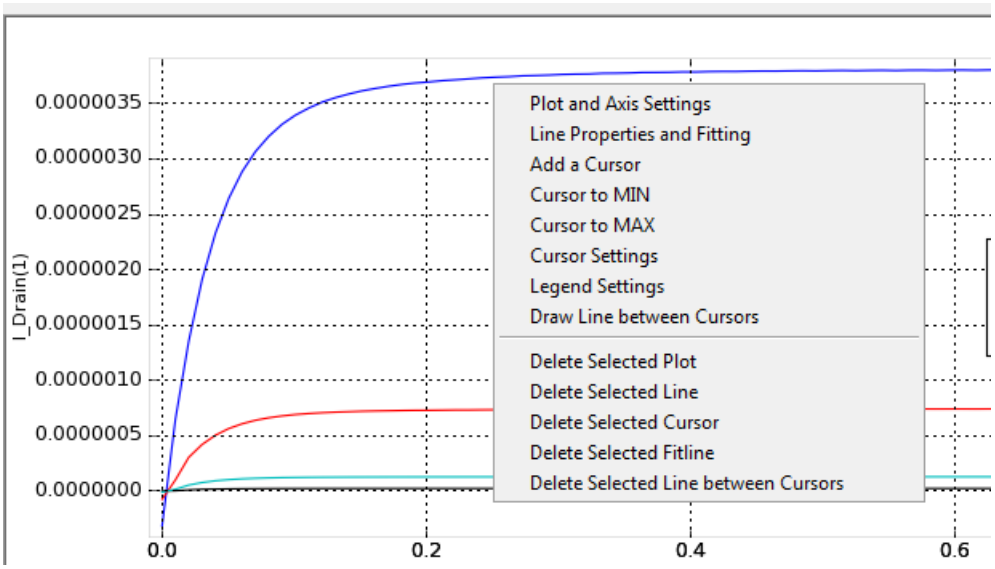


	A(X)	B(Y1)	C	D	E	F
1	V_Pos	I_Pos	R			
2	2.180833e-2	9.013203e-3	2.419598			
3	4.361665e-2	1.802641e-2	2.419598			
4	6.542498e-2	2.703961e-2	2.419598			
5	8.723331e-2	3.605281e-2	2.419598			
6	1.090416e-1	4.506602e-2	2.419598			
7	1.3085e-1	5.407922e-2	2.419598			
8	1.526583e-1	6.309242e-2	2.419598			
9	1.744666e-1	7.210563e-2	2.419598			
10	1.962749e-1	8.111883e-2	2.419598			
11	2.180833e-1	9.013203e-2	2.419598			
12	2.398916e-1	9.914524e-2	2.419598			

Access the graph settings menu

To view the Graph Settings menu, right-click any portion of the graph (see next figure).

Figure 99: Graph settings menu



The Graph Settings Menu functions are summarized below.

Plot and Axis Settings: Sets general properties of the selected graph and the parameters of the axis. Many sub-functions are provided. For more information, refer to Define a graph and plotting (on page 4-31).

Line Properties and Fitting: Sets the properties of a selected line and makes the fitting if the cursors are set. For more information, refer to Define a graph and plotting (on page 4-31).

Add a Cursor: Adds a floating cursor to the location where you have your mouse pointer.

Cursor to MIN: Moves the selected cursor to the minimum point of the attached curves.

Cursor to MAX: Moves the selected cursor to the maximum point of the attached curves.

Cursor Settings: Adds a cursor to a graph and sets the cursor properties. For more information, refer to Define a graph and plotting (on page 4-31).

Draw Line between Cursors: Draws a linear line between two selected cursors.

Delete Selected Plot: Deletes the plot below where the menu is displayed from the graph sheet and resizes all other plots.

Delete Selected Line: Deletes a selected line from the plot. To delete a specific line, right-click the line and choose Delete Selected Line from the menu.

Delete Selected Cursor: Deletes the selected cursor. To delete a cursor, right-click the cursor and choose Delete Selected Cursor in the menu.

Delete Selected Fitline: Deletes the selected fitline cursor. To delete a fitline, right-click the fitline and choose the Delete Selected Fitline in the menu.

Delete Selected Line between Cursors: Deletes the selected line between the two cursors. To delete a line between cursors, right-click the line and choose the Delete Selected Line between Cursors in the menu.

Define a graph and plotting

There are two steps to define a graph and plot the data:

1. Define the data to be graphed.
2. Plot the data.

Define the data to be graphed

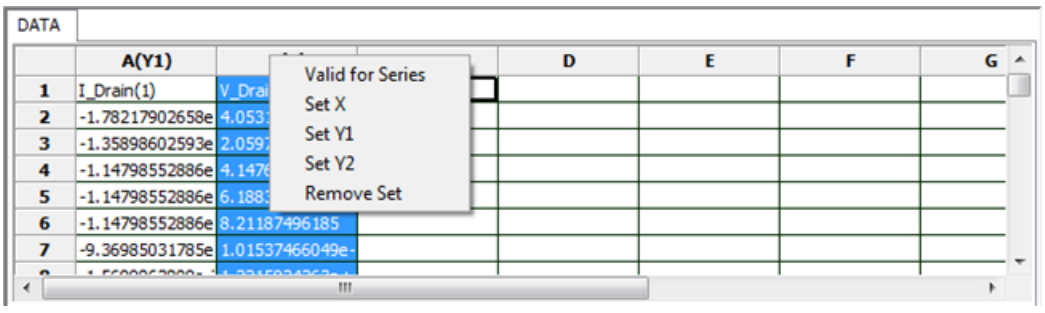
There are two steps to define and graph data:

1. Assign the data to an axis.
2. Assign the axis to a plot.

Assign data to an axis

Right-click the title of any column in the datasheet to display the graph definition menu (see next figure).

Figure 100: Graph definition menu



Graph definition menu descriptions:

- **Valid for Series**: Used when your test is multistep and there are a series of results (see previous figure). If Valid for Series is selected, the settings for any column are valid for all other columns in the same series. For example, if column B (I_Drain(1)) is set as X, column D (I_Drain(2)) is also set as X.
- **Set X**: Sets the selected column as the X Axis on the graph.
- **Set Y1**: Sets the selected column as the Y Axis on the left side of the graph.
- **Set Y2**: Sets the selected column as the Y Axis on the right side of the graph.
- **Remove Set**: Deletes the selected column from the plot.

To draw a graph, at least one X and one Y should be set.

NOTE

ACS Basic supports multiplot using one datasheet. This allows different columns to be plotted to different graphs.

Assign the axis to a plot

After an axis is set, you must choose the data you want plotted on the graph. To choose data, select the title of a column (the column highlights).

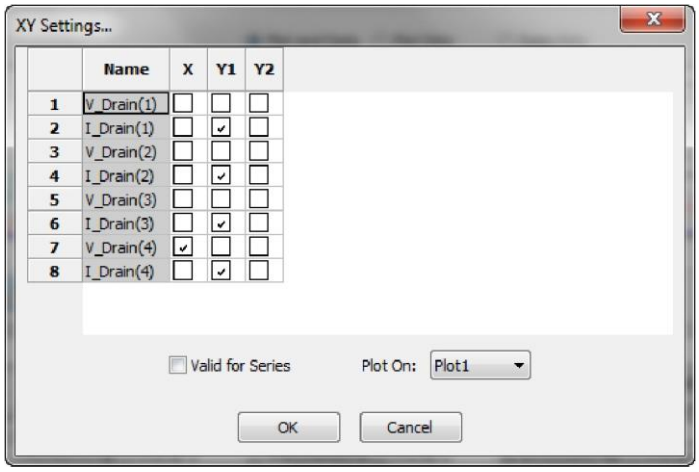
To choose more than one column, drag the mouse over the columns, or execute a Ctrl + on the column. At least one X and one Y axis (either Y1 or Y2) must be selected. The following figure shows the results of choosing multiple columns.

Figure 101: Result of data selection

	A(X)	B(Y1)	C(X)	D(Y1)
1	V_Collector(1)	I_Collector(1)	V_Collector(2)	I_Collector(2)
2	0.000000e+000	1.836736e-007	0.000000e+000	2.466173e-007
3	-2.000000e-002	-5.159855e-006	-2.000000e-002	-7.082264e-006
4	-4.000000e-002	-1.381116e-005	-4.000000e-002	-1.888655e-005
5	-6.000000e-002	-2.545521e-005	-6.000000e-002	-3.462865e-005
6	-8.000000e-002	-3.779190e-005	-8.000000e-002	-5.113309e-005
7	-1.000000e-001	-4.814865e-005	-1.000000e-001	-6.482606e-005
8	-1.200000e-001	-5.537796e-005	-1.200000e-001	-7.429314e-005
9	-1.400000e-001	-5.985659e-005	-1.400000e-001	-8.010770e-005
10	-1.600000e-001	-6.244343e-005	-1.600000e-001	-8.345355e-005

If data has not been selected, select the Plot icon and the XY setting dialog opens. Select the settings that you want to plot at one time (see the following figure).

Figure 102: XY setting dialog



Plot the data

Select the **Plot** icon on the toolbar to display the Plotting menu.

After choosing the data that you want to plot, choose one of the following commands:

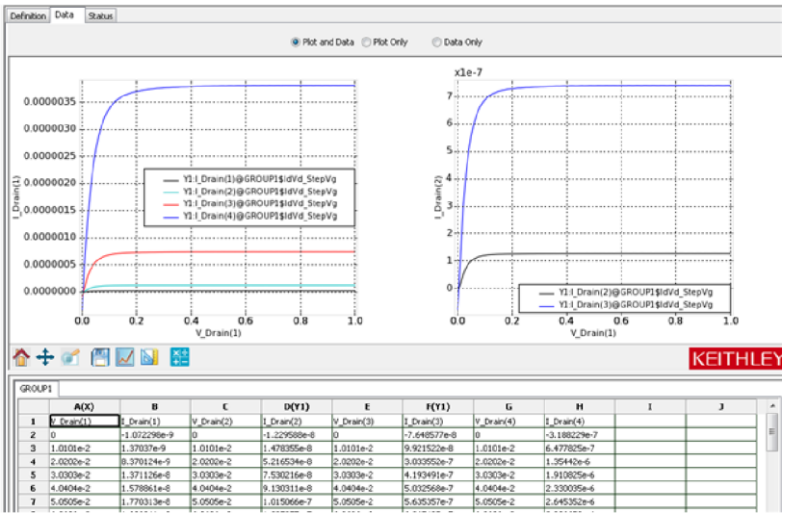
New Plot: Draws a graph on a new plot separate from the existing one.

Plot1: Draws a graph on plot1, which already exists. If plot1 is not blank, the original lines will remain instead of being deleted.

NOTE

If the test is run on multiple groups, the results of all the groups under test are shown on one plot (see next figure).

Figure 103: Plot shows two ids_vd plots

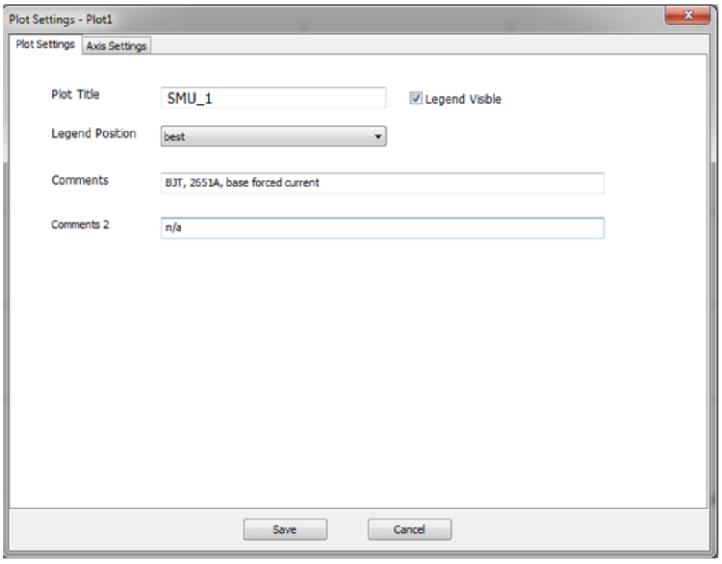


Define properties of a graph and axis

Right-click any portion of the graph to be edited, then choose the **Plot and Axis Settings** command in the Graph Settings menu. The Plot Setting dialog opens. It contains the **Plot Setting** and **Axis Setting** tabs, as shown in the following figures.

Plot setting tab

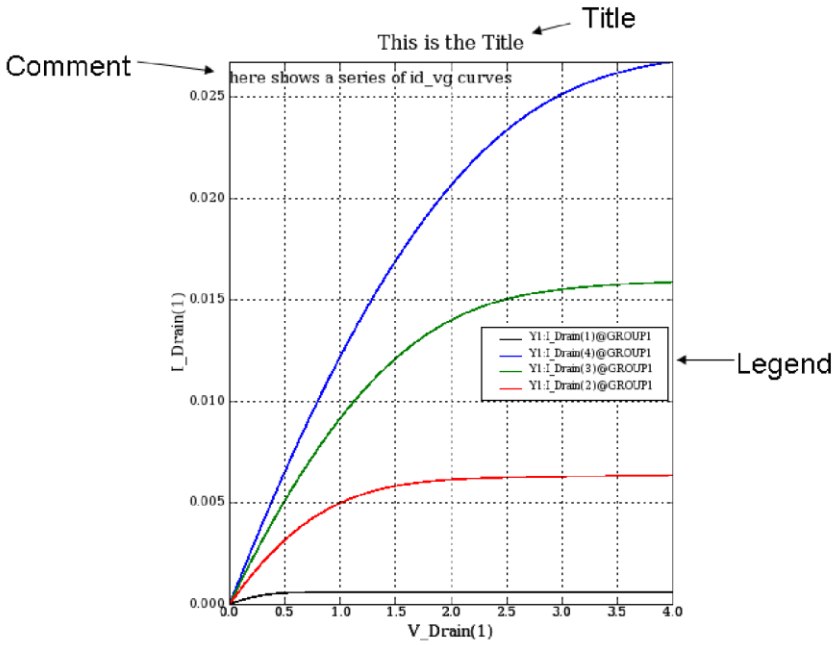
Figure 104: Plot Settings tab



- **Plot Title**: Sets the title of the selected plot.
- **Legend Visible**: Causes the legend to appear on the plot when selected.
- **Legend Position**: Selects the position for the legend on the plot. There are many positions available. However, the recommended position choice is "best."
- **Comments**: Allows you to write comments which will be displayed on the selected plot.

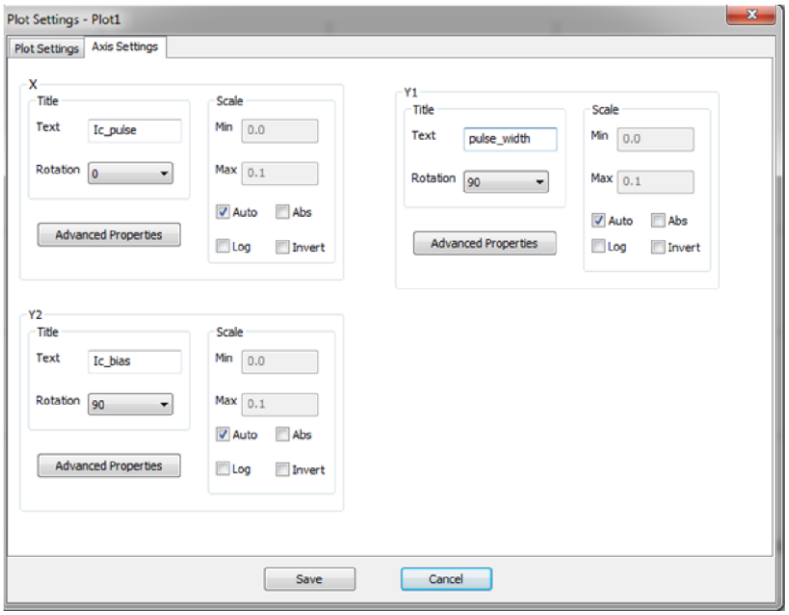
The next figure is an example of a plot with a title on top, legend on the right, and comments on the top left corner.

Figure 105: Results from the plot settings tab



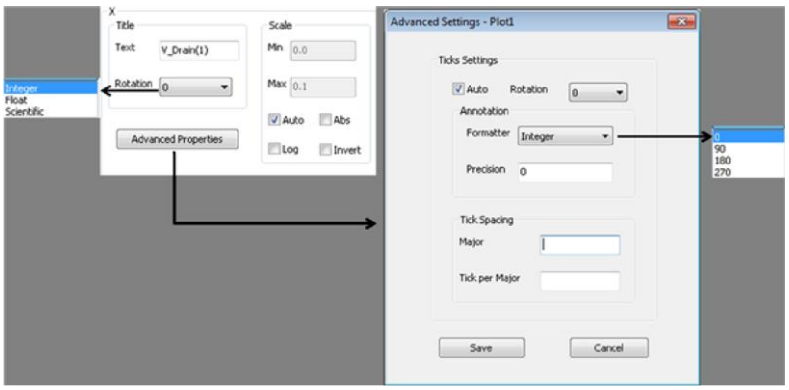
Axis setting tab

Figure 106: Axis Settings tab



The axis setting tab contains three areas that correspond to the settings of the X, Y1, and Y2 axes. The items in these three areas are the same for each axis. Only one axis is described in the following figure. It shows all the items and sub-items for one axis on the Axis Settings tab.

Figure 107: Items on the axis settings tab



Title section

Sets the properties for the axis title. The options are:

- **Text:** The name of the axis.
- **Rotation:** Rotates the title if necessary. The X-axis default is zero. The Y1 or Y2-axis default is a 90º rotation.
- **Scale section:** Sets the scale of the chosen axis.
- **Min:** Sets the minimum value of the axis. Valid when Auto is cleared.
- **Max:** Sets the maximum value of the axis. Valid when Auto is cleared.
- **Auto:** Autoscales the axis to fit the data. Auto is the default setting.

NOTE

To specify exact numbers for the minimum and maximum values of the scale, clear Auto and enter the values in the Min and Max text boxes.

- **Abs:** Plots the absolute value of the data along the axis.
- **Log:** Arranges the axis in a logarithmic format if selected. If not selected, the axis is arranged in a linear format. The default setting is cleared (linear format).

NOTE

Log format can only be applied to positive data. If there is negative data, select **Abs** to allow Log format.

- **Invert:** Changes the minimum value of the axis to the maximum value and the maximum value to the minimum value.

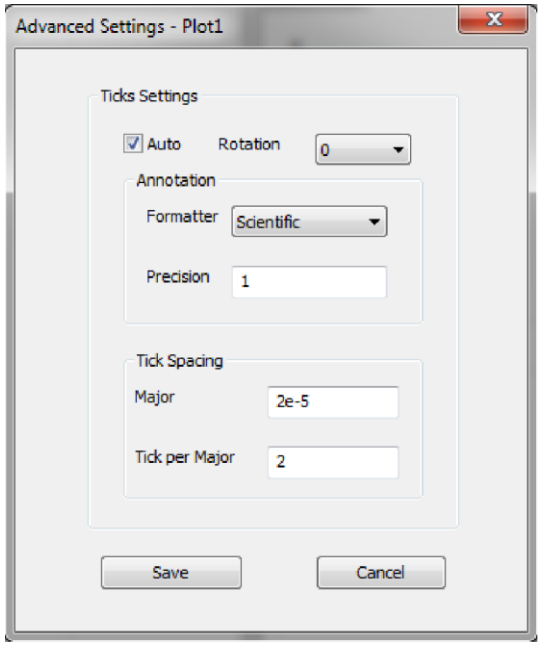
Advanced Properties

Select the **Advanced Properties** button (see next figure) to open the dialog. Items in the Advanced Setting dialog are as follows:

- **Auto:** Sets the ticks (scale) automatically; the other settings are ignored. This is selected by default.
- **Rotation:** Rotates the ticks if necessary. Zero is set by default.
- **Formatter:** Sets the format for the annotation of the ticks. Three formats are provided: Integer, Float, and Scientific.
- **Precision:** Sets the precision of the annotation. Valid when Float or Scientific format is selected.
- **Major:** Sets the value between major ticks along the axis.

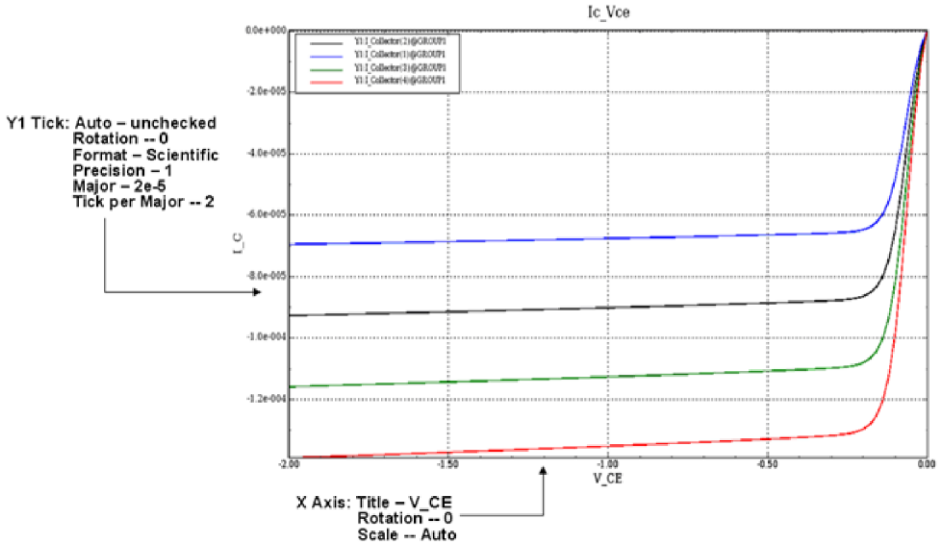
- **Tick per Major:** Sets the number of subticks inside a major tick.

Figure 108: Plot Settings Advanced Properties X axis



The following figure shows a series of Ic_Vce curves with marked features.

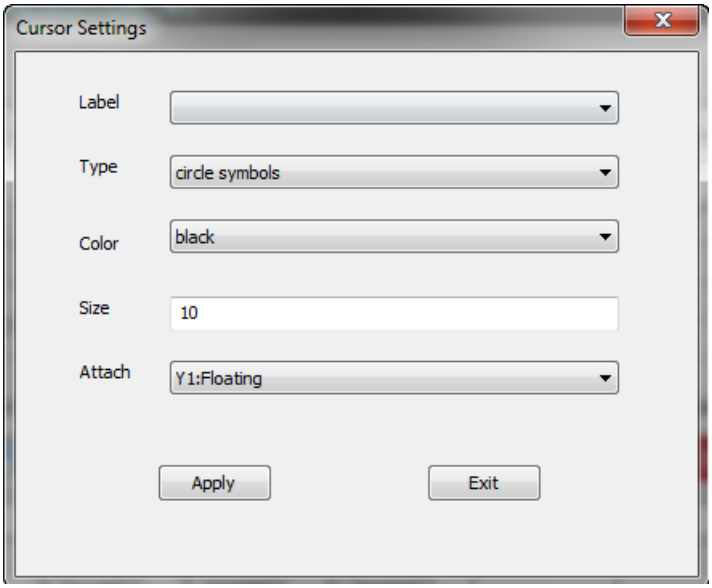
Figure 109: Ic_Vce curves example



Define a cursor

Right-click any portion of the plot and choose Cursor Settings from the Graph Settings menu. The Cursor Settings dialog opens. The following figure shows the Cursor Settings dialog and cursors on the I_V curve of a diode.

Figure 110: Cursor Settings dialog



Items in the Cursor Setting dialog are as follows:

Label: Selects an existing cursor name. Choose a cursor from the list.

Type: Selects the shape of the chosen cursor.

Color: Selects the color of the chosen cursor.

Size: Changes the size of the chosen cursor

Attach: Allows you to select between Floating or a specific curve. If set to Floating, the cursor can go anywhere on the plot, otherwise it can only run along the selected curve.

Apply: Applies the settings for the chosen cursor.

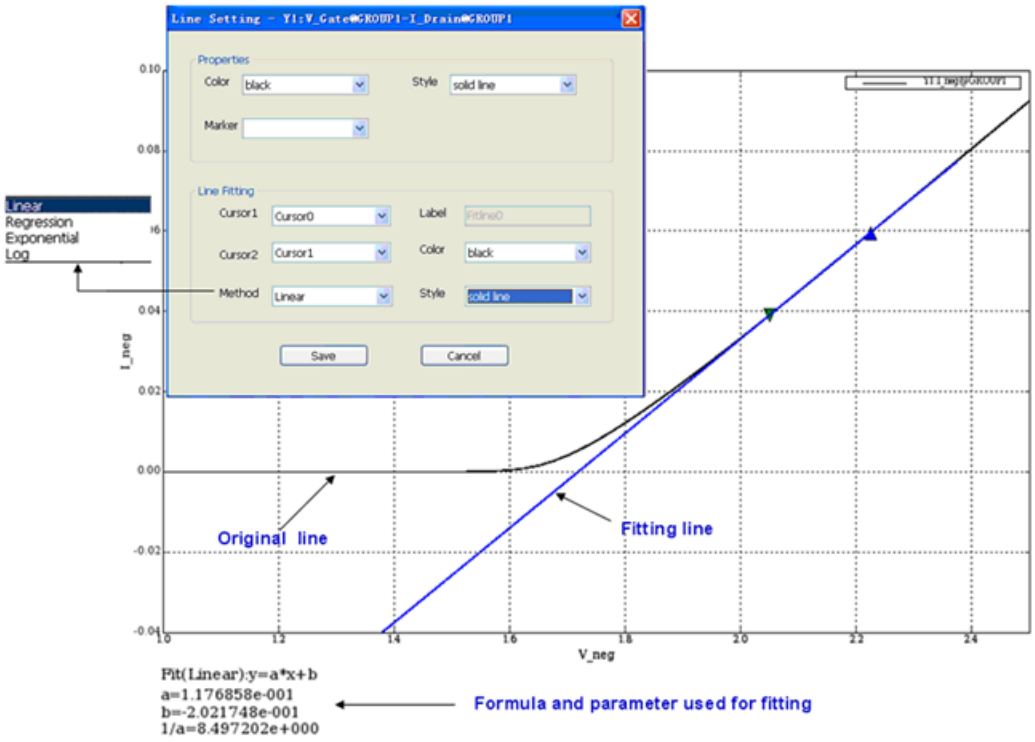
Exit: Exits the dialog. If Apply is not selected before you Exit, all the settings are ignored.

After selecting **Apply**, the cursor displays on the plot. If you select the cursor, the coordinates (X, Y) of the current position display on the bottom plot (see previous figure).

Line properties and fitting

Select the line to be edited, then right-click, and select the **Line Properties and Fitting** command in the Graph Settings menu. The Line Settings dialog opens (see next figure).

Figure 111: I_V curve of a diode with the linear fitting line and the formula used



The previous figure shows the original line (black) together with the fitting line (blue) without features. The formula used is displayed at the bottom left corner of the plot.

The Line Setting dialog contains several items:

Properties

- **Color:** Allows the user to choose the color of the selected line.
- **Style**: Selects the style of the selected line: solid, dashed, dotted, or dash-dot.
- **Line Width**: Sets the width of the selected line.
- **Marker**: Selects the type marker for each data point on the line. There are no markers by default.
- **Marker Color**: Selects the color of the markers.
- **Marker Size**: Selects the size of the markers.

Line fitting

- **Cursor1**: Selects the first cursor to define the line segment to be fitted (pick from the list).
- **Cursor2**: Selects the second cursor to define the line segment to be fitted (pick from list).
- **Method**: Selects the function used to fit the segment of the line (pick from Linear, Regression, Exponential, or Log).
- **Label**: Shows the fit line label.
- **Color**: Selects the fit line color (choose from the list).
- **Style**: Selects the fit line style: solid, dashed, dotted, or dash-dot.

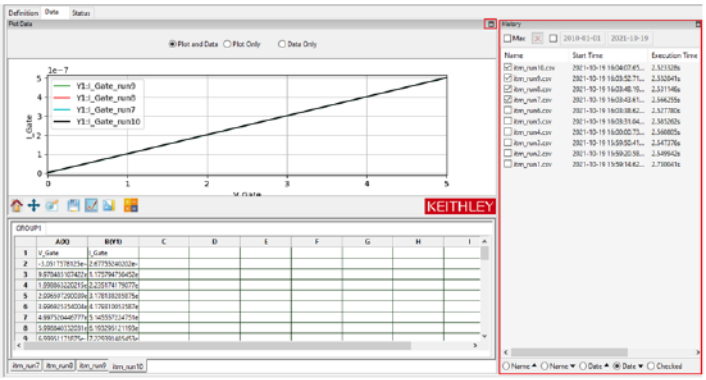
NOTE

An original line can only have one fit line. If the number of fit lines on the graph sheet is two or more, only one formula displays.

History data

The history data shows data generated from running tests. You can view the data by selecting the enlarge button of the Plot and Data section.

Figure 112: History Data panel

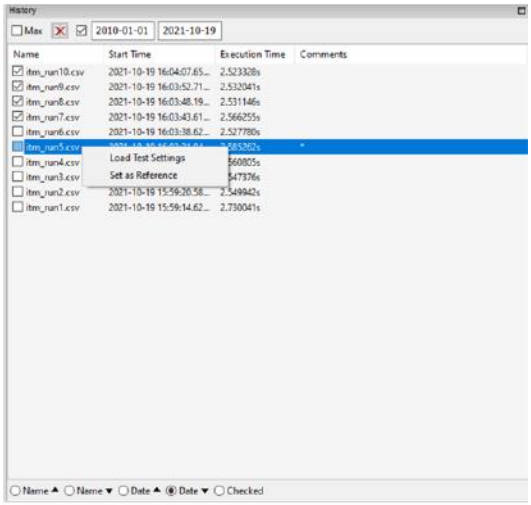


When you select the run button, a new page of test data is appended in the Data panel. The new data is named `testname_runX` (*X* is the count of runs). Each test adds a new plot of test data that is overlayed in the Plot panel. The history data is added in the History section, which is named `testname_runX.csv` (*X* is the count of runs).

The history data item includes the name and information about start time, execution time, and comments. You can edit the comments of each history data item in the interface.

When Max is selected, the first five pages from the run history are displayed. When you select or clear the history data item, the corresponding data and plot can be viewed or hidden in the data and plot area. For example, in the following figure, `itm_run1.csv` is cleared and is not displayed.

Figure 113: History data



There are two parameters that contain test history data. These files are in the directory `C:\ACS\KATS\ACS_settings.ini`.

max_pages:	The number of pages that can be selected and displayed in the plot data panel.
max_histories:	The amount of history data that can be saved.

Figure 114: Parameters settings history data

```
[run_history]
max_pages = 5
max_histories=50
```

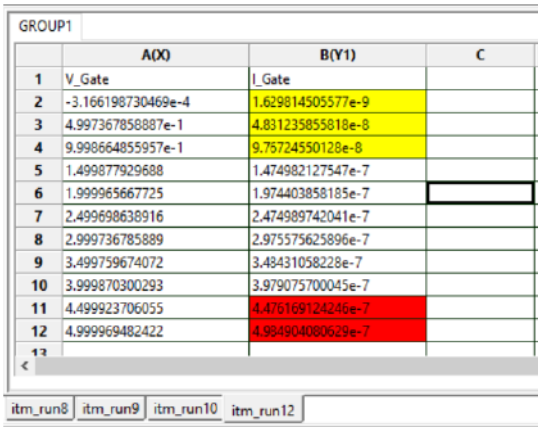
- **Test settings:** To load test settings from your previous testing right-click a history data item and choose **Load Test Settings**. This allows you to use the test settings on the current test module (refer to the previous figure).
- **Reference:** To create a reference, you can right-click a history data item and choose **Select as Reference**. This reference is used for all the selected history data on the Data and Plot area (refer to the previous figure).
- **Date:** To see the date, select the box next to the red X delete button. You can filter the dates by selecting a range of dates (refer to the previous figure).

- **Sort data:** You can sort your test data by name, date, selected or cleared in ascending or descending order.
- **Delete:** To delete a history data item, select the item you want to delete, then select the red X delete button (refer to the previous figure).

Fail data

If a test result is out of specifications for Spec Low or the Spec High range, as defined in the Pattern-Limits page, the background color of the data cell changes to the Manual Run Fail Color Setting defined in the **Tools > Preference > Advanced** panel. For example, in the following figure, the yellow cells indicate the specifications are lower than expected and the red cells indicate they are higher.

Figure 115: Background color fail data



Formulator

The Formulator is accessible in the Data tab. It allows you to perform data calculations on test data and on the results of other Formulator calculations. The Formulator provides a variety of computational functions, common mathematical operators, and common constants.

A formula created by the Formulator is an equation composed from a series of functions, operators, constants, and arguments. For more detailed information about the Formulator functions, refer to "Formulator function reference" in the *Automated Characterization Suite (ACS) Fundamental Reference Manual* (document number ACS-914-01).

Formulator arguments and constants

A formula created using the Formulator function performs calculations on any combination of the following:

- ITM, STM, or PTM test data
- Secondary data created by other Formulator formulas
- Standard constants for the list of constants

Some of the functions operate on columns of values (vectors) only. Others operate on single values (scalars) only. Still others operate on both single values (scalars) and columns of values (vectors).

The results of some calculations may be a column of values (vector) or a column containing only a single value.

Real-time functions, operators, and formulas

The Formulator provides a variety of functions and operators. Some of these may be used for real-time, in-test calculations for ITM data. Others may be used only for post-test data computations.

A formula containing exclusively real-time operators and functions is a real-time formula. If a real-time formula is specified as part of an ITM definition, it runs for each data point generated by the ITM just after the point is generated. The results of a real-time formula can be viewed in the Data sheet or plotted during the test in the same way as test data.

NOTE

Real-time calculations do not apply to PTM data. A PTM Data sheet does not update until the test is completed.

The following operators and functions are real-time operators and functions:

1. Operators: +, -, *, /, ^ (exponential)
2. Functions: ABS, DELTA, DIFF, EXP, LN, LOG10, SQRT

Real-time formulas run as follows:

1. If a real-time formula is created before the ITM has been run, the formula runs automatically during each ITM run.
2. If a real-time formula is created after an ITM has been run, the formula runs initially upon adding it to the ITM and automatically during each subsequent ITM run.

Post-test only functions and formulas

A formula containing any one (or more) of the remaining Formulator functions is a post-test-only formula. It executes only at the end of each run of the ITM, STM, or PTM in which the formula is defined.

The results of a post-test-only formula may be viewed in the Data worksheet or plotted only at the end of a test.

The following functions are post-test-only functions:

AT	AVG	EXPFIT	EXPFITA	EXPFITB
LINFIT	LINFITSLP	LINFITXINT	LINFITYINT	MAX
MAXPOS	MIN	MINPOS	REGFIT	REGFITSLP
REGFITXIN	REGFITYINT	TANFIT	TANFITSLP	TANFITXINT
TANFITYINT	VTLINGM	VTSATGM		

The formula below is a post-test-only formula and MAX is a post-test-only function:

```
RESULT2 = MAX(ABS(DELTA(I_ drain)))
```

Post-test-only formulas execute as follows:

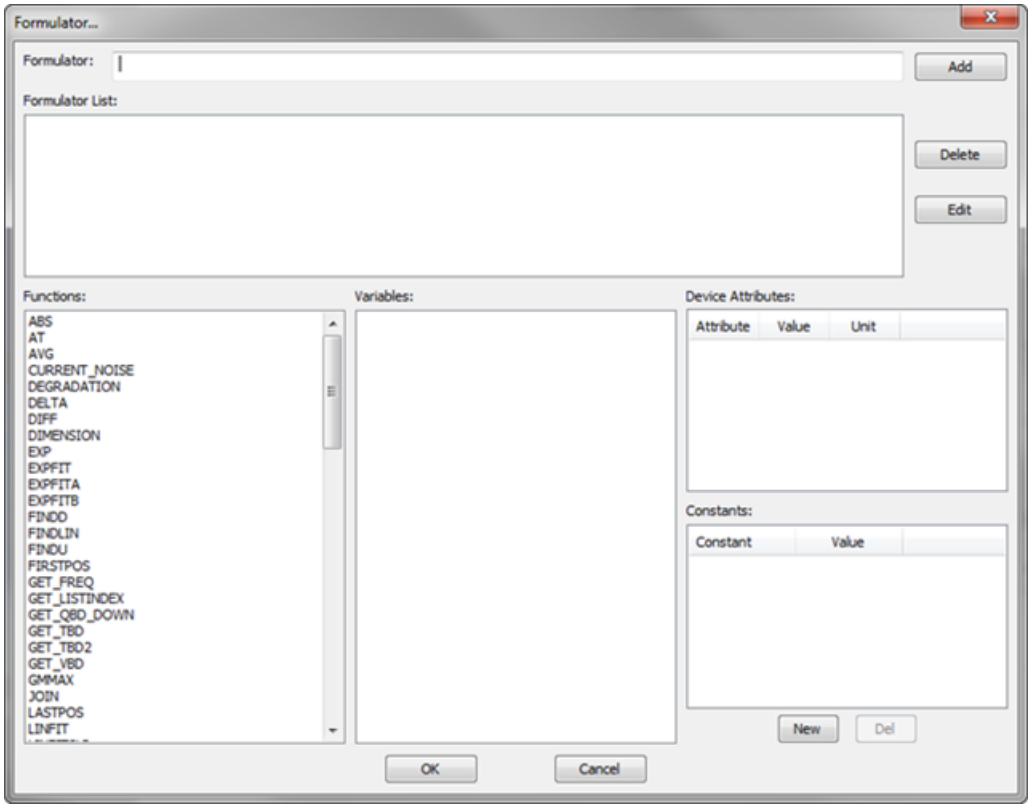
- If a post-test only formula is created before the ITM, STM, or PTM has been run, the formula performs automatically at the conclusion of each run.
- If a post-test only formula is created after an ITM, STM, or PTM has been run, the formula performs initially upon adding it to the ITM, STM, or PTM and automatically at the conclusion of each subsequent run.

Start the Formulator function

To open the Formulator:

1. Open the data tab for the test that you want to analyze.
2. Select the **Formulator** function and the dialog opens (see next figure).

Figure 116: Formulator dialog



The Formulator dialog:

- **Formulator**: Use to create new formulas or edit existing formulas.
- **Formulator List**: Displays previously created formulas for ITMs, STMs, or PTMs.
- **Add**: Starts the calculation for the formula in the Formulator, and moves the formula to the Formulator List.
- **Delete**: Deletes a formula when the formula is selected in the Formulator List.
- **Edit**: Allows you to edit the formula that is currently selected in the Formulator List.
- **Functions**: Displays a list of functions. When you select a function, it is added to the equation and displayed in the Formulator.
- **Variables**: Lists the names of all columns in the Data sheet. Columns created by some functions may contain only a single value.

NOTE

Do not use X, Y1, and Y2 as variables in ACS Basic. These are reserved variables.

- **Constants**: Is where you can insert each constant or value by name. When you select the constant in the constants list, it is added to the equation in the Formulator.
- **New**: Opens a dialog input that allows you to enter the name and value of the new constant. After you enter the name and value, select OK; the new constant is added to the constants list.
- **Del**: Deletes the selected constant

The Formulator functions

The Formulator functions, operators, and constants can be used individually or combination to create simple or complex analysis equations.

Multiple functions can be nested. For example, in one equation you can calculate the following:

1. Find the maximum value of a column using the MAX function.
2. Find the absolute value of the maximum using the ABS function.
3. Multiply the ABS value by a constant.

The equation below illustrates the use of nested Formulator functions:

```
MAXABS = 10*ABS(MAX(ColumnA))
```

The number of levels of nesting is unlimited. The purpose, format, and arguments for these functions and all other functions available in the Formulator are described in Formulator arguments and constants (on page 4-44).

Advanced operations

This section describes the following topics:

- Use an existing project
- Create a new device
- Add a new device
- Add a blank test
- Set preferences
- Run CVU connection compensation
- Graphically design a new device

Projects

This section describes how to use an existing project in the standard ACS Basic library and how to build a new project.

CAUTION

The software connections must accurately reflect the physical hardware connections at the time the test is run. Incorrect configurations result in test errors, and possibly device damage. At every test startup, you must check that the connections in the device view of the ACS Basic interface match the physical connections between the test instruments and the device.

An existing project

Once started, ACS Basic automatically opens the previously opened project by default.

To view the default project:

1. Select the **Preference** item in the Tools menu. The Preferences dialog opens.
2. On the Path tab, you see the Auto-Load Project and the Default Project.

To open a project other than the default:

1. Select the **Open** icon on the toolbar of the ACS Basic GUI.
2. Select the project file you to open.
3. Select and open the `.xml` file project (for example, `trace_mode_lab.xml`).

NOTE

If SMUs are required in the project you are trying to open, the ACS Basic software tries to detect them in the present configuration. Once the SMUs are found, the project loads the project in the configuration navigator.

If the SMU connections for this project do not match the present physical connections, the SMUs Missing dialog opens.

The SMUs Missing dialog informs you that some of the SMUs for a project cannot be found. To correct this, check all module settings in the project and reassign them where necessary until the SMUs Missing GUI no longer displays when opening an existing project.

Create a new project

To create a new test project, refer to Build a project plan (on page 4-12).

Create a new device

Each device has a specific device configuration. The Device Settings GUI is the interface used to set the connections of the device.

For Single-test mode, in the Definition tab of the Test GUI, you can select the Device View icon to view the Device Settings GUI of the currently open test module. For Multitest mode, select the device node on the configuration navigator, and the Device Setting GUI opens (see next figure).

There are two tabs in the device workspace:

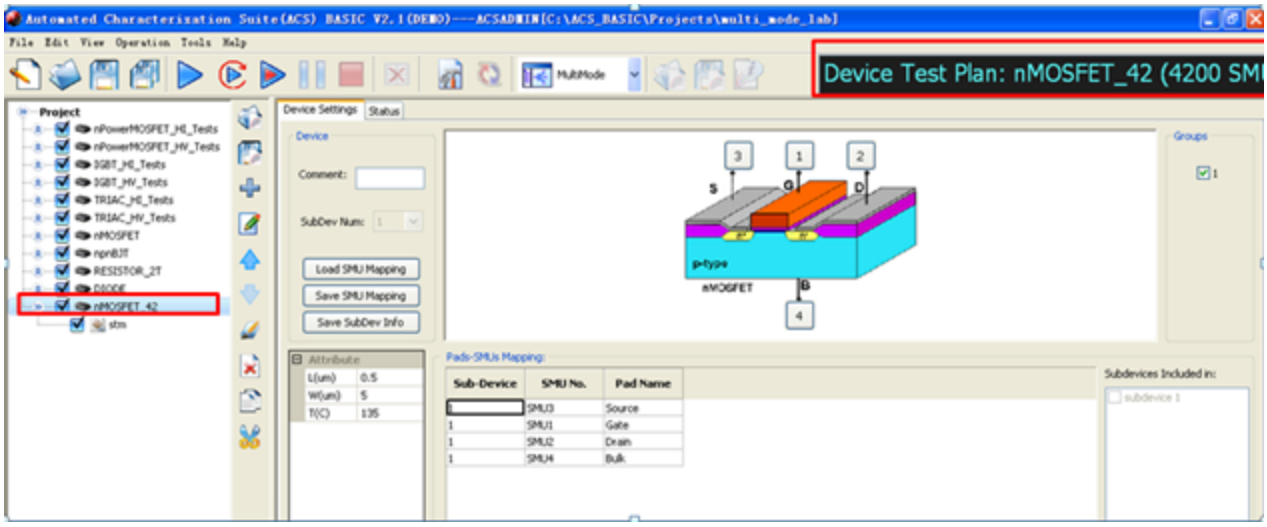
- **Device Settings:** The interface where you can assign the device settings according to the physical connection of the device. This tab opens by default.
- **Status:** Shows the device information.

Add a new device

To add a new device to the configuration navigator in Multitest mode:

1. Select the **Add new test** icon on the vertical toolbar. The Specify the Device/Test Type dialog opens.
2. Select the **Device** option and select **OK**. A SMU Type dialog opens.
3. Select the Model 42xx or 26xx SMU to add a new device to your test. Select **OK**. The Select Device Type dialog opens.
4. Select a device, and then select **OK**. The Image Browser dialog opens, as shown in the following figure.
5. Select a `.bmp` file and select **Open**.
6. The new device is now created and added to the configuration navigator. In this example, the device name is nMosfet_42 and its Device Settings tab opens by default, as shown in the following figure). Select this new device to view its information.

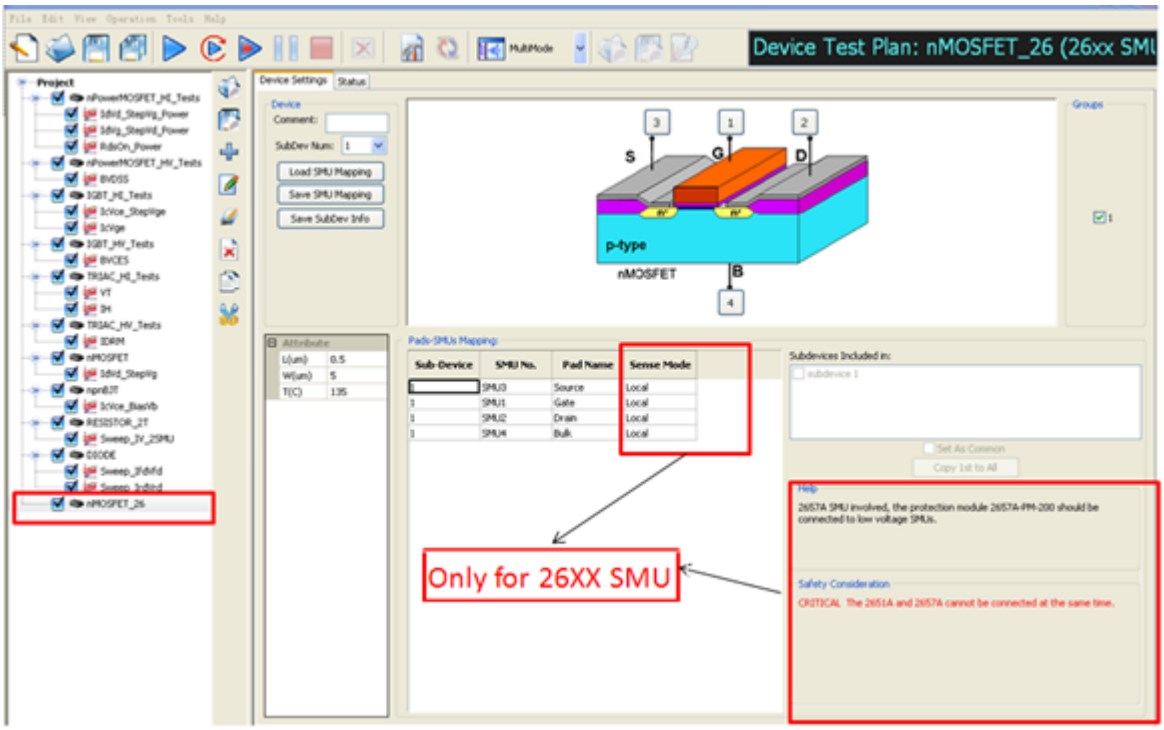
Figure 117: Example 4200 SMU information



7. If you are adding a Model 26xx SMU, you also need to determine the Sense Mode of the device. Refer to the following figure.
   - If your device is local (2-wire): Choose **Local**.
   - If your device is remote (4-wire): Choose **Remote**.

   The new 26xx SMU device in the figure is named nMOSFET_26.

Figure 118: Example 26xx SMU information



NOTE

If a Model 2651A is assigned and set to Remote sense mode on the drain (FET) or collector (BJT or IGBT), other Model 26xx SMUs should be forced to Remote sense mode.

The Device Settings tab for the Model 26xx SMU contains a Help area. This area helps you determine if you have configured your SMU in a way that will not function properly. For instance, here are some messages that you may encounter:

- "2600 non-A/B SMUs are involved, Pulse mode test cannot be supported": If a non-A/B Model 26xx is part of your configuration, pulse mode cannot be applied in the ITM device.
- "2657A SMU involved, the protection module 2657A-PM-200 should be connected to low voltage SMUs": If the Model 2657A is part of your configuration, you need to ensure that any low voltage SMUs are protected by the protection module.
- "CRITICAL The 2651A and 2657A cannot be connected at the same time": The Model 2657A can generate high voltage and should not be connected to a device that also has a Model 2651A connected, since it will not be protected from the Model 2657A high voltage.
- "CRITICAL Mixed use of Model 2601(A/B) and 2602 (A/B) with Model 2657A is not supported. The Model 2657A-PM-200 Protection Module only protects up to 200V": The Model 2657A can generate high voltages and should not be connected to a system that also contains Models 2601(A/B) and 2602(A/B).
- "Cannot place a low current SMU in the path of high current": You should not combine the use of a high current SMU (2651A) and a low current SMU on the Drain-Source of the MOSFET, Collector-Emitter of the BJT, Collector-Emitter of the IGBT, or on a two-terminal resistor.

NOTE

Limitations exist in ACS Basic when assigning SMUs to device terminals to prevent instrument damage due to incompatible instruments.

If a connection to the device is deemed unsafe by the software, you cannot run the test and you see an error message in the Help area of the GUI. Here are some items to consider when configuring your hardware:

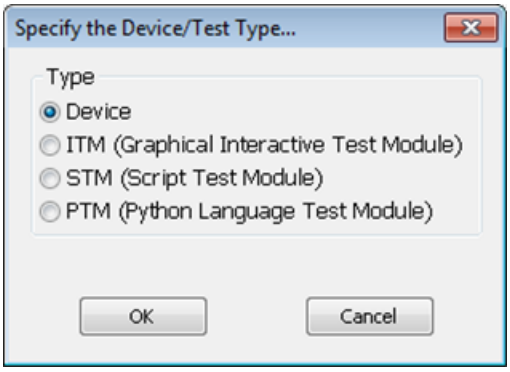
- Non-A 26xx SMUs cannot be included in a system with Model 265xA instruments.
- A Model 2657A SMU cannot be included in a system with Models 2651A, 2601B, or 2602B.
- A Model 2651A SMU cannot be included in a system with low-current SMUs.

- For two-terminal devices, a Model 2651A cannot be included in a system with low-current SMUs.
- For three-terminal and four-terminal devices, if a Model 2651A SMU is connected to the device's collector or drain terminal, then the emitter or source terminal, must be connected to ground or another Model 2651A SMU. If a Model 2651A SMU is connected to the base, or gate, then the other device terminals must be connected to ground or another Model 2651A.

Add a blank test

1. Select the device you want to create the test under, then select the **Add new test** icon on the vertical toolbar. The Specify the Device/Test Type dialog opens, as shown in the following figure.
   a. Select the test module type, ITM, STM, or PTM.
   b. Select **OK**. A test module is inserted to the configuration navigator.

Figure 119: Specify the Test Type dialog



2. If needed, change the name of the new test in the configuration navigator.
3. Select which Groups the test will include in the Groups area in the ITM Definition tab or on the STM Setup tab.
4. Save the test project by selecting the **Save** icon on the toolbar. You can also use the Save function in the File menu.

Setting Preferences

Select Preferences in the Tools menu to customize your settings. The Preferences dialog box opens.

There are three tabs in the Preferences setting dialog box: Path, Instruments, and Advanced. The Path tab opens by default.

The preferences settings are used to apply customized settings for manual or automation tests.

Path tab

Items on the Path tab are shown on the following figure.

Project Repository: Shows the default directory path where all projects are saved. You can also select browse and choose another directory, which changes the default path.

Auto-Load Project: Determines which ACS Basic project opens during startup.

Default Project: Shows the default project path.

Summary Reports Path: Shows the default directory path where test reports are located.

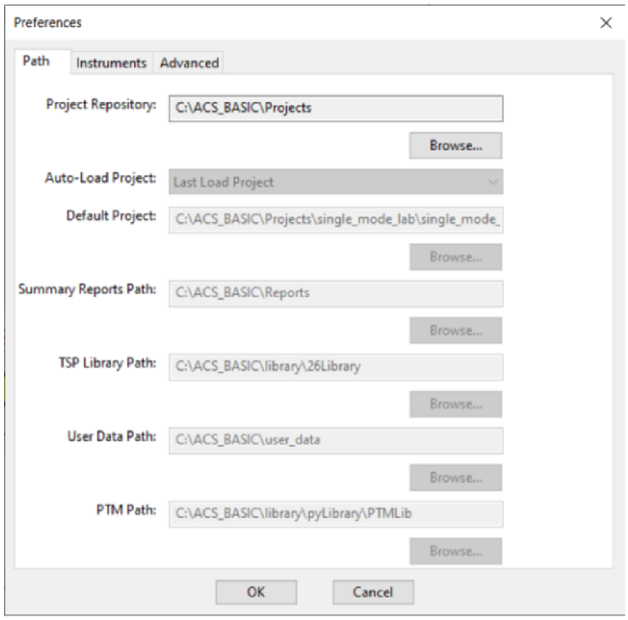
TSP Library Path: Shows the default TSP library directory path used to store the STM library.

User Data Path: Shows the default path where binning files and data files are located.

PTM Library Path: Shows the default PTM library directory path that is used to store the Python library.

Reference Data Path: Shows the default directory where the reference data is saved.

Figure 120: Preferences Path tab



Instruments tab

Items on the Instruments tab are shown on the following figure.

TSP-Link SMU Settings:

- **Default Remote Sense:** Allows you to set the default Sense Mode of each SMU that is selected (see next figure).
- **Check all:** Selects all the SMU Remote Sense boxes. It is cleared by default.
- **Source Auto Delay for 24xx:** Adds an autosource delay to 24xx for each test. It is selected by default.
- **Measure Auto Delay for 26xx:** Adds an automeasure delay to 26xx for each test. It is selected by default.

Initialize Unused SMU indicates the status setting for unused SMUs in an ITM:

- Default
- High Impedance
- Force Current 0A

Script cleaning:

- Clears 26xx scripts when a project is opened.

Output Enable Action (OE Output Off):

- Select to enable; disables the Series 2600B, Models 2601B, 2602B, and the Series 2400 SMUs output when the interlock is disengaged.

Login Bypass Option

- Select to enable; shows the login at start up.

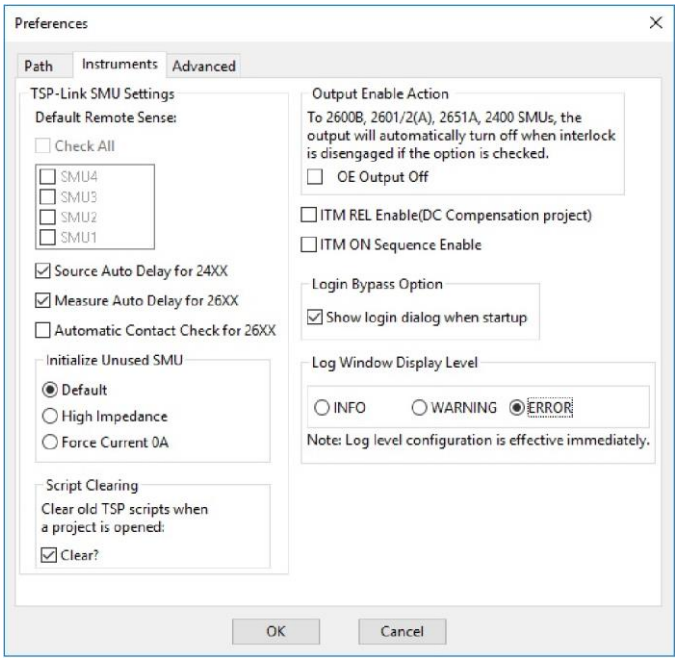
Log Window Display Level

- **INFO:** The message area displays each command, including the WARNING and ERROR level.
- **WARNING:** The message area displays error messages as they occur, including the ERROR level.
- **ERROR:** The message area displays error messages when they occur

NOTE

Autodelay times are measurement delay times that are controlled by the SMU instrument and may vary by range. For more information regarding auto delay settings, refer to the manual for the specific instrument.

Figure 121: Instruments tab



Advanced tab

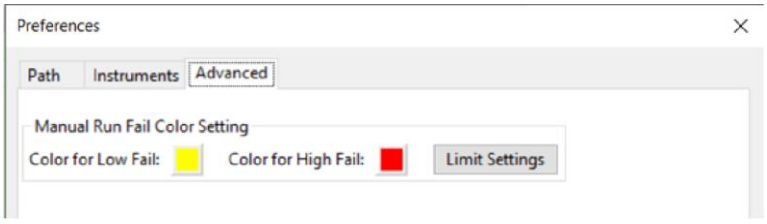
Items on the Advanced tab are shown on the following figure.

Manual Run Fail Color Setting:

- Color for Low Fail and Color for High Fail; select the background color for fail data that is below the stated specifications or above the stated specifications during a manual test.
- Limit Settings; select to configure the .csv data format (tabulated) that defines specifications that are considered "low" and "high" (for the low fail data and high fail data).

For more detail refer to Fail data (on page 4-43).

Figure 122: Advanced tab



CVU connection compensation

It is recommended that you perform open and short compensations measurements before measuring the device capacitance. Once the compensation data is stored, it can be used repetitively across different devices if there is no change in instrument and cable connections.

You can correct offset and gain errors caused by the connections between the Model 4210-CVU and the device under test (DUT) by using CVU connection compensation. Correction is a two-part process:

1. Generate CVU connection compensation data for an open or short connection.
2. Enable open or short correction values before executing a test.

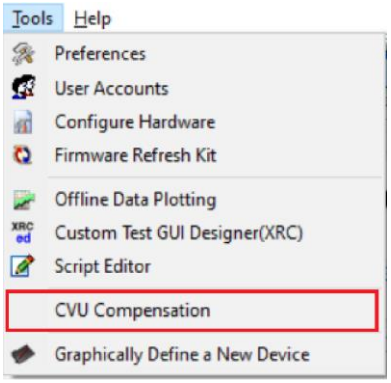
NOTE

Perform CVU connection compensation any time the connection setup is changed. CVU connection compensation is not affected by changes in either temperature or humidity.

Generate connection compensation data

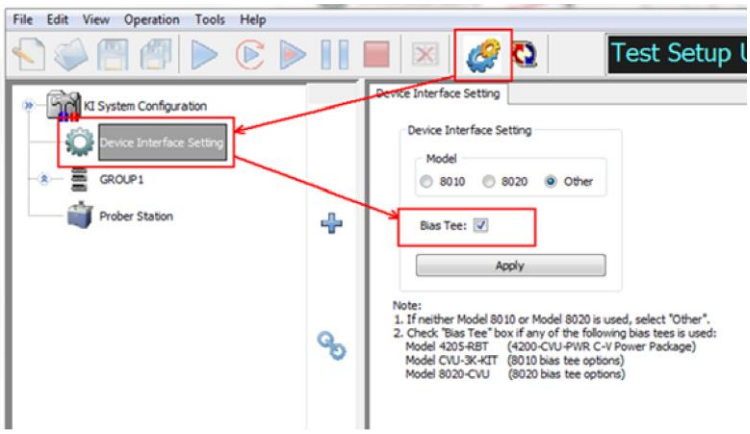
Open the CVU Compensation GUI from the Tools menu on the ACS Basic menu bar.

Figure 123: Open CVU Compensation GUI



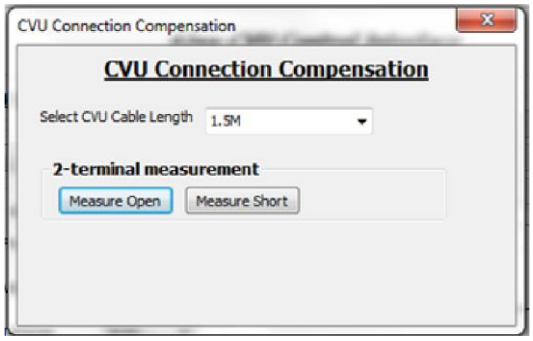
The device interface settings will determine the CVU Compensation GUI that appears.:

Figure 124: Device Interface Setting GUI



If the Bias Tee option is not selected, the CVU compensation GUI appears as follows:

Figure 125: CVU Compensation GUI without bias tee



2-terminal measurement (without bias tee)

This measurement is applied to a CV test without a bias tee (for example, the CVITM; see Enable compensation (on page 4-61)).

2-terminal measurement (with bias tee)

This measurement is applied to a Generic HV CV PTM (see Enable open or short correction for Generic HV CV PTM (on page 4-62)).

- **Measure Open**: **Select** the button to generate open correction values.
- **Measure Short**: **Select** the button to generate short correction values.

3-terminal measurement

This measurement is applied to both Generic HV CV PTM and device specific PTM libraries.

Level Setting:

- **Circuit-level parameters**: This compensation data is applied to the following devices and libraries.

MOSFET	BJT	IGBT
Ciss	**Cib**	**Cies**
Coss	**Cob**	**Coes**
Crss		**Cres**

- **Component-level parameters**: This compensation data is applied to the following devices and libraries.

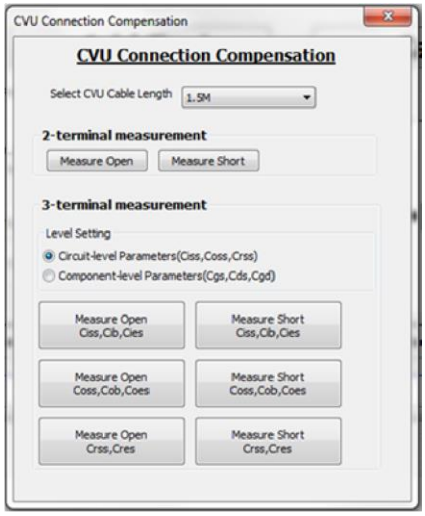
MOSFET	BJT	IGBT
Cgd	**Ccb**	**Cgc**
Cgs	**Ceb**	**Cge**
Cds	**Cce**	**Cce**

There are certain connection configurations that you can use with different devices, and the compensation will only need to be performed once. The Measure Open and Measure Short buttons group the same capacitance terminologies for MOSFET, BJT and IGBT.

For example, the input capacitances Ciss, Cib, and Cies for MOSFET, BJT, and IGBT share the same connection configuration. The same compensation data can be used for them all.

However, if the Bias Tee option is selected, the CVU compensation GUI appears as follows:

Figure 126: CVU Connection Compensation



Set the CVU cable length

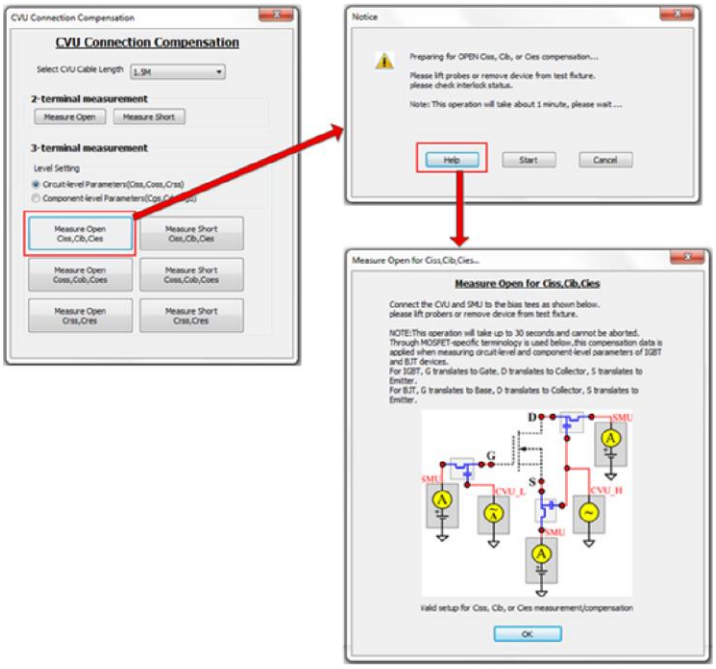
Use the menu to set the default SMA cable length.

- **0 M**: Use to make measurements at the terminals of the Model 4210-CVU (no cabling). This disables compensation.
- **1.5 M**: Use with the standard red SMA cables (1.5 m) supplied with the CVU (Keithley Instruments part number CA-447).
- **3.0 M**: Use the standard red SMA cables (3.0 m) (Keithley Instruments part number CA-446). This setting can also be used when using a switching matrix.

Measure open and measure short buttons

Based on the test that you want to perform, select a Measure Open, or Measure Short button (see below for options). When you select a Measure Open or Measure Short button, a dialog box opens. Select the Help button and you will see connection information:

Figure 127: Dialog for Measure Open Ciss, Cib, Cies



Make sure to set up the appropriate connection that appears once you select the Help button. For example, the previous figure shows the Measure Open Ciss, Cib, Cies CVU connection.

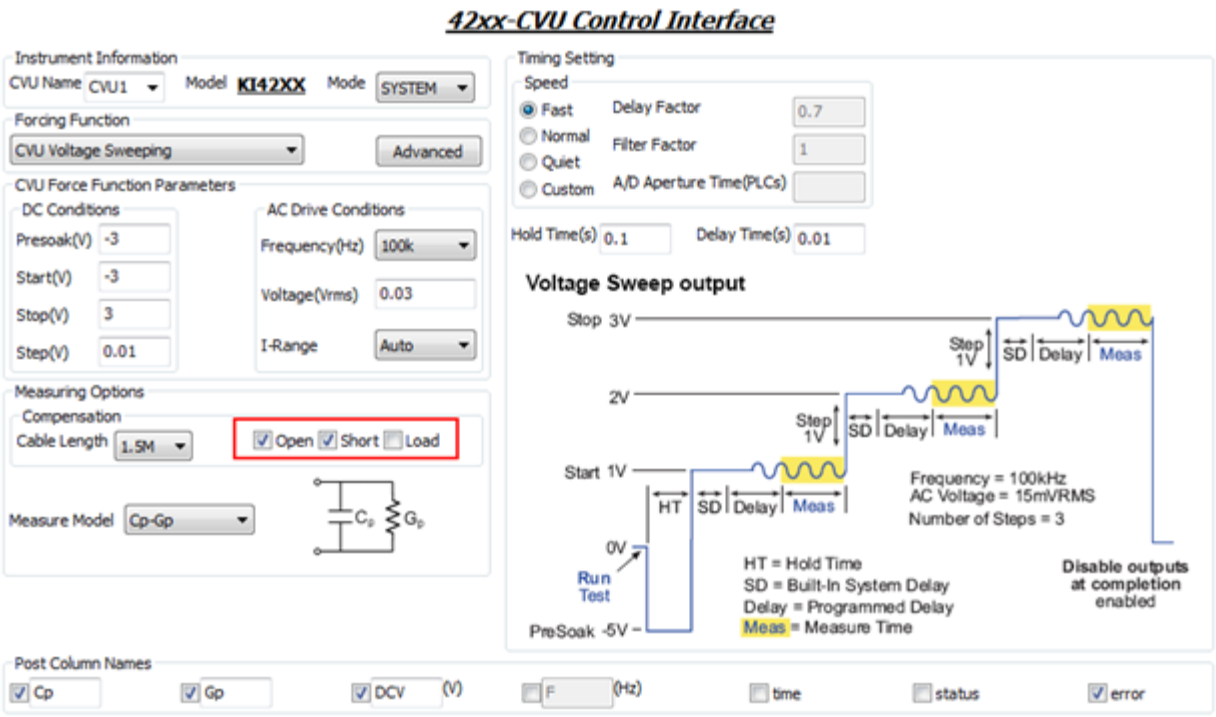
Once you have the test connections completed, select the Start button to begin generating open or short compensation data. Note that the compensation data file that is created is saved in the following computer directory: `C:\ACS_BASIC\compensation`.

If you are testing 2-terminal devices, the name of the generated file will be 2TComp.csv. Note that if you are testing 3-terminal devices, the name of the generated file will depend on the test that you select. The format of the generated compensation date file name will be similar to this format: `Ciss_Cib_Cies_Comp.csv`.

Enable compensation

Compensation is done during test execution with open or short correction enabled. Open or short correction is enabled from the GUI Generic HV CV PTM or device-specific PTM, or the CVITM (ki42xxCVU). The compensation data can be seen in the Data tab for the Generic HV CV PTM or device-specific PTM:

Figure 128: CVU Enable Open or Short Correction for CVITM



NOTE

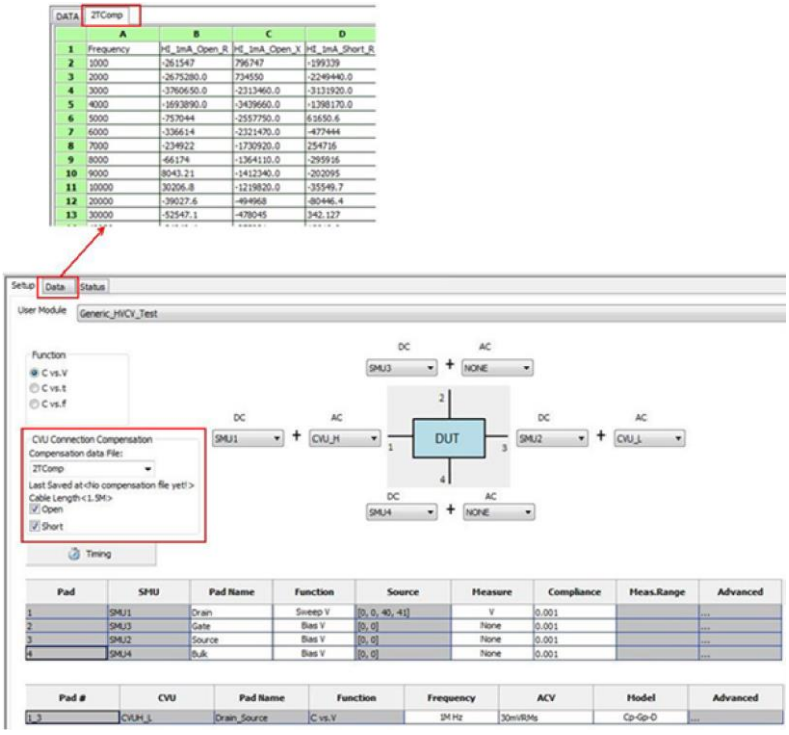
In ACS Basic, the CVU Connection Compensation (accessible from the Tools menu) does not contain a Load compensation measurement (see Measure Open and Measure Short buttons graphic). If you want to measure Load compensation (see the previous graphic) you will have to use the 4200 software KITE to collect load compensation data first, then use ACS Basic to make the measurement. In KITE, the compensation data is stored in the CVU card (the hardware). This means that when you return to ACS Basic for the measurement, you do not need to import data.

Load compensation requires connecting to a known standard load close to the value of the measured device. It may be impractical to find loads if the device sizes vary considerably.

Enable open or short correction for Generic HV CV PTM

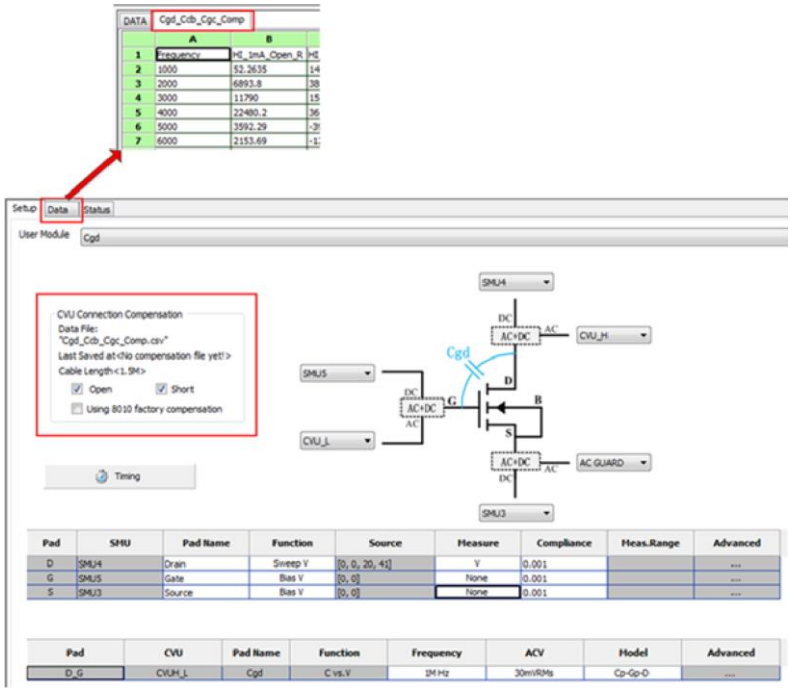
The following GUI shows where to enable compensation in the specific test module. Additionally, compensation data can be viewed in the Data tab of the test module. Compensation data appears as a second worksheet in that tab.

Figure 129: Enable open or short correction for Generic HV CV PTM



Enable open or short correction for device-specific PTM

Figure 130: Enable open or short correction for device-specific PTM



Within each test module, you have the option to select Using 8010 factory compensation. ACS Basic stores default compensation data that was generated using cables and accessories supplied with remote bias tee kits (Models CVU-3K-KIT and CVU-200-KIT and PCT configurations).

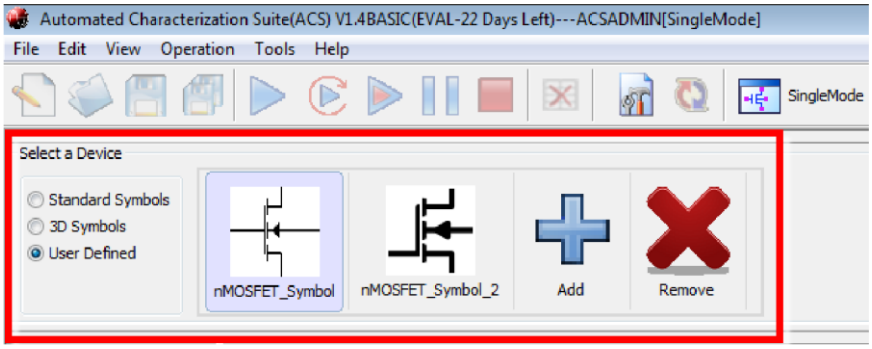
You can load the Generic HV CV PTM using the default compensation data file from the Generic_HVCV_Test GUI. For a device specific PTM you can use the default compensation data file by selecting the Using 8010 factory compensation option. Note the name of the default compensation data file will depend on the test selected. The name begins with `factory_8010_`, for eample, it will be similar to the file name `factory_8010_Ciss_Cib_Cies.csv`.

Graphically define a new device

To open the Graphically Define a New Device dialog, do one of the following:

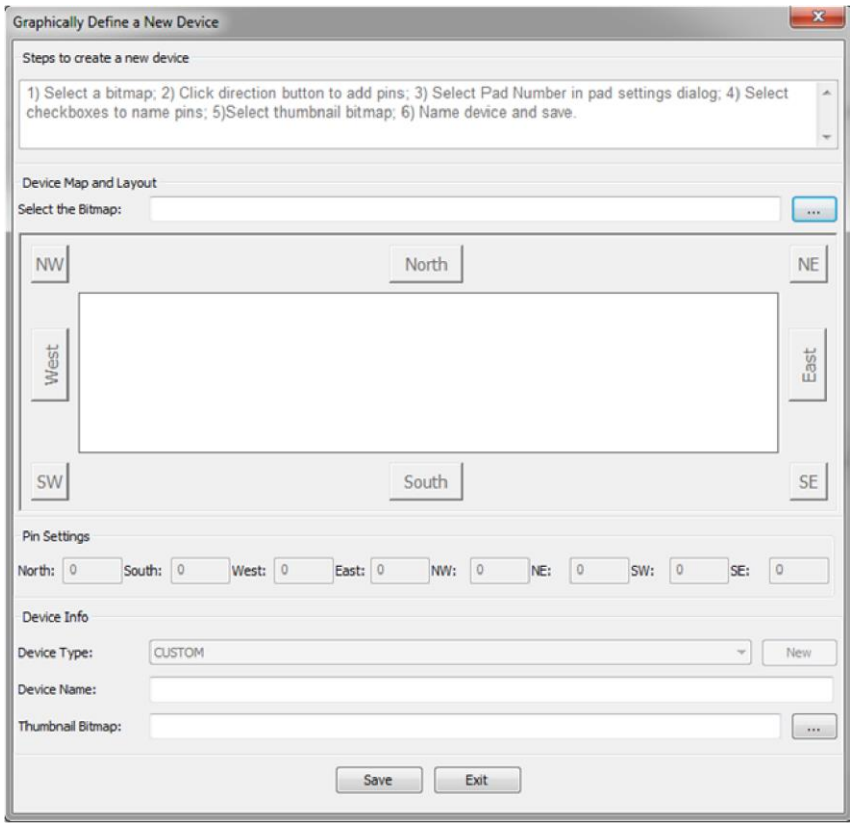
- In the Tools menu, select **Graphically Define a New Device**.
- In the selected test module dialog, select User Defined in the Select a Device area, as shown in the following figure, and select **Add**.

Figure 131: Add a User Defined device



The Graphically Define a New Device dialog is displayed, as shown in the following figure.

Figure 132: A blank Graphically Define a New Device



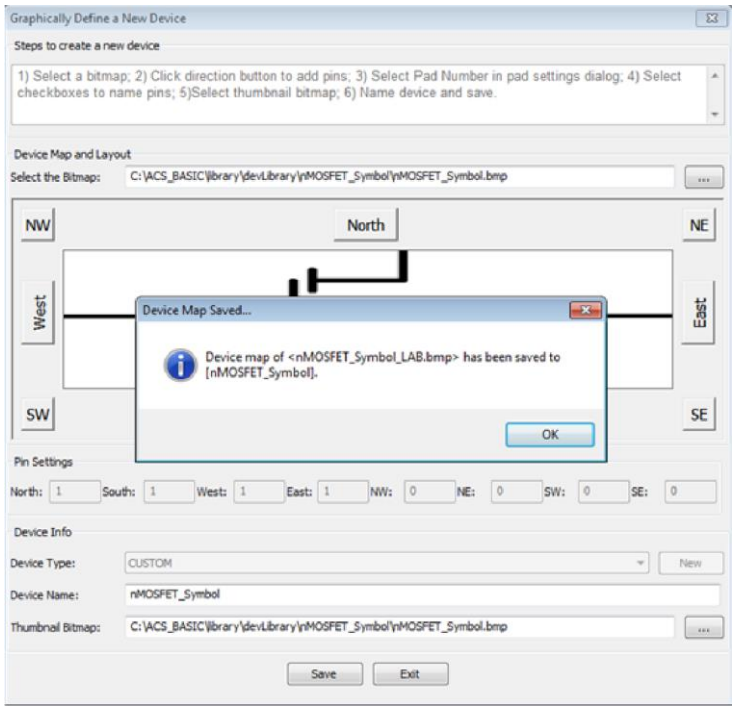
To define a new device:

1. Select a bitmap to represent the device by entering the file address or selecting the ellipsis next to the Select the Bitmap area. The selected bitmap loads in the Device Editor. The bitmap's name will appear in the Device Name edit box as the default name of this device. You can change this name by entering a new name. This device map is used as a graphic representation of the device workspace, plan, and module GUI.

2. Select the thumbnail bitmap of the device by entering the file address or selecting the ellipsis function next to the Thumbnail Bitmap area. This thumbnail is used as a graphic representation in the device selection area.
3. Enter the Pin Settings. The locations on the map are defined as North, South, West, East, NE (Northeast), SE (Southeast), NW (Northwest), and SW (Southwest).
   a. Select the location and a dialog opens for setting the number of pads from a list.
   b. After you select the number of pads, select the check boxes under the number of pads. This will allow you to set the pad name for each pad in this location.
   c. After all the positions are set, select **Save**.

The following figure shows the device nMOSFET_Symbol is saved and ready to use in ACS Basic.

Figure 133: Save the User Defined device



After you have defined and saved the device, it displays in the GUI. Now you can build and configure the test library for this device.

Section 5

ACS Basic test modules

In this section:

Configure a graphical interactive test module (ITM)

An ITM allows you to define a test interactively using a graphical user interface (GUI). ITMs are available for the Model 4200A-SCS SMUs or Series 2600B System SourceMeter instruments.

After inserting an ITM into the configuration navigator, it must be configured to meet your test requirements.

ITM

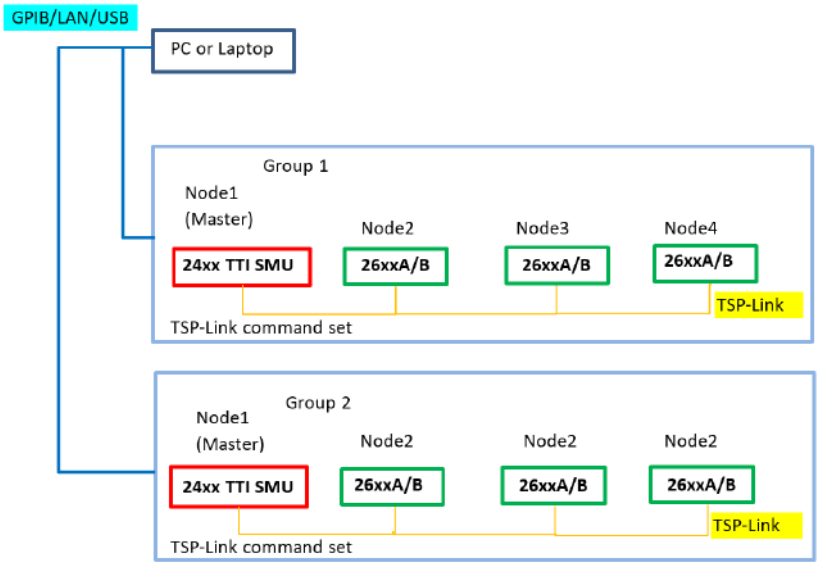
NOTE

If you are using a Series 2400 Graphical Series SourceMeter, you need to set the TSP mode to ITM.

NOTE

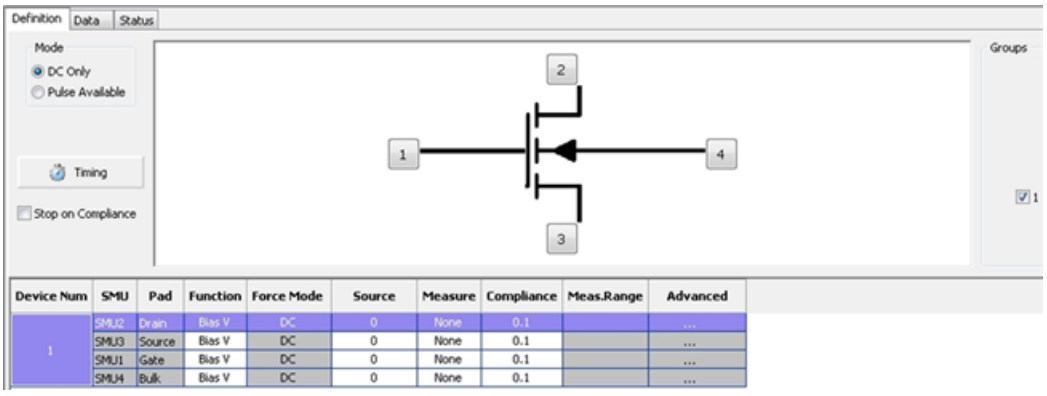
If you are using a mix of Series 2600B and 2400 Graphical Series SMU instruments, the master node must be on the 2400 SMU. See the following figure for an example.

Figure 134: Test system with a mix of 26xxB and 24xx Graphical Series SMUs



Each ITM is associated with a specific ITM work area. An ITM work area displays the definition of the ITM (configuration), test data, and status information, as shown in the following figure.

Figure 135: Definition tab of the Model 26xxB and 24xx SMU ITM



The Definition tab contains the following elements:

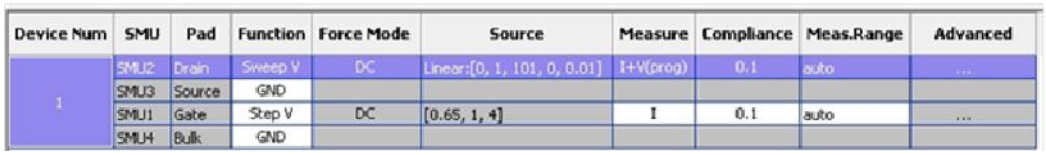
- **Mode**:
  - **DC Only:** Select if you want all SMUs to only operate in DC mode.
  - **Pulse Available:** Select if you want SMUs to operate in DC or pulse mode. Pulse timings for each SMU may be assigned to one of two Pulse Timing configurations. Because accurate timing is required for pulsing, some features, such as autoranging, are disabled when you select Pulse Available.

- **Timing**: Opens the ITM timing dialog, where you can set detailed timing settings for pulse and synchronization between SMUs.
- **Stop on Compliance**: Select if you want to abort the test and turn off the SMU output when the test reaches compliance.
- **Device schematic**: Shows the device image and pad settings according to the device level settings in the configuration navigator.
- **Groups**: Select the group or groups where you want to run the test. The group is determined by the GPIB or ethernet address. Make sure that all the groups you select have the same hardware configurations as your ITM.
- **SMU parameters table**: Set the parameters for testing.

Model 26xxB SMU parameters

The following figure displays the items to set up testing for forcing and measuring functions for each SMU.

Figure 136: SMU parameters table



Device Num	SMU	Pad	Function	Force Mode	Source	Measure	Compliance	Meas.Range	Advanced
1	SMU2	Drain	Sweep V	DC	Linear:[0, 1, 101, 0, 0.01]	I+V(prog)	0.1	auto	...
	SMU3	Source	GND						
	SMU1	Gate	Step V	DC	[0.65, 1, 4]	I	0.1	auto	...
	SMU4	Bulk	GND						

Device Num: Sub-device number defined in the device work area.

SMU: SMU number defined in the device work area.

Pad: Pad name defined in the device work area.

Function: Force function type.

Force Mode: DC for DC Only. DC, pulse timing 1, or pulse timing 2 for pulse available.

Source: The source value of SMU.

Measure: Selects the measurement function option:

- **I:** Actual measured current values.
- **V:** Actual measured voltage values.
- **V (prog):** Reports the programmed voltage. No measurement is made.
- **I (prog):** Reports the programmed current. No measurement is made.
- **I+V:** Measures both current and voltage.
- **I+V (prog):** Measures current and returns voltage (prog).
- **V+I (prog):** Measures voltage and returns current (prog).

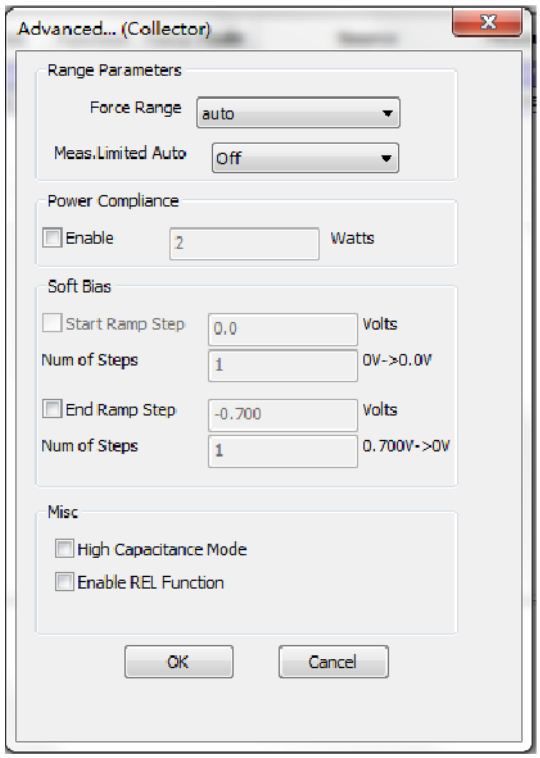
Compliance: Sets current limit for voltage source or voltage limit for current source.

Meas. Range: Specifies the SMU range used. The current measure range is listed for voltage forcing functions, and the voltage measure range is present for current forcing functions only. If current is selected in the Measure cell, ACS enforces the Force range first, and ignores the measure range. You can specify the range used by the SMU using the following options:

- Auto commands the SMU to automatically optimize the measurement range. This option provides the best resolution.
- Numerical ranges allow you to manually select a fixed measurement range.

Advanced: Double-click the cell to display a dialog for advanced settings for this SMU.

Figure 137: SMU Advanced settings dialog



Range Parameters:

- **Force Range**: Specifies the SMU force range used to force voltage or current, including auto, Best Fixed, and numerical range options.
- **Meas. Limits Auto**: When measuring current with Meas.Range set to auto, the application enables this option. This parameter is used with autoranging to put a lower bound on the range used. Since lower ranges generally require greater settling times, setting the lowest range limit might make measurements require less settling time. This option is a compromise between the full auto option and a fixed range option. You can specify the minimum range that the SMU uses when automatically optimizing the current measurements. This option saves time when maximum resolution at minimum currents is not needed.

Power Compliance: Set the power compliance value (watts) for a 2600B SMU. The firmware version of the 2600B SMU must be at least higher than 2.2.1.

Soft Bias: Apply Soft Ramp at the beginning of the bias or first sweep or list point and at the end of the bias or last sweep or list point. Typical use cases are for high power MOSFET measurements. For more information about using Soft Ramp, refer to the following topics:

- SMU forcing functions (on page 5-6)
- General-purpose PTM configuration (on page 5-63)

Misc:

- **High Capacitance Mode:** Sets the SMU to high capacitance mode. This is only available for the Series 2600B SMU and Series 2400 SMU.
- **Enable REL Function**: When Enable REL (relative offset) is selected, the ITM enables the relative offset function for the SMU and sets the global data offset value to the relative offset value. When the relative offset function is disabled, the ITM disables the relative offset. Note that an SMU can execute DC compensation when the fixture opens. The SMU forces 0 volts, measures the offset current at the user specified range, and the ACS Basic software keeps the value as global data.
- **On Sequence for (pad name):** The sequence number for turning on the SMU of the Pad in the test. If set to 1, the SMU of the Pad is the first to turn on. This option is available when the ITM Sequence is selected from the toolbar (**Tools > Preference > Instruments**).
- **Sequence Delay:** The delay before turning on the SMU of the Pad. This option is available when the ITM Sequence is selected from the toolbar (**Tools > Preference > Instruments**).

SMU forcing functions

The SMU forcing functions are selected in the SMU forcing and measuring settings table. The functions are Bias V, Bias I, Sweep V, Sweep I, List V, List I, Step V, Step I, V Meter, GND, and Open. In DC Only mode, there are two more functions: Sweep2 V and Sweep2 I. Due to the selected forcing functions, the forcing value is defined in the Source cell in the SMU forcing and measuring settings table.

Bias V, Bias I

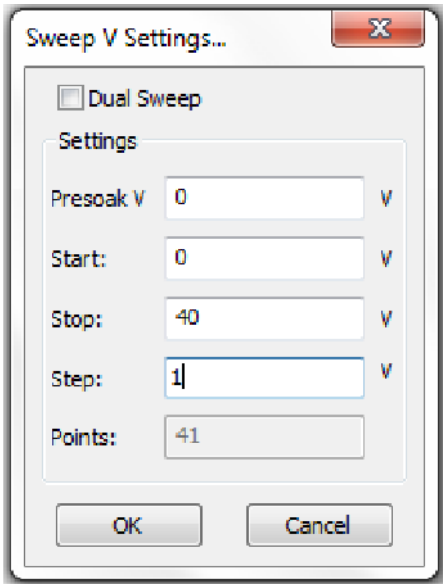
The Bias forcing function maintains a defined constant voltage or current state at the terminal, subject to your specified compliance value of the connected SMU. When Bias V or Bias I is selected, you can set a valid value in the Source cell.

Sweep V, Sweep I

A sweep increments the current or voltage in a series of uniform steps at a speed that is determined by the Speed and Timing settings. When Sweep V or Sweep I is selected, you can set the sweep settings by double-clicking the Source cell and making the necessary changes in the dialog that opens.

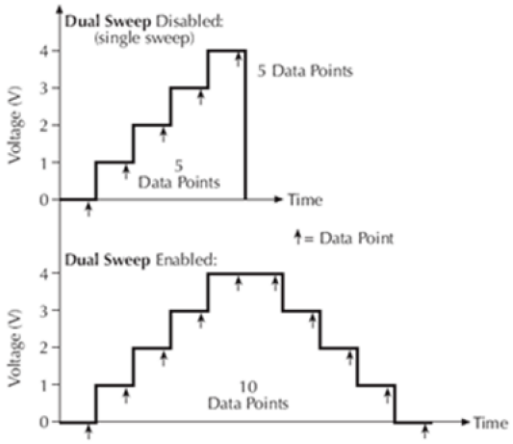
Double-click the Source cell for SMU that is set to Sweep V/Sweep I. A sweep settings dialog opens.

Figure 138: Source settings for SMU sweep functions



Dual Sweep: A SMU that is configured to perform a sweep can also be set to perform a dual sweep. With Dual Sweep enabled, the SMU essentially perform two sweeps. The first sweep steps from the start level to the stop level. The SMU then continues with the second sweep according to the direction chosen (forward or reverse). With Dual Sweep disabled, the SMU performs a single sweep stepping from start to stop.

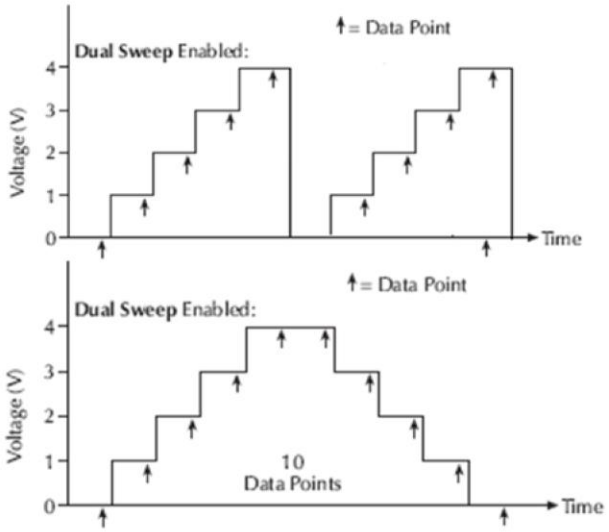
Figure 139: Single and dual sweep examples



Forward: First sweep steps from the start level to the stop level. The SMU continues with the second sweep and proceeds in the same manner as the first sweep.

Reverse: First sweep steps from the start level to the stop level. The SMU continues with the second sweep and steps from the stop level back to the start level.

Figure 140: Dual sweep mode - forward and reverse



Mode:

- **Linear**: Sweep mode based on the linear axis.
- **Log**: Sweep mode based on the logarithm axis.

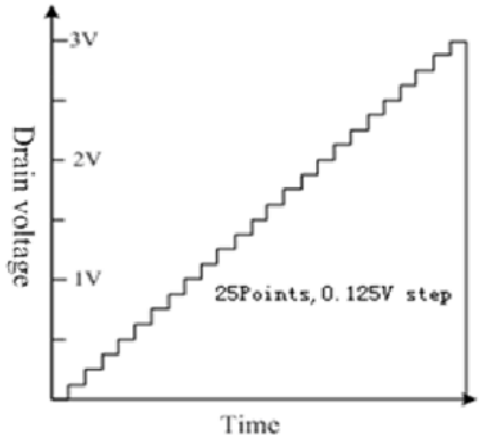
Settings:

- **Start**: Specifies a starting point for a sweep for any valid SMU voltage or current depending on the source function.
- **Stop**: Specifies a stopping point for a sweep for any valid SMU voltage or current depending on the source function.
- **Step**: Specifies the incremental value between every two points. You can edit the step and it is automatically calculated depending on the Start, Stop, and Points values.

Points: Shows the number of data points that are input when a sweep is executed.

Base: Set the Pulse base level when Pulse Available is selected.

Figure 141: Example sweep drain voltage on a MOSFET



Step V, Step I

A step forcing function incrementally steps a current or voltage two or more levels, each of which is held constant during the progress of a voltage or current sweep at another terminal. For each step parametric curve data is recorded in the ITM Data tab worksheet. The combined data can be plotted in the ITM Data tab graph resulting in a series of curves.

More specifically, the Voltage Step forcing function increments through multiple, evenly-spaced constant voltages, and steps over a user-specified range that is subject to a specified current compliance. The time interval for each step is determined automatically by the time required to complete a sweep.

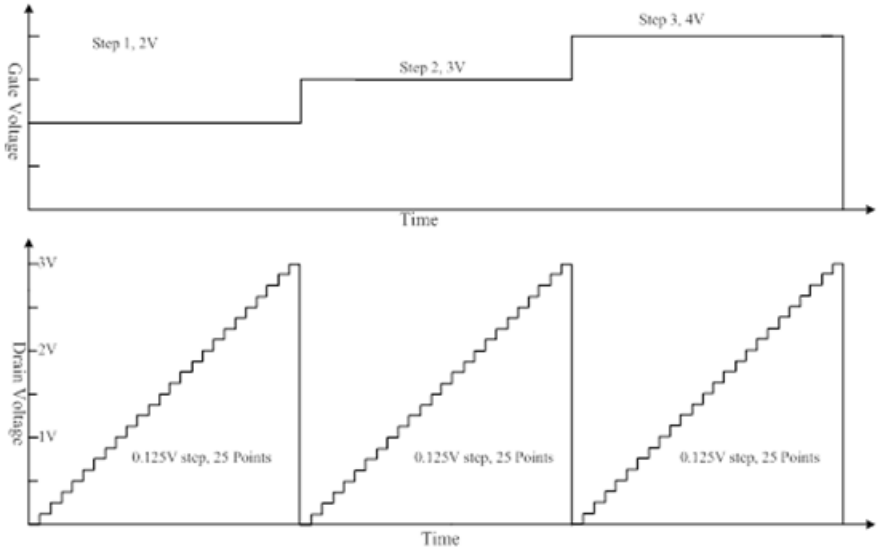
A typical voltage step forcing functions example is shown in the next figure. This figure illustrates how one Definition tab (for the Vds-Id ITM) specifies both Step and Sweep forcing functions.

Figure 142: Typical step forcing parameters

Device Num	SMU	Pad	Function	Force Range	Source	Measure	Compliance	Meas Range	Limits Auto
1	SMU1	Source	Bias V	auto	0.000	None	0.100		
	SMU2	Gate	Step V	auto	[2, 4, 3, 0]	V(prog)	0.100		
	SMU3	Drain	Sweep V	auto	Linear:[0, 3, 25, 0]	I+V(prog)	0.100	auto	
	SMU4	Bulk	Bias V	auto	0.000	None	0.100		

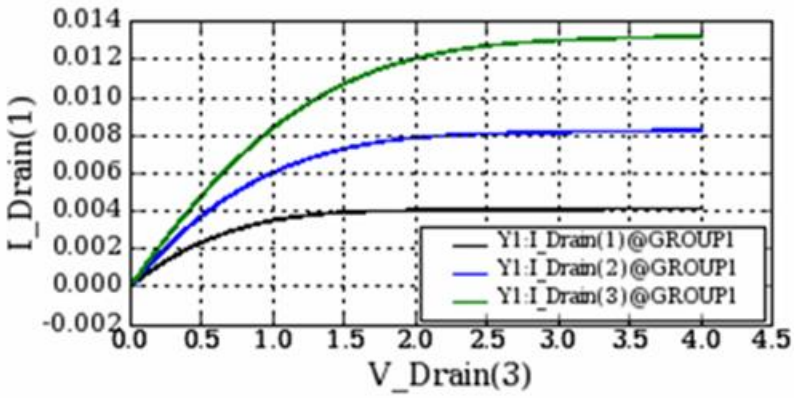
The next figure graphically illustrates the combined Step and Sweep forcing functions specified in the previous figure.

Figure 143: Example MOSFET step gate and sweep drain voltage



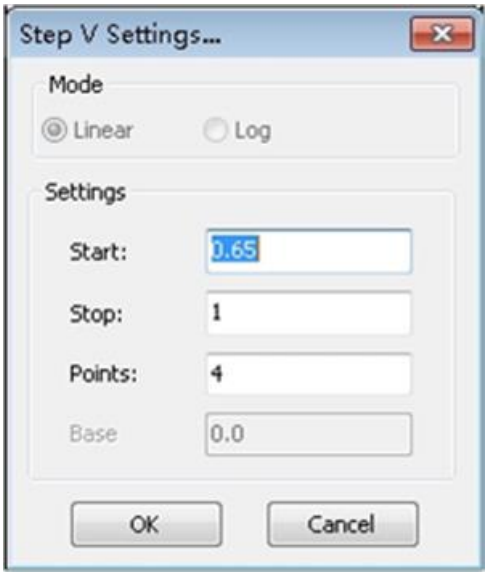
The next figure is a data plot of the graph from the typical forcing parameters figure.

Figure 144: Vds-Id graph



Double-click the Source cell of SMU that set as Step V/Step I to display the step settings dialog.

Figure 145: Source settings for SMU step functions



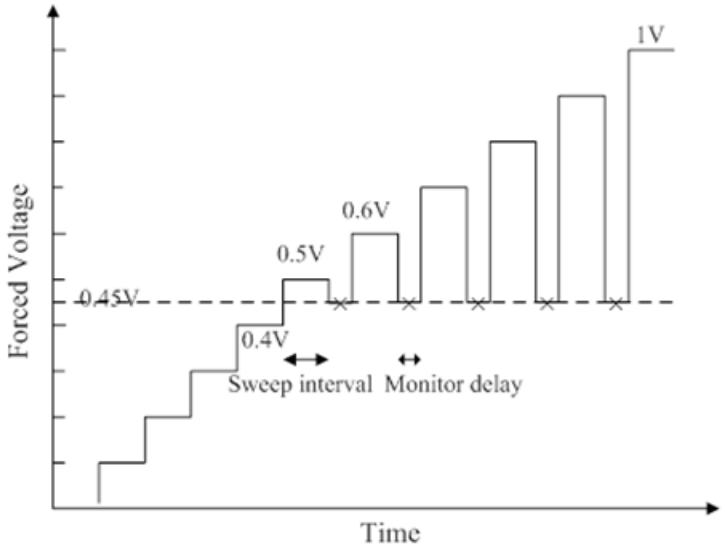
Settings:

- **Start**: Specifies a starting point for a step for any valid SMU voltage or current depending on the source function.
- **Stop**: Specifies a stopping point for a step for any valid SMU voltage or current depending on the source function.
- **Points**: Specifies the number of data points that are input when a step is executed. It also specifies the value of the voltage increments of every step value. More than two points muse be entered in the Data Points cell. For example: If Start = 0V, Stop = 4V, and Points = 5, Step value = (4V-0V)/(5-1)=1V Five values are forced 0V, 1V, 2V, 3V, 4V.
- **Base**: Set the Pulse base level when Pulse Available is selected.

Sweep2 V, Sweep2 I

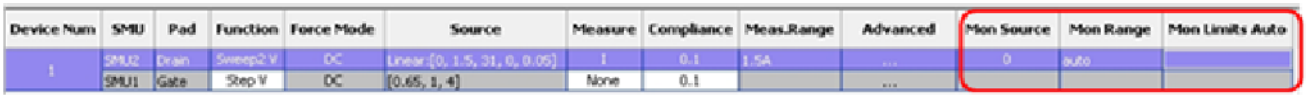
Sweep2 is a special sweep test when used in DC Only mode. When the forced value is greater than the set value, a monitor test adds to each sweep step, thus the test can generate two test results: Measured results and monitored results. More specifically, when each sweep interval increases, it equals the value of the sweep delay and monitor delay (see next figure; the sweep 2 V waveform has a 0.45 V monitor source voltage).

Figure 146: Sweep 2 V waveform



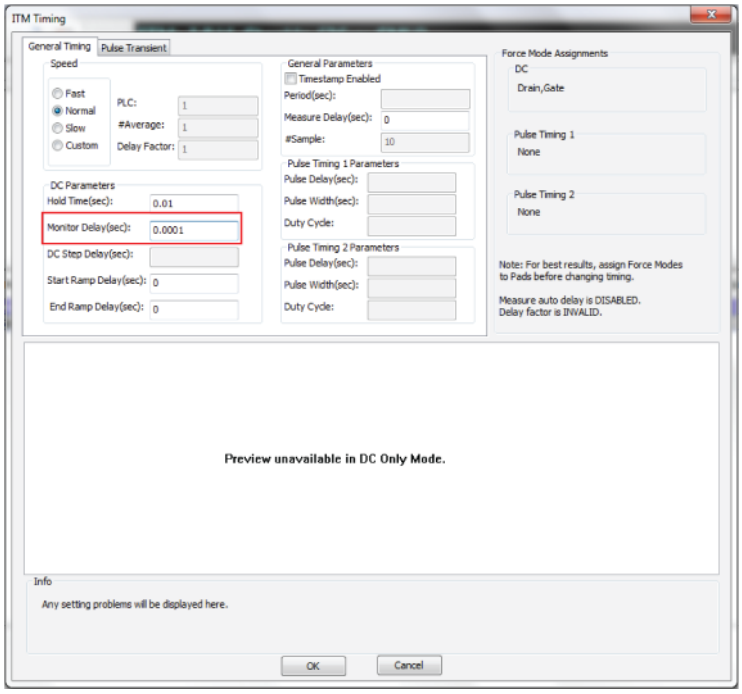
Select the Function cell and select Sweep2 V or Sweep2 I from the list. The monitor items are appended to the Force and Meas cell (see next figure)

Figure 147: The SMU monitor items



Also, the monitor delay in Time dialog is enabled (see next figure).

Figure 148: Enable monitor delay

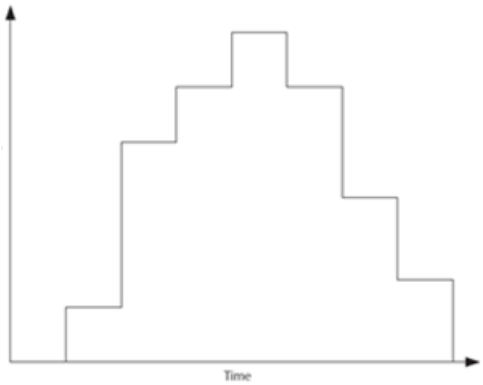


List V, List I

List V and List I are used to input a voltage and current list according to the requirements of the test. The SMU forces the input values one by one like a sweep.

List Sweeps allow you to make measurements only at selected forced voltages and currents. For example, they allow you to skip measurement points or to synthesize a custom sweep that is based on a mathematical equation.

Figure 149: List sweep general illustration



Double-click the Source cell of the SMU and the List Sweep Settings dialog opens (see next figure).

Figure 150: List Sweep Settings dialog

Points: the number of points for the forcing values.

You can import (and then save) a `.csv` file by selecting Import Setting From CSV File (see previous figure).

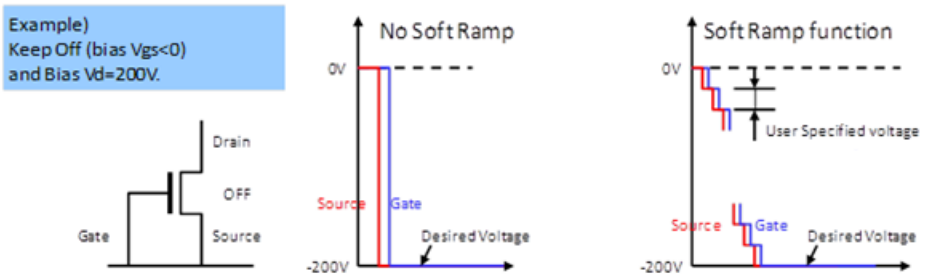
- **Edit item**: The default number can be changed to a different value.
- **New item**: A new item is added when you select this icon.
- **Delete item**: A selected item is deleted when you select this icon.
- **Move up**: A selected item moves up when you select this icon.
- **Move down**: A selected item moves down when you select this icon.
- **Base Level**: The Pulse base level when Pulse Available is selected.

Soft Ramp Bias

The DC Only mode enables you to apply incremental test readings at the beginning of the bias or first sweep or list point tests, and at the end of the bias or last sweep/list point tests. This function allows the forced voltage or current to achieve the set value after a certain number of steps.

The Soft Ramp is from 0 to the set value (bias value or first and last sweep or list value). If the bias or first or last sweep/list point is 0, no soft ramp is necessary. In the "No Soft Ramp" example, the gate breakdown occurs because of the 200 V stress applied. However, with the "Soft Ramp function," you can keep the voltage differences at smaller levels until the set voltage is applied (see next figure).

Figure 151: Comparison of Soft Ramp and without Soft Ramp



Soft Ramp is enabled in the Advanced settings in the SMU forcing and measuring settings table. The Soft Ramp timing is set in the timing settings dialog (see next figures).

Figure 152: Advanced Soft Bias settings

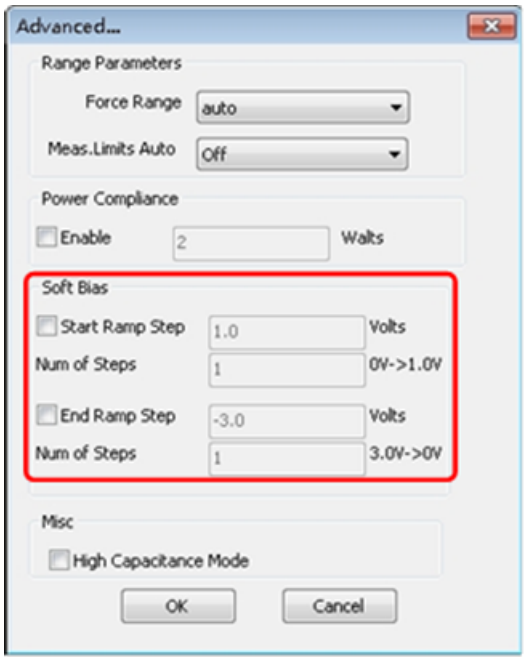
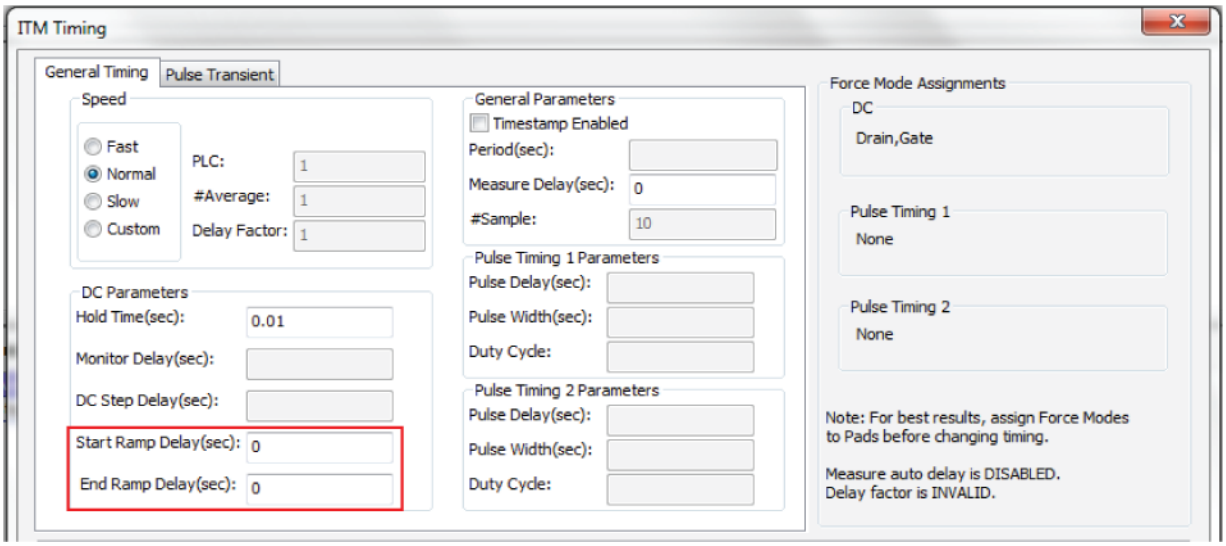


Figure 153: ITM General Timing tab settings



If the Start Ramp and End Ramp are selected, the test applies a Soft Ramp before and after the Bias or Sweep. The Soft Ramp is from 0 to the set value (Bias value or first and last Sweep, List value). If the Bias or first or last Sweep, List point is 0, no Soft Ramp is necessary.

In the Advanced settings dialog, setting the Ramp Step, the Num of Steps is automatically calculated.

Start Ramp Step: The step value determines when to start the ramp.

Stop Ramp Step: The step value determines when to stop the ramp.

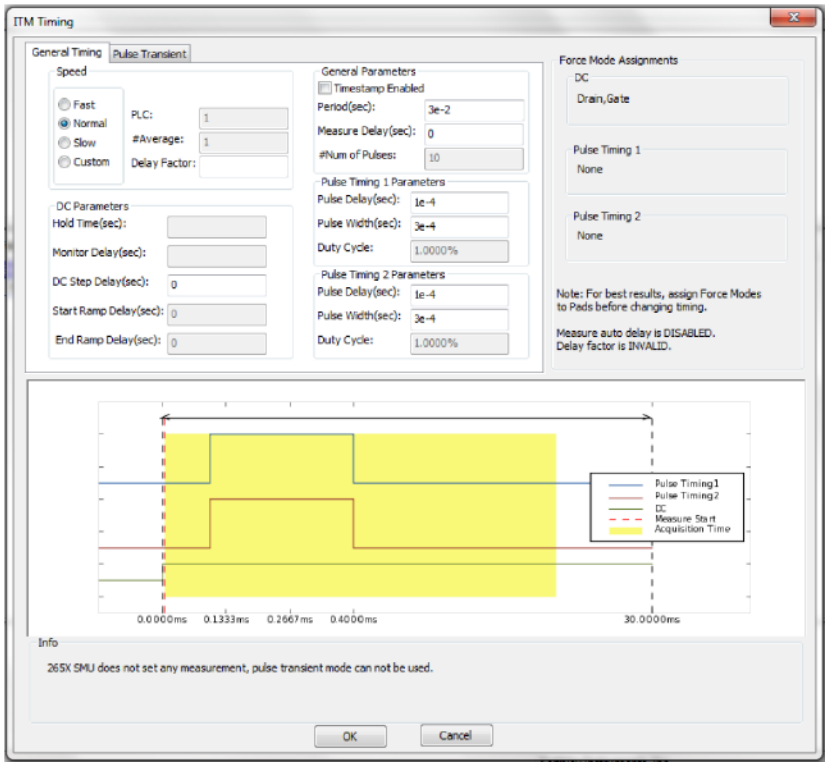
Num of Steps: The number of steps defined for all the SMUs. The start and stop steps are defined separately. The default value for Num of Steps is 1. Its value depends on the Force Value and the Start/Stop Ramp Steps.

In the ITM Timing settings dialog, the Start Ramp Delay and End Ramp Delay determines the step time of the Soft Ramp.

ITM timing settings

When you click the ITM Timing icon in the ITM Definition tab, the ITM Timing GUI opens (see next figure). The ITM Timing GUI is used to configure ITM timing settings.

Figure 154: ITM Timing dialog



Speed

The Speed area of the ITM Timing GUI allows you to:

- Locally select the Fast, Normal, Quiet, or Custom measurement speed mode.
- Configure Custom speed settings in the Custom measurement speed mode.

Fast, Normal, Quiet, and Custom modes are defined as follows:

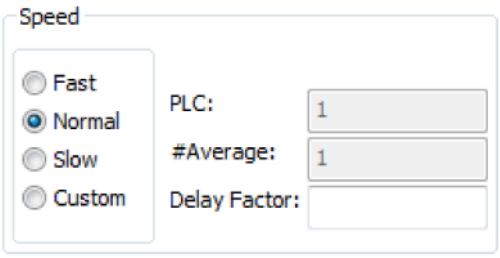
- **Fast:** Optimizes speed at the expense of noise performance. It is a good choice for fast measurements where noise and settling time are not concerns.
- **Normal:** The default and most commonly used setting. It provides a good combination of speed and low noise, and is the best setting for most cases.
- **Slow:** Optimizes the low-noise measurements at the expense of speed. If speed is not a critical consideration, it is a good choice when you need the lowest noise and most accurate measurements.

- **Custom:** Allows you to fine-tune the timing parameters to meet a particular need. With Custom, you can configure the integration time and the number of readings to produce a composite setting. The entries in the PLC and # Average edit fields control the A/D (analog-to-digital) converter integration time used to measure a signal. A short integration time for each A/D conversion results in a relatively fast measurement speed, at the expense of noise. A long integration time results in a relatively low noise reading, at the expense of speed. The integration time setting is based on the number of power line cycles (NPLCs):

  For 60 Hz line power, 1.0 PLC = 16.67 ms (1/60 of a second).

  For 50 Hz line power, 1.0 PLC = 20 ms (1/50 of a second).

Figure 155: Speed settings for an ITM



If you selected the Custom measurement Speed mode, any value between 0.001 and 25 NPLC can be entered. If you selected the Fast, Normal, or Quiet measurement Speed mode, ACS Basic sets the A/D Integration Time correspondingly. The next table summarizes the allowed A/D Integration Time settings for the Measurement Speed modes.

Speed mode	NPLC	# Average
Fast	0.001	1
Normal	1	1
Slow	10	1
Custom	0.001 to 25	≥ 1

General parameters

Timestamp Enabled: If selected, an additional data column (Time) in the Data tab will be added to record the measurement time.

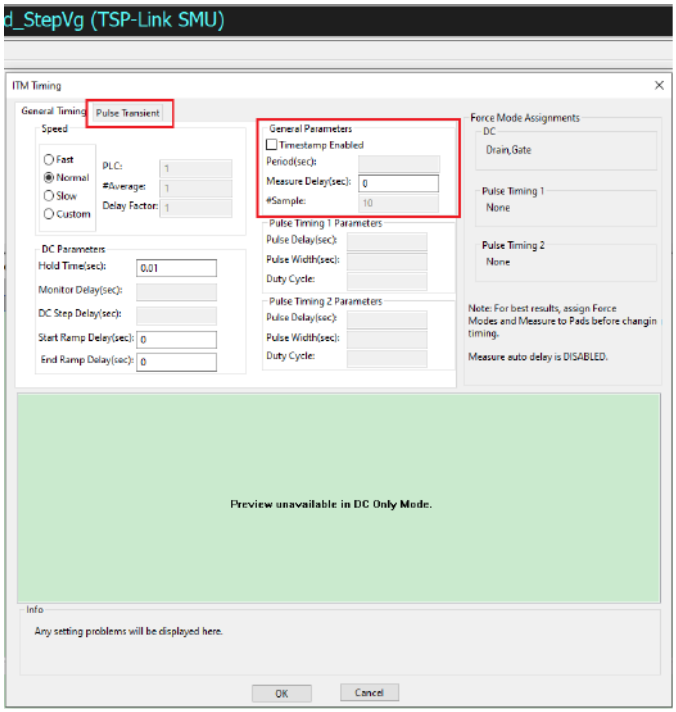
Pulse Transient Mode: if checked, enable transient measurement of pulse mode. And Fast ADC is selected, and PLC is automatically set to 0.001 and #Average to 1.

Period(s): pulse period, unit is second. All the two pulse timing have the same period.

Measure Delay(s): delay time before measurement.

#Num of Pulses: the number of pulses when all the SMUs are set to BIAS in Pulse Available.

Figure 156: General Parameters ITM timing settings



DC Parameters

Hold Time(s): DC Only. When the test is started, the Hold Time (HT) provides extra settling time for the initial step change of the source. The Hold Time is a global setting. Therefore, it is the same for all SMUs in the test system. Note that the Hold Time is applied at the start (only) of each sweep.

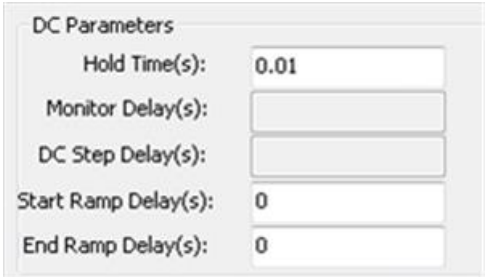
Monitor Delay(s): Sets the monitor delay time in the second sweep phase in of the Sweep2 function.

DC Step Delay(s): Sets the time in seconds from the start of the period to the start of the transition for DC signals.

Start Ramp Delay(s): Sets the delay time of steps in the start ramp of the soft ramp bias.

End Ramp Delay(s): Sets the delay time of steps in the end ramp of the soft ramp bias.

Figure 157: DC Parameters ITM timing settings



Pulse Timing Parameters

ACS Basic allows you to configure a total of two pulse timing configurations.

Pulse Delay(s): Sets the time in seconds from the start of the period to the start of the rising edge for pulse signals assigned to the Force Mode Pulse Timing 2 Parameters.

Pulse Width(s): Sets the width of the pulse in seconds for pulse signals assigned to the Force Mode Pulse Timing 2 Parameters.

Duty Cycle: Displays the duty cycle for pulse signals assigned to the Force Model Pulse Timing 2 Parameters. Duty cycle is calculated by dividing the pulse width by the period. Duty cycle = Pulse width / period.

Figure 158: Pulse Timing Parameters for an ITM

Force Mode Assignments

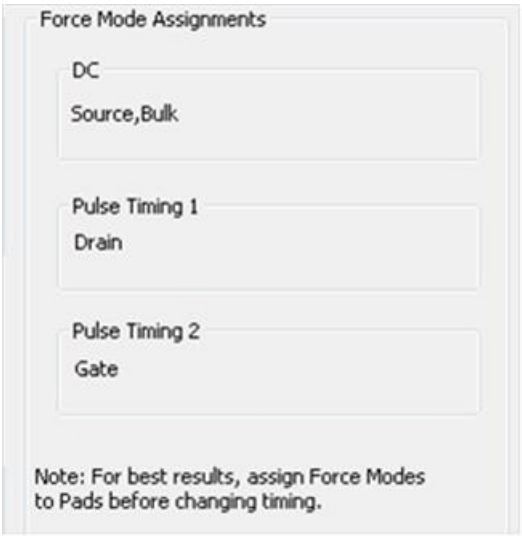
This part displays the force mode assignments for each pad (SMU). It is recommended to assign Force Mode to Pads (SMUs) in the SMU forcing and measuring settings table before changing timing settings.

DC: Displays the SMUs by Pad Name that have their Force Mode assigned to DC.

Pulse Timing 1: Displays the SMUs by Pad Name that have their Force Mode assigned to Pulse Timing 1.

Pulse Timing 2: Displays the SMUs by Pad Name that have their Force Mode assigned to Pulse Timing 2.

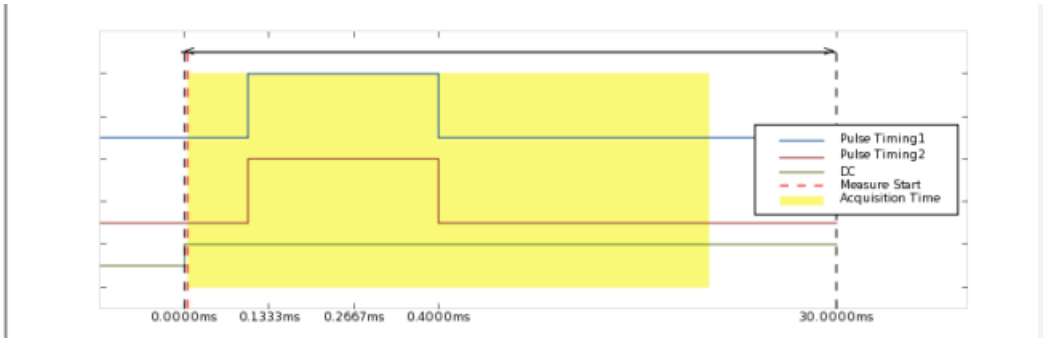
Figure 159: Force Mode Assignments for an ITM



Timing schematic

The timing schematic window is displayed according to the settings and is only visible in Pulse Available mode. DC only mode has no timing schematic.

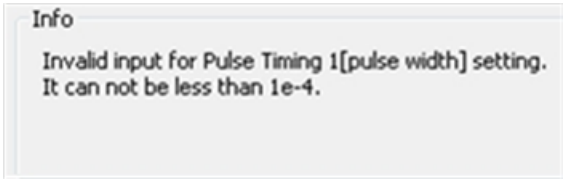
Figure 160: Timing schematic for an ITM



Info

Information about any problems concerned related to the timing settings will be displayed in the Info.

Figure 161: Info in the ITM Timing dialog



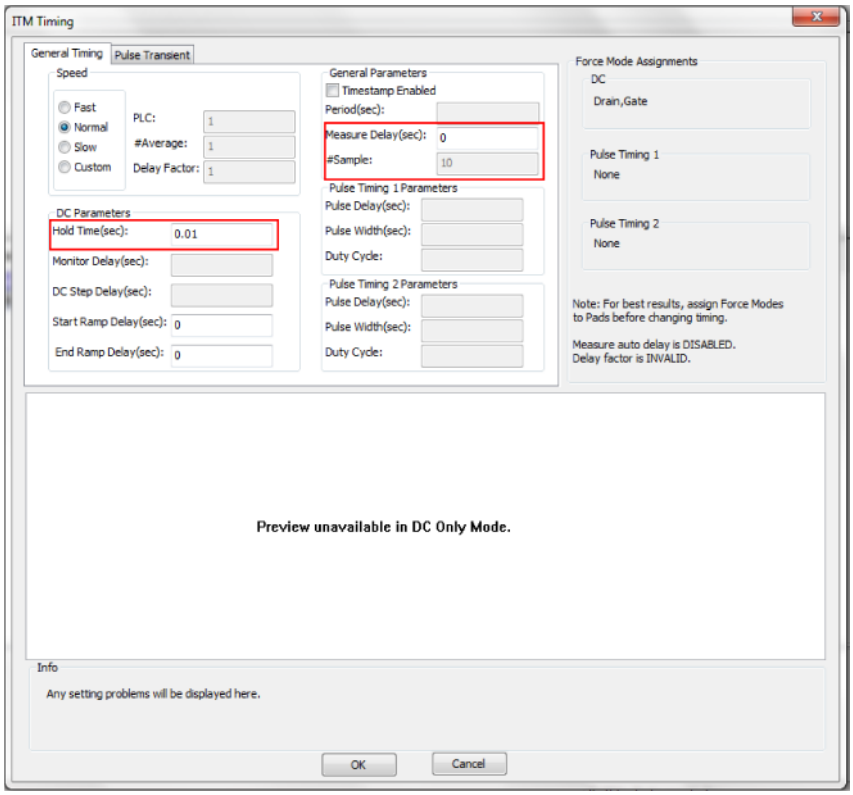
DC Only Timing settings

In DC Only, two types of tests can be implemented: Sweep or Sampling.

#Sample: when all the SMUs are set to Bias, sets the sample number of samples for the sampling measurement. # Samples is only available when all the SMUs are set to the Bias forcing function.

Sweep: If any terminal of the device under test is configured for a dynamic forcing function (step or sweep forcing function), the Sweeping mode is automatically enabled.

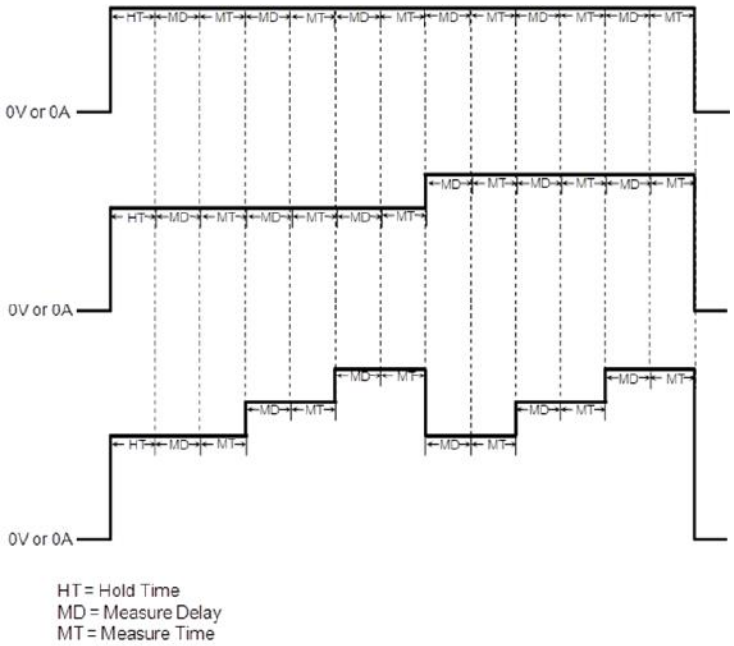
Figure 162: ITM Timing for DC only



Measure Delay: If you are using a sweep forcing function and want additional settling time before each measurement, you can specify an additional delay in the Sweep Delay edit box. You can specify a Measure Delay from 0 to 1000 seconds. The default Measure Delay is 0s.

Hold Time: The starting voltage or current of a sweep may be substantially larger than the voltage/current increments of the sweep. Accordingly, the source settling time required to reach the starting voltage or current of a sweep may be substantially larger than the settling times required to increment the sweep. To compensate, you can specify a Hold Time delay to be applied only at the beginning of the sweep. You can specify a Hold Time delay of 0 to 1000 seconds. The default Hold Time is 0 s.

Figure 163: Sweep timing diagram in DC only

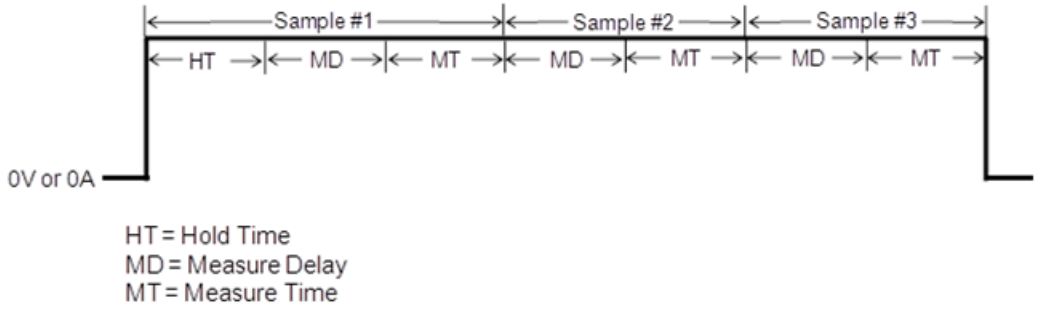


Sampling

The Sampling mode measures voltage and current as a function of time while forcing constant voltages/currents (Bias I or Bias V). For example, the sampling mode would be used to measure voltage while forcing a constant current. Time is measured relative to when the SMU(s) apply the forced voltage or current (that is, t = 0 at the step change from 0.0V/0.0A to the applied voltage/current).

ACS Basic enables the sampling mode option only when all terminals of the device under test are configured for static forcing functions: Voltage Bias or Current Bias, and the mode is must in DC Only mode.

Figure 164: Sampling timing diagram in DC only



Pulse Available Timing settings

When the Pulse Available mode is selected, the ITM will allow DC timing and two pulse timing settings for the SMUs. All the SMUs will have the same measure speed, period, and measure delay. However, they have separate transition delays (DC step delay, pulse delay), and the two pulse timings have their own pulse width.

The next figure indicates the settings for the pulse timing:

1. Measure speed: PLC and average number in acquisition time.
2. Delay time for DC bias SMU.
3. Pulse period and measure delay.
4. Pulse delay and pulse width for each pulse timing. The Duty Cycle is calculated according to pulse period and width.
5. Force mode assignments for each pad.
6. Timing diagram according to the settings.
7. Description of any problems with settings.

Figure 165: Pulse timing settings

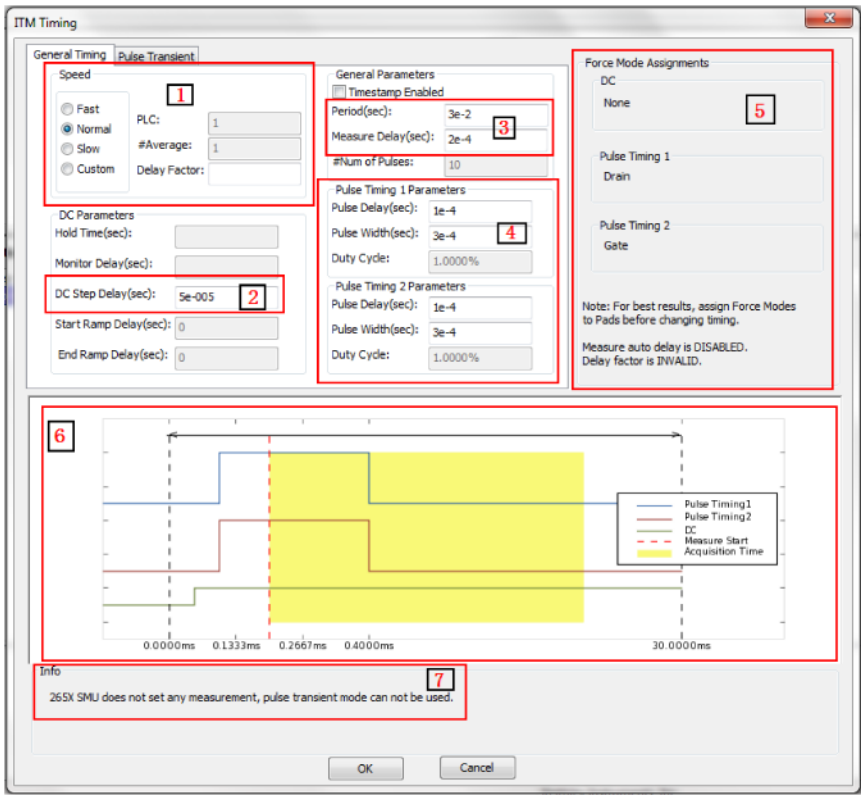
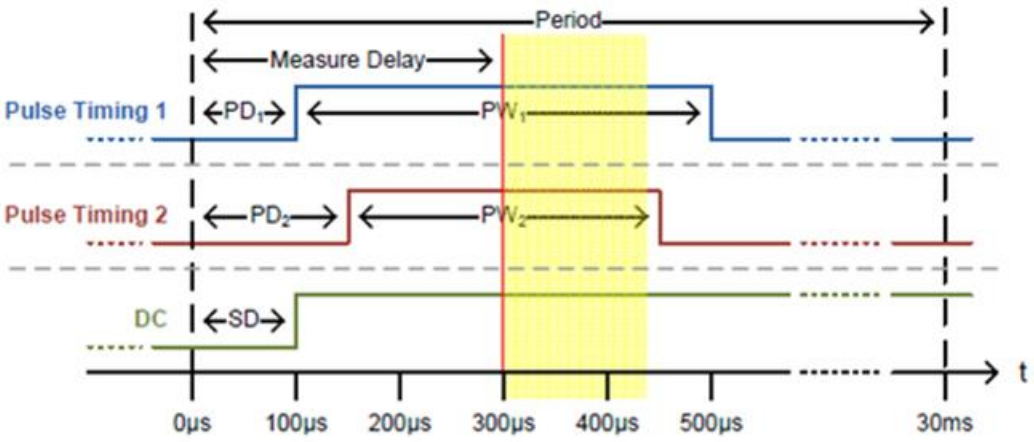


Figure 166: Pulse timing diagram

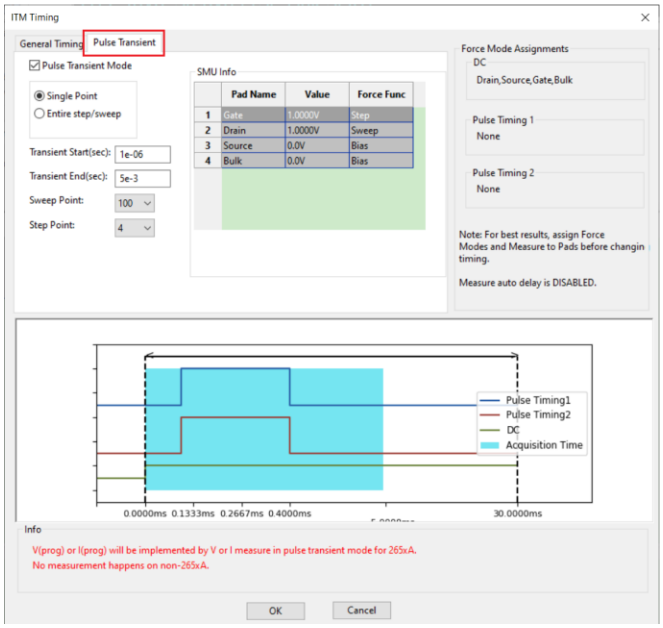


PD_1 – Pulse Delay for Timing 1
PW_1 – Pulse Width for Timing 1
PD_2 – Pulse Delay for Timing 2
PW_2 – Pulse Width for Timing 2
SD – DC Step Delay

Pulse Transient

Pulse Transient mode uses the Fast ADC that is available on some SMU models (for details about Fast ADC, refer to the documentation for the 2651A or 2657A) to capture the waveform for a single point in the step/sweep sequence. It is useful for determining the proper measurement delay setting for the step/sweep sequence by capturing the settling time of the SMU outputs. It is also useful for debugging troublesome data points in the step/sweep sequence without the need for an oscilloscope. It is only available for the 2651A and 2657A SMUs in Pulse Available mode. The Pulse Transient mode timing settings are in the ITM Timing setting window on the Pulse Transient tab (see next figure).

Figure 167: Pulse Transient mode settings



Single Point or Entire step/sweep: Selects whether to output a single point from the step/sweep sequence or the entire step/sweep sequence to perform the pulse transient capture. Single Point only outputs the selected point of the step/sweep sequence making the capture quick because the test does not have to run through the entire sequence. Entire step/sweep outputs the entire step/sweep sequence, and makes the transient measurement only on the selected point. Entire step/sweep is useful when the measurement on the device changes due to effects like device self heating. By outputting the entire step/sweep sequence, the captured waveform will be a better representation of what is happening during that point in the step/sweep sequence.

Transient Start(s): Sets the time in seconds from the start of the period to the start of the acquisition.

Transient End(s): Sets the time in seconds from the start of the period to the end of the acquisition.

Sweep Point: Sets the point in the sweep that will be captured.

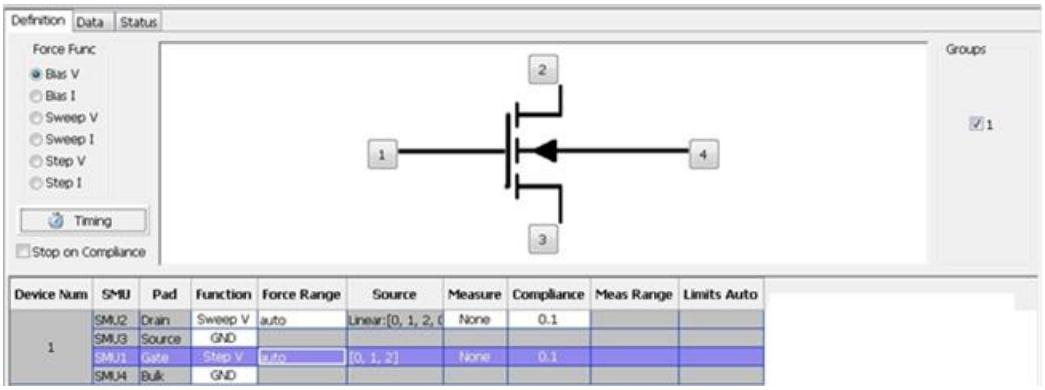
Step Point: Sets the step point that will be captured.

The Sweep Point and the Step Point together appoint a certain period for waveform capture. The SMU info table lists the Pad name, force value and force function of the selected period.

Model 42xx SMU ITM

This ITM is based on the Model 42XX SMU device. It supports both DC and pulse measurement, but it does not have pulse mode. Each ITM is associated with a specific ITM work area. An ITM work area displays the ITM's definition (configuration), test data, and status information (see next figure).

Figure 168: Definition tab of the Model 42XX SMU ITM



Timing: This button opens the ITM timing dialog box where you can set detailed timing settings for pulse and synchronization between SMUs.

Stop on Compliance: Enables the test to abort and turn off the SMU output when it reaches compliance.

Device schematic: Shows the device image and pad settings according to the device level settings in the configuration navigator.

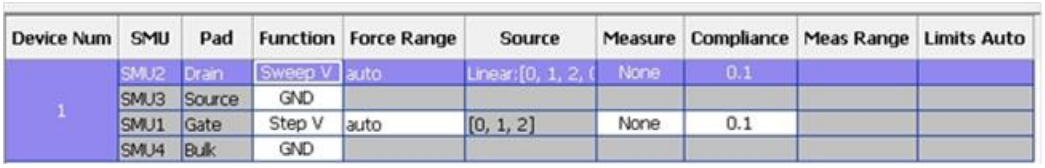
Groups: Select the group or groups where you want to run the test. The group is determined by the GPIB or ethernet address. Make sure that all the groups you select have the same hardware configurations as your ITM.

SMU parameters: The forcing and measuring settings table is where you can set your parameters for testing.

Model 42xx SMU parameters

The following figure displays the items that you can use to set up your testing for forcing and measuring functions for each SMU.

Figure 169: Model 42XX SMU parameters table



Device Num: Sub-device number defined in the Device work area; based on the hardware connection.

SMU: SMU number defined in the Device work area; based on the hardware connection.

Pad: Pad name defined in the Device work area; based on the hardware connection.

Function: Force function type.

Force Range: Specifies the SMU current range used to force the sweep current:

- Auto option commands the SMU to automatically optimize the force range as the sweep progresses. This option provides the best resolution and control when sweeping several decades.
- Numerical range options allow you to manually select a SMU range to meet your needs (the drop list contains all the supported ranges according to the SMU and force function). This range must accommodate the stop or start value, whichever is greater. For example, when forcing stop 1.5 mA, you should choose the 10mA SMU range or the Auto SMU range. The Auto SMU range allows the SMU to select the most appropriate range.

Source: Allows you to set the source value according to the selected Function. When the Function is Bias V or Bias I, it sets the force value in this cell. When the Function is Sweep V, Sweep I, Step V or Step I, the Source settings are similar to the Model 26xxB and 24xx SMU parameters.

Measure: Measure the current unit in amps and the voltage unit in volts. All options are:

- I: Actual measured current values.
- V: Actual measured voltage values.
- V(prog): The programmed voltage, as requested forced voltage values. For example, 2.5V is to be forced at a transistor drain terminal. The programmed value [V(prog)]) is exactly 2.5V, even if the measured value (V) is 2.4997V. Under the programmed option, the reported value is exactly 2.5V, even if the measured value is 2.4997V.
- I(prog): The programmed current, as requested forced current values, when the source function is voltage.
- I+V: Measures both I and V.
- I+V(prog): Measures I and returns V(prog).
- V+I(prog): Measures V and returns I(prog).

Compliance: Specifies any valid SMU current or voltage compliance limit. The current compliance is present for voltage forcing functions and the voltage compliance is used for current forcing functions.

Meas. Range: Specifies the SMU range used. The current measure range is present for voltage forcing functions, and the voltage measure range is present for current forcing functions only. If I is selected in the Measure cell, ACS Basic enforces the Force range first, and ignores the measure range. You can specify the range used by the SMU for current using the following options:

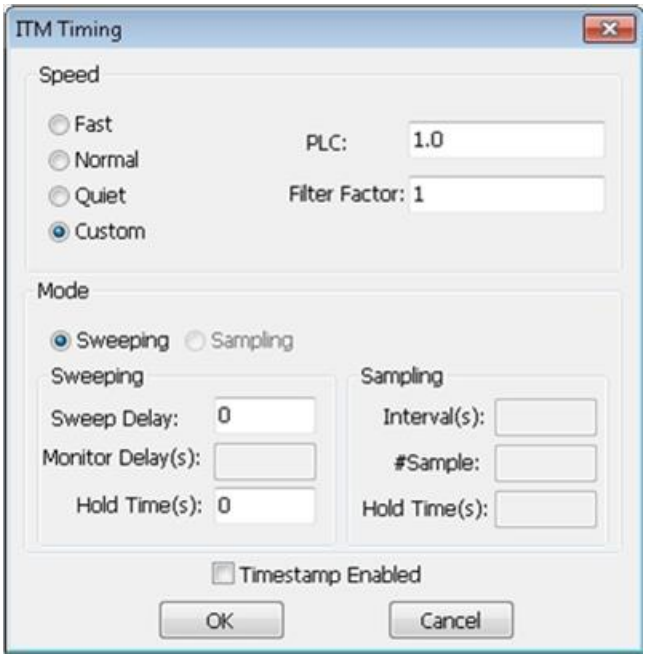
- Auto option commands the SMU to automatically optimize the current measurement range. This option provides the best resolution
- Limits Auto is only enabled when the force function is voltage and there is current measurement. This option is a compromise between the full Auto option and a fixed range option. You can specify the minimum range that the SMU uses when automatically optimizing the current measurements. This option saves time when maximum resolution at minimum currents is not needed.
- Numerical range options allow you to manually select a fixed current measurement range to suit your needs. It selects current ranges when the source function is voltage, and selects voltage ranges when the source function is current. All the supported measurement ranges are listed according to the SMU type

Limits Auto: Enabled after selecting Limits Auto in the Meas Range cell.

Timing

When you click the ITM Timing icon in the ITM Definition tab, the ITM Timing GUI opens. The ITM Timing GUI is where you set the measurement speed and delay time of the test.

Figure 170: 42XX SMU ITM Timing settings



Speed

The Speed area of the ITM Timing GUI allows you to:

- Locally select the Fast, Normal, Quiet, or Custom measurement speed modes.
- Configure Custom speed settings in the Custom measurement speed mode.

Fast, Normal, Quiet, and Custom modes are defined as follows:

- Fast: Optimizes speed at the expense of noise performance. It is a good choice for fast measurements where noise and settling time are not concerns.
- Normal: The default and most commonly used setting. It provides a good combination of speed and low noise, and is the best setting for most cases.
- Slow: Optimizes the low-noise measurements at the expense of speed. If speed is not a critical consideration, it is a good choice when you need the lowest noise and most accurate measurements.
- Custom: Allows you to fine-tune the timing parameters to meet a particular need. With Custom, you can configure the integration time and the number of readings to produce a composite setting. The entries in the PLC and # Average edit fields control the A/D (analog-to-digital) converter integration time used to measure a signal. A short integration time for each A/D conversion results in a relatively fast measurement speed, at the expense of noise. A long integration time results in a relatively low noise reading, at the expense of speed. The integration time setting is based on the number of power line cycles (NPLCs):

  For 60 Hz line power, 1.0 PLC = 16.67 ms (1/60 of a second).

  For 50 Hz line power, 1.0 PLC = 20 ms (1/50 of a second

Mode area

There are two test modes you can select in the Mode area of the ITM Timing GUI: Sweeping and Sampling.

- **Sweeping:** Applies to any ITM where one or more forced voltages/currents vary with time.
- **Sampling:** Applies to any ITM where all forced voltages or currents are static (Bias I or Bias V), with measurements typically made at timed intervals. For example, sampling mode is used to record a few static measurements, or to time-profile the charging voltage of a capacitor while forcing a constant current.

The Mode is used to observe and change the test mode as follows:

For a new ITM, the Mode allows you to select the Sweeping or Sampling test mode. Selecting the appropriate test mode simplifies configuration options and helps to avoid errors:

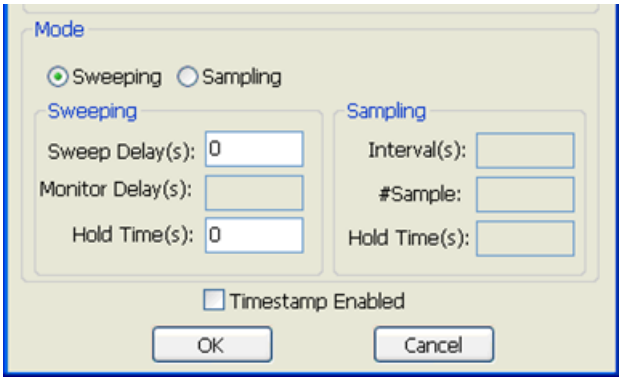
1. Only the static forcing functions are configurable in the Sampling mode.
2. Only mode-appropriate timing options are configurable.

For an existing library ITM that is in the Sampling mode, the Mode area allows you to change to the Sweeping mode if you want to change some of the static forcing functions to dynamic forcing functions.

Sweep mode

If any terminal of the device under test is configured for a dynamic forcing function (step or sweep forcing function), the Sweeping mode is automatically enabled. You can configure two Sweeping mode settings in the ITM Timing GUI (Sweep Delay and Hold Time). You can configure these settings for all measurement Speed modes: Fast, Normal, Quiet, and Custom.

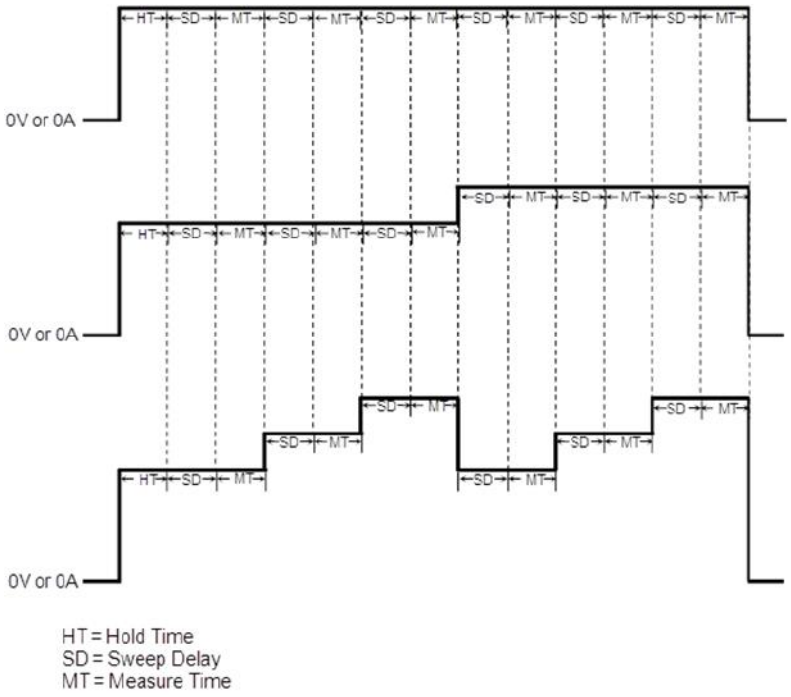
Figure 171: ITM Sweep timing GUI



Sweep Delay: If you are using a sweep forcing function and want additional settling time before each measurement, you can specify an additional delay in the Sweep Delay edit box. You can specify a Sweep Delay from 0 to 1000 seconds. The default Sweep Delay is 0s.

Hold Time: The starting voltage or current of a sweep may be substantially larger than the voltage/current increments of the sweep. Accordingly, the source settling time required to reach the starting voltage or current of a sweep may be substantially larger than the settling times required to increment the sweep. To compensate, you can specify a Hold Time delay to be applied only at the beginning of each sweep. You can specify a Hold Time delay of 0 to 1000 seconds. The default Hold Time is 0s (see next figure).

Figure 172: 42XX sweep mode timing diagram



Hold Time: When the test is started, hold time (HT) provides extra settling time for the initial step change of the source. Hold time is a global setting, therefore, it is the same for all SMUs in the test system. Note that hold time is applied at the start (only) of each sweep.

Sweep Delay: Provides additional settling time for each step in the sweep. It is a global setting and is applied identically to all SMUs in the test system.

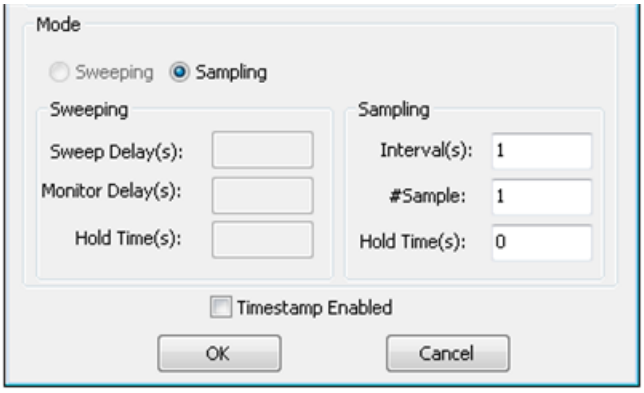
Measure Time: Determined by the # Average and the A/D Integration Time (NPLC). All SMUs in the test system are synchronized. The measure time for the SMU requiring the longest measure time is the same for all SMUs in the test system.

Sampling mode

The Sampling mode measures voltage and current as a function of time while forcing constant voltages/currents (Bias I or Bias V). For example, sampling mode is used to measure voltage while forcing a constant current. Time is measured relative to when the SMUs apply the forced voltage or current (that is, t = 0 at the step change from 0.0V/0.0A to the applied voltage/current).

ACS Basic enables the sampling mode option only when all terminals of the device under test are configured for static forcing functions: Voltage Bias or Current Bias. If any terminal of the device under test is configured for a step or sweep forcing function, the Sampling mode option is unavailable.

Figure 173: ITM sample timing settings



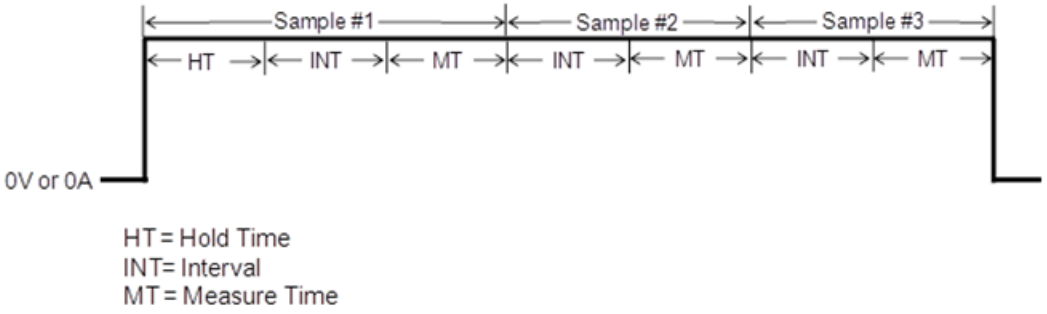
If an ITM is configured for the Sampling mode, you can configure three Sampling mode settings: Interval, #Sample, and Hold Time.

Interval(s): Specifies the time between measurements (also known as data points). Intervals can be set from 0 to 1000 seconds.

#Sample: Specifies the number of data points to be acquired. #Samples can be set from 1 to 4096.

Hold Time(s): After initial application of voltage or current by the SMUs, the source settling times can be substantial. To allow for settling, you can specify extra hold time (delay) to be applied before making the first measurement. You can specify a hold time from 0 to 1000 seconds (see next figure).

Figure 174: 42XX sampling mode timing diagram



Hold Time: If needed, can be used to allow for extra source settling after the initial application of voltage or current from the SMU. Hold Time is a global setting, and is the same for all SMUs in the test system.

Interval: Specifies the time between measurements (also known as data points). Interval (INT) can be set from 0 to 1000 seconds.

Measure Time: Determined by the A/D Integration Time. All SMUs in the test system are synchronized. The measure time for the SMU requiring the longest measure time is the same for all SMUs in the test system.

Configure a script test module (STM)

A STM is a test module that supports Test Script Processor (`.tsp`) files. A `.tsp` file is written in TSP for testing a device with a Series 2600B, Model 3706A, or Model 707B/708B in TSP or Instrument Control Library (ICL) mode. Several STM examples and tests are installed with ACS Basic. These scripts can be imported and modified in the STM work area.

The `.tsp` files can also have a GUI associated with them to simplify configuring tests. STM GUIs can either be the standard GUI built into ACS Basic or a user-created XML Resource (XRC) GUI.

The Test Script

A script is a collection of instrument control commands and programming statements. The statements control script execution and provide facilities such as variables, functions, branching, and loop control. Scripts are programs that are written using the Lua programming language. Descriptions of the commands and examples can be found in the *ACS Basic Libraries Reference Manual*.

STM Script examples

Scripts can be written in the STM or using the Test Script Builder (see Script Editor (on page 6-1) for more information). Scripts are loaded into the Series 2600B and can be saved in nonvolatile memory. Scripts not saved in nonvolatile memory will be lost from instrument memory when the Series 2600B is turned off.

The script below is a script used to perform the resistance measurement.

Example:

```
local  v_value = {}
local  i_value = {}
local  sensemode = 0
local  testmode = 0
local  RSMU1 = SMU1
local  RSMU2 = SMU2
local  forcevalue = 1
local  myLIMIT = 0.1
local  myNPLC = 1
local  testdelay = 0.1
local  resetflag = 1
local  Rvalue = {}

setmode(RSMU1, KI_INTGPLC, myNPLC)
 setmode(RSMU1, KI_SENSE, sensemode)

if RSMU2 ~= KI_GND then
   setmode(RSMU2, KI_SENSE, sensemode)
   limiti(RSMU2, 1)
   end –if
if testmode == 0 then
       limiti(RSMU1, myLIMIT)
       forcev(RSMU1, forcevalue)
elseif testmode == 1 then
      limitv(RSMU1, myLIMIT)
      forcei(RSMU1, forcevalue)
end –if
if RSMU2 ~= KI_GND then
     forcev(RSMU2, 0)
end –if

delay(testdelay)
intgv(RSMU1, v_value)
intgi(RSMU1, i_value)
Rvalue[1] = v_value[1]/i_value[1]

postdata("R_single", Rvalue[1])
if resetflag == 1 then
    devint()
end --if
```

Script function

Lua facilitates grouping commands and statements using the function keyword. Therefore, a script can also consist of one or more functions. Once a script has been run, the host computer can call a function in the script directly. Lua allows you to define functions. A function can take a predefined number of parameters and return multiple parameters.

To define a function and call that function, see the example below.

Example:

```
function add_two(parameter1, parameter2)
return(parameter1 + parameter2)
end
print(add_two(3, 4))
```

Below is a function that returns multiple parameters. It is used to perform the resistance measurements.

Example:

```
function R-single(sensemode, testmode, RSMU1, RSMU2, forcevalue, myLIMIT, myNPLC,
   testdelay, resetflag, Rvalue)
        local v_value = {}
        local i_value = {}

setmode(RSMU1, KI_INTGPLC, myNPLC)
setmode(RSMU1, KI_SENSE, sensemode)
if RSMU2 ~= KI_GND then
   setmode(RSMU2, KI_SENSE, sensemode)
   limiti(RSMU2, 1)
end --if
if testmode == 0 then
   limiti(RSMU1, myLIMIT)
   forcev(RSMU1, forcevalue)
elseif testmode == 1 then
   limitv(RSMU1, myLIMIT)
   forcei(RSMU1, forcevalue)
end --if
if RSMU2 ~= KI_GND then
   forcev(RSMU2, 0)
end --if
delay(testdelay)
intgv(RSMU1, v_value)
intgi(RSMU1, i_value)
Rvalue[1] = v_value[1]/i_value[1]
postdata("R_single", Rvalue[1])
if resetflag == 1 then
   devint()
end --if
end --function
```

Call the function as shown below:

```
local sensemode = 0
local testmode = 0
local RSMU1 = SMU1
local RSMU2 = SMU2
local forcevalue = 1
local myLIMIT = 0.1
local myNPLC = 1
local testdelay = 0.1
local resetflag = 1
local Rvalue = {}
R-single(sensemode, testmode, RSMU1, RSMU2, forcevalue, myLIMIT, myNPLC, testdelay,
   resetflag, Rvalue)
```

Script header

It is recommended that you add an OUTPUT list header at the top of each STM. This causes the following:

- In the Data tab of a test module in Test Setup, you will see the variable sequence is the same as in the OUTPUT list.
- This OUTPUT list is of the highest priority compared to "postdata." If you add the OUTPUT list, postdata will not be scanned or used.
- If a parameter is not in the OUTPUT list but you still post it, it will be added in the datasheet as well.

Add the following OUTPUT header:

```
--[[
--OUTPUT--
I_drain
V_gate
--End of OUTPUT--
]]--
```

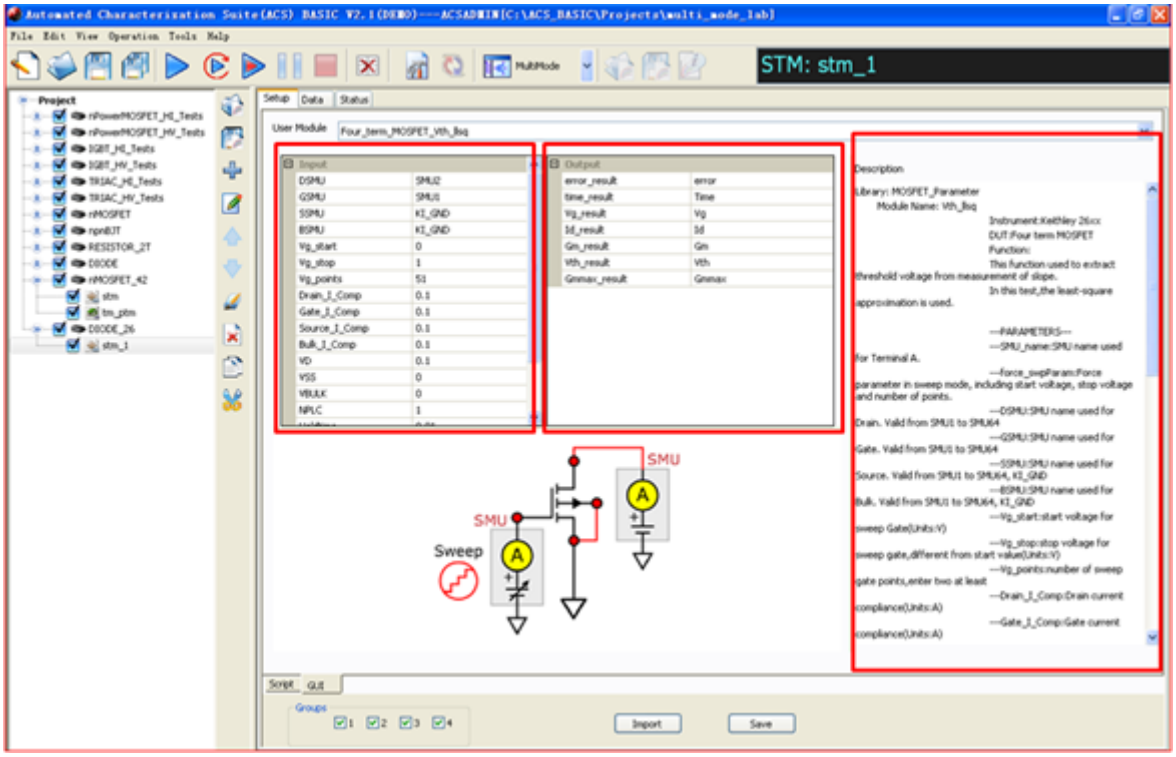
The GUI

STMs can include either the standard GUI or a custom built XRC GUI. Both GUIs provide an interface to change parameters in the script without editing the script itself. If a GUI is associated with your STM, you can view both the GUI and the script in the STM work area. See Setup tab (on page 5-40) for more information.

Standard GUI

A standard GUI can be created using the Script Editor tool. In the standard GUI, you can input the parameters directly without editing the script itself. There are three areas in the standard GUI Setup tab: Input, Output, and Description (see next figure).

Figure 175: Standard STM GUI



XRC GUI

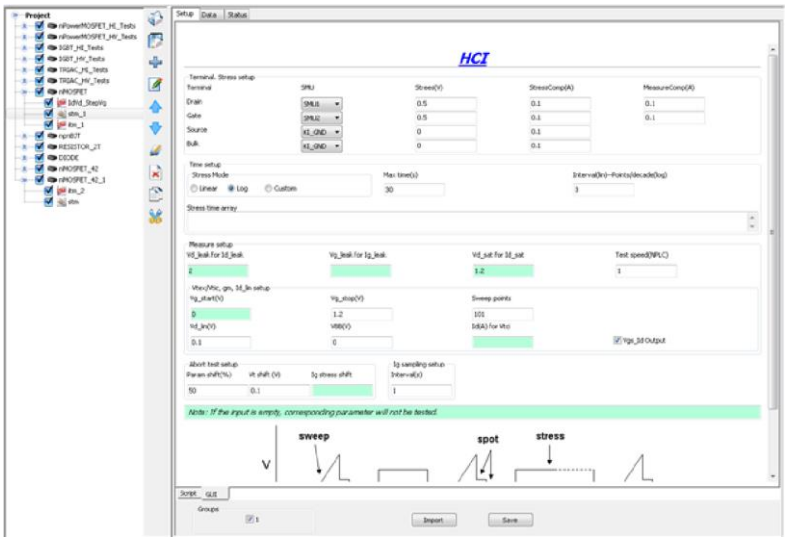
XRC GUIs are created using the XRCed tool and are saved as a file with the `.xrc` extension. If the imported TSP file has a corresponding `.xrc` GUI file, ACS Basic automatically loads and opens the GUI. You can set parameters on the GUI and you do not need to edit the parameters within the script itself.

XRCed can be accessed by clicking the ACS Basic Tools menu, then the Custom Test GUI designer command. For more information on how to generate the `.xrc` file, refer to Create a XRC GUI file (on page 5-40).

An XRC GUI is completely customizable, and can include windows, menus, and a variety of input controls. The HCI STM in the next figure is one example of a STM that includes a XRC GUI.

You must include the `.xrc` file in the header of the script to include it in the STM.

Figure 176: HCI module opened in STM work area



All files in the WLR folder (the path is \\ACS_BASIC\library\26Library\WLR) have a XRC GUI. The GUI file is installed in the XRC folder: \\ACS_BASIC\library\26Library\xrc.

If an input parameter in the GUI is left empty, the value of this input parameter shows as "nil" in the function call of the script. After the correct input is entered, the function call area of the Setup tab of this test module automatically updates (see next figure).

Figure 177: Change of function call

Open a STM work area

The STM work area allows you to enter information that defines a given STM. The work area also allows you to view and analyze test data (both numerically and graphically) that is created by the STM and check its status.

To open a blank STM work area in Multitest mode:

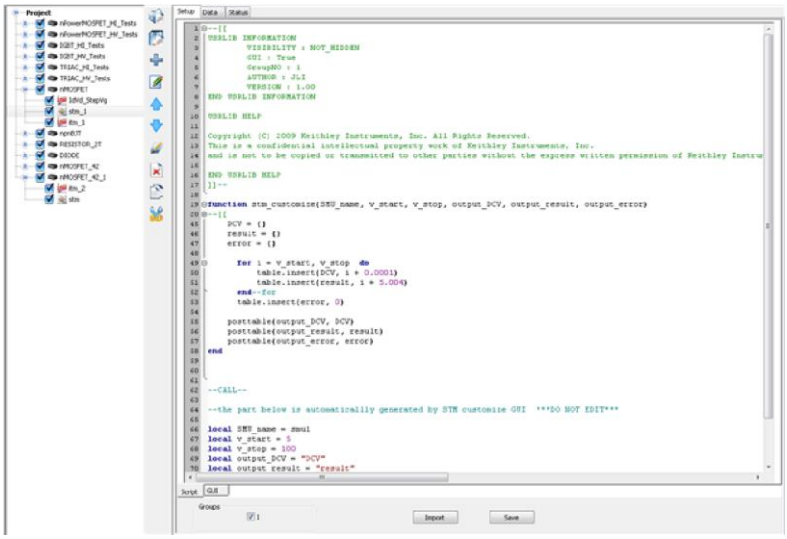
1. Select the device in the configuration navigator where you want to create the STM.
2. Select the **Add new test** icon.
3. Select **STM** in the Specify the Test Type dialog. Select **OK**. A new test named "stm" should be added to your device.
4. If you would like to run or modify an existing TSP file, select **Import** at the bottom of the STM work area in the Setup tab. Refer to the *Automated Characterization Suite* (*ACS) Basic Edition Libraries Reference Manual* (document number ACSBASIC-908-01) for more information.

You can also access existing STMs from the test selection area of Multitest or Single-test mode. Most of the available STMs in this area have a XRC GUI and are in the wafer level reliability (WLR) library.

As with all test modules, make sure to check that all your SMU connections match what is written in the script or displayed in the GUI before you run the module. If you have written your own script or modified and existing file, you can save the configuration by clicking the **Save** icon on the toolbar or by clicking the **Save** button at the bottom of the screen.

The STM work area contains the tabs Setup, Data, and Status (see next figure).

Figure 178: The Setup tab of a XRC STM



STM Setup tab

In this tab, you can:

- Specify a TSP-created user library and user module
- Specify a limited number of test parameters
- Import and save the STM

To configure or reconfigure an STM, you need to configure the features of the Setup tab, which is the default tab of the STM work area. The STM Setup tab area contains the following:

TSP Script area: Displays a blank page for new STMs. It Is the main work area for viewing and editing a script test module (STM).

Groups information area: Allows you to select the groups where you would like the present STM to be assigned.

STM Data tab

The Data tab displays the STM results in the data worksheet in a spreadsheet format. This spreadsheet allows you to perform data analysis. It displays the same output information as the Setup tab. It also allows you to create and export graphs of the test and test-analysis results. It also provides for flexible plot-data selection, formatting, annotation, and numerical coordinate display (using precision cursors).

STM Status tab

The Status tab monitors the current configuration status of the STM, reporting its readiness for use, and recommending additional preparations, if necessary. A test-ready report for a STM is the same as for an ITM.

Create a XRC GUI file

XRC GUIs are created using the XRC editor tool (XRCed) and saved as `.xrc` files. The XRC GUI can only be used to assign input values in the script. It cannot be used to display any outputs of the script.

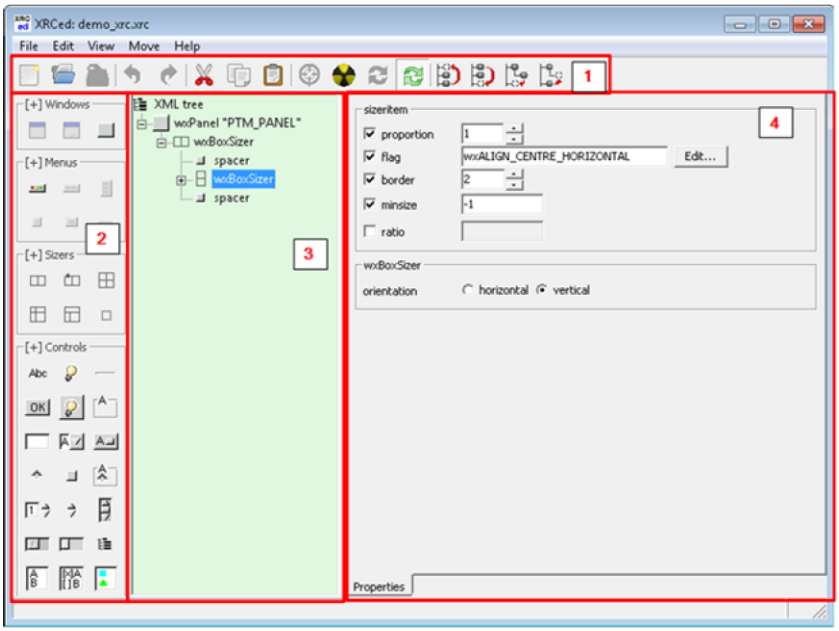
XRCed uses a tree format to create the GUI. As components are added, they must be placed as the child of a parent component within the XML tree, and their placement in the tree determines where they will be positioned in the GUI.

XRCed GUI

To start XRCed tool, select the ACS Basic Tools menu, then select Custom Test GUI Designer (XRC). When the XRCed GUI opens, you see four areas, as shown in the following figure:

1. Toolbar
2. Component selection area
3. XML tree
4. Work area

Figure 179: XRCed GUI



At the top of the XRCed, the toolbar allows you to execute several common commands and commands specific to the XRCed. The toolbar is grouped into four different sets of commands:

1. New, Open, and Save file
2. Undo and Redo
3. Cut, Copy, Paste

4. Locate, Test, Refresh, and Auto-refresh
   - **Locate**: Finds a control in the XML tree. Click **Locate**, then click the control in the test window to view the control properties.
   - **Test**: Creates a test window.
   - **Refresh**: Updates the test window to show any changes you have made to the GUI. This button is only available when a test window is open.
   - **Auto-refresh**: Updates the test window to show any changes whenever a new component is selected in the XML tree, if enabled.
5. Move up, Move down, Make sibling, Make child
   - **Move up**: Moves the selected component to the next position in the tree.
   - **Move down**: Moves the selected component to the position in the tree.
   - **Make sibling**: Moves the selected component up to the next level in the tree hierarchy.
   - **Make child**: Moves the selected component down to the next level in the tree hierarchy.

On the left side of the XRCed GUI is the component selection area. These components are grouped into Windows, Menus, Sizers, and Controls:

- **Windows**: The largest component of the GUI; other components are grouped under windows:
  - **wxFrame** or **wxDialog** windows can only be created within the XML tree.
  - **wxPanel** is the only window that can be created within another component.
- **Menus**: Create an area to group together controls.
- **Sizers**: help you organize the components in your window.
- **Controls**: make up the main content of the GUI. In addition to creating data input fields and descriptions for them, you can add more interactive ways to display the content.

The center column of the XRCed displays the XML tree. This is where you can navigate through different components of the GUI. To add a component to the tree, select the component in the component selection area. It appears in the tree as a child to the component that was previously selected in the XML tree. Double-click the component in the tree to change its configuration.

The work area in the right column allows you to modify the components of the GUI. All components have a Properties tab in the work area that allows you to change settings specific to the component. Window and Control type components also have a Style tab that lets you change properties such as the font and layering settings.

Create a new XRC

The following instructions are a step-by-step guide on how to create an XRC GUI. For example, consider the following function within a script:

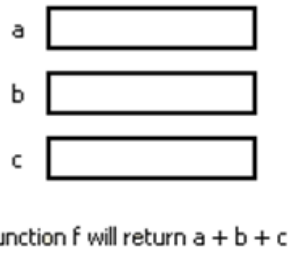
```
function f (a, b, c, output)
   output = a + b + c
   postdata ("output", output)
end
```

To create an XRC GUI for this function, use the codes that are in a TSP file. The TSP and XRC GUI file are linked by the `.xrc` file name that appears at the top of the TSP file, for example:

```
----<<xrc=example.xrc>>----
```

Before you start building your XRC GUI, you should have a draft of what you would like the GUI to look like. Consider what controls you would like to include, such as labels, static text, text boxes, and images. This makes the process faster and more efficient. The next figure is a draft of what the example GUI will look like:

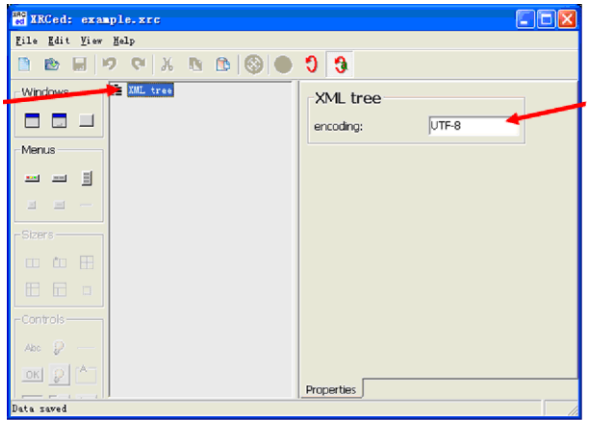
Figure 180: An example GUI image



Once you have decided on a layout, open the XRCed program:

1. Click the ACS Basic **Tools** menu.
2. Select **Custom Test GUI Designer(XRC)**.
3. Click the **File** menu, then select **New** to open a new file.
4. Click the **XML tree** and change the encoding to UTF-8 in the encoding edit box (see next figure).
5. Click the **File** menu, then select **Save As** and save your `.xrc` file in the `C:\ACS\library\26Library\xrc` folder to set the directory path.

Figure 181: TSP GUI builder interface



The first thing you can add to your GUI is a window, which is the outermost shell of the GUI. You must use a wxPanel window for the GUI to work in ACS Basic.

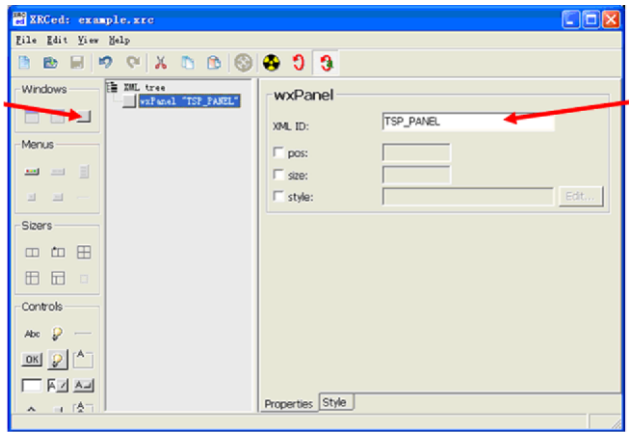
For our example, add a panel under the XML tree:

1. Click the Add wxPanel icon on the left panel to add a panel under the XML tree (see next figure).
2. Select the wxPanel in the XML tree to bring up its workspace, then change the XML ID to "TSP_Panel" (see next figure).

NOTE

Naming the panel "TSP_PANEL" for the new object is essential for creating a GUI related to a TSP script. You should only use this name in the XML ID edit box. If you create an XRC GUI for a PTM, enter PTM_PANEL in the XML ID edit box (see Configure an XRC PTM (on page 5-59) for more information).

Figure 182: Add panel for new object

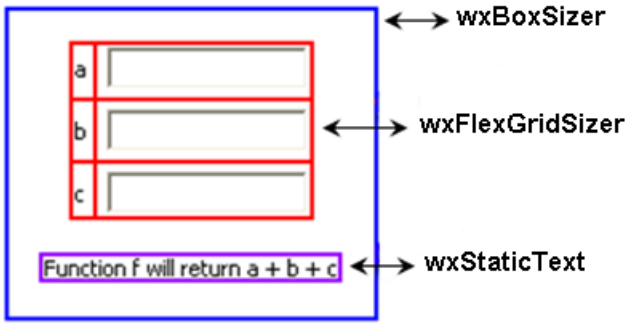


After creating your panel, start adding Sizers to organize the layout of the GUI. Sizers generally do not create visuals that are seen in the GUI, they only create a structure to organize the components that they contain. Once you add a Sizer in the XML tree, you can then add components underneath it that will be arranged in the GUI based on the Sizers configuration. These components can either be Controls or other Sizers used to create further substructures. There are several types of sizers that can be added to an XRC GUI.

- **wxBoxSizer**: The most basic Sizer. Components that are listed under a wxBoxSizer in the XML tree will be displayed in a horizontal row or vertical column, depending on the configuration.
- **wxStaticBoxSizer**: Similar to the wxBoxSizer, but it also creates a visual border around the entire Sizer in the GUI. You can also add text to embed in the upper right corner of the border.
- **wxGridSizer**: Allows you to create multiple rows and columns. Components listed under this Sizer in the XML tree fill in the GUI grid from left to right, starting at the top row and proceeding to the one below after each row is full.
- **wxFlexGridSizer**: Similar to the wxGridSizer, but offers more flexibility in shaping the grid.
- **wxGridBagSizer**: Disabled in ACS Basic
- **wxSpacer**: Used as a place holder to create a specifically shaped space in the GUI.

To continue our example, add Sizers to create the overall structure for the GUI (see next figure).

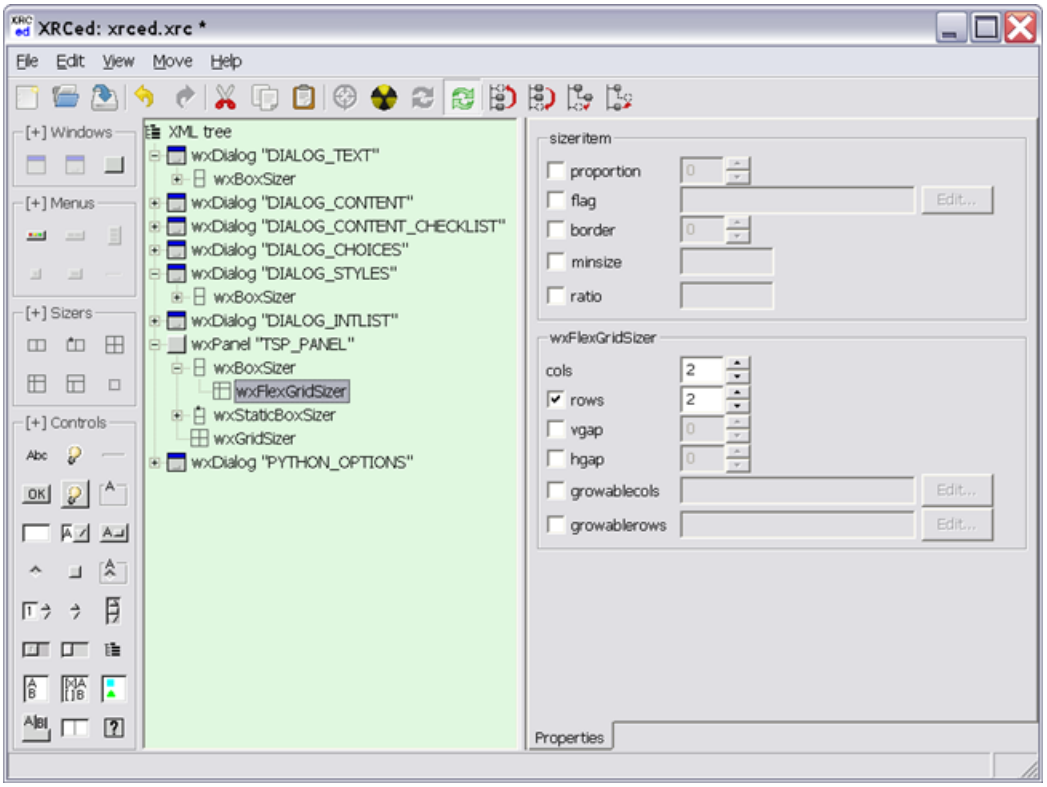
Figure 183: Design GUI structure



1. Click the **wxPanel** node of the XML tree.
2. Click the **wxBoxSizer** icon to add a wxBoxSizer under the wxPanel.
3. Select **wxBoxSizer** orientation: vertical or horizontal.
4. Double-click the **wxBoxSizer** node to make sure it is selected in the XML tree. Click the **wxFlexGridSizer** icon to add a wxFlexGridSizer under wxBoxSizer.

5. Double-click the **wxBoxSizer** node to make sure it is selected in the XML tree. Click the **wxStaticText** icon to add a wxStaticText under wxBoxSizer after the wxFlexGridSizer.

Figure 184: Add Sizers under the TSP_PANEL



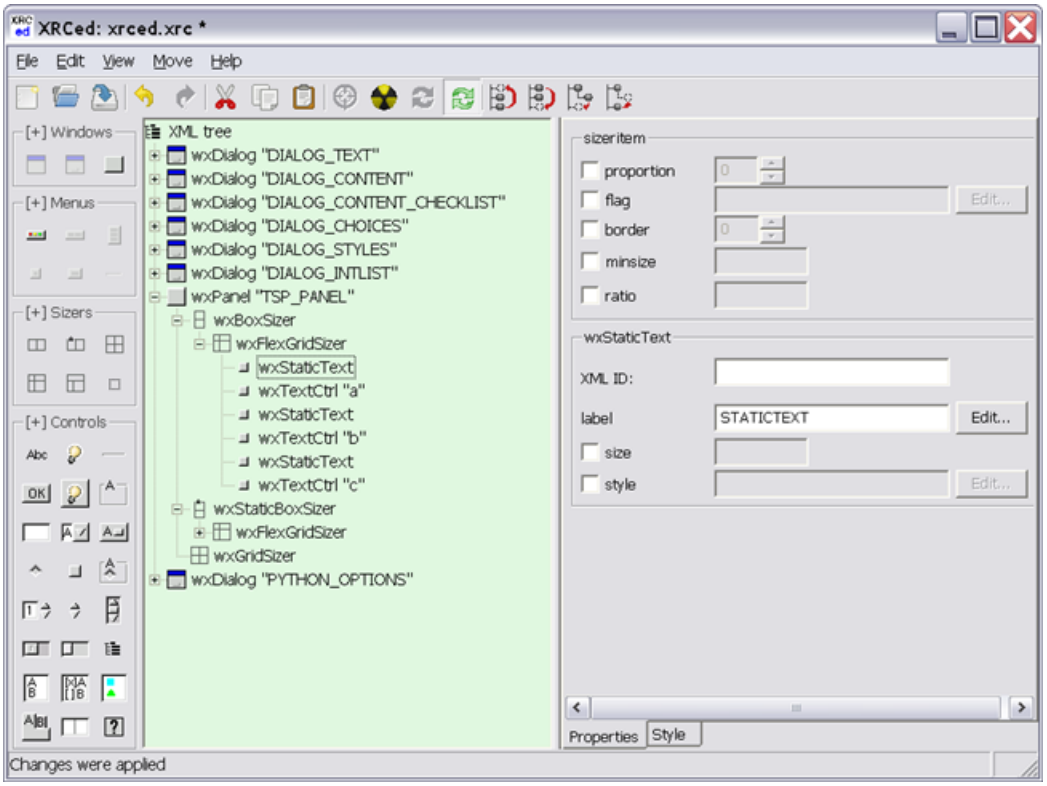
Note that the example will not be fully developed; the only visible component is the wxStaticText. After more components are added, the structure created by the Sizers becomes visible.

Once the structure is complete, add Controls to fill in the GUI. Add labels and text box under the wxFlexGridSizer, and then modify the information for each:

1. Click the **wxFlexGridSizer** node of the XML tree.
2. Set cols (columns) and rows: item on the right of interface to match the number of rows in the GUI. For example, in the previous figure, the rows should be changed to three in order to match the GUI draft.
3. Click the **wxStaticText** icon and the **wxTextCtrl** icon in the Controls box to add a **wxStaticText** (label) and a **wxTextCtrl** (text box) under the wxFlexGridsizer. Repeat this step two times to complete all three rows.
4. Click each **wxStaticText** component in the **wxFlexGridSizer** and set the label in the work area. In this example, set the label edit box to a, b, and c respectively (see next figure).

5. Click each **wxTextCtrl** component in the **wxFlexGridSizer** and set the XML ID. For example, set the XML ID edit box to a, b, and c respectively. This step is important because it links the data input in these boxes to be used as variables in the script.
6. Click the **wxStaticText** component in the **wxBoxSizer** and set the label at the bottom of the GUI. For our example, enter the text "Function f will return a + b + c."

Figure 185: Modify information



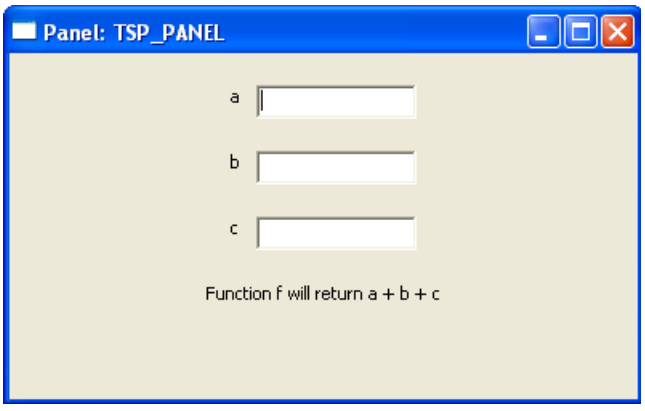
Now all the content has been added to your GUI. The last thing to do is add formatting to align and space out the components. Most of the formatting will be done in the "sizeritem" area of the work area for the wxFlexGridSizer and the wxStaticText at the end of the GUI. Since we are using a wxFlexGridSizer, the labels and text boxes in the grid will align nicely and do not need any extra formatting.

To add formatting:

1. Click the **wxFlexGridSizer**. Check the border option in the sizeritem area of the work area and increase the value to 20. This border adds space around the grid so that it does not push against the edge of the window.
2. Check the flag option for the wxFlexGridSizer.
    a. Select **Edit…** to the right, which opens the Sizer item flags dialog.
    b. Select **wxTOP** and **wxBOTTOM**. These add the border space to the top and bottom edge of the grid. Alternately, you could select wxLEFT, wxRIGHT, or wxALL for other border configurations.
    c. Select **wxALIGN_CENTER_HORIZONTAL**. This aligns the grid to the horizontal center of the window.
    d. Select **OK**.
3. In the **wxFlexGridSizer** work area, select the **vgap** (vertical gap) and **hgap** (horizontal gap) options. Increase the vertical gap to 20 and the horizontal gap to 10. This will add space between the cells of the column so that it is not so tightly packed.
4. Click the last sibling **wxStaticText** in the GUI. Select the flag option in the sizeritem area of the work area. Select **Edit…** and select **wxALIGN_CENTER_HORIZONTAL** then click **OK**.

Your GUI is now complete, and should look like the next figure. There are many more options available to adjust formatting if you choose to do so. Every component will have a sizeritem area in its work area and settings specific to the type of component. Many components have a style tab in the work area where you can adjust settings such as the font. Once you have finalized your GUI, make sure to click **Save** or **Save As** in the File menu to save the `.xrc` file.

Figure 186: Example GUI



Modify script for an imported GUI

If you want to modify information for an imported GUI, you must modify the script. In the script, there must be four parts: `.xrc` file name, input variables, function, and function call. You can modify different parts depending on your requirements (see next two figures).

Figure 187: Modify Script for import GUI

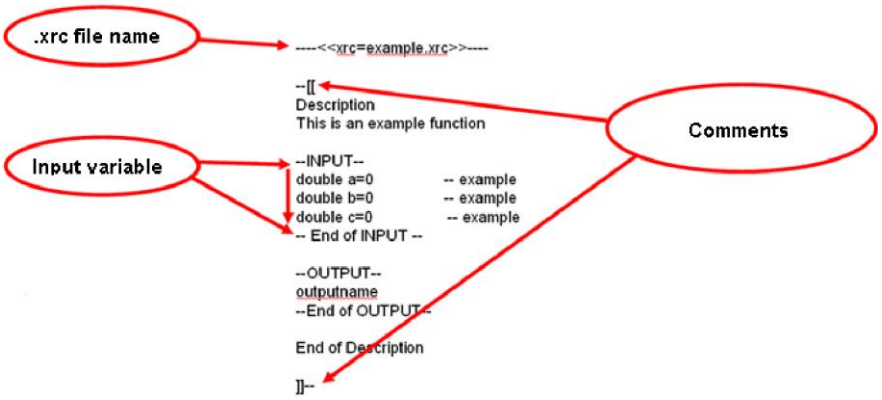
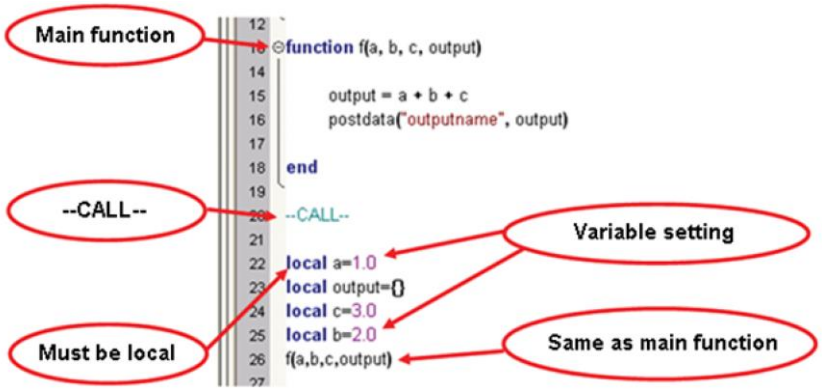


Figure 188: Modify Script for import GUI_2.png

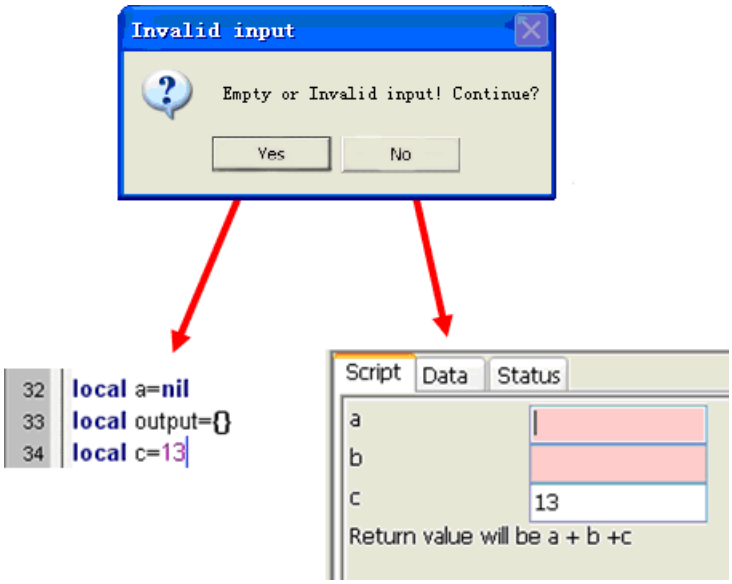


NOTE

In a TSP script, comments are preceded by "- -" on a single line or enclosed in "- - [[" and "]] - -" for a multi-line comment block. In a C++ program, comments are enclosed in "/*" and "*/". Input variable types can be integer, double, string, table, or instID. The function call area is proceeded by the text '--CALL-', and the function name must be the same as the main function's. The following '--CALL --' is auto-generated (see previous figure).

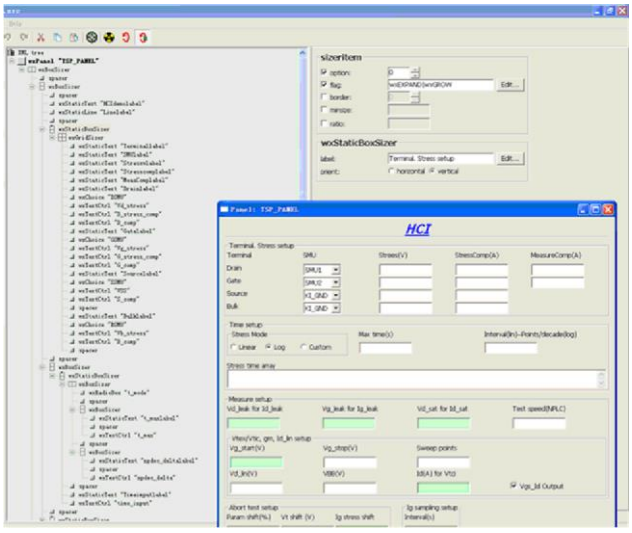
For input variables, you can limit the input value range. For example: double a=0.0 in [-40,40]; double b=0.0 in [-40,]; integer a=0 in [,40].

In the GUI, you can input values of your choice. If an input is left empty, the Invalid input dialog opens when you click the Script tab in the work area (see next figure). Clicking **Yes** sets the variable to "nil" in the script. Clicking **No** changes the invalid or empty input box to a different color.



The next figure shows the XRC GUI of the HCI and the XRC XML tree.

Figure 189: HCI .xrc GUI and its .xml tree



Key points for creating TSP GUI

The XML ID for input components in the GUI must have the same name as the parameters in the script.

The XML ID of the wxPanel must be `TSP_PANEL` (for a PTM XRC GUI, the XML ID must be `PTM_PANEL`).

There must be a `--CALL--` area in the script.

The `--INPUT--` area and the definitions of the input variables should be clear.

Input a range for variables.

There must be a function call. The call function name has to be the same as the main function.

Configure a Python test module (PTM)

A Python language test module (PTM) is a module defined primarily by programming a Python language user module. It can be created by using the script editor in ACS Basic.

ACS Basic supports three types of PTM: script, standard, and `.xrc`. All three types use a work area that contains Setup, Data, and Status tabs:

- **Setup tab:** Allows you to specify a user library and user module and a limited number of test parameters. To configure or reconfigure a PTM, you need to change the settings of the Setup tab, which is the default tab of the PTM work area. The Setup tab differs depending on the type of PTM.
- **Data tab:** Displays the test results in a spreadsheet and allows you to perform data analysis. The Data tab also allows you to create and export graphs of the test and test-analysis results. Graph options include flexible plot-data selection, formatting, annotation, and numerical coordinates display using precision cursors.
- **Status tab:** Monitors the present configuration status of the PTM, reporting its readiness for use and recommending additional preparations, if necessary. A test-ready report for a PTM is the same as for an ITM. To view example Status tab reports for ITMs, refer to "ITM" in the *Automated Characterization Suite (ACS) Fundamental Reference Manual* (document number ACS-914-01).

Configure a PTM script

A PTM script refers to a file or program that can be created using the Python programming language. In ACS Basic, the script module shows a script format with no GUI.

For information about creating a script using the Script Editor, refer to Script editor tutorials (on page 6-13).

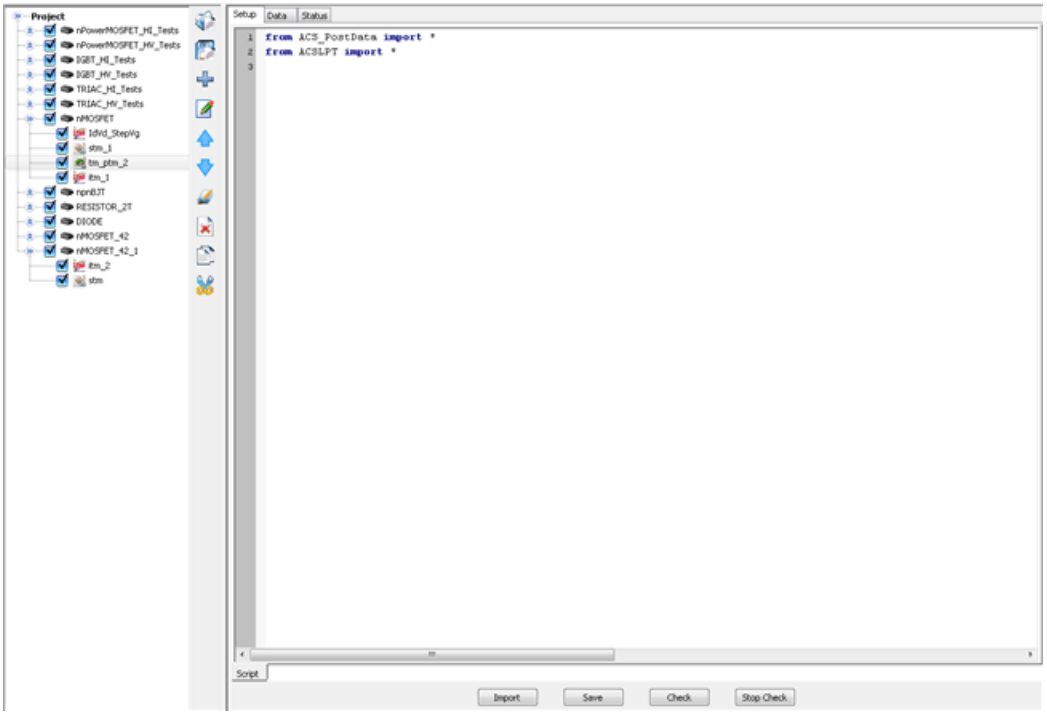
After a new PTM is inserted in a project plan, it must be configured to meet the test requirements. This section describes how to add a script PTM and how to work with Python code.

Insert a new PTM

To insert a new PTM:

1. Select **Add new test**.
2. Select **PTM** in the Specify the Test Type dialog. The PTM Setup tab is the default view. It contains the Python Script work area, which is a blank module and allows you to import one into the user library or into your own library. It also contains buttons at the bottom of the GUI, as shown in the following figure.

Figure 190: A blank PTM



3. To import a PTM, select **Import**. You can import a script, standard, or .xrc PTM. Scripts in the pyLibrary are described in Device library.
4. Verify that the SMU settings match the physical connections. Change the inputs, if necessary.
5. Change the script as needed.

6. Select **Check** to check the Python code for basic errors by checking it in the interpreter (only available in Demo mode). You can stop the check by selecting **Stop Check** (only available in Demo mode).
7. If you need to rename the module, select Rename on the vertical toolbar.
8. Select **Save** to save the PTM.

You can also enter Python program code in the Setup tab.

A Python script

A script is a collection of instrument control commands and programming statements. PTM scripts are written as programming code in Python, in a structure that is framed by commands. Program statements lie within these commands and control script execution, providing facilities such as variables, functions, branching, and loop control.

The table in the next topic explains Python code, displays the lines of code, and corresponding information.

For additional information regarding Python code, refer to the following websites:

- https://www.python.org/downloads/ (python.org/downloads/)
- https://www.activestate.com/products/python/ (activestate.com/products/python/)
- https://www.python.org/doc/ (python.org/doc/)

PTM Script example

Below is an example script that performs a diode measurement.

```
import ACSLPT as lpt
import ACS_PostData as post
def Diode_DynamicZ_I1I2(PSMU, NSMU, I1, I2, RangeV, ComplianceV, nPLC, Holdtime):
    """
    VISIBILITY: NOT_HIDDEN
    TYPE: script
    INPUTLIST
        #Name,  Type,  Default Value,  Min Value, Max Value
        PSMU,enum,'SMU2',('SMU1','SMU2','SMU3','SMU4')
        NSMU,enum,'SMU1',('SMU1','SMU2','SMU3','SMU4','GNDU')
        I1, float,  0.01,  0.1,0.1
        I2, float,  0.02,  0.1,0.1
        RangeV, int,  1,  1,5
        ComplianceV,  float,  20,  200,200
        nPLC,   float,  1.0,   0.01,10
        Holdtime, float,  0.01,  0,100
    END INPUTLIST
```

```
OUTPUTLIST
    #Name,       Type
    error,       int
    DynamicZ,    float
END OUTPUTLIST

DESCRIPTION
    Library: S4200Parlib
    Module Name: Diode_DynamicZ_I1I2
    Instrument:Keithley S4200 SMU.
    DUT: Diode
    Function: Calculates the Dynamic Impedance based on two forward voltage or
two reverse voltage measurements.
                DynamicZ = (v2 - v1) / (I2 - I1)
    Pin Connections: It uses one SMU to force forward current, while the other
terminal is grounded.

    *********
    Input
    *********
    PSMU(enum):     SMU name of P terminal. Valid from SMU1 to SMU8.
    NSMU(enum):     SMU name of N terminal. Valid from SMU1 to SMU8, GNDU.
    I1(float):      P terminal forced current. Valid from -0.1 to 0.1. [A]
    I2(float):      P terminal forced current. Valid from -0.1 to 0.1. [A]
    RangeV(int):          Range of voltage measurement. Valid from 1 to 5.
                            Input             Range
                           ======================
                           1           Full Auto
                           2           Limited Auto 0.2V
                           3           Limited Auto 2V
                           4           Limited Auto 20V
                           5           Limited Auto 200V

    ComplianceV(float):   Compliance of current force. Valid from -200 to 200.
[V]
    nPLC (float):  Number of power line cycles for integration. Valid input
from 0.01 to 10.
    Holdtime (float):     Holdtime before measurement. Valid input from 0 to
100.  [s]

    *********
    Output
    *********
    error(int):
                    0     (OK) normal working condition;
                    -1       Output file open error
                    -10000       (INVAL_INST_ID)An invalid instrument ID was
specified. This generally means that there is
no instrument with the specified ID in your configuration.
                  -10100  (INVAL_PARAM) An invalid parameter was specified.
                  -12021   (KI_COMPLIANCE) Measurement compliance occurred.
                  -12022   (KI_RANGE_COMPLIANCE) Range limit was reached during
measurement.
    DynamicZ(float):             Dynamic impedance of diode.
    END DESCRIPTION
"""

error = 0
```

```
    DynamicZ = 0.0
    lpt.tstse()

    #Some input checking is needed
    if  abs(I1) > 0.1 or DrainIbd == 0:
        error = -10100
        post.ACSPostDataInt("error",error)
        return error, DynamicZ
    if  abs(I2) > 0.1 or DrainIbd == 0:
        error = -10100
        post.ACSPostDataInt("error",error)
        return error, DynamicZ
    if  RangeV < 1 or RangeV > 5:
        error = -10100
        post.ACSPostDataInt("error",error)
        return error, DynamicZ
    if  abs(ComplianceV) > 200:
        error = -10100
        post.ACSPostDataInt("error",error)
        return error, DynamicZ
    if  nPLC < 0.01 or nPLC > 10:
        error = -10100
        post.ACSPostDataInt("error",error)
        return error, DynamicZ
    if  Holdtime < 0 or Holdtime > 100:
        error = -10100
        post.ACSPostDataInt("error",error)
        return error, DynamicZ

    #Map the SMUs with DUT terminals
    PSMUId = lpt.getinstid(PSMU)
    if not PSMUId:
        error = -10000
        post.ACSPostDataInt("error",error)
        return error, DynamicZ
    NSMUId = lpt.getinstid(NSMU)
    if not NSMUId:
        error = -10000
        post.ACSPostDataInt("error",error)
        return error, DynamicZ

    #Set range and compliance
    lpt.limitv(PSMUId, ComplianceV)
    if NSMUId != 4096:
        lpt.limitv(NSMUId, ComplianceV)
    if RangeV == 1:
        lpt.setauto(PSMUId)
    elif RangeV == 2:
        lpt.lorangev(PSMUId, 0.2)
    elif RangeV == 3:
        lpt.lorangev(PSMUId, 2)
    elif RangeV == 4:
        lpt.lorangev(PSMUId, 20)
    elif RangeV == 5:
        lpt.lorangev(PSMUId, 200)
    else:
        lorangev(PSMUId, 2)
```

```
    #Set instrument config
    lpt.setmode(PSMUId, lpt.KI_INTGPLC,nPLC)
    htime = int(Holdtime * 1000)

    #Set stress
    if NSMUId != 4096:
        lpt.forcev(NSMUId, 0)
    lpt.forcei(PSMUId, I1)
    lpt.delay(htime)
    V1 = lpt.intgv(PSMUId)

    lpt.forcei(PSMUId, I2)
    lpt.delay(htime)
    V2 = lpt.intgv(PSMUId)

    #check compliance
    cstatus = lpt.getstatus(PSMUId, KI_COMPLNC)
    if cstatus == 2:
        error = KI_RANGE_COMPLIANCE
        post.ACSPostDataInt("error",error)
        return error, DynamicZ
    if cstatus == 4:
        error = KI_COMPLIANCE
        post.ACSPostDataInt("error",error)
        return error, DynamicZ

    #test complete
    lpt.devclr()
    Idelta = I1 - I2

    if  Idelta == 0:
        Idelta = 1e-37

    DynamicZ = (V1 -V2) / Idelta

    post.ACSPostDataDouble("DynamicZ", DynamicZ)
    post.ACSPostDataDouble("I1", I1)
    post.ACSPostDataDouble("I2", I2)
    post.ACSPostDataDouble("V1", V1)
    post.ACSPostDataDouble("V2", V2)
    post.ACSPostDataInt("error",error)
return error, DynamicZ
```

Python code explanation

Code lines	Element	Description
1 through 2	Import declaration	The keyword `Import` starts the import declaration.
3 through 4	Declaring functions	The keyword `def` starts the function declaration, followed by the function name and the arguments in parentheses. Multiple arguments are separated with commas.
5 through 75	Function's document string (help information)	The function's document string is enclosed within triple quotes. Paragraph rule (Python syntax rule): Python functions have no explicit beginning or end, and no curly braces to mark where the function code starts and stops. The only delimiter is a colon (:) and the indentation of the code itself. Indenting starts a block and not indenting ends it. White space is significant and must be consistent. In this example, the function code (including the `doc string`) is indented four spaces. It does not need to be four spaces; it needs to be consistent. The first line that is not indented is outside the function. Refer to the indentation error examples below this table
77 through 169	Functional code	The functional program code that runs the test.
170 through 176	Post-test data	The post-test data posting. This section sends variable data to the test module data tab.

The following example shows indentation errors:

```
def perm(l):                          # error: first line indented
for i in range(len(l)):                 # error: not indented
    s = l[:i] + l[i+1:]
        p = perm(l[:i] + l[i+1:])   # error: unexpected indent
        for x in p:
              r.append(l[i:i+1] + x)
            return r                     # error: inconsistent indent
```

The correct code would look like this:

```
def perm(l):
   for i in range(len(l)):
          s = l[:i] + l[i+1:]
          p = perm(l[:i] + l[i+1:])
          for x in p:
                r.append(l[i:i+1] + x)
   return r
```

Post data function in PTM

The following functions are used for a PTM to post data to the data tab and plot of the PTM:

- ACSPostDataInt(data_name, data_val)
- ACSPostDataDouble(data_name, data_val)
- ACSPostArrayInt(data_name, data_array, data_num)
- ACSPostArrayDouble(data_name, data_array, data_num)

NOTE

To use these functions in ACS Basic, you must import ACS_PostData at the beginning of the Python source code.

Configure a standard PTM

A standard PTM refers to a script with a standard PTM GUI that can be created by using the Script Editor tool of ACS Basic. A standard GUI allows you to enter organized parameters and values before running a test.

For more information on how to create a standard PTM using Script Editor, you can refer to **Tutorial 2: Create and run a new PTM (on page 6-23).**

After a new standard PTM is inserted to a project plan, it must be configured to meet the test requirements. This section describes the configuration of standard PTMs.

Configure a standard PTM

Import and open a standard PTM. The PTM Setup tab GUI opens (see next two figures).

Figure 191: Import a standard PTM

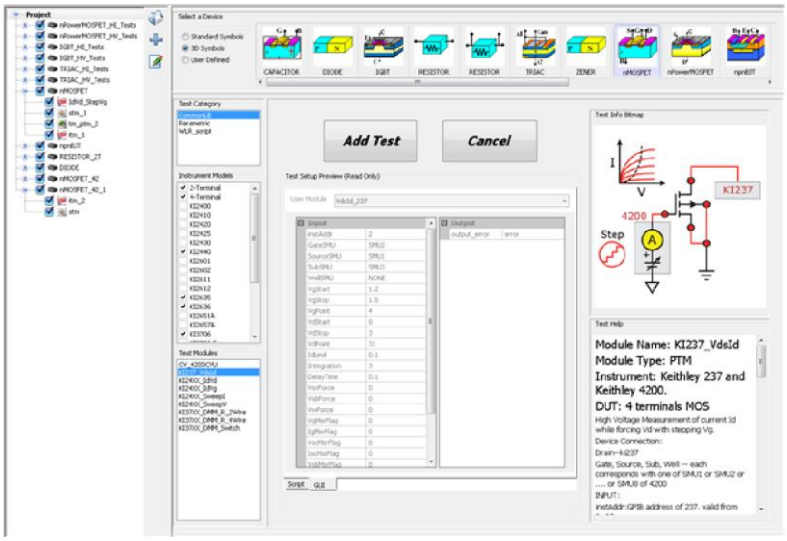
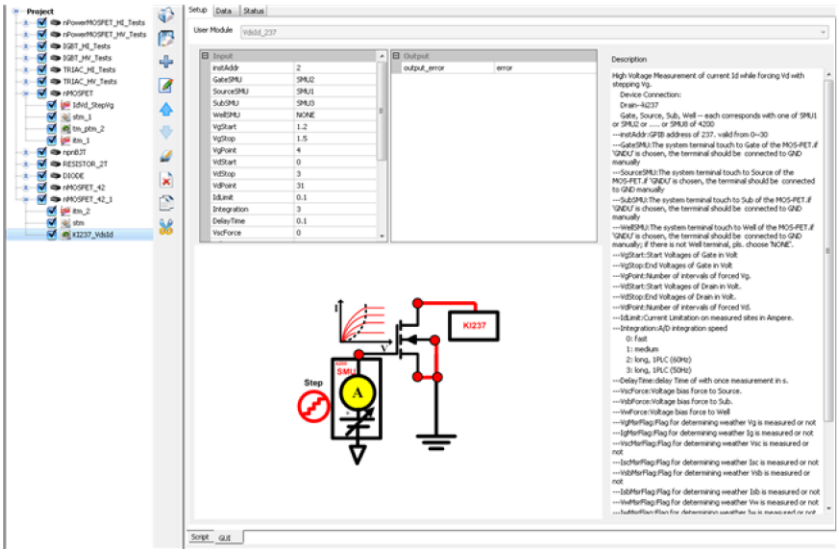


Figure 192: Open a standard PTM



As shown in the previous figure, there are three areas in the Setup tab of a standard PTM: Input, Output, and Description.

Set the Input parameters. Refer to the description in the right panel. If the input parameter cell is an edit box, enter a value. If the input parameter cell is a list, select the item.

Set the output parameter names. Enter a string and it will appear on the Data tab as an output. Save the configuration.

Configure a XRC PTM

The XRCed is a tool for creating a GUI. The PTM can include a `.xrc` file with a test module that will be the user interface for setting input values.

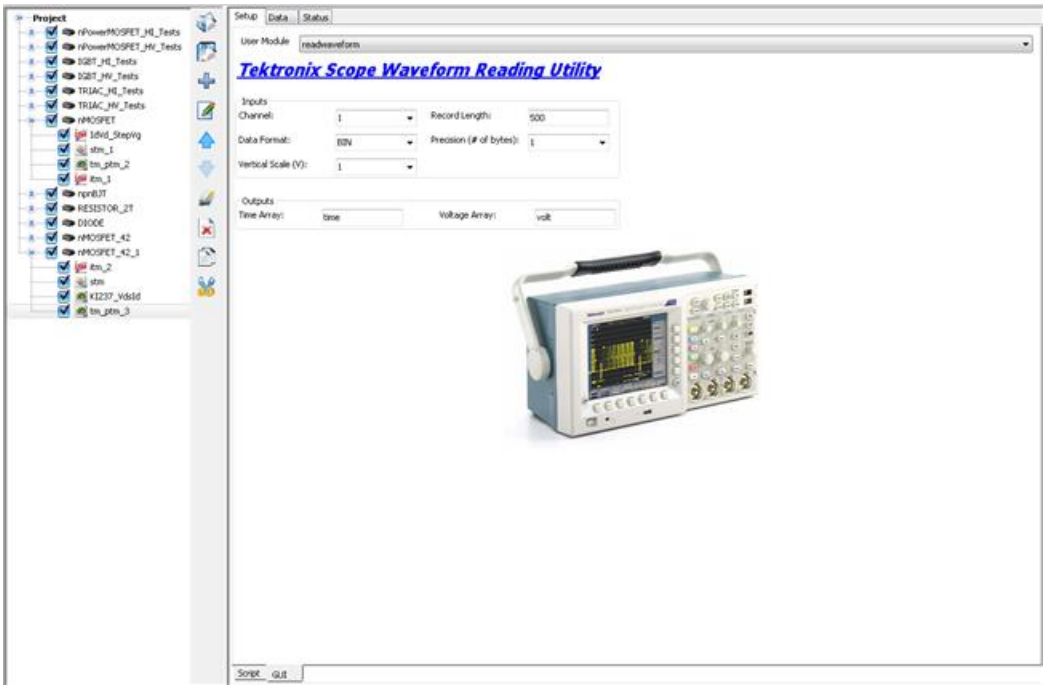
If the imported PTM file has a corresponding `.xrc` GUI file, ACS Basic automatically loads and opens the GUI. You can set parameters on the GUI rather than editing parameters on the Setup tab.

For more information on how you can generate the `.xrc` file, refer to Create a XRC GUI file (on page 5-40).

To configure a PTM with XRC:

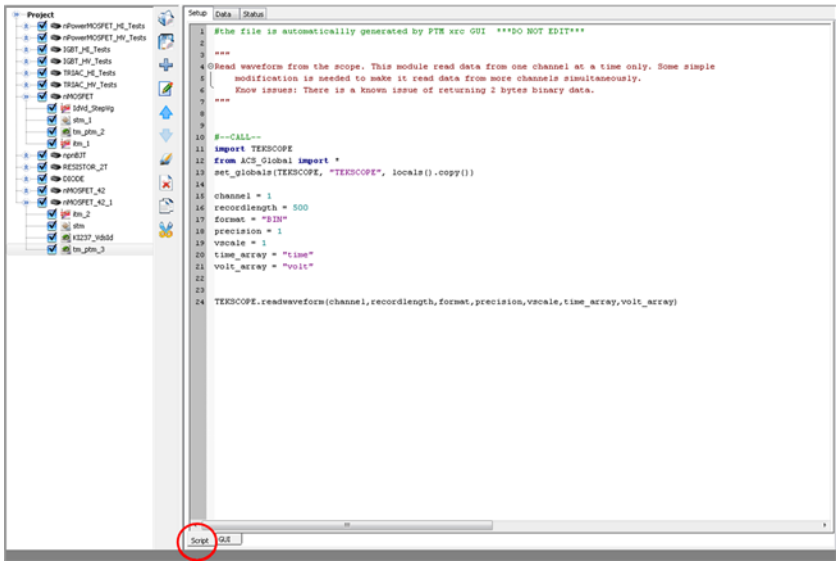
1. After the XRC PTM is imported, click the test name in the configuration navigator. The module opens and the GUI is displayed by default (see next figure). The script code is in the Setup tab.

Figure 193: Import a XRC PTM



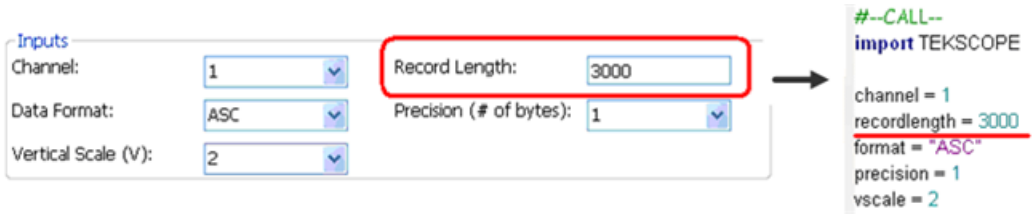
2. Set the Input parameters. Refer to "Common capacitance-voltage (CV) library" in the *Automated Characterization Suite* (*ACS) Basic Edition Libraries Reference Manual* (document number ACSBASIC-908-01) for more information.
3. Save the configuration.
4. The inputs in the GUI are related to the function call in the Setup tab. To open, click the Script tab in the Setup tab (see next figure).

Figure 194: The Setup tab of a XRC PTM



After the correct input is entered in the GUI, the function call area of the Setup tab for this test module is automatically updated (see next figure).

Figure 195: The function call automatic update



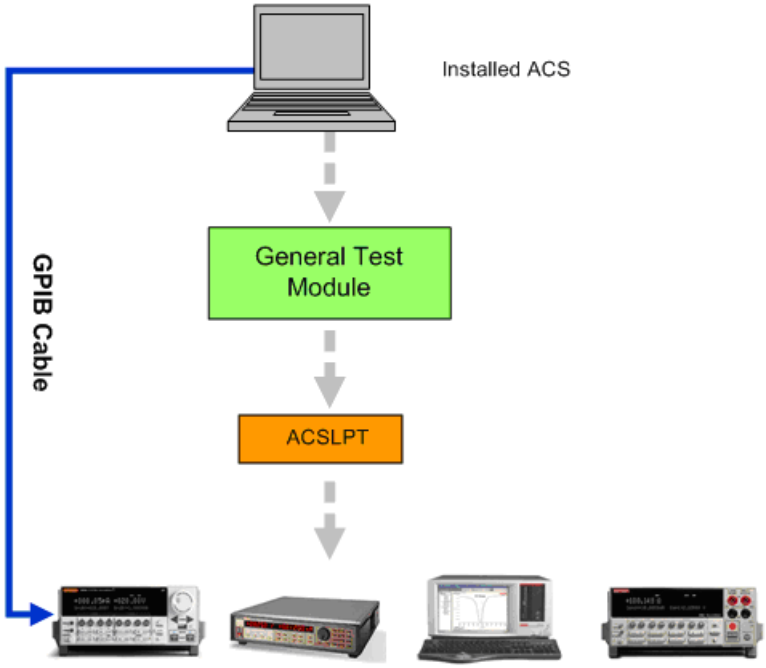
Configure a general-purpose PTM

The general-purpose PTM is used to perform measurements with SMUs from different Keithley product lines. Its GUI provides an interface similar to the standard interactive test modules (ITM) for the Series 2600 and 2600B, therefore, no programming is required.

You can see in the next figure that the general purpose test module can be used with any combination of the following SMUs: Series 2400, Series 2600, Series 2600B, Model 4200-SCS, and Model 237.

For more information on how to configure different GPIB instruments, refer to the ACS Basic hardware configuration (on page 3-1) section.

Figure 196: General purpose module with multiple SMUs



The general purpose module in ACS Basic can provide a total solution for component characterization tests with high power requirements. Because the general purpose module will work with a mix of several different types of SMUs, it can also be used in other applications, such as solar cell and LED testing, among others.

Due to hardware platform differences in the module-compatible SMUs, the general purpose module interface only supports the following forcing functions: BiasV/I, SweepV/I, and StepV/I. Unlike the Series 2600 and 2600B ITM, the general purpose module does not support log sweeps, dual sweeps, or list sweeps, and has limited autoranging.

The external trigger feature in the Common Settings is only used when using multiple Model 237 SMUs with a Model 2361 Trigger Controller. To use this feature, you must configure the Model 2361 as GPI1 (general purpose instrument 1) in the ACS Basic hardware configuration.

The other Common Settings are similar to those in the timing dialog of the standard Series 2600 and 2600B ITM timing settings. The timing and speed of these tests is limited if you combine different generations of hardware. For the fastest test times, you should always use the Series 2600B SourceMeters in the standard tests associated with those instruments.

NOTE

When using only Series 2600/2600B SourceMeter instruments for a particular test, it is recommended that you use either an ITM or STM for optimum SMU performance.

Open the general-purpose PTM

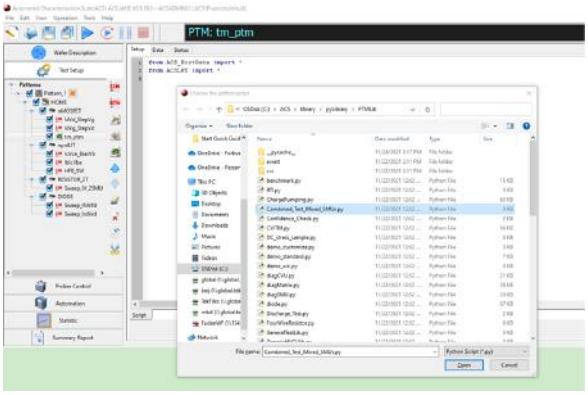
Configure the hardware system

Before you configure a general purpose test module in the system, you need to configure and check the hardware first. For more information on how to configure different GPIB, ethernet, and TSP instruments through ACS Basic, refer to ACS Basic hardware configuration (on page 3-1).

To add a new general purpose module:

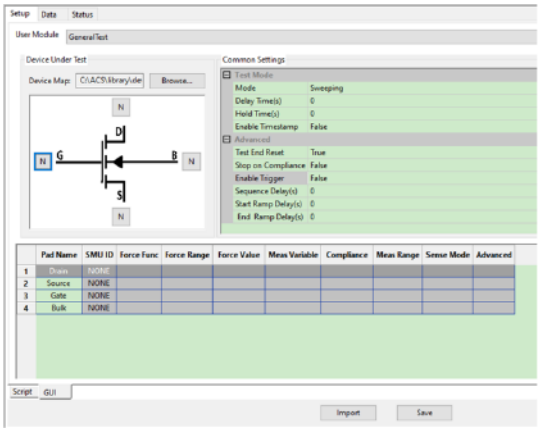
1. Select the **Add new test** icon.
2. Select **PTM** in the Specify the Test dialog box.
3. Open the PTM.
4. Select the **Import** function.
5. In the dialog box that opens, select the `Combined_Test_Mixed_SMUs.py` script (see next figure):

Figure 197: Import the General PTM



The general-purpose PTM GUI opens on the ACS Basic GUI (see next figure).

Figure 198: General purpose PTM GUI



General-purpose PTM configuration

After inserting a general-purpose PTM into your project, you must configure it to meet your test requirements.

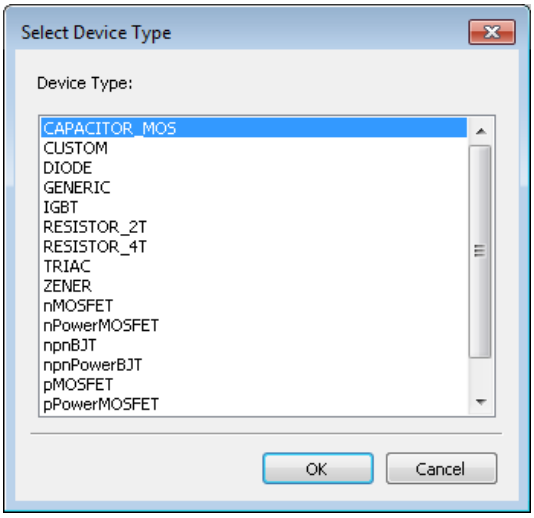
NOTE

The device you select in the general-purpose PTM GUI is independent of the device in the configuration navigator. You must select a device in the module GUI to configure the general-purpose PTM. For consistency with device definitions, you should select the same device for the general purpose module that the PTM exists under in the configuration navigator. By default, the imported PTM opens with the same device as the one in the configuration navigator where the PTM is created.

To configure the general-purpose PTM:

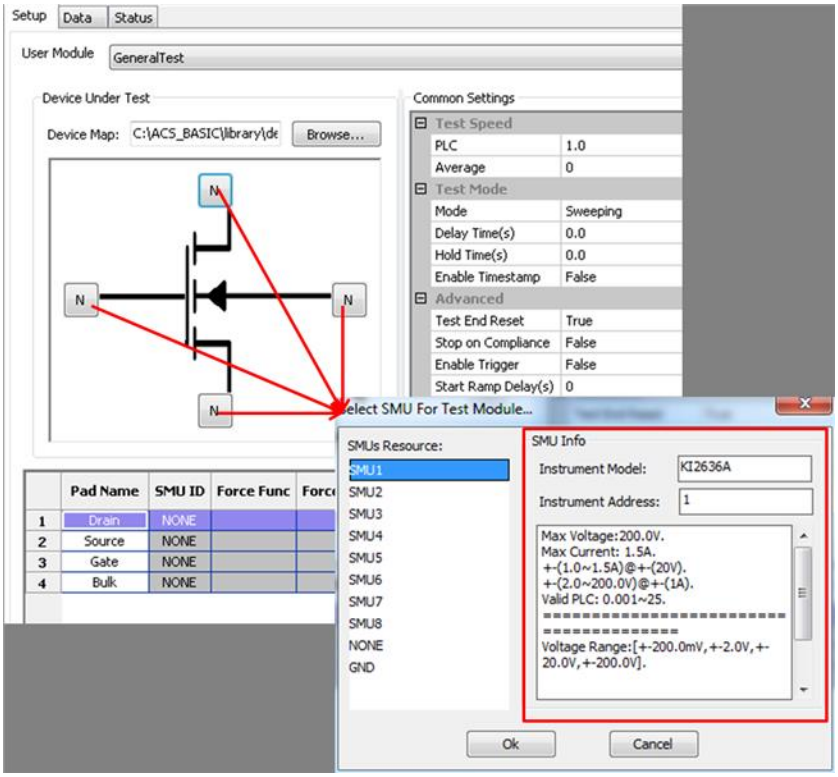
1. Select **Browse**. The Select Device Type dialog displays Device Types (see next figure).

Figure 199: Select Device type dialog



2. Specify which SMU is connected to each device terminal. Select a device terminal to open the Select SMU For Test Module dialog.
3. Select the appropriate SMU. The device terminal reflects your selection (see next figure).

Figure 200: Map SMU and device terminal



NOTE

Limitations exist in ACS Basic when assigning SMUs to device terminals to prevent instrument damage due to incompatible instruments.

If a connection to the device is deemed unsafe by the software, you cannot run the test and you see an error message in the Help area of the GUI. Here are some items to consider when configuring your hardware:

- Non-A 26xx SMUs cannot be mixed with Model 265xA instruments.
- A Model 2657A SMU cannot be mixed with Models 2651A, 2601B, or 2602B.
- A Model 2651A SMU cannot be mixed with low-current SMUs.
- For two-terminal devices, a Model 2651A cannot be mixed low-current SMUs.
- For three-terminal and four-terminal devices, if a Model 2651A SMU is connected to the device's collector, or drain terminal, then the emitter or source terminal, must be connected to ground or another Model 2651A SMU. If a Model 2651A SMU is connected to the base, or gate, then the other device terminals must be connected to ground or another Model 2651A.

Configure the Force and Measure parameters

The Force and Measure settings are associated with an instrument object that is assigned to a device terminal. These settings are used to configure the forcing function and measurements implemented by the instrument.

The following figure indicates all the parameters that need to be configured for forcing and measuring.

Figure 201: Force and Measure parameters

	Pad Name	SMU ID	Force Func	Force Range	Force Value	Meas Variable	Compliance	Meas Range	Sense Mode	Advanced
1	Drain	SMU1	Bias V	auto	0	None	0.01		Local	...
2	Source	SMU3	Bias V	auto	0	None	0.01		Local	...
3	Gate	SMU2	Bias V	auto	0	None	0.01		Local	...
4	Bulk	SMU4	Bias V	auto	0	None	0.01		Local	...

Force Func: You can select the following force functions:

- Voltage bias (Bias V)
- Current bias (Bias I)
- Open
- Voltage sweep (Sweep V)
- Current sweep (Sweep I)
- Voltage step (Step V)
- Current step (Step I)

Sense Mode: There are two choices:

- Local: Two-wire sensing
- Remote: Four-wire sensing

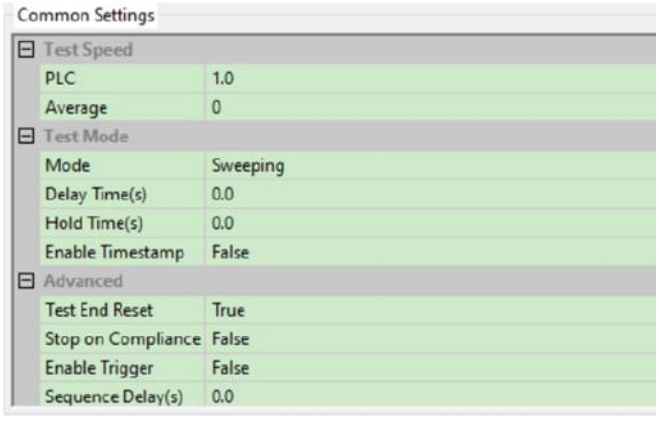
NOTE

If the SMU is a Model 42xx-SMU, the Sense Mode parameter is grayed out and not available to change.

Common settings

The Common Settings are used primarily to configure the timing settings for the General-Purpose PTM. There are three areas in the Common Settings: Test Speed, Test Mode, and Advanced (see the following figure).

Figure 202: General purpose test module Common Settings



Section 6

Script Editor Tool

In this section:

Introduction

The Keithley Script Editor is a tool used to create and manage user libraries.

A user library is a collection of one or more user modules. User modules are Python programming language subroutines (or functions) or the TSP script language. User libraries are created to control instrumentation, analyze data, or perform any other system automation task programmatically. Once a user library has been created using the script editor, its user modules run using ACS Basic software.

The script editor manages user libraries in a structured manner. It also provides a graphical user interface (GUI) that helps you to effectively enter code, compile a user module, and build a user library. It provides user-library management features, such as open, delete, copy module and library, check, edit and browse, save and export functions. You can create your own user libraries.

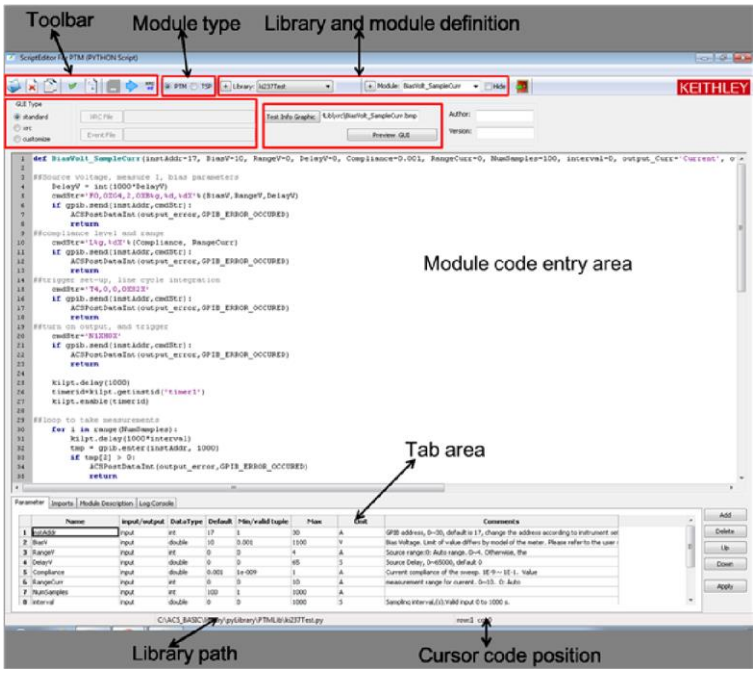
To run script editor user modules in ACS Basic, you must import or create a test module as a TSP or PTM and connect it to the user module. Once the user module is connected to the test module, the following occurs each time ACS Basic runs the STM or PTM:

- Dynamically loads the user module and the appropriate user library.
- Passes the user-module parameters (stored in the STM or PTM) to the user module.
- Generated data by the user module is returned to the STM or PTM for interactive analysis.

Script Editor GUI

The Script Editor GUI provides all the icons, controls, and user-entry areas needed to create, edit, view, and build a user library and to create, edit, view, and compile a user module (see next figure).

Figure 203: Script Editor GUI



The module identification area

The module type, library, and module name are next to the toolbar. The components of this area are used as follows (see previous figure).

- **Module Type**: Displays the type of the active (presently open) user module. There are two types of supported modules: STM and PTM. For more information on STMs and PTMs, refer to Configure a script test module (on page 5-33) (STM) and Configure a Python test module (on page 5-51) (PTM).
- **Library Name**: Displays the name of the active (presently open) user library. Use the Import Library icon to open a specific user library. Use the Delete icon to delete a library. Use the Copy icon to copy a library. Use the Add a new library icon to add a new library (refer to The toolbar (on page 6-9) for more information).

NOTE

When naming a user module, remember to conform to the case-sensitive Python or TSP programming language naming conventions. Do not duplicate names of existing user modules or user libraries.

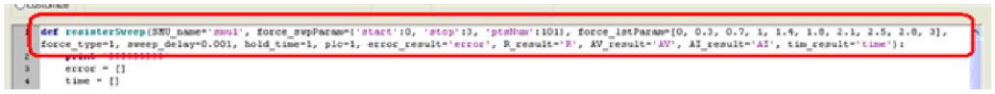
- **Module Name**: Displays the name of the active (presently open) user test module.
- **Hide (Library and Module)**: Hides the Library and Module information so that it is not displayed in the GUI. To display the Library and Module information in the GUI, clear the Hide option.
- **GUI type**: Displays the GUI type of the active (presently open) user module. There are three types of GUIs available: Standard, XRC, Custom. For more information about a STM with a standard GUI, refer to Configure a script test module (on page 5-33) (STM). For more information about a STM with a XRC GUI, refer to Create a XRC GUI file (on page 5-40).
- **Version info**: Displays creator of the user module and the version information.

The user module code-entry area

The user module code-entry area is where you enter, edit, or view the user module code (see previous figure). The scroll bars to the right and below the module code-entry area let you move through the code.

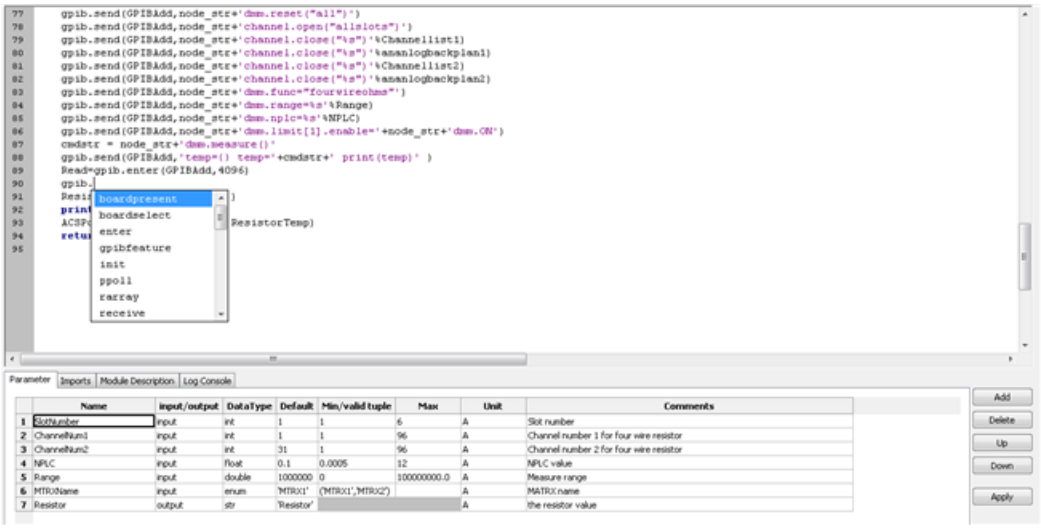
The beginning of the module code contains the function definition area. This area contains the language function prototype for the user module that reflects the parameters that are specified in the Parameter tab area (see next figures).

Figure 204: Function definition area



The parameters that the module depends on to operate are configured in the Parameter tab area. After you add the parameters (by selecting the Add function) and entering the configurations, select **Apply** so the updates to the module take affect immediately (see next figure).

Figure 205: Script editor enter Python code



NOTE

Do not enter the following Python code items in the module code-entry area (script editor enters these at special locations based on information in other places in the script editor GUI):

- Import and global data
- The function prototype

Do not enter the following TSP code items in the module code-entry area (script editor enters these at special locations based on information in other places in the script editor GUI):

- The function call
- The function prototype

To control internal or external instrumentation, use functions from the Linear Parametric Test Library (LPTLib). Refer to the *Automated Characterization Suite* (*ACS) Basic Edition Libraries Reference Manual* (document number ACSBASIC-908-01) for more information.

The tab areas

There are four tabs:

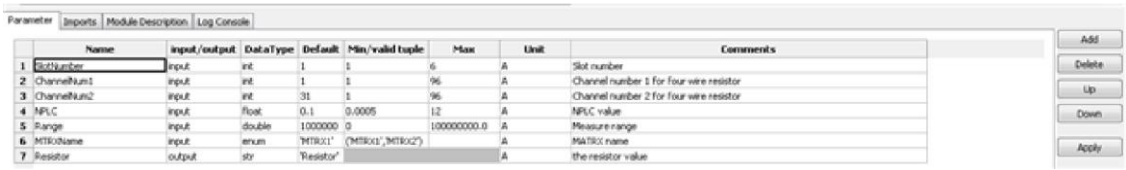
- Parameter
- Imports (for PTMs only)
- Module Description
- Log Console

Parameter tab

The Parameter tab area is used to configure and display the following for each parameter that is included in the user module (see next figure).

- Name
- Input/output
- Data Type
- Default
- Min/valid tuple
- Max
- Unit
- Comments

Figure 206: Parameter tab area with examples

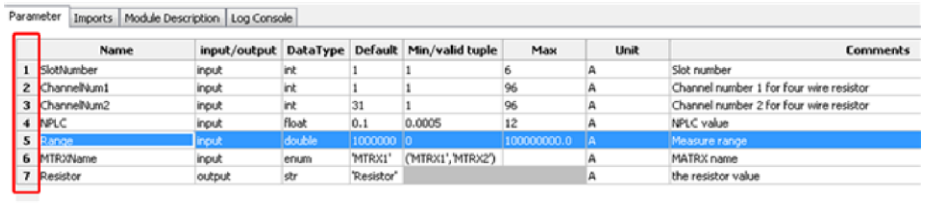


	Name	input/output	DataType	Default	Min/valid tuple	Max	Unit	Comments
1	SlotNumber	input	int	1	1	6	A	Slot number
2	ChannelNum1	input	int	1	1	96	A	Channel number 1 for four wire resistor
3	ChannelNum2	input	int	31	1	96	A	Channel number 2 for four wire resistor
4	NPLC	input	float	0.1	0.0005	12	A	NPLC value
5	Range	input	double	1000000	0	100000000.0	A	Measure range
6	MTRXName	input	enum	'MTRX1'	('MTRX1','MTRX2')		A	MATRX name
7	Resistor	output	str	'Resistor'			A	the resistor value

Add, delete, or apply a parameter:

- Click **Add** and enter the information as indicated in the field descriptions.
- Click the parameter name or any of the adjacent fields and click **Delete**.
- Select the row you want to move by clicking the row number, then click **Up** or **Down** (see next figure).
- Click **Apply** to apply changes made in the Parameter tab area.

Figure 207: Select a row of parameters to move



	Name	input/output	DataType	Default	Min/valid tuple	Max	Unit	Comments
1	SlotNumber	input	int	1	1	6	A	Slot number
2	ChannelNum1	input	int	1	1	96	A	Channel number 1 for four wire resistor
3	ChannelNum2	input	int	31	1	96	A	Channel number 2 for four wire resistor
4	NPLC	input	float	0.1	0.0005	12	A	NPLC value
5	Range	input	double	1000000	0	100000000.0	A	Measure range
6	MTRXName	input	enum	'MTRX1'	('MTRX1','MTRX2')		A	MATRX name
7	Resistor	output	str	'Resistor'			A	the resistor value

- **Name**: Identifies the parameters that are passed to the user module.
- **input/output**: Determines the direction of the data.

NOTE

The string (str) data type can only be used as output.

- **Data Type**: Specifies the parameter data type:

  For PTM and TSP:

  - **enum** - enumerable

NOTE

When the data type is enumerable (enum) data, the value of Min/valid tuple field in the parameter tab is a valid tuple. You must set the Min/valid tuple first so that the Default device can be selected in the drop-down list (see next figure).

  - **str** - string
  - **float** - single precision, floating data point
  - **double** - double precision
  - **int** - integer
  - **list** - list
  - **dict** - dictionary

  For TSP only:

  - **table** - table type

- **Default**: Specifies the default value for the input parameter
- **Min/valid tuple**: Specifies the minimum recommended value for the input parameter (except list, dict, table, and str type). When the user module is used in ACS Basic, configuration of the PTM/STM with a parameter value smaller than the minimum value causes ACS Basic to display an out-of-range message.
- **Max**: Specifies the maximum recommended value for the input parameter (except list, dict, table, and str type). When the user module is used in ACS Basic, configuration of the PTM/STM with a parameter value larger than the maximum value causes ACS Basic to display an out-of-range message.
- **Unit**: Area to enter a label for the default, minimum, and maximum values.
- **Comment**: Location where you can save important information about the input or output.

Imports tab

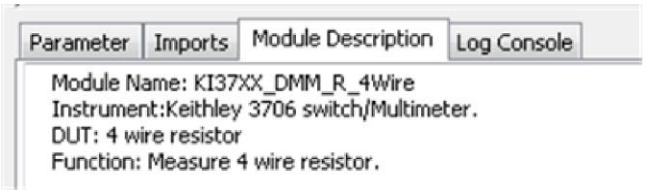
The Imports tab area only displays for a PTM. It lists the imported files used in the Python user module. This area can be used to add import statements to the active (presently open) user module (see next figure). The code displayed in the figure (from ACS_PostData import*) is essential to the ACS Basic post data commands. It allows the PTM to return data to the module's Data tab and should be included in every test module. This tab can also be used to import the global statement.

Figure 208: Imports tab

Module Description tab

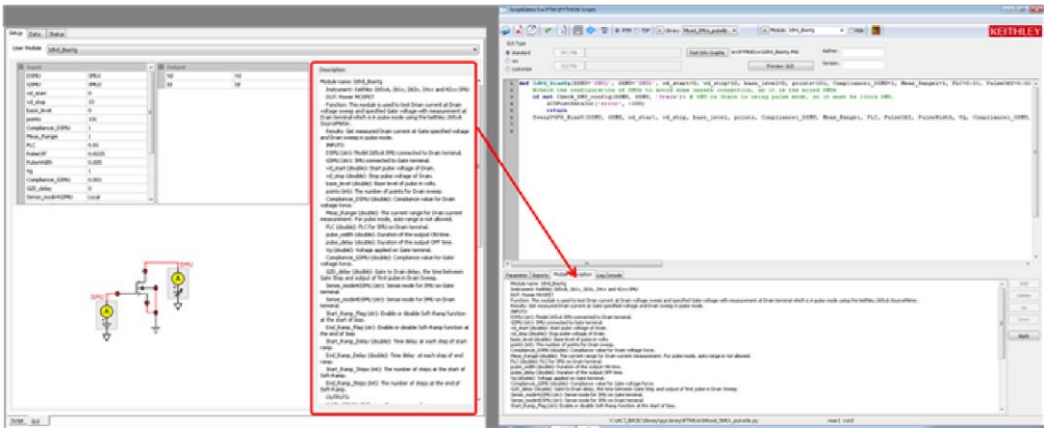
The Module Description tab area allows you to enter descriptive information for the user module (see next figure).

Figure 209: Module Description tab



Information entered in this area documents the module for the ACS Basic user and is used to create the user library help (see next figure).

Figure 210: Description tab in ACS Basic



NOTE

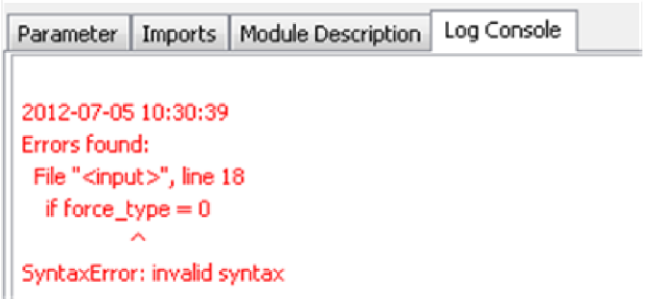
Do not use comment designators (−−) in the Description tab area. When the user-module code is compiled, Script Editor also evaluates the text in this area. Code comment designators in the Description tab area can be misinterpreted, which will cause errors.

Log Console tab

The Log Console tab area displays any error syntax messages that are generated during a code check operation.

When you select the **Check** icon on the toolbar, a message is displayed in the Log Console tab. When an error is displayed in the Log Console tab area, Script Editor shows the line of code where the error occurred or the next line, depending on how the compiler caught the error (see next figure). This facilitates error corrections. If no errors are found, the Build tab area displays a message stating that fact.

Figure 211: Log Console tab



Status bar

The Status bar at the bottom of the Script Editor GUI displays the module directory path and the cursor location in the code-entry area. For example, if the cursor is in row 42, column 4, in the code-entry area, the status bar indicates that information (see next figure).

Figure 212: Status bar

The toolbar

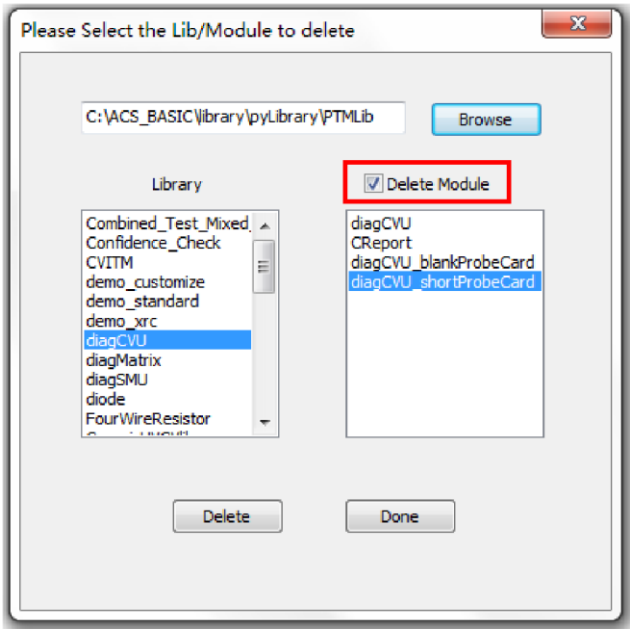
The icons on the toolbar (see next figure) are on the top of the Script Editor GUI.

Figure 213: Script Editor Toolbar

The Toolbar icon descriptions:

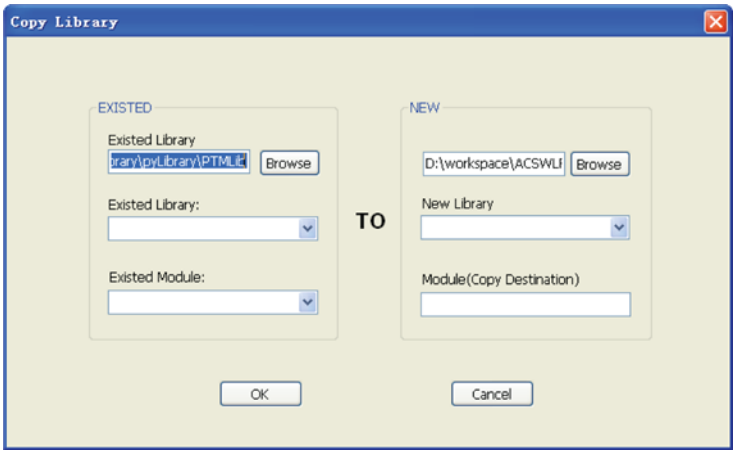
- **Delete:** Deletes an existing user library or a user module from the indicated user library. Selecting **Delete** displays the “Please Select the Lib/Module to delete” dialog, as shown in the following figure. To delete a selected library and all its contents, select **Delete**. To delete a particular test module from the selected library, select **Delete Module**, then select **Delete**.

Figure 214: Choose a module to delete



- **Copy Library and Module:** Creates a copy of any existing library and module. Selecting Copy Module and Library displays the Copy Library dialog, as shown in the following figure. Select the user library and module to copy. Enter the new library and module name in the New section. You must enter a unique user-module name. Select **OK**. The existing library or module is copied (see next figure).

Figure 215: Copy library and module

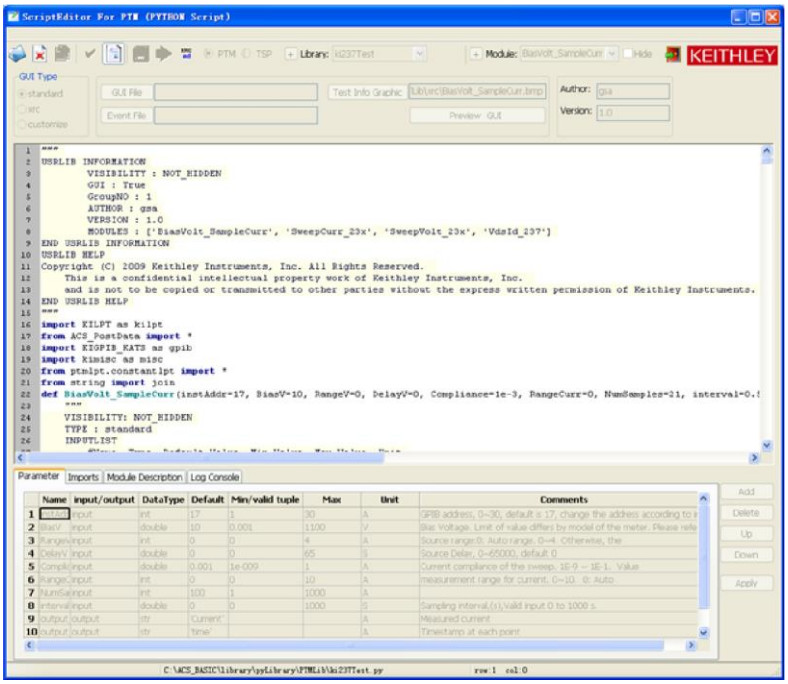


NOTE

If you enter an existing library name to the New section, the module will be copied from one existing library into another existing library. If you enter a new library name, the module is copied into the new library. The module name in the existing area can be empty (all the modules in the source library are copied to the destination library).

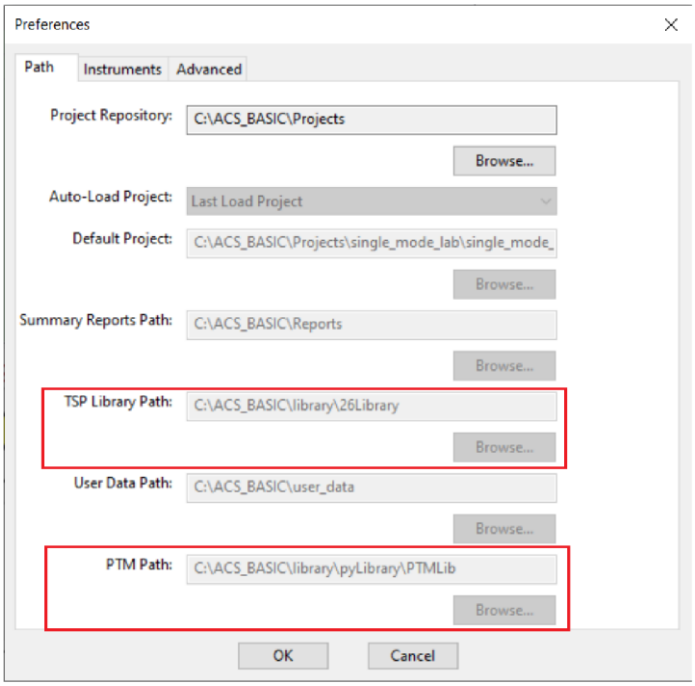
- **Check**: Select this icon to check the source files of the open user module for syntax errors.
- **Browse or Edit the whole file**: Select this icon to see the code-entry area that contains the whole file with the tab content disabled (see next figure). When you import a TSP or Python source code (which is not generated by the Script Editor), it may start this function automatically if the file format does not match the Script Editor file format. You can use this feature to edit tests that were not created with structured inputs, outputs, and descriptions.

Figure 216: Browse the entire file



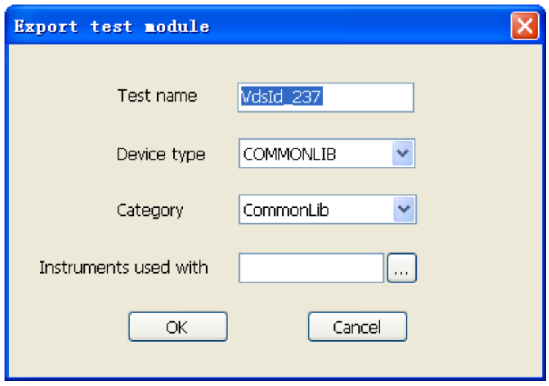
- **Save**: Click this icon to save the opened user module source code. The default directory path is `\\ACS_BASIC\library\26Library\TSPLib` for a TSP script and `\\ACS_BASIC\library\pyLibrary\PTMLib` for a PTM. The path is assigned in the Preference setting dialog (see next figure).

Figure 217: Choose the TSP and PTM path



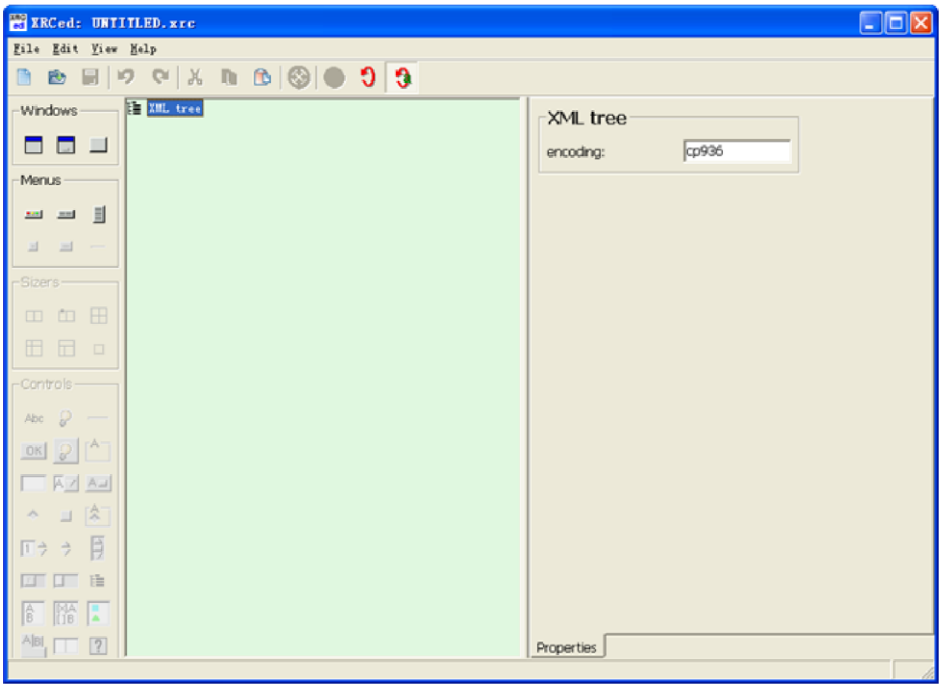
Export: Click this icon to open the Export test module dialog. Enter the name in the Test name edit box; select the Device type and Category for the Test. Click **OK**. The test module is exported and saved. The saved module is now available to load in ACS Basic.

Figure 218: Export test module dialog



- **XRC Builder**: Enables you to build GUIs for custom test modules written in either Python or TSP. The TSP scripts work with the Series 2600, Series 2600B, and Model 3706A instruments (see next figure). The Python scripts communicate over GPIB to any of these instruments and to any general purpose hardware a user might want to control. Change the GUI Type to XRC to take advantage of these GUI design features. For more information, refer to Create a XRC GUI file (on page 5-40).

Figure 219: XRC GUI builder



Script Editor tutorials

This section includes two tutorials. One is for creating and running a new TSP script and the other is for a PTM script. Each tutorial provides systematic instructions for accomplishing common tasks with the Script Editor.

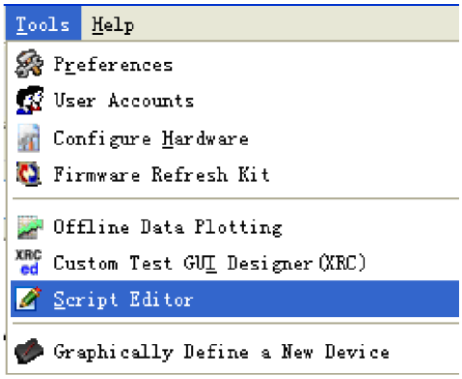
Tutorial 1: Create and run a new TSP script

Script Editor is a tool that facilitates the development of user libraries. Each user library is comprised of one or more user modules, and each user module is created using test script processor (TSP) or Python test module (PTM). For more information about TSP, refer to Multiple TSP instruments (on page 3-23) and for PTM refer to Configure a PTM script (on page 5-51).

This hands-on tutorial is provided to show you how to create a new user library and new user module.

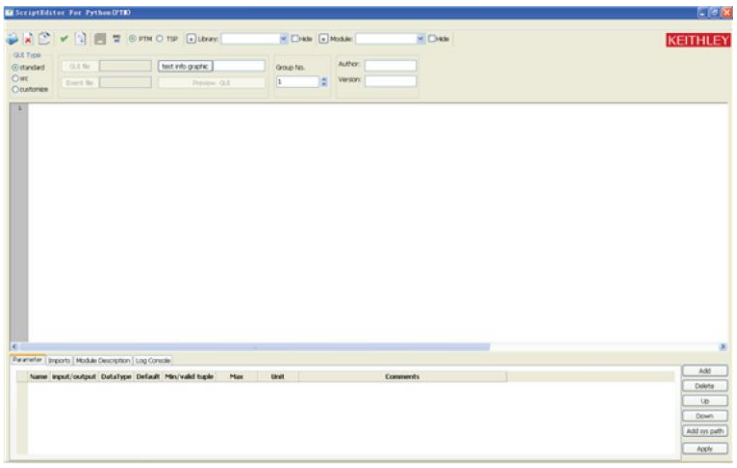
1. Open the Script Editor by selecting **Script Editor** in the Tools Menu (see next figure) or by clicking the Script editor icon on the vertical toolbar in the ACS Basic GUI.

Figure 220: Select Script Editor in the Tools menu



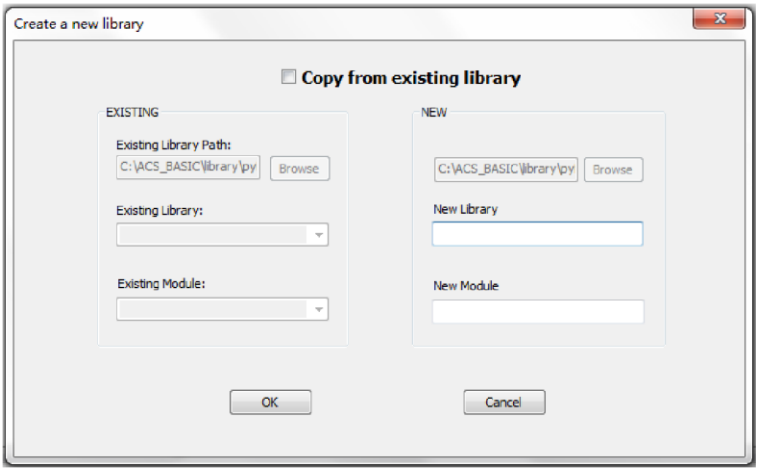
2. Select TSP in the module type area when Script Editor opens (see next figure).

Figure 221: Blank Script Editor GUI



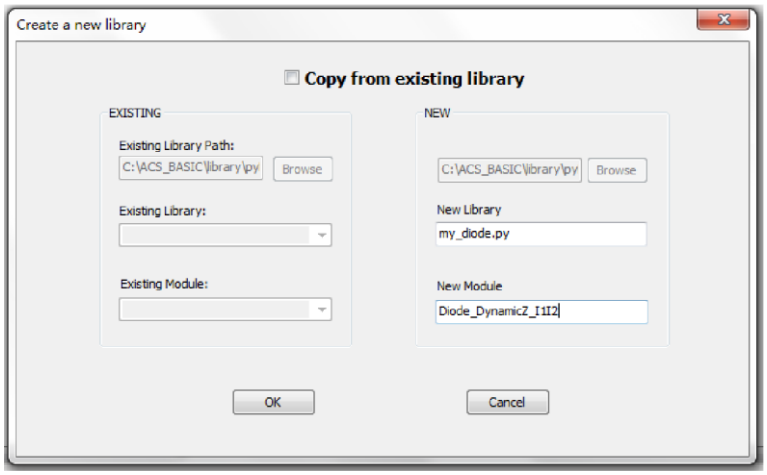
3. Create a new user library by clicking the Add new library icon. The Create a new library dialog opens (see next figure).

Figure 222: Create a new library dialog



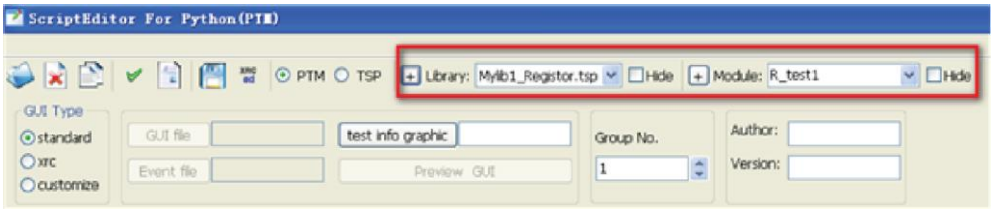
4. Enter the new user library name in the New Library edit box. The library name must have `.py` or `.tsp` extension. For this tutorial, enter `mylib1_Resistor.tsp`.
5. Enter `R_test1` as the New Module name (see next figure).

Figure 223: Enter new library and module name



6. Select **OK**. The new library and module names will appear in the script editor GUI (see next figure).

Figure 224: New Library name in Script Editor GUI



7. Select a GUI for TSP:
   a. Select a GUI type. If selecting the XRC GUI type, you can select the **GUI File** and the **Event File**. For more information about a STM with a standard GUI, refer to Configure a script test module (STM) (on page 5-33). For more information about a STM with a XRC GUI, refer to Create a XRC GUI file (on page 5-40).
   b. Select a graphic for the TSP script by clicking on the **Test Info Graphic** button.
   c. Enter the **Author** and **Version** information.
8. Enter the required parameters for the code as follows:
   a. Select the **Parameter** tab (if the Parameter tab area is not displayed).
   b. Select the **Add** button.
   c. Enter the first parameter name (or accept the default). For the **mytsp1** user module, replace the default name with **RSMU1**.
   d. Specify the I/O of the first parameter in the input/output cell.

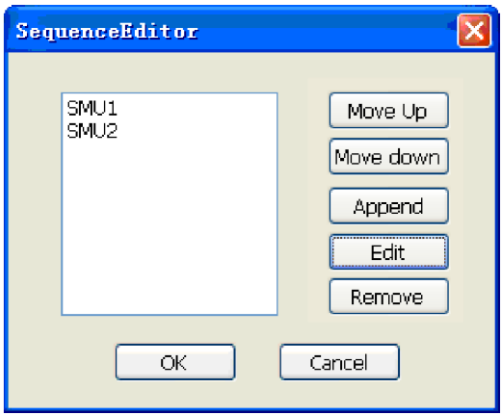
NOTE

For an output parameter, only the string data type is acceptable.

e. Enter the data type for the first parameter. For this tutorial, enum was selected for the first parameter:
   - **Int**: Integer number
   - **str**: String; can only be enclosed in single quotes ()
   - **double**: 64-bit float point number
   - **float**: 32-bit float point number
   - **enum**: Supplies an enumerator that can list the components of a composite moniker
   - **table**: Tables are created using table constructors, which are defined using curly brackets {}

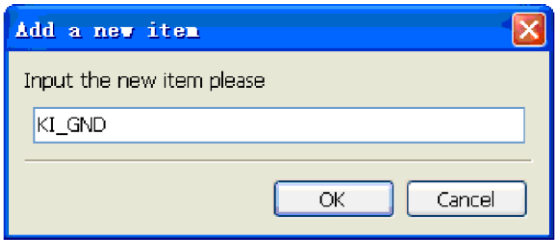
f. Enter the default, minimum, maximum, and unit values for the parameter:
   - If the data type is enum, you should set the Min/valid tuple value first: Right-click the **Min** area, and the Sequence Editor dialog opens (see next figure). Select the **Append** button.

Figure 225: Sequence Editor for setting the enumerable parameter



   - Input the items in the Add a new item dialog (see next figure). Select **OK**. The enumerable data is set.

Figure 226: Add a new item for setting the enumerable parameter



- For table type (often used in TSP), add the items in the table.
- For the `mylib1` user module, enter `SMU1`, `SMU2`, and `KI_GND` for `RSMU1` enum data.

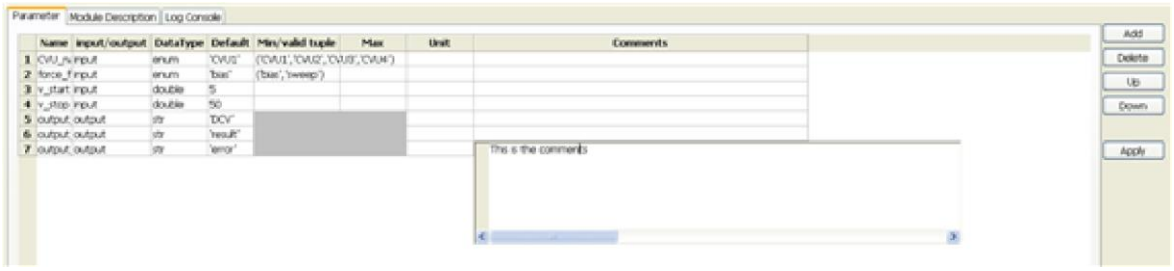
g. After the Min/valid tuple value is set, the default value can be selected. In the next figure, **SMU1** for **RSMU1** was selected.

h. Enter comments for the parameter, if needed. Select the comment area to display an edit box. Enter the parameter description (see next figure).

NOTE

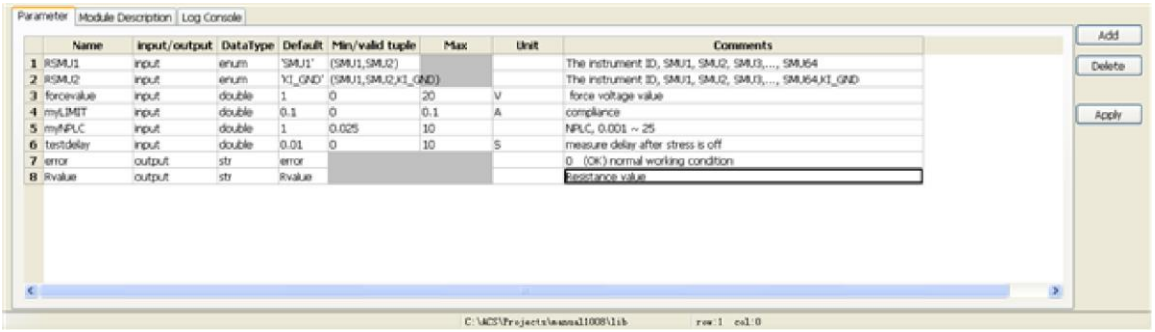
When the user module code is compiled, Script Editor evaluates the text in the Module Description area. Therefore, do not use comment designators (--) in the Module Description tab area because the designators can be misinterpreted which can cause errors.

Figure 227: Enter comments



i. Repeat steps b through g for the remaining user module input and output parameters. For the `R_test1` module, add eight parameters (see next figure).

Figure 228: Parameter tab for TSP



9. Select **Apply**. The parameters are displayed in the Parameter tab (see next figure).

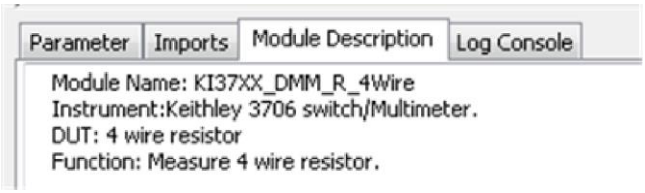
Figure 229: Script Editor with parameters

10. Enter the module description:
    a. Select the **Module Description** tab (see next figure).
    b. Enter any comments that are needed to document the user module.

NOTE

The comments that you include with the code in Script Editor cannot be viewed by other users of Script Editor.

Figure 230: Module Description tab

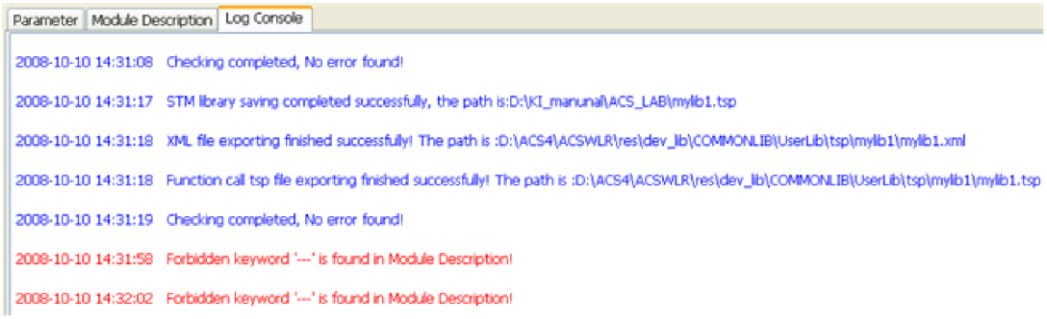


11. Enter the new TSP code to the module-code entry area. Refer to the *Automated Characterization Suite* (*ACS) Basic Edition Libraries Reference Manual* (document number ACSBASIC-908-01) for a complete list of supported I/O and SMU commands.
12. For the **R_test1** user module, enter the following code:

```
local v_value = {}
local i_value = {}
local error code = {}
local R = {}
setmode(RSMU1, KI_INTGPLC, myNPLC)
limiti(RSMU1, myLIMIT)
forcev(RSMU1, forcevalue)

if RSMU2 ~= KI_GND then
   forcev(RSMU2, 0)
end --if

delay(testdelay)
intgv(RSMU1, v_value)
intgi(RSMU1, i_value)

R[1] = v_value[1]/i_value[1]
posttable(Rvalue, R)
table.insert(error_code, 0)
posttable(error, error_code)

devint()
end
```

13. Compile the user module. For TSP, the Script Editor checks the user module code as follows:
    a. In the Toolbar, Select the **Check** icon and the following occurs:
        - The user-module source-code file is compiled.
        - The Log console message tab indicates the compilation progress and, if problems are encountered, displays error messages. For example, when you check the `mytsp1` user module, the Log console gives you instant feedback (see next figure).

Figure 231: The log console message tab

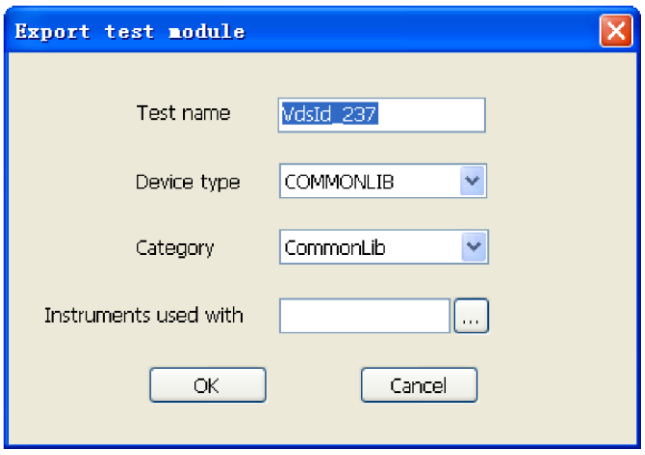


14. Make changes and continue checks until no errors exist.
15. Save the user module:
    a. Select **Save**. The source code information is saved to the directory `C:\ACS_BASIC\library\26Library\TSPLib`.
    b. For example, in the next figure, after clicking the Save icon, the following files are saved:
        - The source code file (`.tsp`). The default directory is `C:\ACS_BASIC\library\26Library\TSPLib`
        - The `.bmp` file and `.xrc` file. The default directory is `C:\ACS_BASIC\library\26Library\TSPLib\xrc`
        - The events file. The default directory is `C:\ACS_BASIC\library\26Library\TSPLib\event`

Figure 232: File save in the Log Console tab

16. Export the user module to the ACS Basic device library:
    a. Select the **Export** icon, and the Export test module dialog opens (see next figure).

Figure 233: Export test module dialog



b. Enter a **Test name** in the edit box.

c. Select a **Device type** using the drop-down arrow.

d. Select a **Category** using the drop-down arrow.

e. Select the **ellipsis** so that you can choose an "Instrument used with" to export your test module for the library.

f. Select **OK**.

The exported test module parameter files (`.xml`, `.bmp`, `.tsp`, `.csv`, and help file) are saved in the directory `C:\ACS_BASIC\library\devLibrary\Device_name\Category_name\language\test_module_name`.

The exported information displays in the Log Console tab (see next figure).

Figure 234: Export test module Log Console tab

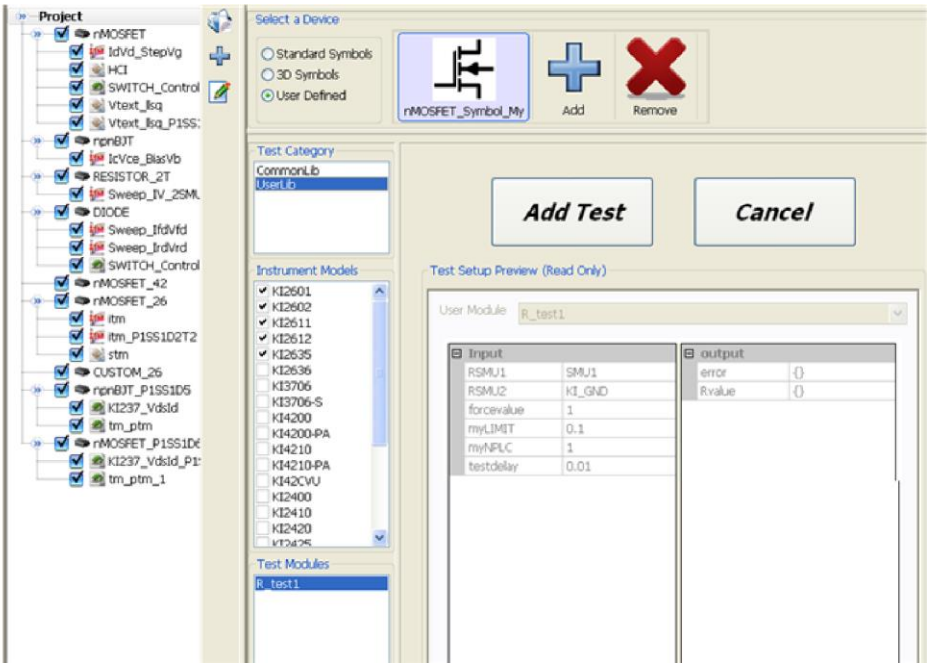
The default directory path is `C:\ACS_BASIC\library\devLibrary\RESISTOR_2T\UserLib\tsp\R_test`. The new standard TSP script is created and saved in the default directory. You can open and run this module in ACS Basic.

Run the STM

To run the STM:

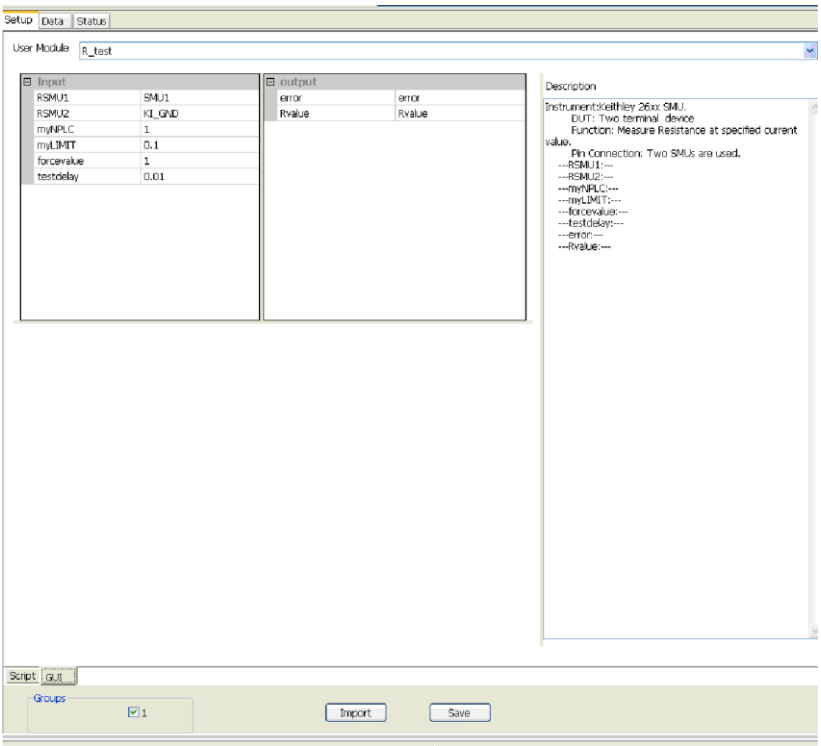
1. Click the **Return to Library View** icon on the ACS Basic toolbar. If ACS Basic is in Multitest mode, you must build a configuration navigator in order to add the test. For more information on how to build the configuration navigator in Multitest mode, refer to Build a project plan (on page 4-12).
2. Select the device where the user module `R_test1` is saved. In the next figure, the device Resistor_2T is selected.
3. Select **UserLib** in the Test Category. The user module library opens in the Test modules area. Select the user module library that you want to test. In the next figure, `mylib1_Resistor` is selected.

Figure 235: Open the UserLib module



4. Click the **Open Test** button (see next figure). Accept the default parameters for this example.

Figure 236: UserLib module opening



5. Save the STM and the project by clicking the **Save All** button.
6. Run the STM by clicking the Run icon.
7. If the user module that you created generates data, check the execution results in the STM Data worksheet. To view the Data worksheet, click the **Data** tab.

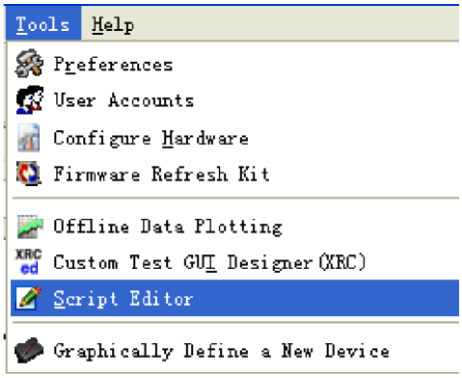
Tutorial 2: Create and run a new PTM

This hands-on tutorial is provided to show you how to create a new PTM. For more information on PTMs, refer to Configure a PTM script (on page 5-51).

To create a new PTM:

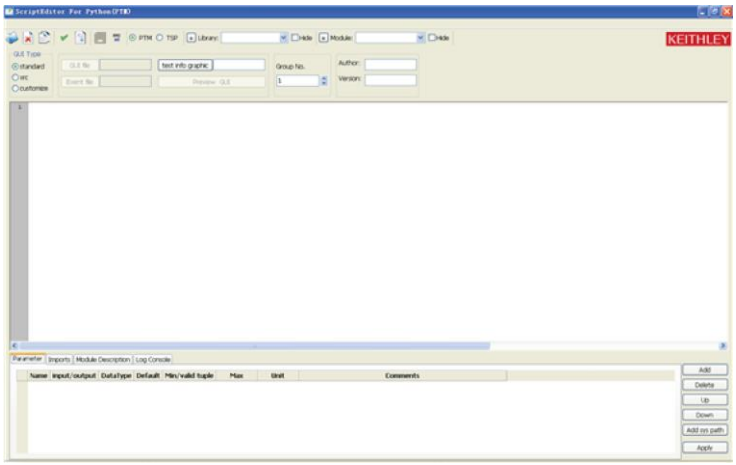
1. From the Tools Menu, select **Script Editor** (see next figure) or by selecting the Script editor icon on the vertical toolbar in the ACS Basic GUI.

Figure 237: Select Script Editor in the Tools menu



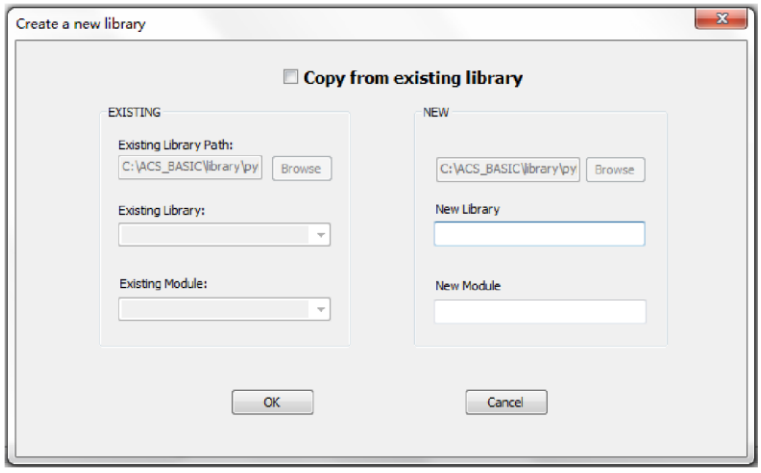
2. Select PTM in the module type area when the Script Editor opens (see next figure).

Figure 238: Blank Script Editor GUI



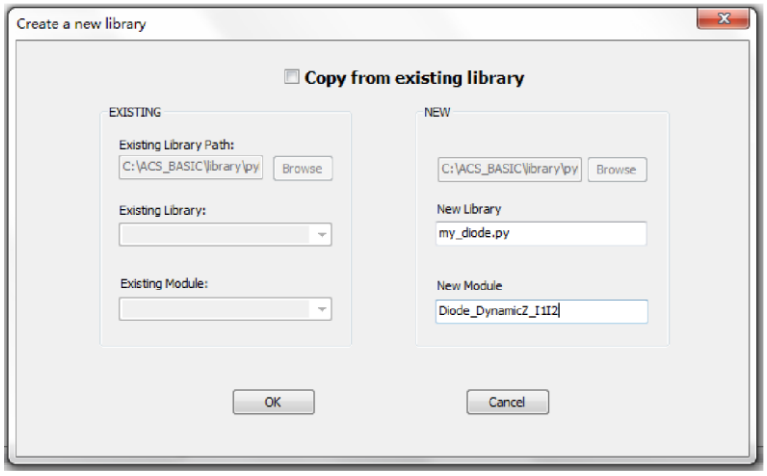
3. Create a new user library by clicking the Add a new library icon. The Create a new library dialog opens (see next figure).

Figure 239: Create a new library dialog



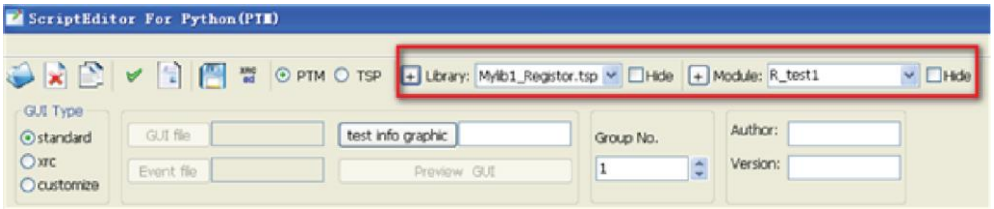
4. Enter the new user library name in the New Library edit box. The library name must have a `.py` or `.tsp` extension. For this tutorial, enter `my_diode.py`.
5. Enter `Diode_DynamicZ_I1I2` as the New Module name (see next figure).

Figure 240: Enter new library and module name



6. Select **OK**. The new library and module names appear in the Script Editor (see next figure).

Figure 241: New Library name in Script Editor GUI



Select a GUI for the PTM:

1. Select the **standard** GUI type. If selecting the XRC GUI type, you can select the **GUI File** and the **Event File** buttons. For more information on the STM with XRC GUI, refer to Configure a script test module (STM (on page 5-33)).
2. Select a graphic for the PTM script by clicking **test info graphic**.
3. Enter the **Author** and **Version** information.

Enter the required parameters for the code:

1. Select the **Parameter** tab (if the Parameter tab area is not displayed).
2. Select **Add**.
3. Enter the first parameter name. For the **mytsp1** user module, replace the default name with **RSMU1**.
4. Specify the I/O of the first parameter in the input/output cell.

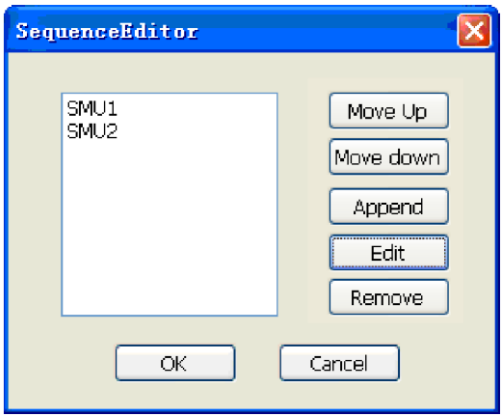
NOTE

The string data type is only valid for output parameters.

5. Enter the data type for the first parameter. For this tutorial, enum was selected for the first parameter. The options are:
   - **Int**: Integer number
   - **str**: String; can only be enclosed in single quotes ()
   - **double**: 64 bit float point number
   - **float**: 32 bit float point number
   - **enum**: Supplies an enumerator that can list the components of a composite moniker
   - **list**: Can be written as a list of comma-separated values (`.csv`); it is not necessary to have all the same type and are defined using square brackets [ ].
   - **dict**: Defines one-to-one relationships between keys and values; first, you create a new dictionary with two elements and assign it to the variable; each element is a key-value pair, and the whole set of elements is enclosed in curly brackets { }.

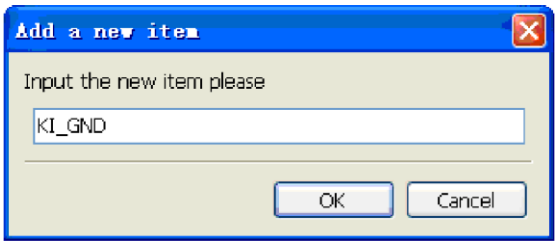
6. Since the data type is enum, set the minimum/valid tuple value first. Select the **Min** area to open the Sequence Editor dialog opens (see next figure). Select **Append**.

Figure 242: Sequence Editor for setting the enumerable parameter



7. Input the items in the Add a new item dialog (see next figure). Select **OK**. The enumerable data is set.

Figure 243: Add a new item for setting the enumerable parameter

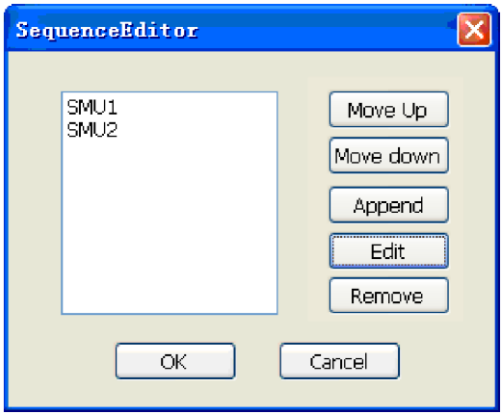


8. Enter the default, minimum, maximum, and unit values for the parameter.

NOTE

If the data type is a list, set the default value as follows: Select the default value (the Sequence Editor dialog opens). Add the item and then Select **OK**. The value appears in the default box with the format [a,b,c].

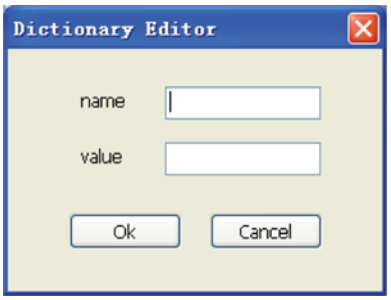
Figure 244: Sequence Editor for setting the enumerable parameter



NOTE

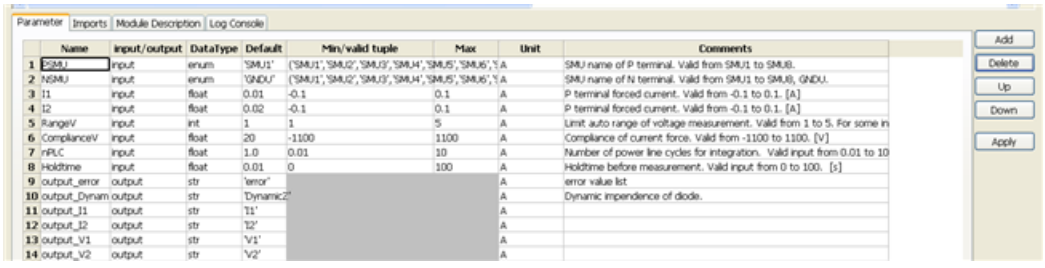
If the data type is dict type (dictionary type, only used in PTM): Select the default value and the Sequence Editor opens (see next figure). Select the Append function and the Dictionary Editor dialog opens (see next figure). Input the name and value in the Sequence Editor. The value is formatted as {a:1,b:2,c:3}.

Figure 245: Dictionary Editor dialog



The next figure shows the PTM parameters.

Figure 246: Enter parameters for PTM



	Name	input/output	DataType	Default	Min/valid tuple	Max	Unit	Comments
1	PSMU	input	enum	'SMU1'	('SMU1','SMU2','SMU3','SMU4','SMU5','SMU6','N		A	SMU name of P terminal. Valid from SMU1 to SMU8.
2	NSMU	input	enum	'GNDU'	('SMU1','SMU2','SMU3','SMU4','SMU5','SMU6','N		A	SMU name of N terminal. Valid from SMU1 to SMU8, GNDU.
3	I1	input	float	0.01	-0.1	0.1	A	P terminal forced current. Valid from -0.1 to 0.1. [A]
4	I2	input	float	0.02	-0.1	0.1	A	P terminal forced current. Valid from -0.1 to 0.1. [A]
5	RangeV	input	int	1	1	5	A	Limit auto range of voltage measurement. Valid from 1 to 5. For some in
6	ComplianceV	input	float	20	-1100	1100	A	Compliance of current force. Valid from -1100 to 1100. [V]
7	nPLC	input	float	1.0	0.01	10	A	Number of power line cycles for integration. Valid input from 0.01 to 10
8	Holdtime	input	float	0.01	0	100	A	Holdtime before measurement. Valid input from 0 to 100. [s]
9	output_error	output	str	'error'			A	error value list
10	output_Dynam	output	str	'Dynamic2'			A	Dynamic impendence of diode.
11	output_I1	output	str	'I1'			A	
12	output_I2	output	str	'I2'			A	
13	output_V1	output	str	'V1'			A	
14	output_V2	output	str	'V2'			A	

If needed, import other libraries to use in the current library:

1. Select the **Imports** tab. The Imports tab area opens (see next two figures).

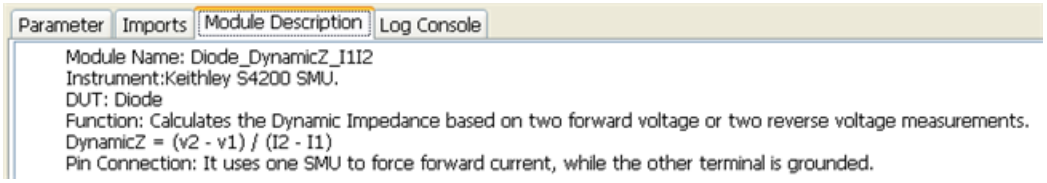
Figure 247: Imports tab

2. Enter any additional import files that are needed for the user module.

Enter a Module Description:

1. Select the **Module Description** tab.
2. Enter needed text to document the user module. A Script Editor user will not see the comments that you include with the code (see next figure).

Figure 248: Module Description tab area DynamicZ



NOTE

When the user module code is compiled, Script Editor evaluates the text in the Module Description area. Therefore, do not use comment designators (---) in the Module Description tab area because the designators can be misinterpreted which can cause errors.

3. Select the **Apply** button.
4. Enter the new PTM code to the module-code entry area. Refer to the *Automated Characterization Suite* (*ACS) Basic Edition Libraries Reference Manual* (document number ACSBASIC-908-01) for the supported I/O and SMU commands.
5. For the `DynamicZ` user module, enter the following code.

```
def Diode_DynamicZ_I1I2(PSMU='SMU1', NSMU='GNDU', I1=0.01, I2=0.02, RangeV=1,
ComplianceV=20, nPLC=1.0, Holdtime=0.01, output_error='error',
output_DynamicZ='DynamicZ', output_I1='I1', output_I2='I2', output_V1='V1',
output_V2='V2'):
    Get4200HWCtrl()
    error = [ ]
    DynamicZ = 0.0
    #tstsel()
    #Some input checking is needed
    if  abs(I1) > 0.1:
          error.append(INVAL_PARAM)
          ACSPostArrayInt(output_error, error, len(error))
          return INVAL_PARAM
    if  abs(I2) > 0.1:
          error.append(INVAL_PARAM)
          ACSPostArrayInt(output_error, error, len(error))
          return INVAL_PARAM
    if  RangeV < 1 or RangeV > 5:
          error.append(INVAL_PARAM)
          ACSPostArrayInt(output_error, error, len(error))
          return INVAL_PARAM
    if  abs(ComplianceV) > 1100:
          error.append(INVAL_PARAM)
          ACSPostArrayInt(output_error, error, len(error))
          return INVAL_PARAM
    if  nPLC < 0.01 or nPLC > 10:
          error.append(INVAL_PARAM)
          ACSPostArrayInt(output_error, error, len(error))
          return INVAL_PARAM
    if  Holdtime < 0 or Holdtime > 100:
          error.append(INVAL_PARAM)
          ACSPostArrayInt(output_error, error, len(error))
          return INVAL_PARAM
    #Map the SMUs with DUT terminals
    PSMUId = getinstid(PSMU)
    if not PSMUId:
          error.append(INVAL_INST_ID)
          ACSPostArrayInt(output_error, error, len(error))
          return INVAL_INST_ID
    NSMUId = getinstid(NSMU)
    if not NSMUId:
          error.append(INVAL_INST_ID)
          ACSPostArrayInt(output_error, error, len(error))
          return INVAL_INST_ID
    #Set range and compliance
    limitv(PSMU, ComplianceV)
    if NSMU != "GNDU":
    #if NSMUId != 4096:
          limitv(NSMU, ComplianceV)
    if RangeV == 1:
          setauto(PSMU)
    elif RangeV == 2:
          lorangev(PSMU, 0.2)
    elif RangeV == 3:
          lorangev(PSMU, 2)
    elif RangeV == 4:
          lorangev(PSMU, 20)
```

```
elif RangeV == 5:
      lorangev(PSMU, 200)
else:
      lorangev(PSMU, 2)
#Set instrument config
setmode(PSMU, KI_INTGPLC, nPLC)
htime = int(Holdtime * 1000)
#Set stress
if NSMU != "GNDU":
#if NSMUId != 4096:
      forcev(NSMU, 0)
forcei(PSMU, I1)
delay(htime)
V1 = intgv(PSMU)
forcei(PSMU, I2)
delay(htime)
V2 = intgv(PSMU)
#check compliance
cstatus = getstatus(PSMU, KI_COMPLNC)
if cstatus == 2:
      error.append(KI_RANGE_COMPLIANCE)
      ACSPostArrayInt(output_error, error, len(error))
      return error[len(error)-1]
if cstatus == 4:
      error.append(KI_COMPLIANCE)
      ACSPostArrayInt(output_error, error, len(error))
      return error[len(error)-1]
#test complete
devint()
Idelta = I1 - I2
if  Idelta == 0:
      Idelta = 1e-37
DynamicZ = (V1 -V2) / Idelta
error.append(0)
ACSPostDataDouble(output_DynamicZ, DynamicZ)
ACSPostDataDouble(output_I1, I1)
ACSPostDataDouble(output_I2, I2)
ACSPostDataDouble(output_V1, V1)
ACSPostDataDouble(output_V2, V2)
ACSPostArrayInt(output_error, error, len(error))
return  DynamicZ
```

Complete user module configuration:

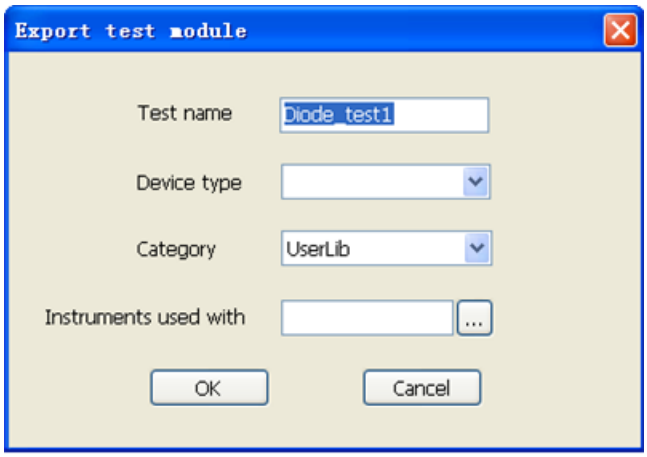
1. Compile the user module.
2. Save the user module. Select the **Save** icon. The source code information is saved to the directory `C:\ACS_Basic\library\pyLibrary\PTMLib`. For example, in the next figure, after clicking **Save**, the following files are saved:
   - The source code file (`.py` file). The default directory: `C:\ACS_Basic\library\pyLibrary\PTMLib`
   - The `.bmp` file. The default directory: `C:\ACS_Basic\library\pyLibrary\PTMLib\xrc`
   - The `event` file. The default directory: `C:\ACS_Basic\library\pyLibrary\PTMLib\event`

Figure 249: File save in the Log Console PTM

Export the user module:

1. Select the **Export** icon, and the Export test module dialog opens (see next figure).

Figure 250: Export test module PTM



2. Enter a **Test name** in the edit box.
3. Select a **Device type** from the list.
4. Select a **Category** from the list.
5. Select the **ellipsis** so that you can choose an "Instrument used with" to export your test module for the library.
6. Select **OK**.

The exported test module parameter files (`.xml`, `.bmp`, `.tsp`, `.csv`, and help file) are saved in the following directory: `C:\ACS_BASIC\library\devLibrary\Device_name\Category_name\language\test_module_name`.

The exported information displays in the Log Console tab (see next figure).

Figure 251: Export PTM information

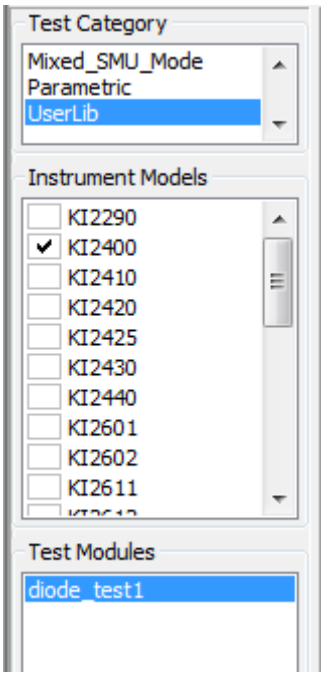
The default directory path is:
`C:\ACS_BASIC\library\devLibrary\RESISTOR_2T\UserLib\ptm\Diode_test1.`
The new standard PTM script is created and saved in the default directory. You can open and run this module in ACS Basic.

Run the PTM

To run the PTM:

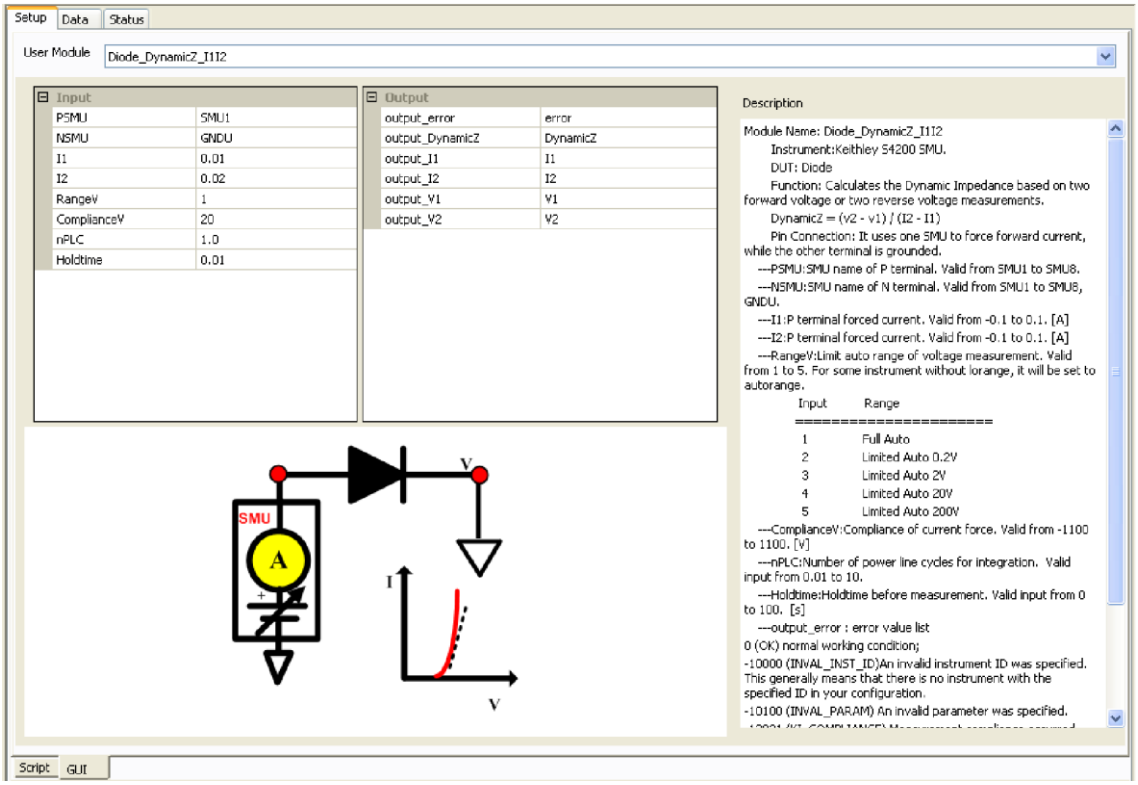
1. Click the **Return to Library View** icon on the ACS Basic toolbar. If ACS Basic is in Multitest mode, you must build a configuration navigator in order to add the test. For more information on how to build the configuration navigator in Multitest mode, refer to Build a project plan (on page 4-12).
2. Select the device where the user module `Diode_test1` is saved.
3. Select **UserLib** in the Test Category. The user module library opens in the Test modules area. Select the user module library that you want to test. In the next figure, `mylib1_Diode` is selected.

Figure 252: Open UserLib in Test Category



4. Click the **Open Test** button (see next figure). Accept the default parameters for this example.

Figure 253: PTM UserLib module



5. Save the PTM and the project by clicking the **Save All** button.
6. Run the PTM by clicking the Run icon.
7. If the user module that you created generates data, check the execution results in the PTM Data worksheet. To view the Data worksheet, click the **Data** tab.

Section 7

ACS Basic Formulator functions

In this section:

ABS

Purpose: Calculates the absolute value of each value in the designated column (vector) or the absolute value of any operand.

Format: ABS(X)

X = The name of any column (vector) listed under Columns or any operand.

Example: F2 = ABS(I_GATE)

Remarks: This function can be used to perform calculations in real time, while a test is executing.

AT

Purpose: Extracts and returns a single value from a column (vector).

Format: AT(V, POS)

V = The name of any column (vector) listed under Columns.

POS = The row number or position of column V where the single value is located.

Example: IDSAT = AT(I_DRAIN, 36)

AVG

Purpose: Returns the average of all values in the column (vector).

Format: AVG(V)

V = The name of any column (vector) listed under Columns.

Example: LEAKAGE = AVG(I_GATE)

CURRENT NOISE

Purpose: This formula is used to calculate noise current from Ig.

Format: CURRENT_NOISE(IMEAS,POINT_NUM)

IMEAS = measured current

POINT_NUM = The noise points number for calculate.

Example: Ig_noise=CURRENT_NOISE(Ig,50)

DELTA

Purpose: Returns the differences between the adjacent values in a column (vector). For example, for column V, DELTA returns (V2 − V1), (V3 − V2) and so on.

Format: DELTA(V)

V = The name of any column (vector) listed under Columns.

Example: GM = DELTA(I_DRAIN)/DELTA(V_GATE)

Remarks: This function performs calculations in real time, while a test is executing.

DIFF

Purpose: For all values in two selected columns (vectors), DIFF returns a third column (vector) containing the coefficient differences. That is, for columns V1 and V2, DIFF returns the following: (V12 − V11)/(V22 − V21), (V13 − V12)/(V23 − V22) and so on.

Format: DIFF(V1, V2)

V1 = The name of any column (vector) listed under Columns.

V2 = The name of any column (vector) listed under Columns.

Example: GM = DIFF(I_DRAIN, V_GATE)

Remarks: This function performs calculations in real time, while a test is executing.

DIMENSION

Purpose: Returns the number of occurrences of the last elements of the array.

Format: DIMENSION(V)

V = The name of any column (vector) listed under Columns.

Example: V2 = DIMENSION(V1)

Remarks: Returns the occurrence number.

EXP

Purpose: Returns the exponential, e_value, for each value in a column (vector) or for any operand.

Format: EXP(X)

X = The name of any column (vector) listed under Columns or any operand.

Example: NEWCURRENT = CURRENT*EXP(ANODEV)

Remarks: This function performs calculations in real time, while a test is executing.

EXPFIT

Purpose: Performs an exponential fit.

Fits the following exponential relationship to a specified range of values in two columns (vectors): One column, VX, containing X values and the other column, VY, containing Y values: Y = EXPFITA * e(EXPFITB * X) where EXPFITA and EXPFITB are fit constants.

Using the above exponential relationship, returns a new column (vector) containing Y values calculated from all X values in column VX.

Format: EXPFIT(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = For the range of X and Y values to be exponentially fitted, the row number (index) of the starting values.

ENDPOS = For the range of X and Y values to be exponentially fitted, the row number (index) of the ending values.

Example: DIODEI = EXPFIT(ANODEV, ANODEI, 2, LASTPOS(ANODEV))

Remarks: If a VX or VY value at either STARTPOS or ENDPOS is an invalid number (that is, the value is #REF), the function will not return a valid result.

EXPFITA

Purpose: Performs an exponential fit.

Fits the following exponential relationship to a specified range of values in two columns (vectors): One column, VX, containing X values and the other column, VY, containing Y values:

Y = EXPFITA * e(EXPFITB * X) (EXPFITA and EXPFITB are fit constants. Returns the value of the constant EXPFITA.

Format: EXPFITA(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = For the range of X and Y values to be exponentially fitted, the row number (index) of the starting values.

ENDPOS = For the range of X and Y values to be exponentially fitted, the row number (index) of the ending values.

Example: DIODEOFFSET = EXPFITA(ANODEV, ANODEI, 2, LASTPOS(ANODEV))

Remarks: If a VX or VY value at either STARTPOS or ENDPOS is an invalid number (that is, the value is #REF), the function will not return a valid result.

EXPFITB

Purpose: Performs an exponential fit.

Fits the following exponential relationship to a specified range of values in two columns (vectors): One column, VX, containing X values and the other column, VY, containing Y values: Y = EXPFITA * e(EXPFITB * X) where EXPFITA and EXPFITB are fit constants.

Returns the value of the constant EXPFITB in the relationship above.

Format: EXPFITB(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = For the range of X and Y values to be exponentially fitted, the row number (index) of the starting values.

ENDPOS = For the range of X and Y values to be exponentially fitted, the row number (index) of the ending values.

Example: DIODEIDEALITY = 1/(EXPFITB(ANODEV, ANODEI, 2, LASTPOS(ANODEV))*0.0257)

Remarks: If a VX or VY value at either STARTPOS or ENDPOS is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

FINDD

Purpose: Given a column (vector) V, beginning at START, FINDD searches down the column until it finds a value that matches the specified value X. Then it returns the row number (index) of that value. If FINDD does not find an exact match for X, it returns the row number (index) of the V value that is closest to X.

Format: FINDD(V, X, START)

V = The name of any column (vector) listed under Columns.

X = Any value (which may be the result of other calculations).

START = The row number (index) of the starting value for the search.

Example: IF = AT(ANODEI, FINDD(ANODEV, 0.7, FIRSTPOS(ANODEV)))

Remarks: Refer to the similar functions FINDLIN (on page 7-6) (Find using linear interpolation) and FINDU (on page 7-7) (Find up).

FINDLIN

Purpose: Find using linear interpolation. For example, given a column (vector) V, beginning at START, FINDLIN searches down the column until it finds a value that is closest (but does not exceed) the specified value X. Linear interpolation is then used to determine its decimal location between the found value and the next value in the column (vector). The returned index number (in decimal format) indicates the position of the specified value.

For example, assume you want to use FINDLIN to locate value 6 in the following array:

(Index 1)V

(Index 2)1

(Index 3)4

(Index 4)8

The search finds the index marker that is closest to (but does not exceed) 6. In this case, Index 3 is the closest. Linear interpolation is then used to determine the decimal position of the specified value (6) which is between Index 3 (value 4) and Index 4 (value 8). Value 6 is halfway between Index 3 and Index 4. Therefore, FINDLIN returns Index 3.5.

Format: FINDLIN(V, X, START)

V = The name of any column (vector) listed under Columns.

X = Any value (which may be the result of another calculation).

START = The row number (index) of the starting value for the search.

Example: IF = AT(ANODEI, FINDLIN(ANODEV, 0.7, FIRSTPOS(ANODEV)))

Remarks: Refer to the similar functions FINDD (on page 7-6) (Find down) and FINDU (on page 7-7) (Find up).

FINDU

Purpose: Given a column (vector) V, beginning at START, FINDU searches up the column until it finds a value that matches the specified value X. It then returns the row number (index) of that value. If FINDU does not find an exact match for X, it returns the row number (index) of the V value that is closest to X.

Format: FINDU(V, X, STARTPOS)

V = The name of any column (vector) listed under Columns.

X = Any value (which may be the result of another calculation[s]).

STARTPOS = The row number (index) of the starting value for the search.

Example: IF = AT(ANODEI, FINDU(ANODEV, 0.7, LASTPOS(ANODEV)))

Remarks: Refer to the similar functions FINDLIN (on page 7-6) (Find using linear interpolation) and FINDD (on page 7-6) (Find down).

FIRSTPOS

Purpose: Returns the row number (index) of the first value in a column (vector), typically the number 2.

Format: FIRSTPOS(V)

V = The name of any column (vector) listed under Columns.

Example: STARTOFARRAY = FIRSTPOS(I_DRAIN)

Remarks: Refer to the function LASTPOS (on page 7-11).

GET_FREQ

Purpose: Get the FFT frequency for an input time list.

Format: GET_FREQ(time_source)

Time_source = The name of time column (vector) listed under Columns.

Example: Fre1=GET_FREQ(time1)

Remarks: This Formulator function only used in the 1/f noise test project.

GET_TBD

Purpose: This formula is used to get breakdown(BD) time of TDDB test.

Format: GET_TBD(IS, TIME, CUR_J, SLOPE_J, NOISE_J, FIX_CUR_J, SLOPE_P_NUM, NOISE_P_NUM, AND_OR,METHODS)

It includes the following parameters.

- two variables: IS and TIME.
- four methods for BD judgment: CUR_J, SLOPE_J,NOISE_J, IX_CUR_J.
- two parameters for calculation: SLOPE_P_NUM, NOISE_P_NUM.
- one logic relationship: AND_OR.

You must choose the methods:

- IS = gate current.
- TIME = test time.
- CUR_J= judgment current times for breakdown, as |Ig[n]/Ig[0]| >= CUR_J.
- SLOPE_J= judgment slope times for breakdown, as |slope[n]/slope[n-1]| >= SLOPE_J.
- NOISE_J= judgment noise times for breakdown, as |Inoi[n]/Inoi[n-1]| >= NOISE_J.
- FIX_CUR_J=fixed cursor for judgment, as |Ig[n]| >= FIX_CUR_J.
- SLOPE_P_NUM= the slope points number for each step calculate. If equals to 5, then slope of Ig is calculated every 5 points.
- NOISE_P_NUM= the noise points number for each step calculate. If equals to 5, then noise current of Ig is calculated every 5 points.
- AND_OR= '0' or '1'. '1' stands for 'and', means all methods selected are achieved then BD happens; '0' stands for 'or', means any of the methods selected are achieved then BD happens.
- METHODS= the used methods serial number. '1234' means all four methods are chosen. '14' means only first and fourth are chosen.

Example: TBD=GET_TBD(Ig, time, 100, 50, 10000, 1E-4, 5, 5, 0,14)

Remarks: The above example means calculating TBD using Ig and time array. The judgment current times is 100, judgment slope times is 50, the judgment noise times is 10000, the fixed cursor data for judgment is 1E-4. Slope of Ig is calculated every 5 points and noise current of Ig is calculated every 5 points. Using the first and fourth methods, and these two methods are achieved then BD happens.

GET_TBD2

Purpose: This formula is used to get breakdown(BD) time of TDDB test.

Format: GET_TBD2(IS, IS_TIME, ISILC, SILC_TIME, IS_CUR_J, ISILC_CUR_J, SLOPE_J, NOISE_J, FIX_CUR_J, SLOPE_P_NUM, NOISE_P_NUM,METHODS)

It includes the following parameters:

- four variables: IS and IS_TIME, ISILC, SILC_TIME.
- four methods for BD judgment: CUR_J, SLOPE_J,NOISE_J, FIX_CUR_J.
- two parameters for calculation: SLOPE_P_NUM, NOISE_P_NUM.

You must choose the methods:

- IS = gate stressed current.
- IS_TIME = stressed time.
- ISILC = gate leakage current.
- ISILC_TIME = measure ISILC time.
- CUR_J= judgment current times for breakdown, as |Ig[n]/Ig[0]| >= CUR_J.
- SLOPE_J= judgment slope times for breakdown, as |slope[n]/slope[n-1]| >= SLOPE_J.
- NOISE_J= judgment noise times for breakdown, as |Inoi[n]/Inoi[n-1]| >= NOISE_J.
- FIX_CUR_J=fixed cursor for judgment, as |Ig[n]| >= FIX_CUR_J.
- SLOPE_P_NUM= the slope points number for each step calculate. If equals to 5, then slope of Ig is calculated every 5 points.
- NOISE_P_NUM= the noise points number for each step calculate. If equals to 5, then noise current of Ig is calculated every 5 points.
- METHODS= the used methods serial number. '1234' means all 4 methods are chosen. '14' means only first and fourth are chosen.

Example: TBD=GET_TBD2(Ig, time_str, I_silc, time_silc, 100, 50, 10000, 1E-4, 5, 5, 0,14)

Remarks: The above example means calculating TBD using two group of arrays: Ig and time_str, I_silc and time_silc. The judgment current times is 100, judgment slope times is 50, the judgment noise times is 10000, the fixed cursor data for judgment is 1E-4. Slope of Ig is calculated every five points and noise current of Ig is calculated every five points. Using the first and fourth methods, and these two methods are achieved then BD happens.

GET_VBD

Purpose: Get breakdown (BD) voltage in VRamp test.

Format: GET_VBD(IS, TIME, CUR_J, SLOPE_J, SLOPE_P_NUM, AND_OR, METHODS)

You must choose the methods:

- IS = gate current
- TIME = test time
- CUR_J = judgment current times for breakdown, as |Ig[n]/Ig[0]| >= CUR_J
- SLOPE_J = judgment slope times for breakdown, as |slope[n]/slope[n-1]| >= SLOPE_J
- SLOPE_P_NUM= the slope points number for each step calculate. If equals to 5, then slope of Ig is calculated every 5 points.

- AND_OR= '0' or '1'. '1' stands for 'and', means all methods selected are achieved, then BD happens; '0' stands for 'or', means any of the methods selected are achieved, then BD happens.
- METHODS= the used methods serial number. '12' means two methods are chosen.

Example: VBD=GET_VBD(Ig, time, 100, 50, 5, 5, 0,1)

Remarks: The above example means calculating VBD using Ig and time array. The judgment current times is 100, judgment slope times is 50. Slope of Ig is calculated every 5 points. Using the 1st methods, and it achieved then BD happens.

GMMAX

Purpose: Returns the maximum value from the two columns(vectors).

Format: GMMAX(Id, Vg)

Id= the I_drain vector.

Vg= the V_gate vector.

GM= DIFF(Id, Vg).

Example: Neg= GMMAX(DIFF(I_drain, V_gate))

JOIN

Purpose: To join multiply arrays together, and return a new array.

Format: JOIN(array1,array2)

Example: newarray=JOIN(array1,array2)

LASTPOS

Purpose: Returns the row number (index) of the last value in a column (vector).

Format: LASTPOS(V)

V = The name of any column (vector) listed under Columns.

Example: NUMSWEEPPTS = LASTPOS(COLLECTORI)

Remarks: Refer to the function FIRSTPOS (on page 7-8).

LINFIT

Purpose: Finds a linear equation of the form Y = a + bX from two sets of X and Y values selected from two columns (vectors), VX and VY. This equation corresponds to a line drawn through two points on a curve that is created by plotting the values in VY against the values in VX. The two points are specified by the arguments STARTPOS and ENDPOS.

Using the linear equation, returns a new column (vector) containing Y values calculated from all X values in column VX.

Format: LINFIT(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = The row number (index) of the first set of X and Y values applied to an exponential operation; row number or position (index) of the starting values.

ENDPOS = The row number (index) of the second set of X and Y values applied to an exponential operation; row number or position (index) of the ending values.

Example: RESISTORFIT = LINFIT (RESV, RESI, FIRSTPOS(RESV), LASTPOS(RESV))

Remarks: If a VX or VY value at either STARTPOS or ENDPOS is an invalid number (that is, the value is `#REF`), the function will not return a valid result. To return a linear regression fit for two columns (vectors), use the REGFIT function.

LINFITSLP

Purpose: Finds a linear equation of the form Y = a + bX from two sets of X and Y values selected from two columns (vectors), VX and VY. This equation corresponds to a line drawn through two points on a curve that is created by plotting the values in VY against the values in VX. The two points are specified by the arguments STARTPOS and ENDPOS.

Returns the slope of the linear equation (value of "b" in Y = a + bX).

Format: LINFITSLP(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = The row number (index) of the first set of X and Y values.

ENDPOS = The row number (index) of the second set of X and Y values.

Example: RESISTANCE = 1/LINFITSLP(RESV, RESI, FIRSTPOS(RESV), LASTPOS(RESV))

Remarks: If a VX or VY value at either STARTPOS or ENDPOS is an invalid number (that is, the value is `#REF`), the function will not return a valid result. To return the slope of a linear regression fit for two columns (vectors), use the REGFITSLP function.

LINFITXINT

Purpose: Finds a linear equation of the form Y = a + bX from two sets of X and Y values selected from two columns (vectors), VX and VY. This equation corresponds to a line drawn through two points on a curve that is created by plotting the values in VY against the values in VX. The two points are specified by the arguments STARTPOS and ENDPOS.

Returns the X intercept of the linear equation:

(value of "-a/b" in Y = a + bX).

Format: LINFITXINT(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = The row number (index) of the first set of X and Y values.

ENDPOS = The row number (index) of the second set of X and Y values.

Example: EARLYV = LINFITXINT(COLLECTORV, COLLECTORI, 56, 75)

Remarks: If a VX or VY value at either STARTPOS or ENDPOS is an invalid number (that is, the value is `#REF`), the function will not return a valid result. To return the X intercept of a linear regression fit for two columns (vectors), use the REGFITXINT function.

LINFITYINT

Purpose: Finds a linear equation of the form Y = a + bX from two sets of X and Y values selected from two columns (vectors), VX and VY. This equation corresponds to a line drawn through two points on a curve that is created by plotting the values in VY against the values in VX. The two points are specified by the arguments STARTPOS and ENDPOS.

Returns the Y intercept of the linear equation (value of "a" in Y = a +bX).

Format: LINFITYINT(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = The row number (index) of the first set of X and Y values.

ENDPOS = The row number (index) of the second set of X and Y values.

Example: OFFSET = LINFITYINT(GATEV, GATEI, FIRSTPOS(GATEV), LASTPOS(GATEV))

Remarks: If a VX or VY value at either STARTPOS or ENDPOS is an invalid number (that is, the value is `#REF`), the function will not return a valid result. To return the Y intercept of a linear regression fit for two columns (vectors), use the REGFITYINT function.

LN

Purpose: Returns the base-e (natural, Napierian) log of each value in a designated column (vector) or the Napierian log of any operand.

Format: LN(X)

X = The name of any column (vector) listed under Columns or any operand.

Example: DIODEV = LN(ANODEI)*0.026

Remarks: This function performs calculations in real time, while a test is executing.

LOG

Purpose: Returns the log value of each value in a designated column (vector) or the log of any operand. Note that it cannot accept two arrays in its parameters.

Format: LOG(V,VBASE)

V = column name

VBASE = logarithm base value

Example: F1 = LOG(I_DRAINI, 2)

Remarks: This function can be used to perform real-time calculations, while a test is executing.

LOG10

Purpose: Returns the base-10 log of each value in a designated column (vector) or the base-10 log of any operand.

Format: LOG10(V)

V = The name of any column (vector) listed under Columns or any operand.

Example: F2 = LOG10(I_DRAIN)

Remarks: This function performs calculations in real time, while a test is executing.

LOGFIT

Purpose: Finds the linear equation Y = a + b*log10(X) from two sets of X and Y values selected from two columns (vectors), VX and VY. The two points are specified by the arguments STARTPOS and ENDPOS.

Format: LOGFIT(VX, VY, STARTPOS, ENDPOS, POINTS=0)

VX = column name or operand

VY = column name or operand

STARTPOS = range of X and Y values applied to an exponential operation; row number or position (index) of the starting values

ENDPOS = range of X and Y values applied to an exponential operation; row number or position (index) of the ending values

Example: F3=LOGFIT(I_FM_pre1,I_Neg_pre1,23,56)

Remarks: If a VX or VY value at either STARTPOS or ENDPOS contains an invalid number (that is, the value is `#REF`), the function will not return a valid result.

MAX

Purpose: Searches all values in a column (vector) and returns the maximum value.

Format: MAX(V)

V = The name of any column (vector) listed under Columns.

Example: MAXGM = MAX(DIFF(DRAINI, GATEV))

MAXPOS

Purpose: Searches all values in a column (vector), finds the maximum value, and returns the row number (index) of the maximum value.

Format: MAXPOS(V)

V = The name of any column (vector) listed under Columns.

Example: PEAKSTRESS = AT(GATEV, MAXPOS(SUBSTRATEI))

MIN

Purpose: Searches all values in a column (vector) and returns the minimum value.

Format: MIN(V)

V = The name of any column (vector) listed under Columns.

Example: SMALLESTI = MIN(DRAINI)

MINPOS

Purpose: Searches all values in a column (vector), finds the minimum value, and returns the row number (index) of the minimum value.

Format: MINPOS(V)

V = The name of any column (vector) listed under Columns.

Example: LOCATION = MINPOS(DRAINI)

MOBILITY

Purpose: Mobility at Vg_tar by Id-Vg curve.

Format: MOBILITY(w,l,ci,Vg_tar,Vth,Id,Vg)

- w = Width of device gate
- l = Length of device gate
- ci = Capacitance of gate
- vg_tar = Target VG
- Vth = Threshold voltage
- Id = Measured Drain current
- Vg = Measured Gate voltage

Example: U1= MOBILITY(10,0.03,1,0.05,0.078,Id,Vg)

NOISE UTM

Purpose: This formula is used to generate correct time array for noise current

Format: NOISE_UTIM(MEASTIMES,POINT_NUM)

POW

Purpose: Returns the power value of each value in a designated column (vector) or any operand. Note that it cannot accept two arrays in its parameters.

Format: POW(VBASE, VPOW)

VBASE = The name of any column (vector) listed under Columns, or a number as base value.

VPOW = The name of any column (vector) listed under Columns, or a number as power value.

Example: I2 = POW(I_DRAIN, 2) or I2pow = POW(2, I_DRAIN)

POWERFFT

Purpose: Perform the forward FFT for input data source, get the power spectrum.

Format: POWERFFT(data_source, dialog box)

data_source: The value list of the sample data . a list

dialog box: The name of the dialog box function. integer

Value mappings of the parameter dialog box:

POWERFFT value mappings table

Value	Dialog box	Function execution
1	Rectangular	1
2	Hanning	w(n) = 0.5*(1-cos(2*pi*n/(N-1)))
3	Hamming	w(n) = 0.54-0.46*cos(2*pi*n/(N-1))
4	Blackman	w(n) = 0.42-0.5*cos(2*pi*n/(N-1))+0.08*cos(4*pi*n/(N-1))
5	Welch	w(n) = 1-((n-0.5*(N-1))/0.5*(N+1))^2

Example: pow1=POWERFFT(IDrain, 2)

Remarks: Return the power spectrum calculated for input source, this function only used in 1/f noise test project

REGFIT

Purpose: Performs a linear regression fit.

Fits the relationship of the form Y = aX + b to a specified range of values in two columns (vectors): column VX containing X values and column VY containing Y values:

Y = REGFITYINT + REGFITSLP * X (REGFITSLP and REGFITYINT are slope and Y-intercept constants). Using the above linear relationship, returns a new column (vector) containing Y values calculated from all X values in column VX).

Format: REGFIT(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = For the range of X and Y values to be fitted, the row number (index) of the starting values.

ENDPOS = For the range of X and Y values to be fitted, the row number (index) of the ending values.

Example: COLLECTORFIT = REGFIT(COLLECTORV, COLLECTORI, 25, LASTPOS(COLLECTORV))

Remarks: If a VX or VY value at either STARTPOS or ENDPOS is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

REGFITSLP

Purpose: Fits the relationship of the form Y = a + bX to a specified range of values in two columns (vectors): column VX containing X values and column VY containing Y values: Y = REGFITYINT + REGFITSLP * X where REGFITSLP and REGFITYINT are slope and Y-intercept constants.

Returns the value of the slope constant REGFITSLP in the relationship above.

Format: REGFITSLP(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = For the range of X and Y values to be fitted, the row number (index) of the starting values.

ENDPOS = For the range of X and Y values to be fitted, the row number (index) of the ending values.

Example: COLLECTORRES = 1/REGFITSLP(COLLECTORV, COLLECTORI, 25, LASTPOS(COLLECTORV))

Remarks: If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is #REF), the function will not return a valid result.

REGFITXINT

Purpose: Fits the relationship of the form Y = a + bX to a specified range of values in two columns (vectors): column VX containing X values and column VY containing Y values: Y = REGFITYINT + REGFITYINT * X where REGFITSLP and REGFITYINT are slope and Y-intercept constants.

Returns the value of the X intercept for relationship above (–REGFITYINT/REGFITSLP).

Format: REGFITXINT(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = For the range of X and Y values to be fitted, the row number (index) of the starting values.

ENDPOS = For the range of X and Y values to be fitted, the row number (index) of the ending values.

Example: EARLYV = REGFITXINT(COLLECTORV, COLLECTORI, 25, LASTPOS(COLLECTORV))

Remarks: If a VX or VY value at either STARTPOS or ENDPOS is an invalid number (that is, the value is #REF), the function will not return a valid result.

REGFITYINT

Purpose: Fits the relationship of the form Y = a + bX to a specified range of values in two columns (vectors)—column VX containing X values and column VY containing Y values: Y = REGFITYINT + REGFITSLP * X where REGFITSLP and REGFITYINT are slope and Y-intercept constants.

Returns the value of the Y intercept (REGFITYINT) for relationship above.

Format: REGFITYINT(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = For the range of X and Y values to be fitted, the row number (index) of the starting values.

ENDPOS = For the range of X and Y values to be fitted, the row number (index) of the ending values.

Example: OFFSET = REGFITYINT(COLLECTORV, COLLECTORI, 25, LASTPOS(COLLECTORV))

Remarks: If a VX or VY value at either STARTPOS or ENDPOS is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

REGFIT_LGX_LGY

Purpose: Performs a linear regression fit.

Fits the relationship of the form LGY = aLGX + b to a specified range of values in two columns (vectors):

column VX extracting LGX values and column VY extracting LGY values: LGY = REGFITYINT + REGFITSLP * LG X where REGFITSLP and REGFITYINT are slope and LGY-intercept constants.

Using the above linear relationship, returns a new column (vector) containing LGY values calculated from all LGX values. That is, input value is normal VX, VY, but the fit relationship is built on LGY and LGX.

Format: REGFIT_LGX_LGY(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = For the range of X and Y values to be fitted, the row number (index) of the starting values.

ENDPOS = For the range of X and Y values to be fitted, the row number (index) of the ending values.

Example: COLLECTORFIT1 = REGFIT_LGX_LGY(COLLECTORV, COLLECTORI, 25, LASTPOS(COLLECTORV))

Remarks: If a *`VX`* or *`VY`* value at either *`STARTPOS`* or *`ENDPOS`* is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

REGFIT_LGX_Y

Purpose: Performs a linear regression fit.

Fits the relationship of the form Y = aLGX + b to a specified range of values in two columns (vectors): column VX extracting LGX values and column VY containing Y values: Y = REGFITYINT + REGFITSLP * LGX where REGFITSLP and REGFITYINT are slope and Y-intercept constants.

Using the above linear relationship, returns a new column (vector) containing Y values calculated from all LGX values in column VX. That is, input value is normal value VX,VY, but the fit relationship is built on Y and LGX.

Format: REGFIT_LGX_Y(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = For the range of X and Y values to be fitted, the row number (index) of the starting values.

ENDPOS = For the range of X and Y values to be fitted, the row number (index) of the ending values.

Example: COLLECTORFIT 2 = REGFIT_LGX_Y(COLLECTORV, COLLECTORI, 25, LASTPOS(COLLECTORV))

Remarks: If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

REGFIT_X_LGY

Purpose: Performs a linear regression fit.

Fits the relationship of the form LGY = a X+ b to a specified range of values in two columns (vectors)—column VX containing X values and column VY extracting LGY values: LGY = REGFITYINT + REGFITSLP * X where REGFITSLP and REGFITYINT are slope and LGY-intercept constants.

Using the above linear relationship, returns a new column (vector) containing LGY values calculated from all X values in column VX. That is, input value is normal value VX, VY, but the fit relationship is built on LGY and X.

Format: REGFIT_X_LGY(VX, VY, STARTPOS, ENDPOS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

STARTPOS = For the range of X and Y values to be fitted, the row number (index) of the starting values.

ENDPOS = For the range of X and Y values to be fitted, the row number (index) of the ending values.

Example: COLLECTORFIT3= REGFIT_X_LGY (COLLECTORV, COLLECTORI, 25, LASTPOS(COLLECTORV))

Remarks: If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is #REF), the function will not return a valid result.

RES

Purpose: Calculate resistance vector value by giving V and I.

Format: RES(V,I))

V=voltage vector listed in Variables box

I=current vector listed in Variables box

Example: RG_1=RES(V_Gate_1,I_Gate_1)

Remarks: If a I or V value is an invalid number (that is, the value is #REF), the function will not return a valid result.

RES_4WIRE

Purpose: calculate 4-wire resistance vector value by giving V and I.

Format: RES_4WIRE(VPOS,VNEG,I)

VPOS=positive voltage vector listed in Variables box

VNEG=negative voltage vector listed in Variables box

I=current vector listed in Variables box

Example: RGD=RES_4WIRE(V_Gate_1,V_Drain_1,I_Gate_1)

Remarks: If VPOS, VNEG, or I value is an invalid number (that is, the value is #REF), the function will not return a valid result.

RES_4WIRE_AVG

Purpose: calculate 4-wire resistance average value by giving V and I.

Format: RES_4WIRE_AVG(VPOS,VNEG,I)

VPOS=positive voltage vector listed in Variables box

VNEG=negative voltage vector listed in Variables box

I=current vector listed in Variables box

Example: RGD_AVG=RES_4WIRE_AVG(V_Gate_1,V_Drain_1,I_Gate_1)

Remarks: If VPOS, VNEG , or I value is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

RES_AVG

Purpose: calculate resistance average value by giving V and I

Format: RES_AVG(V,I)

V=voltage vector listed in Variables box

I=current vector listed in Variables box

Example: RG_1=RES_ AVG (V_Gate_1, I_Gate_1)

Remarks: If a I or V value is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

SMOOTH

Purpose: Smooth the vector using FFT.

Format: SMOOTH (V, RANK)

V = The name of any column (vector) listed under Columns.

RANK = The rank number of smooth level.

Example: I2 = SMOOTH (I1, 2)

Remarks: If a V value is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

SQRT

Purpose: Returns the square root of each value in a designated column (vector) or the square root of any operand.

Format: SQRT(X)

X = The name of any column (vector) listed under Columns or any operand.

Example: TWO = SQRT(4)

Remarks: A negative value of X returns #REF in the Data worksheet. This function performs calculations in real time, while a test is executing.

SS

Purpose: Extract the sub-threshold swing between a certain duration of current. The sub-threshold swing is defined as the deferential of Vg and log Id, for example, SS=dVg/dlogId.

Format: SS (I_DRAIN, V_GATE, START_I, STOP_I)

I_DRAIN = the drain current measured in the test.

V_GATE = the gate voltage forced in the test.

START_I = the start current to define the calculation region.

STOP_I = the stop current to define the calculation region.

Example: S0=SS (I_Drain_1, V_Gate_1, 1.3e-5, 1.5e-5)

Remarks: If an I_DRAIN or V_GATE value is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

SSVTCI

Purpose: Extract the sub-threshold swing from the certain duration between left offset and right offset of the constant current threshold voltage.

Format: SSVTCI(I_DRAIN, V_GATE, VTCI,LEFT_OFFSET,RIGHT_OFFSET)

I_DRAIN = the drain current measured in test.

V_GATE= the gate voltage.

VTCI= constant current threshold voltage.

LEFT_OFFSET,RIGHT_OFFSET= left offset and right offset of the constant current threshold voltage

Example: SLOPE=SSVTCI(I_Drain_1,V_Gate_1,VTH,-0.1,-0.1)

Remarks: If a I_DRAIN or V_GATE value is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

SUBARRAY1

Purpose: Extract a subarray based on full array.

Format: SUBARRAY1(V, start, period)

V: vector array

start: start index

period: extraction period

Example: VGate2=SUBARRAY1(VGate, 1, 5)

SUBARRAY2

Purpose: Extract a subarray based on full array.

Format: SUBARRAY2(V, start, stop)

V: vector array

start: start index

stop: stop index

Example: VGate2=SUBARRAY1(VGate, 1, 50)

TANFIT

Purpose: Finds a linear equation of the form Y = aX + b from two columns (vectors), VX and VY. This equation corresponds to a tangent of the curve that is created by plotting the values in VY against the values in VX. The value at which the tangent is found is specified by the argument POS.

Using the linear equation, returns a new column (vector) containing Y value calculated from all X values in column VX.

Format: TANFIT(VX, VY, POS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

POS = The row number (index) of the value where the tangent is to be found.

Example: VTFIT = TANFIT(GATEV, DRAINI, MAXPOS(GM))

Remarks: If a VX or VY value at POS is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

TANFITSLP

Purpose: Finds a linear equation of the form Y = aX+ b from two columns (vectors), VX and VY. This equation corresponds to a tangent of the curve that is created by plotting the values in VY against the values in VX. The value at which the tangent is found is specified by the argument POS.

Returns the slope of the linear equation (value of "b" in Y = aX + b).

Format: TANFITSLP(VX, VY, POS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

POS = The row number (index) of the value where the tangent is to be found.

Example: VTSLOPE = TANFITSLP(GATEV, DRAINI, MAXPOS(GM))

Remarks: If a VX or VY value at POS is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

TANFITXINT

Purpose: Finds a linear equation of the form Y = aX + b from two columns (vectors), VX and VY. This equation corresponds to a tangent of the curve that is created by plotting the values in VY against the values in VX. The value at which the tangent is found is specified by the argument POS.

Returns the X intercept of the linear equation (value of "-a/b" in Y = aX + b).

Format: TANFITXINT(VX, VY, POS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

POS = The row number (index) of the value where the tangent is to be found.

Example: VT = TANFITXINT(GATEV, DRAINI, MAXPOS(GM))

Remarks: If a VX or VY value at POS is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

TANFITYINT

Purpose: Finds a linear equation of the form Y = aX + b from two columns (vectors), VX and VY. This equation corresponds to a tangent of the curve that is created by plotting the values in VY against the values in VX. The value at which the tangent is found is specified by the argument POS.

Returns the Y intercept of the linear equation:

(value of "a" in Y = aX + b).

Format: TANFITYINT(VX, VY, POS)

VX = The name of any column (vector) listed under Columns.

VY = The name of any column (vector) listed under Columns.

POS = The row number (index) of the value where the tangent is to be found.

Example: OFFSET = TANFITYINT(GATEV, DRAINI, GMMAX)

Remarks: If a VX or VY value at POS is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

TAR_YatX

Purpose: Find y_tar in y corresponding to x_tar in x's index

Format: TAR_YatX(x, y, x_tar)

x x array

y y array

x_tar target x

Example: y0=TAR_YatX(V, I, 0.1)

TTF_DID_LGT

Purpose: TTF means time to failure. In HCI test, you may want to analyze two arrays, time array and degradation (in percentage), and find the specific time corresponding to the aimed degradation. This is accomplished by choosing different fitting Models.

In this Formulator, the fitting relationship is built on time log scale and degradation of drain current.

Format: TTF_DID_LGT(DID, T, START_POS, END_POS, GOAL)

DID = degradation (in percentage) of drain current listed under Columns

T= time array listed under Columns

STARTPOS = For the range of T and DID values to be fitted, the row number (index) of the starting values.

ENDPOS = For the range of T and DID values to be fitted, the row number (index) of the ending values.

GOAL = the aim degradation (in percentage) value.

Example: F1= TTF_DID_LGT (I_DRAIN_DEG, TIME, 5, 11, 10)

Remarks: If a T or DID value at pos is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

TTF_ LGDID_T

Purpose: TTF means time to failure. In HCI test, you may want to analyze two arrays – time array and degradation (in percentage), and find the specific time corresponding to the aimed degradation. This is accomplished by choosing different fitting Models.

In this Formulator, the fitting relationship is built on time array and log value of degradation of drain current.

Format: TTF_LGDID_T(DID, T, START_POS, END_POS, GOAL)

DID = degradation (in percentage) of drain current listed under Columns

T = time array listed under Columns.

STARTPOS = For the range of T and DID values to be fitted, the row number (index) of the starting values.

ENDPOS = For the range of T and DID values to be fitted, the row number (index) of the ending values.

GOAL = the aim degradation(in percentage) value

Example: F2 = TTF_LGDID_T (I_DRAIN_DEG, TIME, 5, 11, 10)

Remarks: If a T or DID value at pos is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

TTF_DID_T

Purpose: TTF means time to failure. In HCI test, you may want to analyze two arrays, time array and degradation (in percentage), and find the specific time corresponding to the aimed degradation. This is accomplished by choosing different fitting models.

In this Formulator, the fitting relationship is built on time array and degradation of drain current.

Format: TTF_DID_T(DID, T, START_POS, END_POS, GOAL)

DID = degradation (in percentage) of drain current listed under Columns.

T = time array listed under Columns.

STARTPOS = For the range of T and DID values to be fitted, the row number (index) of the starting values.

ENDPOS = For the range of T and DID values to be fitted, the row number (index) of the ending values.

GOAL = the aim degradation (in percentage).

Example: F3= TTF_DID_T (I_DRAIN_DEG, TIME, 5, 11, 10)

Remarks: If a T or DID value at pos is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

TTF_LGDID_LGT

Purpose: TTF means time to failure. In HCI test, you may want to analyze two arrays, time array and degradation (in percentage), and find the specific time corresponding to the aimed degradation. This is accomplished by choosing different fitting models.

In this Formulator function, the fitting relationship is built on time log scale and degradation of drain current.

Format: TTF_LGDID_LGT(DID, T, START_POS, END_POS, GOAL)

DID = degradation (in percentage) of drain current listed under Columns.

T = time array listed under Columns.

STARTPOS = For the range of T and DID values to be fitted, the row number (index) of the starting values.

ENDPOS = For the range of T and DID values to be fitted, the row number (index) of the ending values.

GOAL = the aim degradation(in percentage) value.

Example: F4= TTF_LGDID_LGT (I_DRAIN_DEG, TIME, 5, 11, 10)

Remarks: If a T or DID value at pos is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

VTCI

Purpose: Algorithm by using u, w, l to calculate Id_target.

Format: VTCI(u, w, l, I_DRAIN,V_GATE)

u: The current density

w: The device width

l: The device length

I_DRAIN : The I_drain vector

V_GATE: The V_gate vector

Example: VTH1 = VTCI(0.001,1,0.03,I_DRAIN,V_GATE)

Remarks: If an I_DRAIN or V_GATE value at POS is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

VTLINGM

Purpose: Extrapolates threshold voltage from measurement of maximum slope of the ID-VGS curve, as below:

Vth = VGS(@Gmmax)-ID(@Gmmax)/Gmmax (VGS(@Gmmax is the gate voltage at the point of the maximum slope of the ID-VGS curve; ID(@Gmmax is the drain current at the point of the maximum slope of the ID-VGS curve; Gmmax is the maximum slope of the ID-VGS curve).

Format: VTLINGM(I_DRAIN, V_GATE)

I_DRAIN is the drain current measured in test. V_GATE is the gate voltage.

Example: VTH0 = VTLINGM(I_DRAIN, V_GATE)

Remarks: DC bias voltages for Vth measurements are VDS = VDS_lin, VBS = VBB typically, for PMOS, VDS_lin=-0.1 V(@VDD=5V); for NMOS, VDS_lin=0.1V(@VDD=5V). If a I_DRAIN or V_GATE value is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

VTSATGM

Purpose: Extrapolates threshold voltage from measurement of the IDS-VGS curve. In saturation region of MOSFET, Vth=Vgs(@Ids=0).

DC bias voltages for Vth saturation measurements are

VDS = VGS = VDD, VBS = VBB

Format: VTSATGM(I_DRAIN, V_GATE)

Example: VTH1 = VTSATGM(I_DRAIN, V_GATE)

Remarks: If an I_DRAIN or V_GATE value at POS is an invalid number (that is, the value is `#REF`), the function will not return a valid result.

VT SAT TRI

Purpose: Vt algorithm by triple divide Id square root curve.

Format: VT_SAT_TRI(Vg_tar, Id, Vg)

Vg_tar Target VG

Id Measured Drain current

Vg Measured Gate voltage

Example: `VT0 = VT_SAT_TRI(0.08, Id, Vg`

Keithley Instruments • 28775 Aurora Road • Cleveland, Ohio 44139 • 1-800-833-9200 • tek.com/keithley

KEITHLEY
A Tektronix Company

04/2022