Tektronix

Model 4200A-SCS

Clarius

USER'S MANUAL

4200A-914-01F April 2025

KEITHLEY
A Tektronix Company

Model 4200A-SCS

Clarius

User's Manual

Document number: 4200A-914-01 Rev. F April 2025

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

Introduction

In this section:

Get started with Clarius

Clarius is the primary application of Clarius+ and is the primary user interface for the 4200A-SCS. Clarius is a versatile tool that helps you characterize individual parametric test devices or automate testing of an entire semiconductor wafer. It allows you to create, execute, and evaluate tests and complex test sequences without programming.

The Clarius Software user interface provides touch-and-swipe or point-and-click control for advanced test definition, parameter analysis, graphing, and automation capabilities for modern semiconductor, materials, and process characterization.

Key features:

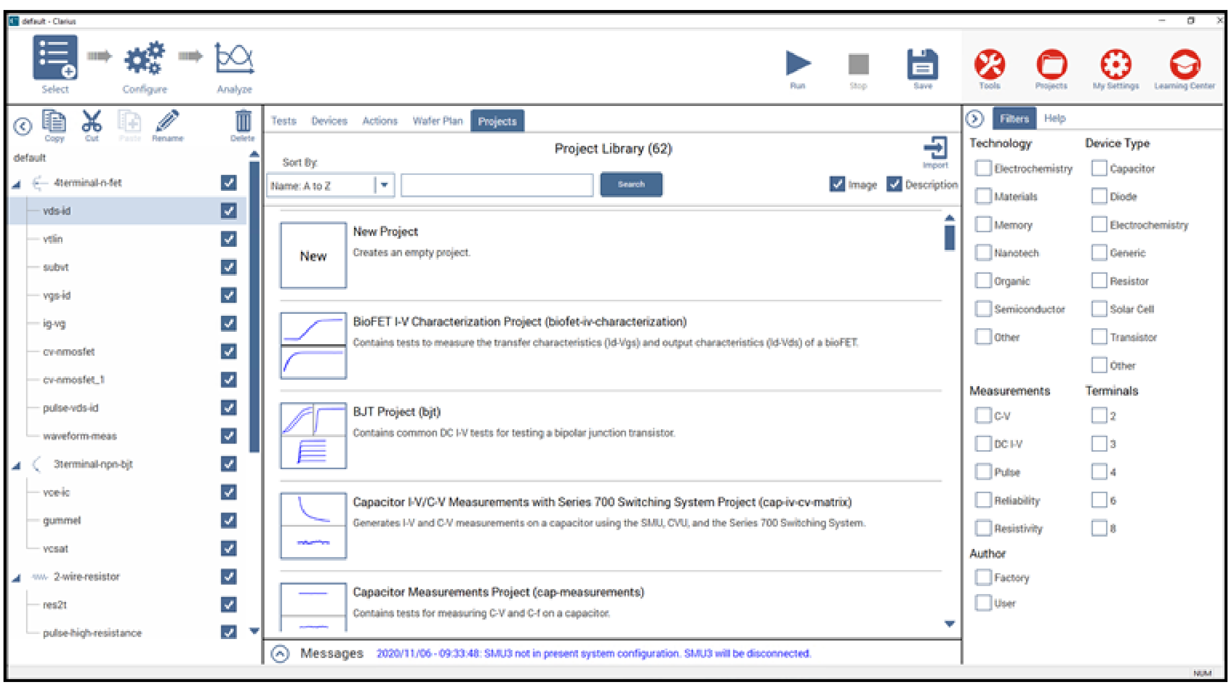
- Ready-to-use, modifiable application tests, projects, and devices that reduce test development time
- Built-in measurement videos from world-wide Application Engineers in four languages
- Pin-to-pad contact check ensures reliable measurements
- Multiple measurement functions
- Data display, analysis, and arithmetic functions

Clarius interface

The Clarius interface allows you to:

- Build and edit project and execution sequences.
- Configure tests.
- Execute tests and actions, such as switching matrix connections and prober movements, including:
    - A single test for one device (such as a transistor, diode, resistor, capacitor).
    - A test sequence for one device.
    - Test sequences for multiple devices. For example, test all the devices contacted by a prober at a location on a semiconductor wafer.
    - The test sequences of an entire project, which may include multiple prober touchdowns for a single semiconductor die.
- View test and analysis results.
- Analyze test results using built-in parameter extraction tools.

Figure 1: Clarius interface



Touchscreen basics

You can operate the 4200A-SCS using the touchscreen. You can use your fingers, clean room gloves, or any stylus manufactured for capacitive touchscreens.

To select and move on the screen:

- To scroll, swipe up or down on the screen.
- To select an item, touch it on the screen.
- To double-click an item, touch it twice.
- To right-click an item, touch and hold, then release to see the options.

To enter information, you can use the on-screen keyboard. Swipe from the left side of the display to open the keyboard.

The touchscreen uses standard Microsoft™ Windows™ touch actions. For additional information on the actions, refer to the Microsoft help information, available from the on-screen keyboard window menu option **Tool > Help Topics**.

You can also adjust the touch settings using the Pen and Touch options in the Windows Control Panel.

Choose the project phase

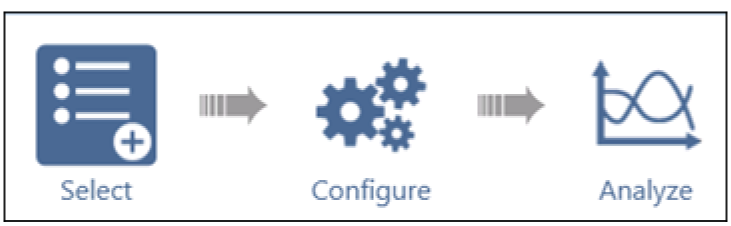
The options on the left side of the top pane of Clarius determine which phase of the project you are working on and allow you to select options to support your tests.

Select displays the libraries, which you can use to add existing projects, tests, devices, actions, and wafer plans to your project. You can also create your own tests, actions, and projects.

Configure displays the parameters for the item you selected in the project tree. For example, if you selected a test, the parameters for each terminal of the test and the entire test are available.

Analyze displays the results of the test in a spreadsheet and graph. You can also access analysis tools to explore and export your data.

Figure 2: Select, Configure, and Analyze



Run tests and set up your workspace

Figure 3: Clarius run test and workspace options

The options on the right side of the top pane of Clarius include options that allow you to run tests, configure instruments, manage projects, set up your workspace, and learn about the 4200A-SCS.

Run runs the highlighted item. You can run an individual test by highlighting only that test. You can run all the tests for a device, subsite, site, or project by highlighting the device, subsite, site, or project. Only items that are checked and below the selected item in the hierarchy are run.

Stop stops all running items.

Save saves the project configuration.

Tools provides module-specific tools and data export options. For source-measure units (SMUs), you can run autocalibration. For capacitance-voltage units (CVUs), you can set up connection compensation, do real-time measurements, and perform a confidence check. For pulse measure units (PMUs) and pulse generator units (PGUs), you can set up connection compensation. The Data Export options allow you to export Run History data files to Microsoft Excel.

Use **Projects** to manage your projects. It includes options to create, import, export, copy, cut, paste, edit, and delete projects. Projects are automatically stored in Projects when you create them in the project tree.

Use **My Settings** to customize Clarius to better meet your needs. You can change environment settings, run settings, GPIB abort settings, and error and warning logging. It also includes the About Clarius option, which lists the Clarius version and copyright information.

Use the **Learning Center** to access complete 4200A-SCS documentation, including online help, videos, instructions, application notes, white papers, and other materials to help you use your 4200A-SCS.

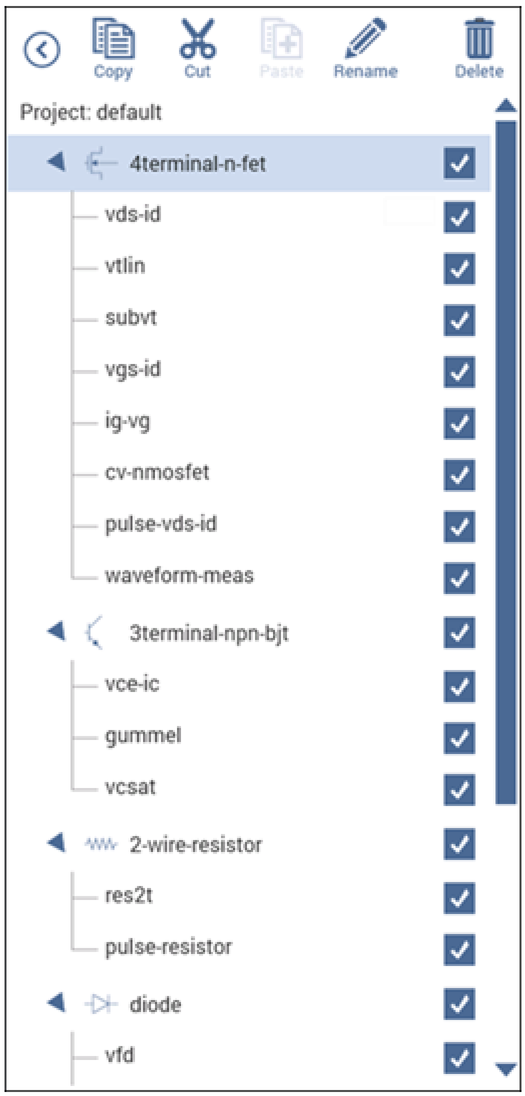
Organize items in the project tree

The project tree on the left side of the Clarius window displays the items in your project, including devices, tests, actions, and sites. The project tree for the default project is shown in the following figure.

The settings for the item you select in the project tree are displayed when you select Configure from the top bar. The test data for the item is displayed when you select Analyze.

When you run a test, the item that is highlighted runs. If the item is a project, site, subsite, or device, all checked items in the hierarchy below the highlighted item run.

Figure 4: Project tree for the default project

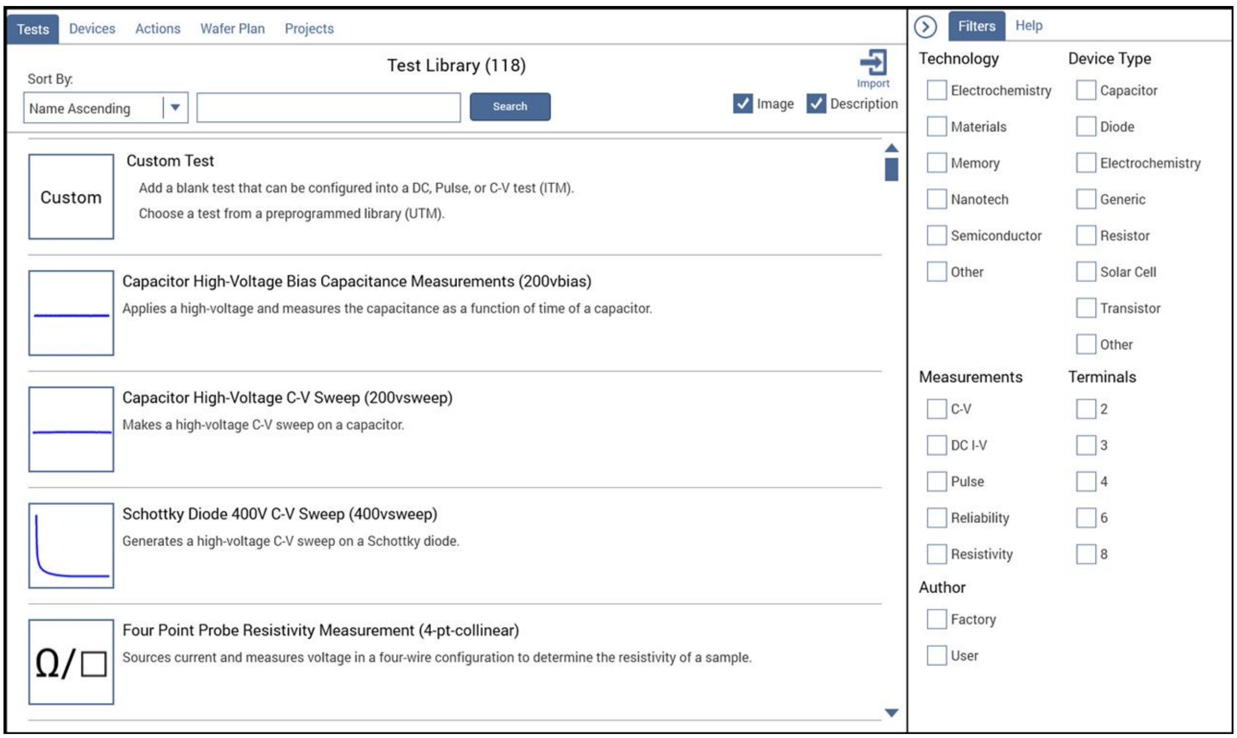


Select items from the libraries

When you choose Select, the center pane displays libraries of tests, devices, actions, wafer plans, or projects that you can add to the project tree. These libraries are templates that you can copy from to create your own tests, devices, actions, wafer plans, and projects. When you copy an item from the library to the project tree, the item in the project tree is a copy. The item in the library is not affected by any changes you make to the copy.

The **Test Library** contains predefined tests. The predefined tests contain detailed definitions that tell Clarius how to characterize a device, including associated data analyses and parameter extractions. Clarius comes with a library of tests for commonly used devices, including transistors, diodes, resistors, and capacitors. You can also create your own tests. The following figure shows the Test Library.

Figure 5: Test library



The **Device Library** contains the devices that need to be characterized, such as transistors, diodes, resistors, or capacitors. Each test must be in the project tree under a device. The devices available in the library include the standard set of devices that come installed on the 4200A-SCS and any custom-name devices that you have submitted; refer to Submitting devices to a library (on page 4-2).

The **Action Library** contains items that support the tests and help control the project. Actions can generate dialog boxes to prompt test operator action, control prober movements, and manage switching. You can also create your own actions from user libraries.

The **Wafer Plan Library** contains sites and subsites. A site is used if you are testing a repeating pattern of dies and test structures on a wafer. Every wafer location that a prober can move to and contact at any one time is a subsite. There are typically multiple subsites for each site. Subsites typically correspond to a single test structure or other combination of devices that are tested as a group.

The **Project Library** contains predefined projects. Projects include the devices, tests, actions, sites, and subsites organized for testing a single device, group of devices, or wafer. You can also create your own empty project.

For most of the libraries, the right pane displays filters that you can use to reduce the list of library items to the items you need. You can also use the Search option at the top of the library to type a search term to reduce the number of items.

You can sort the libraries by name or title.

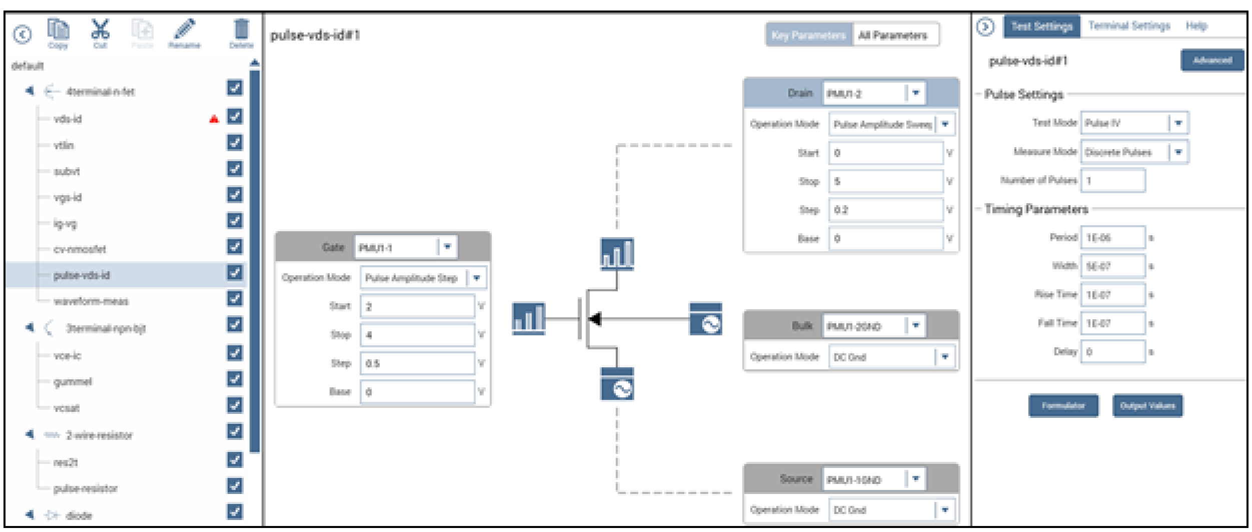
The **Image** and **Description** options at the top of the library allow you to turn the images and descriptions that are shown in the library on or off. Turning them off allows more items to be visible on the screen.

Import allows you to import items into the 4200A-SCS.

Configure the project

Select **Configure** for an item in the project tree to display the settings for that item. Depending on the item, settings are available in the center and right panes. Help for the selected item is also available in the Help pane.

Figure 6: Configure pane for the pulse-vds-id test



Analyze data

When you run tests, you can display and analyze test results and test definitions in the Analyze pane. The Analyze pane displays data in a spreadsheet and as a graph.

The **View** options change the view from both spreadsheet and graph to only the spreadsheet or only the graph. You can use **Save Data** to save the graph to an image file or save the data into a spreadsheet file.

If the Formulator was used to calculate data for this test, the **Run Formulas List** displays the calculations. You can select **Formulator** to open the Formulator and edit the formulas or create new ones.

The **Graph Definition Menu** and other options on the graph allow you to change the display of the data.

The **Run History** pane on the right displays the time and name of each test run. When you select a run from Run History, the sheet and graph in the center pane change to show the data from that test run.

The **Terminal Settings** pane, also on the right, displays the settings for the presently selected test run.

Figure 7: Analyze showing the vds-id test

Messages

Messages regarding the test and execution are displayed at the bottom of the Clarius window. To expand the Messages area to view more detail, select the up arrow to the left of the Messages heading.

You can right-click a message to copy it or to select and copy all messages to the clipboard.

You can also right-click and select **Clear All** to remove the existing messages.

Help pane

The Help pane displays information that is related to the library item or project tree item that is selected.

If you have the Select pane open, the help describes the item that is selected in the Library.

If you have the Configure or Analyze pane open, the help describes the item that is selected in the project tree.

Additional Clarius+ applications

Two of the Clarius+ applications support Clarius:

- The Keithley User Library Tool (KULT) allows you to create libraries of test modules using the C programming language. These test modules are executed by Clarius.
- The Keithley Configuration Utility (KCon) manages the configuration and interconnections between the test system components that are controlled by Clarius.

To control the 4200A-SCS remotely using an external GPIB controller, you can use another Clarius+ software tool, the Keithley External Control Interface (KXCI). You cannot run KXCI and Clarius simultaneously.

To configure and control the optional pulse cards, you can use the Keithley Pulse tool (KPulse). A pulse card is a dual-channel pulse card that is integrated inside the 4200A-SCS mainframe. Although KPulse can be launched at the same time as Clarius, KPulse and Clarius cannot communicate with hardware simultaneously.

For information about these applications, refer to:

- *Model 4200A-SCS KULT and KULT Extension Programming* (4200A-KULT-907-01)
- “Keithley Configuration Utility (KCon)” in *Model 4200A-SCS Setup and Maintenance*
- *Model 4200A-SCS KXCI Remote Control Programming*
- “KPulse (for Keithley Pulse Cards)” in the *Model 4200A-SCS Pulse Card (PGU and PMU) User's Manual*

Embedded computer policy

CAUTION

If you install software that is not part of the standard application software for the 4200A-SCS, the nonstandard software may be removed if the instrument is sent in for service. Back up the applications and any data related to them before sending the instrument in for service.

CAUTION

Do not reinstall or upgrade the Microsoft™ Windows™ operating system (OS) on any 4200A-SCS unless the installation is performed as part of authorized service by Keithley. Violation of this precaution will void the 4200A-SCS warranty and may render the 4200A-SCS unusable. Any attempt to reinstall or upgrade the operating system (other than a Windows service pack update) will require a return-to-factory repair and will be treated as an out-of-warranty service, including time and material charges.

Although you must not attempt to reinstall or upgrade the operating system, you can restore the hard drive image (complete with the operating system) using the AOMEI OneKey Recovery software tool, described in "System-level backup and restore software" in *Model 4200A-SCS Setup and Maintenance*.

You can install Windows quality updates. The version of Windows installed on 4200A-SCS systems only supports installation of quality updates. Feature updates are not supported.

As shipped, the 4200A-SCS automatically runs Windows quality updates when the 4200A-SCS is restarted. They are prevented from occurring during operation.

Section 2

Projects and tests

In this section:

Introduction

This chapter describes how to set up projects and tests in Clarius.

Set up a simple project

To start testing, you can start with a new project or use an existing project. A project consists of items such as devices and tests.

The order of operations of a test is determined by the order and selection of items in the project tree.

The settings for the project are saved so you can use them multiple times or adjust them for future tests.

The following topics describe how to set up and run a simple project using an existing project from the Project Library.

Select project components

Use the Select pane to add items to the project tree. When Select is active, the center pane contains libraries for tests, devices, actions, wafer plans, and projects. You can use filters and search options to help you find the items you need for your test.

To clear filters, select **Clear Filters** at the bottom of the Filters pane. To clear the search, select **Clear** next to the Search button.

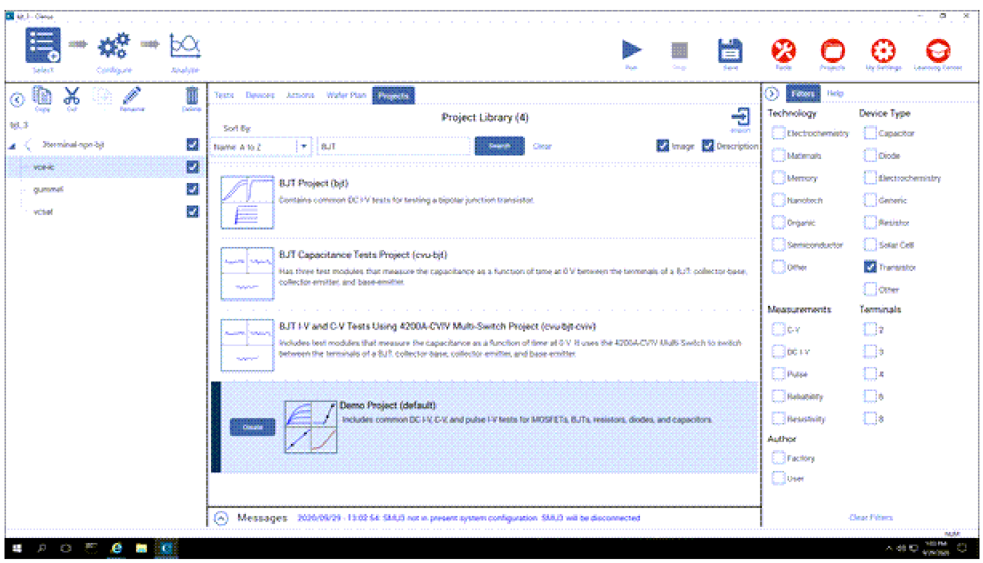
You can use Ctrl+click and Shift+click to select multiple items in the project tree.

The following example shows you how to select tests for bipolar junction transistors (BJTs).

To set up a test of BJTs:

1. Select **Save** to save your existing project.
2. Choose **Select**.
3. Select the **Projects** tab.
4. In the Filters pane, select **Transistor**.
5. In the Search box, type **BJT** and select **Search**. The Project Library displays projects that are intended for BJT transistor testing.
6. Select **Create** for the project you want to open. The project replaces the previous project in the project tree.

Figure 8: Filter and search for the bjt project



Add a device and test to the project

You can add additional items to a project. When you add a project from the library to the project tree, it is copied from the project in the library. Any changes you make do not affect the original project. The new project in the project tree is automatically stored in Projects.

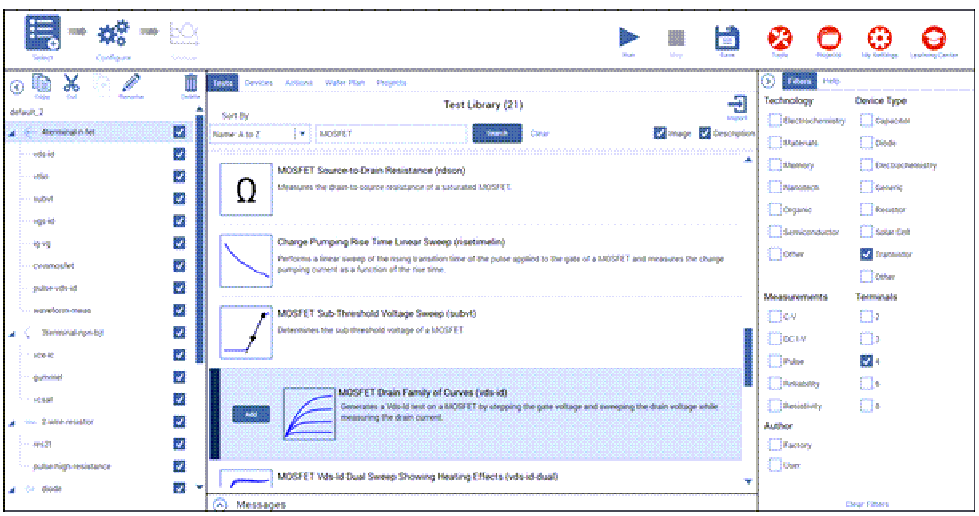
This example shows you how to add a predefined test to the project. Predefined tests are configured with commonly used parameter settings and a set of typical data. Once they are in a project, you can change the parameters as needed. They can be an efficient way for you to add a test to your project.

You can use the basic procedure described here to find any items in the library.

To add a four-terminal MOSFET device and test to the project:

1. In the center pane, select **Tests**.
2. In the Filters pane, select **Transistor** and **4** Terminals.
3. In the search box, type **MOSFET** and select **Search**.
4. Scroll to the **MOSFET Drain Family of Curves (vds-id)** test.
5. Select **Add**. The selected test and the device are added to the project tree under the previous item that was highlighted.
6. To move the device and test, drag the device to a new location.
7. Select **Save**.

Figure 9: Add a MOSFET test and device to the project



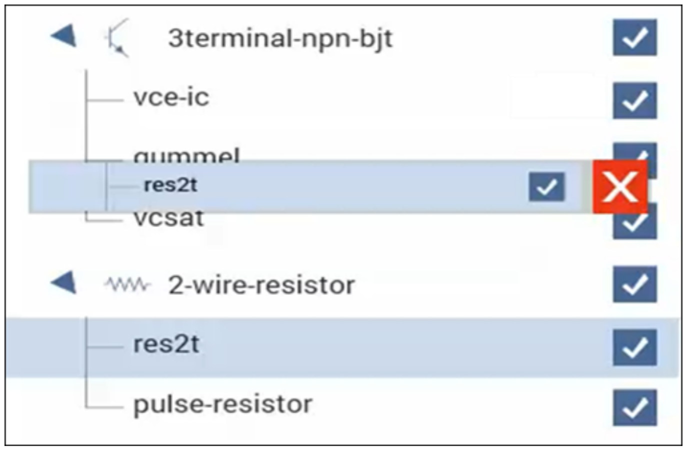
NOTE

If the device for a test is not in the project tree, Clarius adds the appropriate device when you add a test to the project tree. You can also add the device and test separately.

Rearrange items in the project tree

To rearrange items in the project tree, drag the items to the new location. If the item cannot be placed in the selected location, a red X is displayed. In the example below, a resistor test cannot be placed under a BJT device.

Figure 10: Object not allowed at this location in the project tree



For actions, if they are at the bottom of the project tree, you can promote or demote them to move them in the tree structure. For example, if the action is under a device, you might want to move it to be at the project level. To promote or demote an action, right-click the action and select **Promote Action** or **Demote Action**.

Delete objects in the project tree

CAUTION

If you delete an object, other items may also be deleted. For example, if you delete a subsite, all device and tests in the subsite are also deleted. If you delete a device, all tests in the device are deleted.

To delete an object:

1. In the project tree, select the item you want to delete.
2. Select the object.
3. Select **Delete** at the top of the project tree. A confirmation message is displayed.
4. Select **OK**.

Configure a simple test

Use the Configure pane to set up your test. For interactive test modules (ITMs), the Configure pane displays a schematic of the test device. The schematic is connected to an object that shows the operation mode and the type of instrument that is connected to the terminal.

The settings for the test are saved so you can use them multiple times.

NOTE

The following topics discuss the Test Settings pane for interactive test modules (ITMs). For tests that are based on user modules (UTMs), you use the options in the Test Settings pane to select the User Library and User Module for the test. Refer to Create a custom test (on page 2-21) for information on settings available for UTMs.

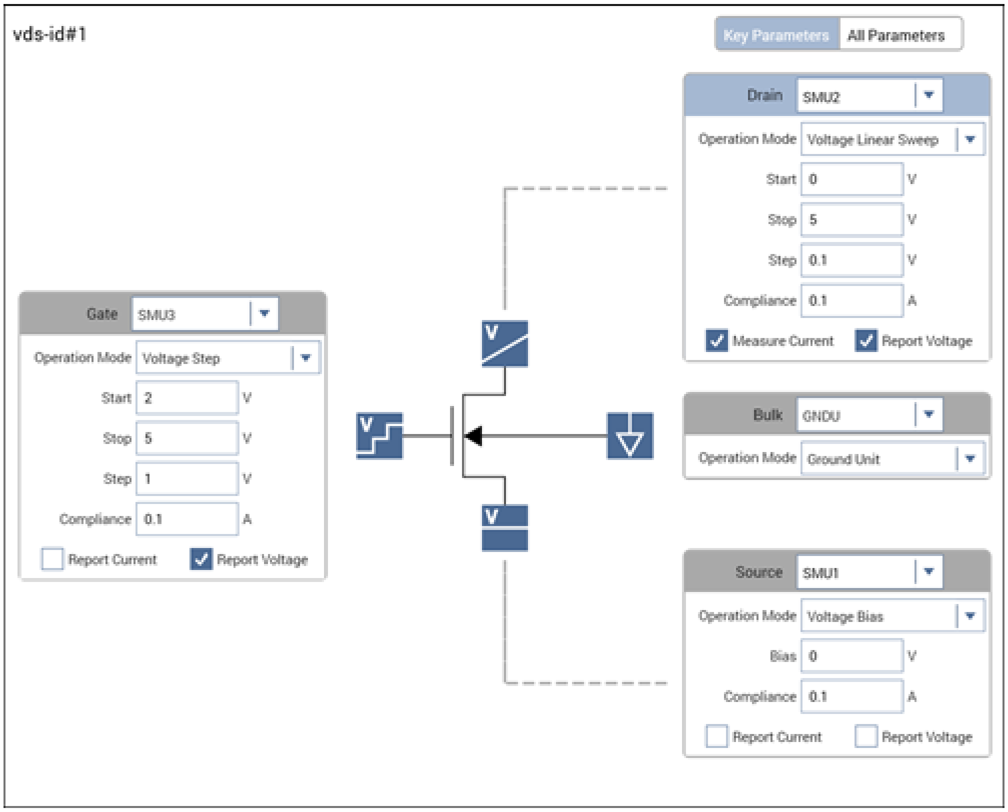
CAUTION

The connections selected in the Clarius software must accurately reflect the physical hardware connections when the test is executed. Incorrect terminal configurations can result in anomalous test results and device damage.

The key parameters for each terminal are displayed near the terminal. The key parameters include:

- The type of terminal, such as gate, drain, source, or collector.
- The instrument that is attached to the terminal. You assign the instrument, ground unit, or open circuit that is physically connected to the terminal during the test.
- The operation mode and basic settings for that mode. For example, the start and stop values are displayed if a sweep operation mode is selected.

Figure 11: Configure pane



Set the key parameters

The Key Parameters are the most commonly used parameters.

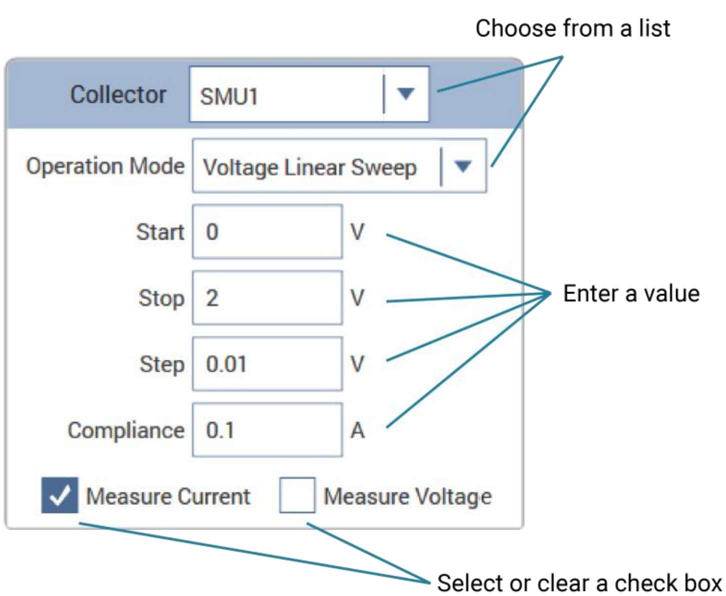
The parameters that are available depend on the instrument that is selected. For descriptions of parameters, refer to:

- "SMU - all parameters" in *Model 4200A-SCS Source-Measure Unit (SMU) User's Manual*
- "CVU - all parameters" in *Model 4200A-SCS Capacitance-Voltage Unit (CVU) User's Manual*
- "PMU - all parameters" in *Model 4200A-SCS Pulse Card (PGU and PMU) User's Manual*

To set the Key Parameters:

1. Select the field that you want to change.
2. If there is a:
   - **Down arrow to the right of the field:** Select a value from the list.
   - **Field:** Type the value. Error messages are displayed if you type an out-of-range value.
   - **Checkbox:** Select or clear the checkbox to enable or disable an option.

Figure 12: Clarius selection options



3. Select **Save**.

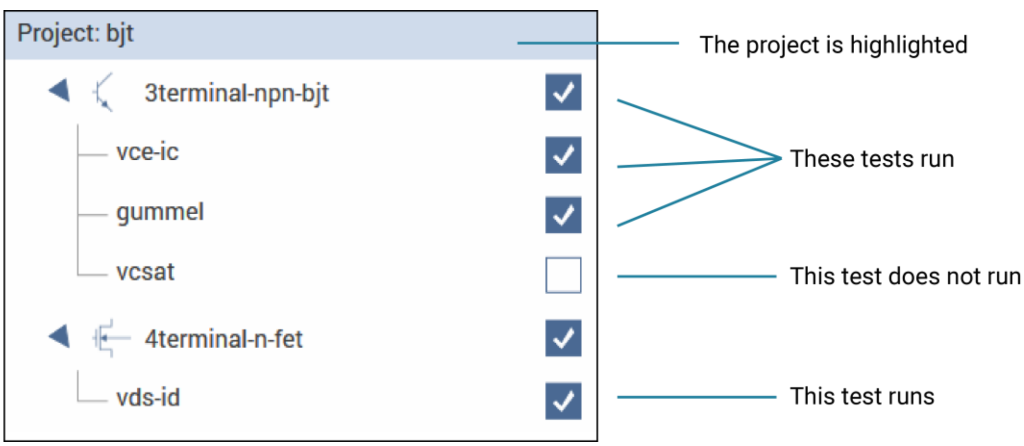
Run a simple test

When you select Run, selected tests and actions at a lower level than the highlighted item in the project tree are executed from top to bottom in the project tree. If you want to run an entire project, make sure the project name is highlighted. Running a project saves the configuration settings and the existing run history of the project.

In the following example, when you select **Run**, the following occurs:

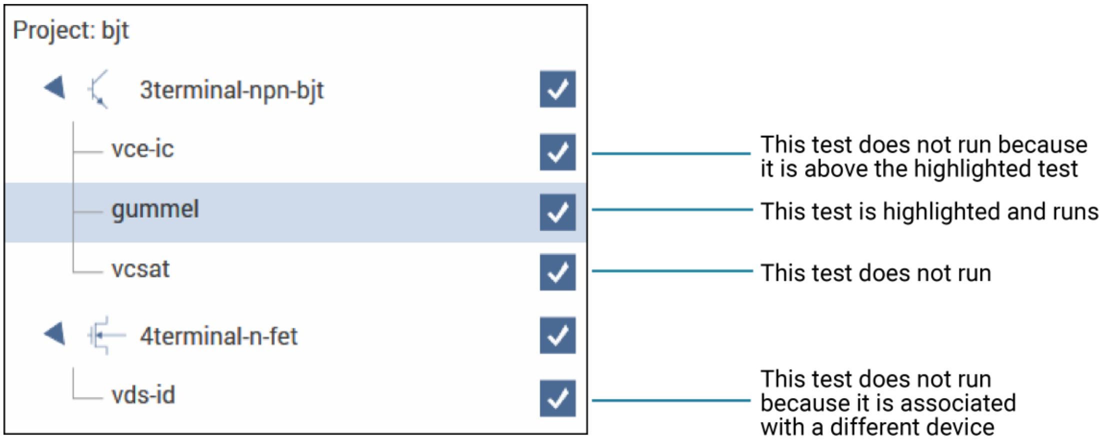
- The `vce-ic` test runs.
- The `gummel` test runs.
- The `vcsat` test is skipped.
- The `vds-id` test runs.

Figure 13: Run a test at the project level



In the following example, only the `gummel` test runs. Even though the other tests are selected, they are not below the `gummel` test in the hierarchy.

Figure 14: Run specific tests



To run a test in Clarius:

1. In the project tree, select that tests and actions that you want to run or execute.
2. Highlight the item where you want the test to start. For example, if you want to run the entire project, select the project.
3. Select **Run**.
4. Select **Analyze** to view the results.

NOTE

To abort a test, select **Stop**. All test and action execution stops immediately.

Working with the Projects dialog

The Projects option in the Clarius ribbon allows you to work with the projects you have created. Projects from all user accounts and any projects added to the project directory, which is defined in My Settings, are available.

You can use the Projects dialog to create new projects, import and export projects, and to copy, cut, paste, edit, delete, search for, and open projects.

NOTE

To change the project directory, see My Projects Directory (on page 4-7).

Open a project

Your projects are automatically added to the Projects dialog when they are added to the project tree. This procedure describes how to retrieve a project.

To open a project:

1. Choose **Projects** from the ribbon.
2. Type the project name in the Search box.
3. Select **Search**.
4. Select the project.
5. Select **Open Project** or double-click the project name. If the project in the project tree has unsaved changes, you are prompted to save the changes.

Edit project information

You can edit information that is displayed in the Project Library for a specific project. You can also edit the filters and keywords that are used.

To edit project information:

1. In Clarius, select **Projects** from the ribbon.
2. Select the project to be edited.
3. Select **Edit**. The Project Information Editor opens.
4. In the Basic tab, complete the information as needed. Refer to the following table for the options.
5. Select the **Filters** tab. These options set the filters that will cause this item to appear in the library when you select the right-pane filters.
6. Select the filters that help a user find this item in the library.
7. Select the **Keywords** tab. These options determine what you can type in the library Search field to locate this item. You can use the Sort By options at the bottom of the lists to change the order of the entries in the Information Editor. It does not affect the order in the library.
8. Drag a keyword from the left to the right to add a keyword.
9. To remove a keyword, select the keyword and select **Delete**. This does not remove the keyword from the Global Keywords list.
10. To add a keyword, select **New** and type the keyword.
11. Select **OK** to save the changes.

Options in the Information Editor	
Preview	Displays the changes you make as they will appear in the library.
Name	Type the new name. This is the name that is used in the library and the project tree.
Title	Type the title. This is used in the library.
Description	Type a brief description of the item. This is displayed in the library.
Author	Type information that identifies who created this item. This is available only through the Project Information Editor.
Help	Cannot be changed. This is the help file that is displayed in the right pane when Configure or Analyze is selected. It is also displayed when the item is selected in the library.
Library Image	The image that is displayed in the library. Select the image to choose a different image. Images should be 400×400 pixels in `png` format. Larger images display, but anything larger than 400×400 is cut off in the library display. To re-use an image from an older project, you may need to save the existing `bmp` image to `png` format. You can use a tool such as Microsoft™ Paint to convert the image. To leave the image area blank, select **Clear**.

Add notes to a project

You can add notes to a project in the Projects dialog.

You can copy content from another application, such as Microsoft Word, and text formatting will be maintained. You can include images and tables.

You can also use the following keyboard shortcuts to format text in the Project Notes dialog:

- **Ctrl + A:** Select all text or items in a document or webpage.
- **Ctrl + B:** Apply or remove bold formatting.
- **Ctrl + C:** Copy selected text or items to the clipboard.
- **Ctrl + I:** Apply or remove italics formatting.
- **Ctrl + U:** Apply or remove underlining.
- **Ctrl + V:** Paste copied text or items from the clipboard.
- **Ctrl + X:** Cut selected text or items to the clipboard.
- **Ctrl + Z:** Undo the last action.

Project Notes are stored with the project and are exported and imported. If you import a KITE project that includes notes, the notes are stored here.

To add notes to a project:

1. Choose **Projects** from the ribbon.
2. Select the project.
3. Select **Project Notes**.
4. Enter notes as needed.
5. Select **OK**.

Create a new project from the Projects dialog

You can create a new project from the Projects dialog. This is the same as creating a new empty project using the Project Library "New Project" option.

To create a new project:

1. Open **Projects** from the ribbon.
2. Select **New**. A confirmation message is displayed.
3. Select **Yes**. The new project opens in Clarius and the existing project is closed. The Projects dialog closes.
4. In the project tree, select **Rename** to assign a new name to the project.
5. Press **Enter** to accept the new name.

Export a project

You can export a project. An exported project can be imported into another 4200A-SCS.

The export includes all Run History data for each test in the project.

The exported file has the extension `.kzp`.

To export a project:

1. In Clarius, select **Projects** from the ribbon.
2. Select the project to be exported.
3. Select **Export**. The Export Project dialog opens.
4. Select the location for the exported file. You can right-click to create a new folder, rename an existing folder, or delete a folder.
5. If you are exporting data for import into a version of Clarius that is prior to version 1.12, select **Export run data in prior to Clarius 1.12 format**.
6. Select **Export**.

Import a project

You can import an exported project or project directory from another 4200A-SCS.

Exported projects have the extension `kzp`. Refer to Export a project (on page 2-12) for instructions.

Project directories have the extension `kpr`. If you are importing a project directory, you import the `kpr` file from that directory. The import includes all files from the project directory, assuming that the project directory is valid.

NOTE

Make sure that the files that you are importing are not set to read-only or run-only.

NOTE

If you are importing a project from a 4200-SCS, see Migrate projects from 4200-SCS systems (on page 2-17).

To import a project:

1. In Clarius, select **Projects**.
2. Select **Import**. The Import Project dialog opens.
3. Locate and select the exported project (`kzp`) or project directory (`kpr`).
4. Select **Import**. The Project Information Editor opens.
5. In the Basic tab, complete the information as needed. Refer to the following table for the options.
6. Select the **Filters** tab. These options set the filters that will cause this item to appear in the library when you select the right-pane filters.
7. Select the filters that help a user find this item in the library.
8. Select the **Keywords** tab. These options determine what you can type in the library Search field to locate this item. You can use the Sort By options at the bottom of the lists to change the order of the entries in the Information Editor. It does not affect the order in the library.
9. Drag a keyword from the left to the right to add a keyword.
10. To remove a keyword, select the keyword and select **Delete**. This does not remove the keyword from the Global Keywords list.
11. To add a keyword, select **New** and type the keyword.
12. Select **Add Project** to add the new object to the library.
13. To open the new project, select the project and select **Open Project**.

Options in the Information Editor	
Preview	Displays the changes you make as they will appear in the library.
Name	Type the new name. This is the name that is used in the library and the project tree.
Title	Type the title. This is used in the library.
Description	Type a brief description of the item. This is displayed in the library.
Author	Type information that identifies who created this item. This is available only through the Project Information Editor.
Help	Only editable when adding an object from the project tree to the library. This is the help file that is displayed in the right pane when Configure or Analyze is selected. It is also displayed when the item is selected in the library.
Library Image	The image that is displayed in the library. Select the image to choose a different image. Images should be 400×400 pixels in `png` format. Larger images display, but anything larger than 400×400 is cut off in the library display. To re-use an image from an older project, you may need to save the existing `bmp` image to `png` format. You can use a tool such as Microsoft™ Paint to convert the image. To leave the image area blank, select **Clear**.

Copy or cut a project

You can copy or cut a project. The new project does not maintain any links to the old project. Changes to the new project do not affect the old project and changes to the old project do not affect the new project.

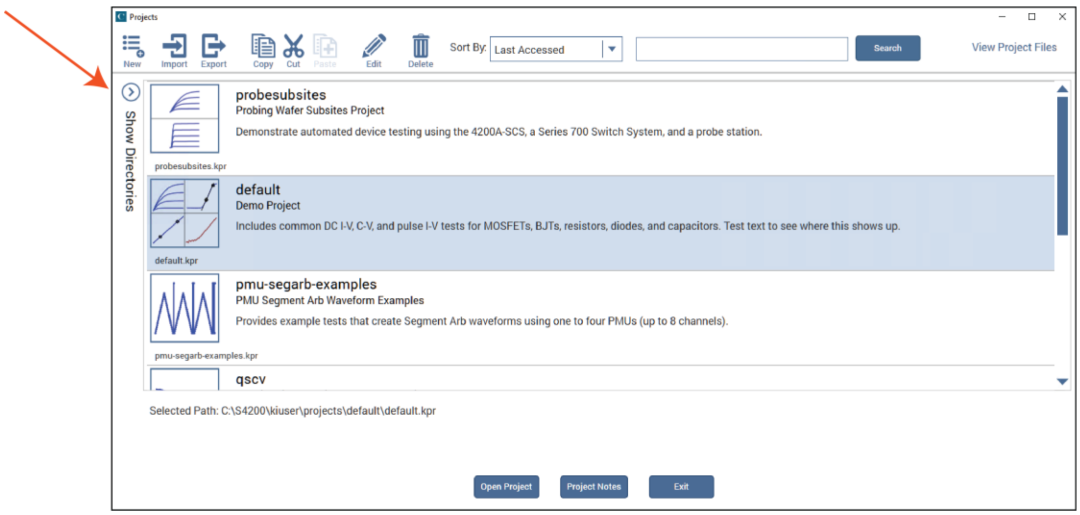
To copy a project:

1. Select **Projects** from the ribbon.
2. Select the project to be copied.
3. Select **Copy** or **Cut**.
4. Select **Paste**.
5. If you are copying a file, to:
   - Copy the project and load it as the active project: Select **Copy and Load**.
   - Copy the project only in Projects: Select **Copy Only**.

Show Directories

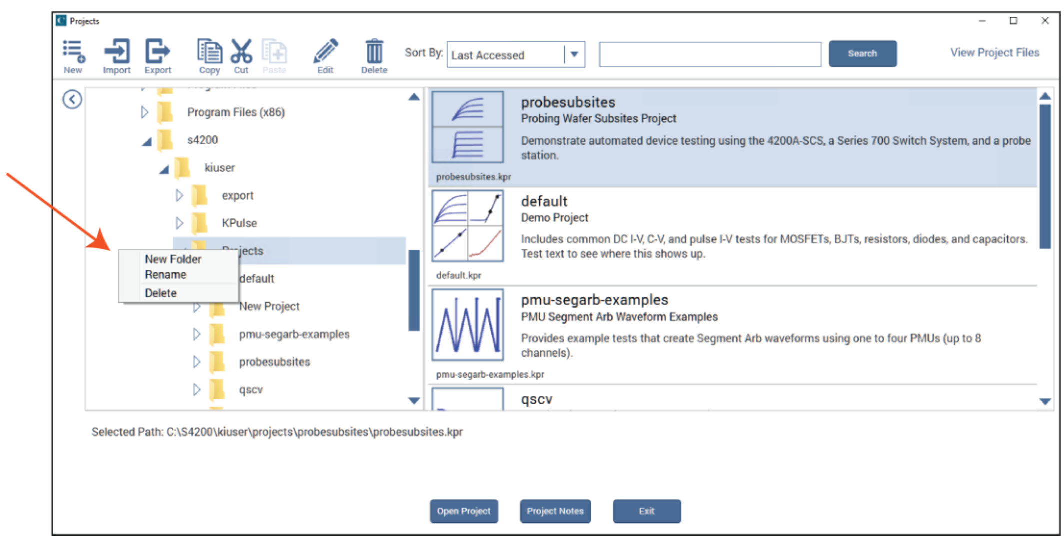
To display projects in a Microsoft File Explorer view, you can select **Show Directories** in the Projects dialog box.

Figure 15: Show Directories pane



When Show Directories is selected, you can right-click in the directory to add new folders, rename folders, or delete folders.

Figure 16: Show Directories open



Delete a project

CAUTION

Before deleting a project, ensure that you and others will not need it in the future.

When you delete a project, all files associated with the project in the project directory that is set in My Settings are also deleted. If the deleted project is open in the project tree, the project tree is cleared.

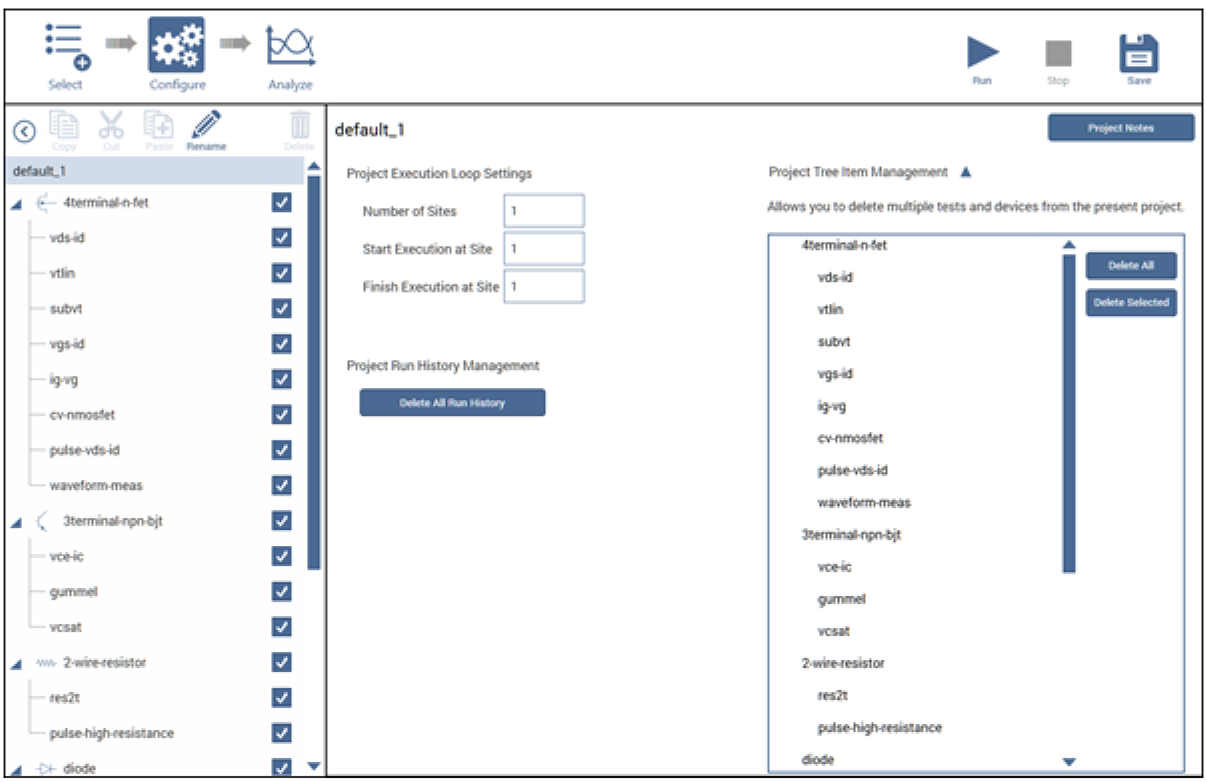
To delete a project:

1. Select **Projects**.
2. Select the project.
3. Select **Delete**. A confirmation message is displayed.
4. Select **Yes**.

Delete multiple objects from a project

If you need to delete multiple items from a project, you can use Project Tree Item Management. This option is available from Configure when a project or site is selected.

Figure 17: Project Tree Item Management



You can select all items under a device or site by selecting the device or site. You can select multiple items using Ctrl+click or a group of items using Shift+click.

If you delete all subsites from a site, the site is also deleted.

View Projects

You can change the view of the Projects window. To:

- **Change the sort order:** Select an option from the Sort By list.
- **Search for a specific project:** Type a keyword from the name or title of the project (descriptions are not searched) and select **Search** to display only the projects that match the keyword.

Migrate projects from 4200-SCS systems

You can use information from older 4200-SCS systems in the 4200A-SCS.

When you bring in information from the 4200-SCS, be aware:

- Copy all the files in the project directory for the project. Make sure the files in the project are kept together when you copy the files. By default, projects are stored in the `C:\s4200\kiuser\Projects` directory.
- If you used Segment Arb waveform files from KPulse in your projects, you need to manually copy and paste the waveform files from the 4200 to the 4200A-SCS. Segment Arb waveform files have the extension `.ksf` and are normally stored in the folder `C:\s4200\kiuser\KPulse\SarbFiles`.
- If your project contains user modules or user libraries that were created in KULT, those user modules are not included when you copy the project directory. See *Model 4200A-SCS Parameter Analyzer KULT and KULT Extension Programming*, “Copy user libraries and files” for instructions on how to import the user libraries and user modules.
- Make sure the files to be imported are not set to read-only or run-only.
- Initialization steps and termination steps will be converted to actions.
- Data that was appended in the 4200-SCS is stored as separate test runs in the Run History tab in the 4200A-SCS.

You cannot migrate from a 4200A-SCS to a 4200-SCS.

To migrate a project from a 4200-SCS system:

1. On the 4200-SCS system, copy the directory for the project you want to transfer.
2. On the 4200A-SCS, paste the project directory in `C:\s4200\kiuser\Projects`.
3. Open Clarius.
4. Select the **Projects** tab.
5. Verify that the project is available. If you sort by Last Accessed, the imported projects are displayed at the bottom of the project list.

Manage projects for multiple users

You cannot use multiple directories for the 4200A-SCS.

If you have multiple users that are using one 4200A-SCS, you can use options in the Project dialog and Library Information Editor to assign unique keywords to each project. These keywords can be used in the library and project search fields to locate your projects. For information on adding keywords through the Projects dialog, refer to Edit project information (on page 2-10). For information on adding keywords to projects that are added to the library, refer to Edit a library object you added (on page 4-3).

When adding projects, you can also assign project names that help you identify the project.

You can also use the import, export, and delete features in the Projects dialog to manage multiple users.

To use import, export, and delete to manage projects, each user will:

1. Create and use a project.
2. When work is complete, in Projects, export the project. Refer to Export a project (on page 2-12).
3. Delete the project from Projects. Refer to Delete a project (on page 2-16).
4. If you need to use the project again, import the project into the 4200A-SCS. Refer to Import a project (on page 2-13).

Set up a complex project

Set up a simple project (on page 2-1) describes how to set up a project with devices and tests for those devices. However, if your system includes wafers, external equipment, or custom tests, you need to add additional items to your project tree to accommodate them.

You can include the following operations and objects in the project:

- Custom tests or actions.
- Switch matrices to cycle electrical connections from the 4200A-SCS between the devices of a subsite. Refer to “Using Switching Matrices” in *Model 4200A-SCS Prober and External Instrument Control* for details.

This section describes how to add custom tests and actions to Clarius and to the project tree.

NOTE

For information on adding, duplicating, and importing projects, refer to Working with the Projects dialog (on page 2-9).

Customize tests

There are two types of tests in Clarius:

- **Interactive Test Modules (ITM):** Predefined tests that you can select and configure through the Clarius interface. They are used exclusively for parametric testing. You can create blank ITMs in Clarius that you can customize.
- **User Test Module (UTM):** A test that is based on a user module. Once the user module is incorporated into a test or action in Clarius, you can select and configure it in the Clarius interface. In addition to controlling tests, UTMs can control internal instrumentation or external instrumentation that is connected through the GPIB bus or RS-232 port. They can also be used for other tasks in the project, such as displaying prompts for test operators.

User modules are created in Keithley User Library Tool (KULT). Clarius^{+} comes with many predefined user modules, organized into user libraries. Refer to User library descriptions (on page 10-1) for descriptions of the pre-built user libraries and modules. You can also use KULT to create your own user modules or modify the source code for a module supplied by Keithley. Refer to *Model 4200A-SCS KULT and KULT Extension Programming* (4200A-KULT-907-01) for details.

Both ITMs and UTMs share common data analysis functions, such as the Analyze spreadsheet and graph.

You can customize tests in the following ways:

- Start with a predefined test and modify it.
- Start with a blank ITM test and modify it.
- Start with a blank UTM test, define the user module, and modify it.

After modifying a test, you can save it to the test library as a predefined test that can be used in other projects.

Modify a predefined test

You can modify an existing test that you added using the steps in Add a device and test to the project (on page 2-3).

Settings that you make to a test that is in the project tree are stored with the project. If you need to return to the settings of the test that is in the library, you can add the test from the library again.

Create a custom ITM

You can create a custom interactive test module (ITM) in Clarius. You do not need to create any external files (such as user modules) to create a custom ITM.

When you create a blank ITM, the number of terminals in the new test are determined by the type of device the test is placed under.

To create an ITM custom test:

1. Choose **Select**.
2. Highlight a device in the project tree or add a device.
3. If you need to add a device, open the Devices tab and select a device.
4. Select the **Tests** tab.
5. In the Test Library, select **Custom Test**.
6. Select **Add a blank test that can be configured into a DC, Pulse, or CV test (ITM)**.
7. Drag **Custom Test** to the project tree. The test has a red triangle next to it to indicate that it is not configured.
8. Select **Rename**.
9. Type a name for the test and press **Enter**.
10. Select **Configure** to set up the test.
11. Select the instrument.
12. Configure the options as needed.
13. For each device terminal, ensure that the physical device connections match the device connections defined in Clarius. If necessary, shut down the instrumentation and correct the physical connections.

CAUTION

Physical device-terminal connections must accurately match virtual connections to avoid inaccurate test results and potential device damage.

The options that are available depend on the instrument that is selected. For descriptions of parameters, refer to:

- “SMU Test Settings” in the *Model 4200A-SCS Source-Measure Unit (SMU) User's Manual*
- “CVU Test Settings” in the *Model 4200A-SCS Capacitance-Voltage Unit (CVU) User's Manual*
- “PMU Test Settings” in the *Model 4200A-SCS Pulse Card (PGU and PMU) User's Manual*

Create a custom UTM

User test modules (UTMs) are created from user modules. Many user modules are provided with the 4200A-SCS in the user libraries. You can also create your own user modules. For information on creating your own user modules, refer to *Model 4200A-SCS KULT and KULT Extension Programming* (4200A-KULT-907-01).

You can use one user module for multiple UTMs. Each instance of the user module is treated separately.

Data generated by a UTM is displayed in the Analyze sheet and graph.

NOTE

When you are building a project, it may be convenient to add all new UTMs first without immediately connecting them to user modules. This allows you to focus on project structure without being distracted with configuration details. To add a UTM without connecting it to a user modules, stop the following procedure after renaming the test.

To create a UTM:

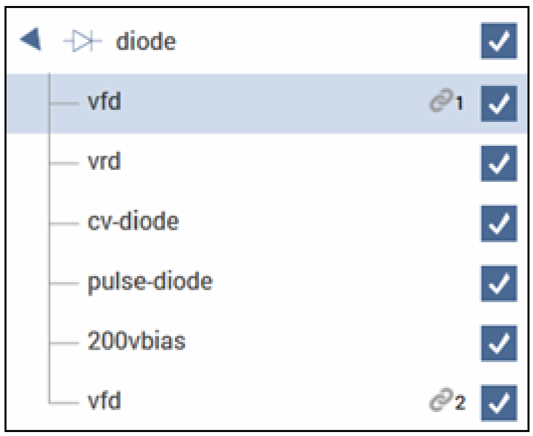
1. Choose **Select**.
2. Select the **Tests** tab.
3. For the Custom Test, select **Choose a test from the preprogrammed library (UTM)**.
4. Drag **Custom Test** to the project tree. The test has a red triangle next to it to indicate that it is not configured.
5. Select **Rename**.
6. Type a name for the test.
7. Select **Configure**.
8. In the right pane, from the User Libraries list, select the user library that contains the user module that contains the test.
9. From the User Modules list, select the user module.
10. Enter the parameters in the Configure pane. Refer to the Help pane for descriptions of the options in the UTM.
11. You can use the Formulator to do calculations on the test results. See Formulator (on page 8-1) for additional information.
12. If you are working with a subsite, select **Output Values** to specify output values to export into the subsite datasheet.
13. You can edit the user interface (UI) of the UTM. Refer to Define the user interface for a user test module (on page 7-1) for instruction.

Link tests or actions

You can use the Linked Copy option to insert multiple instances of test or action in the project tree. When ITMs are linked, Clarius automatically keeps the configurations of the linked ITMs identical. This allows you to have multiple tests that perform identically. When UTMs or actions are linked, the user libraries, user modules, Formulator formulas and constants are identical. Note that parameter settings are not identical for linked UTMs or actions.

When tests are linked, the tests in the project tree display a chain-link icon, as shown in the following figure.

Figure 18: Linked ITMs in the project tree



When using linked copies:

- For ITMs, the settings in the Configure pane (including Formulator formulas and Output Values) are kept in synchronization. When you change a setting in once instance of a linked test, all the linked tests are changed.
- For UTMs and actions, the user libraries, user modules, Formulator formulas and constants, and Output Values are identical between the tests. Parameter settings can be different.
- The data for each linked test remains independent. The Analyze sheet, graph, and graph settings for each linked test are different.

If you migrated projects from a 4200, project components that were linked with Unique IDs (UIDs) are converted to linked tests or actions in Clarius.

To insert linked tests:

1. Add a test to the project tree (refer to Add a device and test to the project (on page 2-3) or Create a custom test (on page 2-21)).
2. Right-click the test and select **Linked Copy**.
3. Select an item in the project tree that you want the test to follow. The item is highlighted in green if this is a valid place to copy the test. It is red if you cannot copy the test to this location.
4. Select **Paste**. The linked test is added to the project tree.

NOTE

You can create multiple linked copies of tests by using Linked Copy at the device or subsite level. In this case, all ITMs associated with the device are copied with the new device and become linked copies of the tests in the original device. UTMs are copied as independent tests or actions. Note that the devices and subsites do not become linked copies, only the ITMs.

Add actions

Actions allow you to move probers, add user notifications such as beepers and dialog boxes, and change switching options. You can add existing actions or create actions based on user modules. When you create an action, you select a user module from a user library to create the action. Clarius supplies user libraries, or you can create your own using KULT.

NOTE

If you are moving from 4200 KITE to 4200A Clarius, actions replace initialization and termination steps. Actions are more versatile than initialization and termination steps. You can place them in the project wherever they are needed instead of being limited to the top and bottom of the project.

To add an action to the project tree, drag it into the tree where the action needs to occur during the test. For example, if you need to sound a beep after a specific test, drag the `Beeper` action to the project tree under that test.

To create an action:

1. Choose **Select**.
2. Select the **Actions** tab.
3. Drag **Custom Action** to the project tree. The action has a red triangle next to it to indicate that it is not configured.
4. Select **Rename**.
5. Enter a name for the action.
6. Select **Configure**.
7. In the Test Settings pane, select the user library.
8. Select the user module.
9. Set other settings as needed. Refer to the Help pane for information.

Example: Creating a project

This section provides an example of how to create a new blank project and configure a new blank test. You will create a test to be performed on a MOSFET, but the procedure is general and can be applied to different devices and applications.

NOTE

The default settings used for the devices, tests, and projects in Clarius are generally sufficient to produce usable data when executing a test. However, you may have additional settings you want to apply when you configure your measurements.

Equipment required

- One 4200A-SCS, with the following instruments:
  - Two medium power (4200-SMU or 4201-SMU) or high power (4210-SMU or 4211-SMU) SMUs
  - Two 4200-PAs
- Three 4200-TRX-2 or 4200-MTRX-2 triaxial cables (supplied with SMU)
- One shielded, three-terminal test fixture with triaxial inputs (such as the 8101-PIV)

Device connections

Using the supplied cables, connect the output terminals of the instruments directly to the MOSFET terminals in the shielded test fixture. The triaxial terminals on the shielded test fixture allow you to connect to the device and maintain a completely shielded and guarded test setup.

WARNING

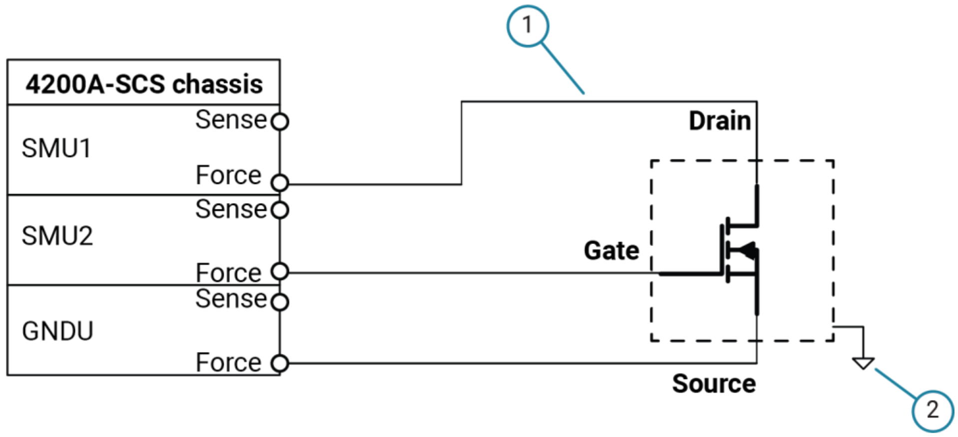
Hazardous voltages may be present on all output and guard terminals. To prevent electrical shock that could cause injury or death, never connect or disconnect from the 4200A-SCS while the output is on. Turn off the equipment from the front panel or disconnect the main power cord from the rear of the 4200A-SCS before handling cables. Putting the equipment into an output-off state does not guarantee that the outputs are powered off if a hardware or software fault occurs.

To prevent electric shock, test connections must be configured such that the user cannot come in contact with test leads, conductors, or any device under test (DUT) that is in contact with the conductors. It is good practice to disconnect DUTs from the instrument before powering up the instrument. Safe installation requires proper shields, barriers, and grounding to prevent contact with test leads and conductors.

Connection schematic

The hardware connections from the output of the instruments in the 4200A-SCS chassis to the test fixture that contains the MOSFET are shown in the following figure. All the connections are 2-wire and only the Force terminal of each SMU is used. The SMUs and GNDU are each connected to a different terminal of the 3-terminal MOSFET.

Figure 19: Connections from the 4200A-SCS to a MOSFET

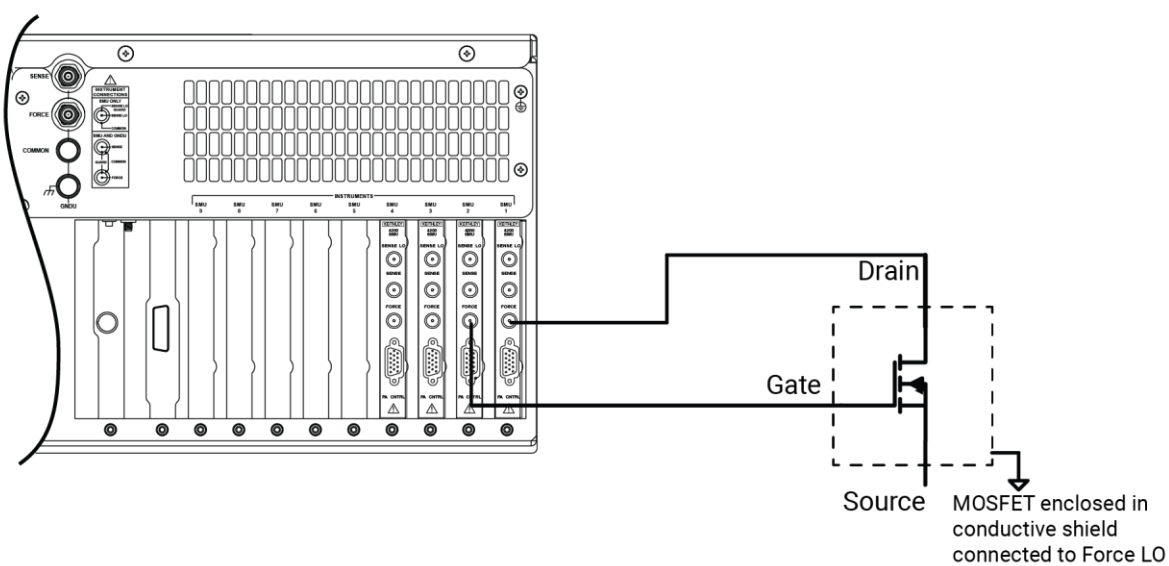


1	4200-TRX-X or 4200-MTRX-X triaxial-to-triaxial cables.
2	MOSFET enclosed in conductive shield connected to FORCE LO.

Connect the 4200A-SCS to the DUT

The hardware connections from the output of the instruments in the 4200A-SCS chassis to the test fixture that contains the MOSFET are shown in the following figure.

Figure 20: Rear-panel connections from the 4200A-SCS to a MOSFET



Set up the measurements in Clarius

This section describes how to set up the 4200A-SCS to generate a V_{ds}-I_d family of curves for a 3-terminal n-type MOSFET. This general procedure can also be used to create tests for other devices and other applications.

For this example, you will use the Clarius application to:

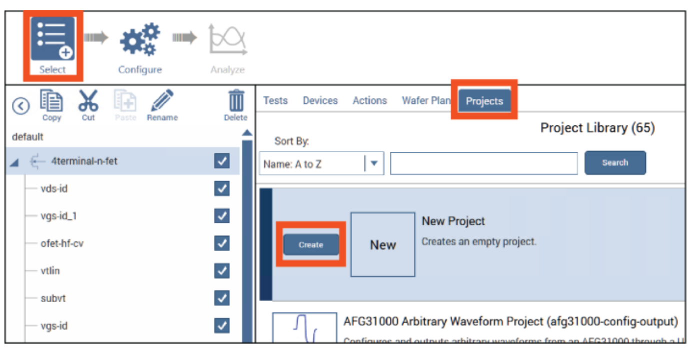
- Select and rename a new project
- Add a device
- Select a custom test
- Configure the test
- Execute the test
- View and analyze the test results

Select and rename a new project

To select and rename a new project:

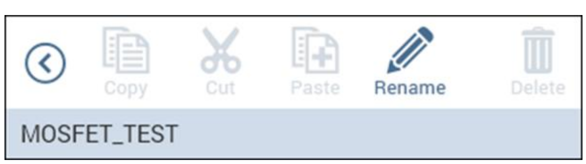
1. Choose **Select**.
2. In the Library, select **Projects**.
3. Select **New Project**.
4. Select **Create**.

Figure 21: Create new project



5. Select **Yes** when prompted to replace the existing project.
6. Assign a title to the project by selecting **Rename** above the project tree.
7. Enter a project name into the text box, then select **Enter**. `MOSFET_TEST` has been chosen for this example.

Figure 22: Toolbar with Rename function



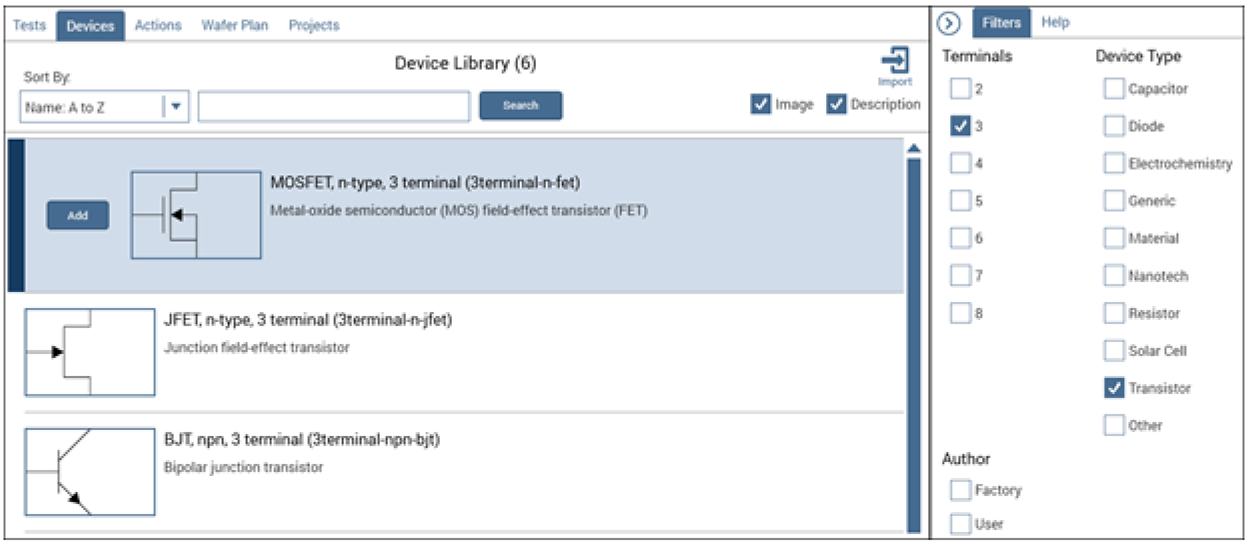
Add a device

Tests must be placed in the project under a device.

To add a device:

1. Select **Devices**.
2. From the Filters pane, select the **3** under the Terminals heading and **Transistor** under the Device Type option.

Figure 23: Searching for a device using Filters



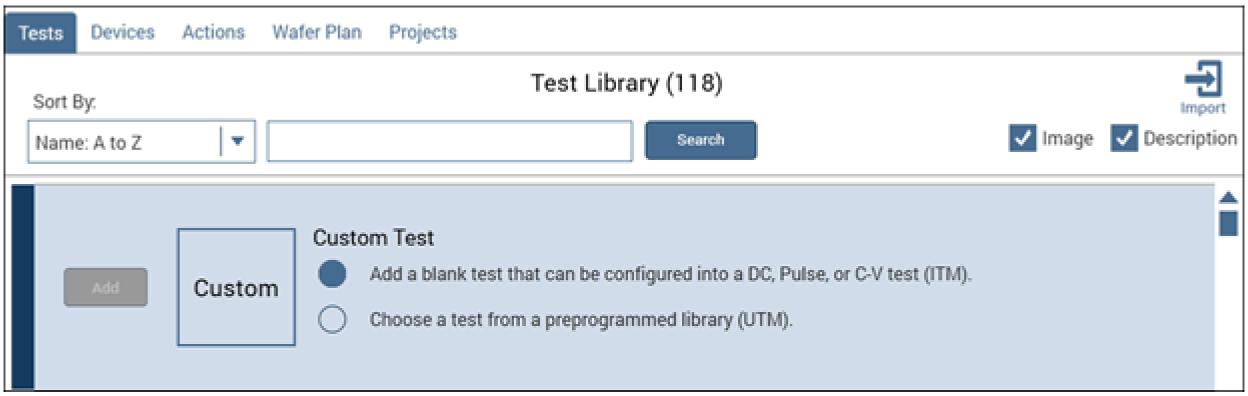
3. Select the `MOSFET, n-type, 3 terminal (3terminal-n-fet)` device.
4. Select **Add** to copy it to the project tree.

Select a custom test

To select a custom test:

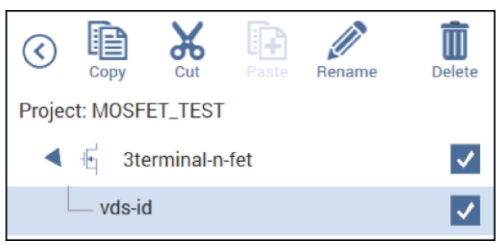
1. Select **Tests**.
2. Select **Custom Test**, then select **Add** to create a new 3-terminal, n-type MOSFET test in the project tree.

Figure 24: Custom Test option



3. Select **Rename** from the toolbar. Enter a test name in the text box, then select **Enter**. `vds-id` was chosen for this example.

Figure 25: MOSFET_TEST project tree with one device and one test

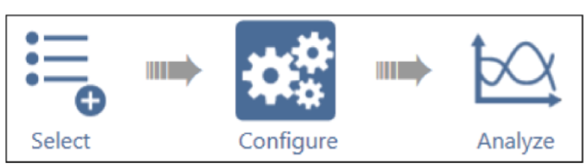


Configure the test

To configure the test:

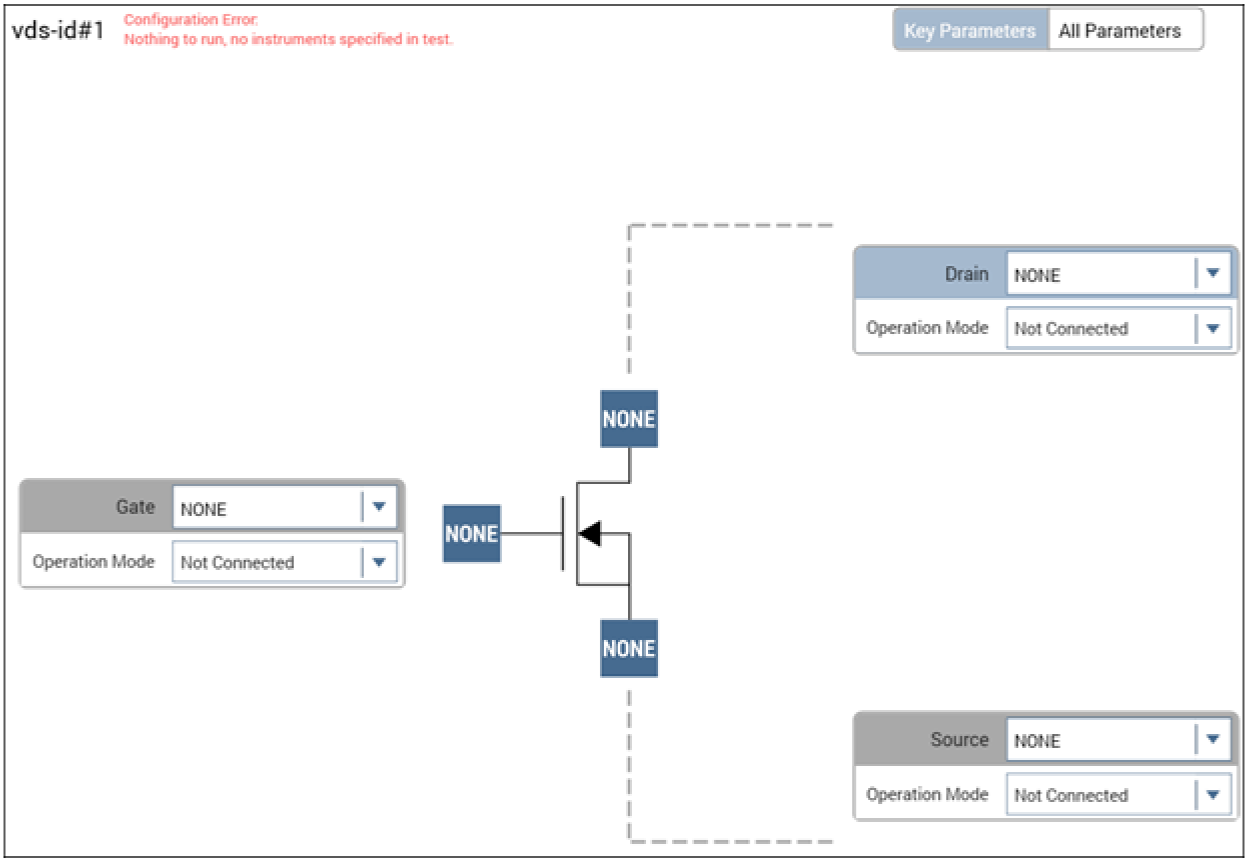
1. Select **Configure**.

Figure 26: Configure highlighted



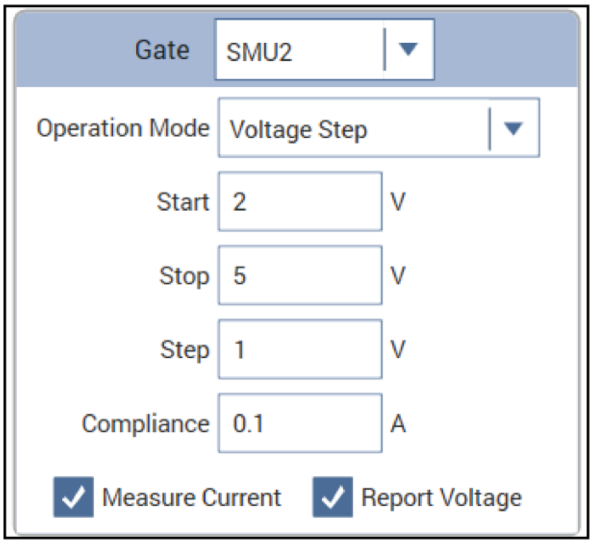
2. In the project tree, select `vds-id`. Because this test is custom, you must assign functions to all terminals connected to the MOSFET before you can run the test.

Figure 27: All MOSFET terminals unassigned in a custom test



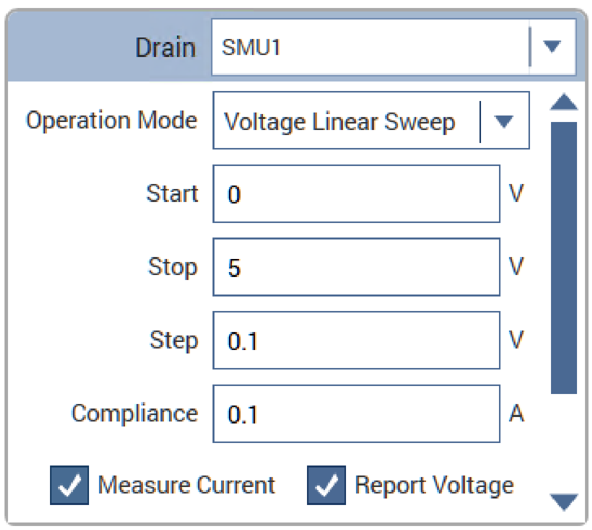
3. Set the Gate terminal connection to **SMU2**.
4. Set the Operation Mode to **Voltage Step**.
5. Change the Start, Stop, Step, and Compliance settings to match the following figure or to the gate settings appropriate for your device.

Figure 28: SMU2 steps from 2 V to 5 V, connected to MOSFET Gate terminal



6. Set the Drain terminal connection to **SMU1**.
7. Set the Operation Mode to **Voltage Linear Sweep**.
8. Change the Start, Stop, Step, and Compliance settings to match the following figure.

Figure 29: SMU1 sweeps from 0 V to 5 V, connected to MOSFET Drain terminal



9. Set the Operation Mode of the Source terminal to **GNDU**.

Execute the test

Select **Run** to execute the test.

Figure 30: Run

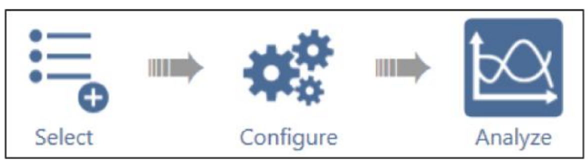
View and analyze the test results

While the test is running, you can view the data in the spreadsheet of the Analyze pane. Because you created a new test, the data must be assigned to the axes of the graph before you can view graphical results.

To view and analyze the test results:

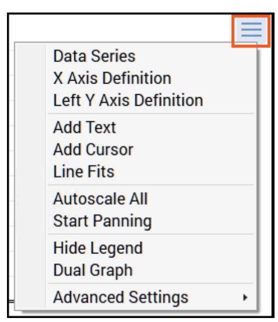
1. Select **Analyze**. The Analyze screen displays data as it is gathered in the spreadsheet and a blank graph with unassigned axes.

Figure 31: Analyze highlighted



2. Select the **Graph Definition Menu**.

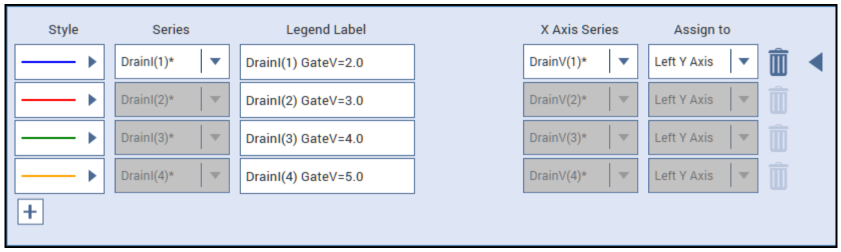
Figure 32: Graph Definition Menu



3. Select **Data Series**.

4. Select **+**. The series are assigned.

Figure 33: Data series selected



5. Select ◄ to close the dialog. The graph displays the `vds-id` family of curves.

NOTE

For more information on setting up a graph, refer to Analyze test data for projects (on page 3-15).

Configure a complex test

For more complex tests, you can:

- Set advanced test and terminal parameters.
- Set up multiple steps or sweeps to track simultaneously using masters and subordinates.
- Configure actions.

Test and terminal settings

You can access test and terminal settings from the center and right panes of Clarius when Configure is selected.

The most common terminal settings are available in the center pane when Key Parameters is selected. Additional common test settings are available in the right Terminal Settings pane.

Less commonly used terminal settings are available in a dialog that you open with the Advanced button on the Terminal Settings pane.

To view all terminal settings, select All Parameters from the center pane. In this view, all terminal settings are displayed for every terminal.

Test settings are also displayed in the right pane. Test settings affect all terminals in the selected test. The most common settings are available in the right pane. Additional settings are available when you select the Advanced button.

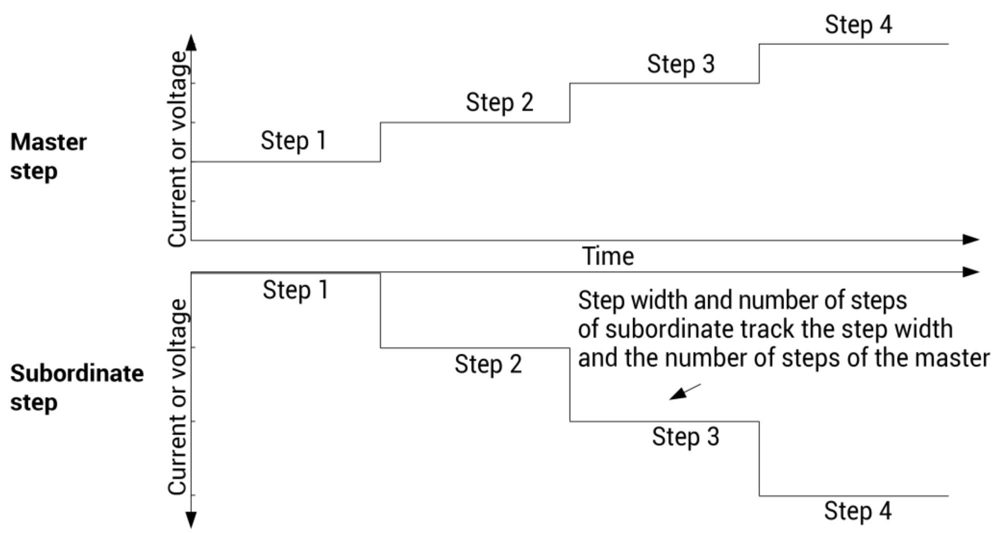
The options that are available depend on the instrument that is selected. For descriptions of parameters, refer to:

- "SMU Test Settings" in the *Model 4200A-SCS Source-Measure Unit (SMU) User's Manual*
- "CVU Test Settings" in the *Model 4200A-SCS Capacitance-Voltage Unit (CVU) User's Manual*
- "PMU Test Settings" in the *Model 4200A-SCS Pulse Card (PGU and PMU) User's Manual*

Step or sweep multiple device terminals in the same test

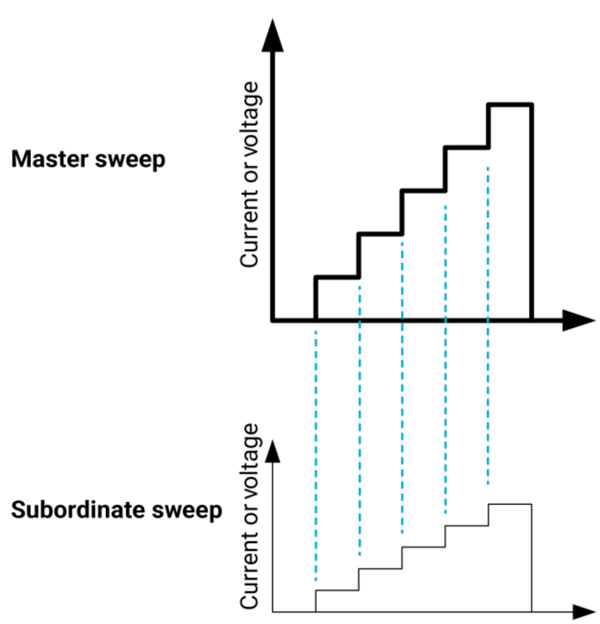
Multiple steps or sweeps in an interactive test module (ITM) must track with regard to step number and duration. For example, you might want to apply multiple steps to multiple device terminals, such as when stepping the biases on two transistor terminals and sweeping voltage or current on the third terminal. In this setup, Clarius automatically sets the step operations to occur simultaneously, with one terminal set up as the master. All other step functions are automatically designated as subordinates. The following figure illustrates this concept.

Figure 34: Master step versus subordinate step



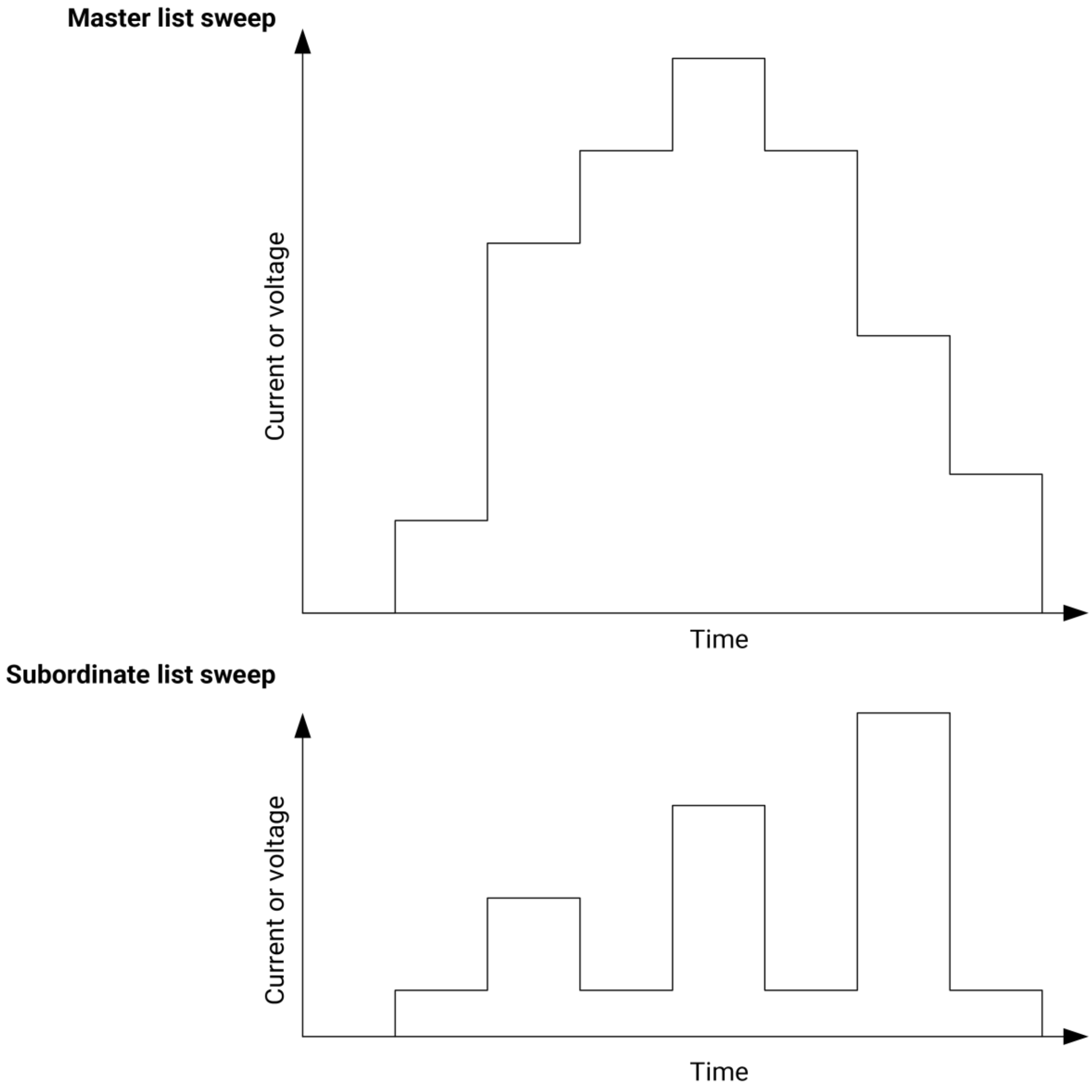
In the following figure, the step width and the number of steps of the subordinate track the step width and number of steps of the master sweep.

Figure 35: Master and subordinate sweeps



In the following figure, the step width and number of steps of the subordinate list sweep tracks the step width and the number of steps of the master list sweep.

Figure 36: Master list sweeps versus subordinate list sweeps



If you do not specify an instrument to be the master, the first instrument that was assigned to the step or sweep operation mode is assigned to be the master. You can change this designation in the Test Settings pane.

When a master is set, Clarius sets the points and step size values for the subordinate terminal to be the same as the settings for the master terminal. You cannot change the subordinate points value for the subordinate terminal. For list sweeps, the number of points in the list items for the subordinate is changed to match the number of points in the master. If points are added to the master, the last point of the subordinate list is repeated. If points are removed from the master, the same number of points are removed from the end of the subordinate list.

You can have a dual sweep on a subordinate terminal even if the master is not set to dual sweep. In this case, the dual sweep of the subordinate terminal has a total number of steps equal to the number of steps in the master terminal. For example, if the master terminal is set to measure ten points, the subordinate dual sweep will measure five points on the first side of the sweep and five points on the second side of the dual sweep. If the master is set to an odd number of points, the subordinate terminal will repeat the measurement of the last sweep point.

The subordinate SMUs are not automatically set for dual sweep when Dual Sweep is enabled for the master SMU. Dual Sweep must be individually enabled for each SMU.

To specify the master sweep:

1. Select the test.
2. In the right pane, select **Test Settings**.
3. For **Sweep Master**, select the instrument that you want to designate as the master.

Configure actions

Settings for actions depend on the type of action that is selected. Actions can generate dialog boxes to prompt test operator action, control prober movements, and manage switching. You can also create your own actions from user libraries.

You can place actions anywhere in the project tree.

Information for actions is available in the Help pane. To open the Help pane, select the action in the project tree and select **Configure**. Select the Help tab in the right pane.

Run a complex test

This section describes how to run individual tests, devices, subsites, and sites. It also describes how to run the stress modes.

NOTE

If Clarius detects an above-normal temperature condition at any SMU, it protects system outputs by preventing or aborting a test run and reporting the condition in the message area of the Clarius window. If the condition occurs when a test is attempted, Clarius prohibits execution. If the condition occurs during a test, Clarius aborts the test.

Run devices and tests

You can execute an entire project or individual parts of the project.

While a test is running, you can view test data in the Analyze pane.

The Message area at the bottom of the center pane of Clarius displays the start time, stop time, and total execution time of the components that were run.

Run projects

Executing an entire project runs its components, including sites, subsites, devices, tests, and actions, in the order in which they appear in the project tree.

Clarius only runs items that are selected in the project tree and are at a lower level than the highlighted item.

When you run a project, each new run creates a new test run in the Run History tab, with its own Analyze spreadsheet. For more about the spreadsheets, refer to Analyze data (on page 3-1).

NOTE

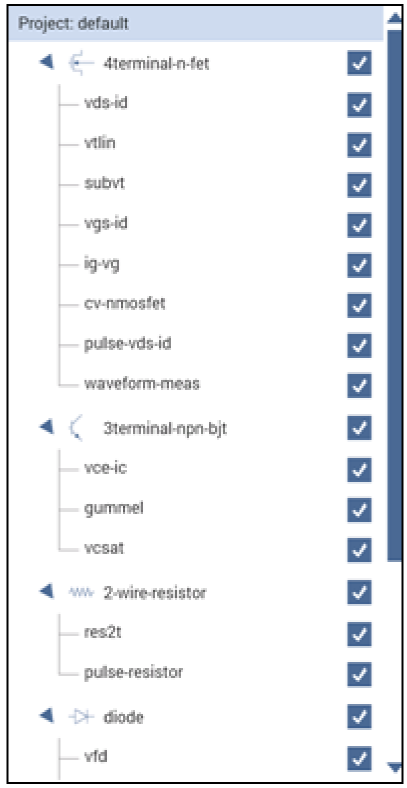
To abort a test, select **Stop**. All test and action execution stops immediately.

The following example uses the Demo Project to demonstrate how to run a project.

To run all objects in a project:

1. Open the **Demo Project**. Refer to Open a project (on page 2-10).
2. Make sure the check boxes are selected for all items in the project tree.
3. Highlight the project name.

Figure 37: Run a project



4. Select **Run**. The Run icon changes as shown below. The active test is listed to the left of Run. The Stop icon changes to red.

Figure 38: Run icon while a test is running

When the test completes, a beep sounds and the run arrows around the Run icon are no longer displayed.

Run individual devices

You can run the tests for a single device in the project tree.

When you run the tests for a single device, the tests are run in the order in which they appear in the project tree. Only the tests that have checkboxes selected are run. In the following example, all the tests for the diode device are run except for `vrd`.

NOTE

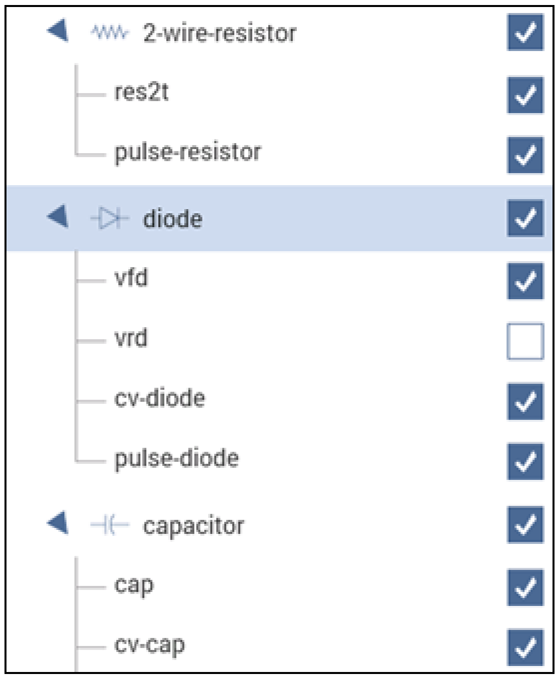
To abort a test, select **Stop**. All test and action execution stops immediately.

The following example uses the Demo Project to demonstrate how to run tests for a device.

To run tests for a device:

1. Open the **Demo Project**. Refer to Open a project (on page 2-10).
2. Make sure the check boxes are selected for all items under the `diode` device except for `vrd`, as shown in the following figure.
3. Highlight the device name, **diode**.

Figure 39: Run tests for a single device



4. Select **Run**. The Run icon changes as shown in the following figure. The active test is listed to the left of Run. The Stop icon changes to red.

Figure 40: Run icon while a test is running

When the test completes, a beep sounds and the run arrows around the Run icon are no longer displayed.

Run an individual test

NOTE

To abort a test, select **Stop**. All test and action execution stops immediately.

To run an individual test in the project tree:

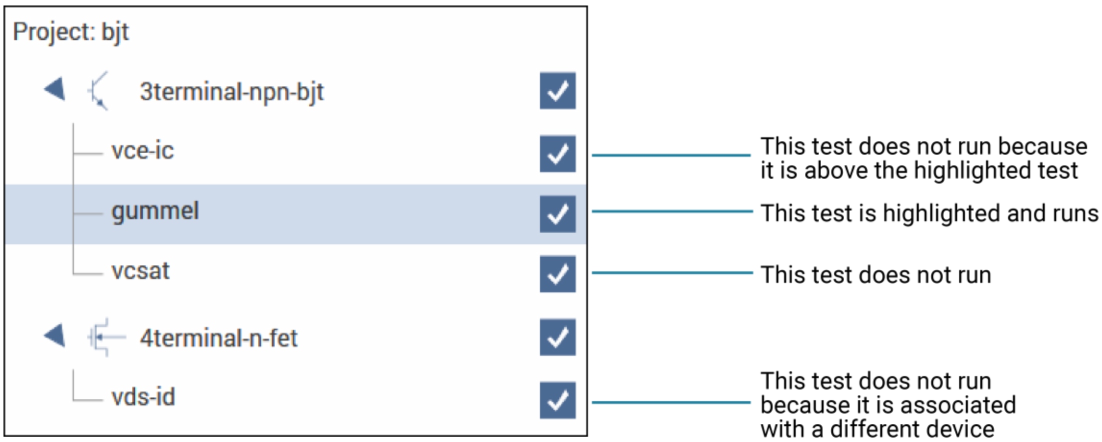
1. In the project tree, make sure the checkbox for the test is selected.
2. Highlight the test.
3. Select **Run**. The Run icon changes as shown below. The active test is listed to the left of Run. The Stop icon changes to red.

Figure 41: Run icon while a test is running

When the test completes, a beep sounds and the run arrows around the Run icon are no longer displayed.

In the following example, only the `gummel` test runs. Even though the other tests are selected, they are at the same level as the `gummel` test in the hierarchy.

Figure 42: Run specific tests



NOTE

You can also use Monitor to run an individual test. For more information, see Monitor a test (on page 2-40).

Monitor a test

You can use the Monitor option to set a test to run continuously until stopped. This option is available for ITMs and UTMs when an individual test is selected and when the Monitor option is enabled.

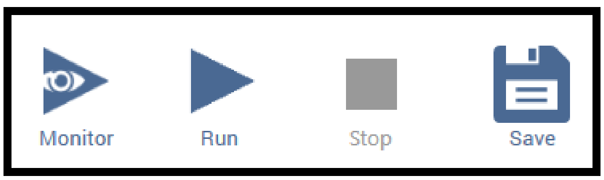
Enable the Monitor option

The Monitor option is enabled in My Settings.

To enable the Monitor option:

1. Select **My Settings**.
2. Select **Run Settings**.
3. Select **Enable Monitor button**.
4. Select **OK**. Monitor is now available to the left of Run.

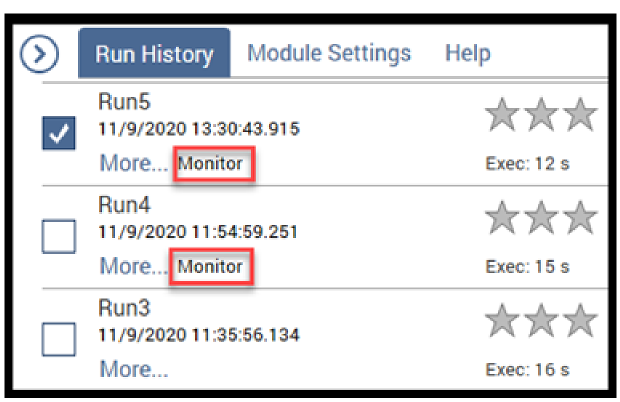
Figure 43: Monitor option



Running a test using Monitor

When a test is run using Monitor, the test loops until you stop it. Only one test run is stored in Run History. The stored run contains the data that was acquired during the last complete test run. The time that is shown for the run is the time of the last completed test run, not the entire time that Monitor was running. Runs that were generated with Monitor include "Monitor" in the description in the Run History tab, as shown in the following figure.

Figure 44: Run History when test is run using Monitor



To use Monitor to run a test:

1. In the project tree, make sure the checkbox for the test is selected.
2. Highlight the test.
3. Select **Monitor**. The Monitor icon changes as shown below. The active test is listed to the left of Monitor, followed by the number of the loop that is running. The Stop icon changes to red.

Figure 45: Monitor when a test is running

4. To stop the test, select **Stop**.

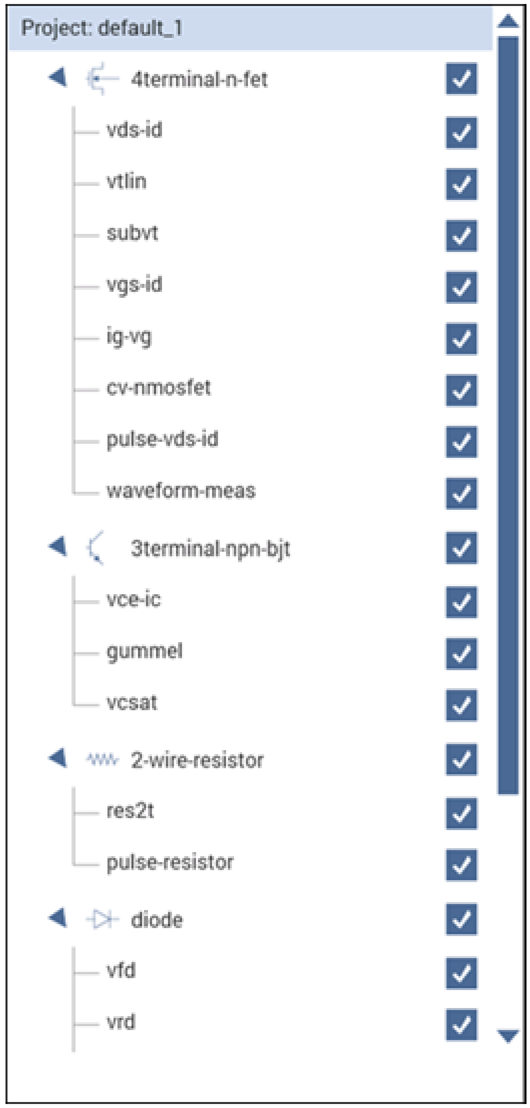
Demo Project overview

The Demo Project includes common DC I-V, C-V, and pulse I-V tests for MOSFETs, BJTs, resistors, diodes, and capacitors. These tests serve as examples and are intended to be copied and modified to work for your own devices. All test parameters in the Demo Project were written for standard discrete parts but can be modified for use with other discrete devices or devices on a semiconductor wafer. These tests demonstrate how to configure tests in the Configure pane, how to use Formulator functions for common mathematical calculations and return them to the data sheet, and how to plot the data in a variety of ways.

The Demo project opens when you first start the Clarius application.

The top portion of the project tree for the Demo project is shown in the following graphic.

Figure 46: Demo project (default) in the project tree



The following topics provide brief descriptions of the tests in the Demo project.

4-terminal n-MOSFET tests

By default, the following tests use three source-measure units (SMUs) and one ground unit (GNDU). You can also use four SMUs, one for each device-under-test (DUT) terminal.

Descriptions of the 4-terminal n-MOSFET tests

Test	Description
`vds-id`	This test generates the standard family of drain current versus drain voltage curves on a FET. For each gate voltage step, the test sweeps the drain voltage and measures the resulting drain current. This test uses either three or four SMUs that are connected to the gate, drain, source, and bulk terminals of the FET.
`vtlin`	Uses a linear curve fit to find the threshold voltage (V_t) of a MOSFET from the generated drain current versus gate voltage data. This test uses three or four SMUs connected to the gate, drain, source, and bulk terminals of the MOSFET.
`subvt`	Executes an I-V sweep and calculates the subthreshold voltage (sub-V_t) of a MOSFET and plots drain current versus gate voltage. This test uses three or four SMUs connected to the gate, drain, source, and bulk terminals of the MOSFET.
`vgs-id`	This test extracts the threshold voltage (V_t) and maximum transconductance (G_m) parameters from a sweep of the drain current versus gate voltage. This test uses either three or four SMUs connected to the gate, drain, source and bulk terminals of a MOSFET.
`ig-vg`	Measures the gate leakage current of the MOSFET as a function of the sweeping gate voltage. The test determines the gate leakage resistance using a linear line fit.
`cv-nmosfet`	Measures the capacitance as a function of the gate voltage between the gate terminal and the drain, source, and bulk terminals tied together. Several parameters are extracted, including the flatband capacitance, doping density, flatband voltage, and oxide thickness.
`pulse-vds-id`	Uses CH1 and CH2 of a PMU to generate a pulse I-V drain family of curves. CH1 outputs a pulse step output to the gate. CH2 outputs a pulsed drain voltage sweep and measures the drain current.
`waveform-meas`	Uses the waveform capture mode of the PMU to show the time-based response of the drain current and drain voltage of a MOSFET. CH1 outputs a single pulse to the gate. CH2 captures the transient response of the drain current and drain voltage.

3-terminal NPN BJT tests

The tests for this device require three SMUs.

Descriptions of the 3-terminal NPN BJT tests

Test	Description
`vce-ic`	As the base current is stepped, this test measures the drain current at each point of the drain voltage sweep. Three SMUs are connected to the base, collector, and emitter terminals of the BJT.
`gummel`	Generates a Gummel plot as it measures both the base current and collector current of a BJT. The currents are measured as a function of the base-emitter voltage. Three SMUs are connected to the base, collector, and emitter terminals of a BJT.
`vcsat`	At a constant base current, this test plots the collector current as a function of the collector voltage. The data is used by the Formulator to calculate the collector saturation voltage ($V_{ce(sat)}$). Three SMUs are connected to the base, collector, and emitter terminals of a BJT.

Resistor tests

The following tests use two SMUs. It is also possible to use one SMU and the GNDU.

Descriptions of the 2-wire resistor tests

Test	Description
`res2t`	Calculates the average resistance from an I-V sweep of sourcing voltage and measuring current. This test uses two SMUs on either side of the resistor. You can also use one SMU and GNDU.
`pulse-resistor`	The resistor is connected between one channel of the PMU and the PMU ground. Applies a pulse voltage sweep to the resistor during the test while the current is measured.
`pulse-high-resistance`	The resistor is connected between CH 1 and CH 2 of the PMU (RPMs). CH1 applies a pulse sweep to the resistor during the test while the current is measured by CH2. The current is derived by averaging four pulses. The Formulator multiples the current of CH2 by −1.

Diode tests

By default, the following tests use up to two source-measure units (SMUs), a CVU, and a PMU.

Descriptions of the diode tests

Test	Description
`vfd`	While a forward-bias voltage sweep is applied to a diode, this test measures the anode current and plots the data on a semi-log scale. A linear line fit is used to derive the slope of the line.
`vrd`	Sweeps the reverse bias voltage and measures the resulting current of a diode. This test uses two SMUs on either side of the diode. You can also use one SMU and GNDU if they are set properly in the Configure pane. For very low current measurements, it is recommended that you use a SMU with the preamp option.
`cv-diode`	Measures the junction capacitance as a function of an applied voltage sweep. The depletion depth (W) and the doping density (N) are calculated as a function of the C-V data.
`pulse-diode`	Applies a pulse voltage sweep to the anode of a diode and measures the resulting anode current. A single channel of the PMU is used to make the measurement.

Capacitor tests

By default, the following tests use up to two source-measure units (SMUs), one CVU, and one PMU.

Descriptions of the capacitor tests

Test	Description
`cap`	Charges a capacitor at a constant current and measures the voltage as a function of time.
`cv-cap`	Measures the capacitance as a function of voltage on a capacitor. The noise is calculated using the standard deviation of the data.
`pulse-cap`	Applies a single voltage pulse to a capacitor and measures the resulting current. A single channel of the PMU is used to make the measurement. One end of the capacitor is connected to PMU1 and the other end is connected to PMU GND.

Analyze data

In this section:

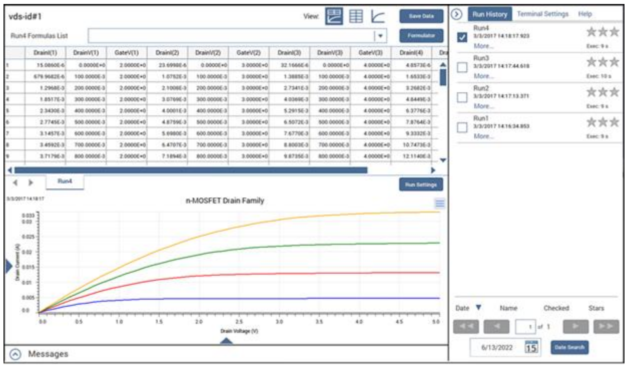
Introduction

When you run tests, Clarius records the data in the Analyze pane. You can display the data in a test as a spreadsheet and as a graph. You can also select data to display and graph at the project level.

Select **Analyze** to view the spreadsheet and graph, shown in the following figure. You can change the display to show only the spreadsheet or only the graph using the View buttons in the upper right of the pane.

You can use the Formulator to have Clarius extract additional parameter information from the data, using formulas that you create. The Formulator calculation results are placed in the Run sheet, in addition to the raw data. Refer to The Formulator (on page 8-1) for more information.

Figure 47: Analyze



NOTE

For information on analyzing data for a subsite, refer to Analyze data for subsites (on page 9-36).

Spreadsheet

The Analyze spreadsheets contain the data generated by the test runs that are selected in the Run History tab. Each run has a spreadsheet that is labeled with the name of the run. The run sheets also record data generated by the Formulator. The run sheets are read-only.

The Analyze spreadsheet is compatible with Microsoft™ Excel™.

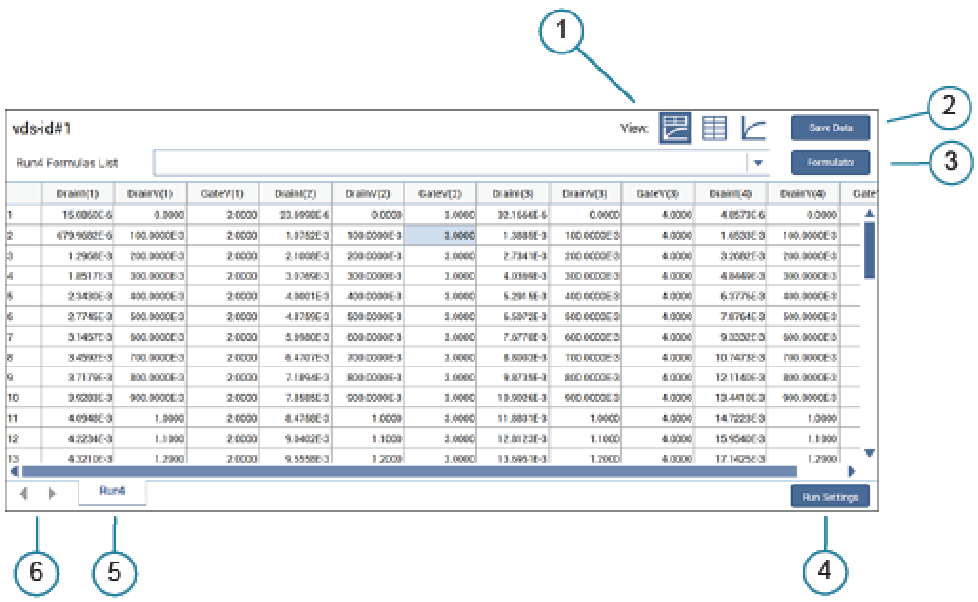
NOTE

If you have tests that were created in earlier versions of Clarius, you may also have a Calc sheet. Calc sheets were used for custom, test-specific data analysis. Calc sheets created for earlier versions are maintained in the present version of Clarius as read-only sheets.

Options on the Run spreadsheet

The following table provides brief descriptions of the options on the Run spreadsheet.

Figure 48: Options on the Run spreadsheet



1	View selections for the Analyze pane. You can display the spreadsheet and the graph, the spreadsheet only, or the graph only.
2	Save Data allows you to save the spreadsheet and graph data for the selected project, test, or action. Refer to Save results and graphs (on page 3-33) for more information.
3	Opens the Formulator, which allows you to make data calculations on test data and on the results of other Formulator calculations. Refer to Formulator (on page 8-1) for details.
4	Run Settings displays the test and terminal settings used to produce the results. This is a summary of the Test Settings and Terminal Settings that were set in Configure. Refer to Run Settings (on page 3-3) for more information.
5	Tab for the test run data. If there are multiple test runs selected, you can select the tab to display the data for that test run.
6	If there are more than three tabs, you can use these arrows to move between the spreadsheet tabs.

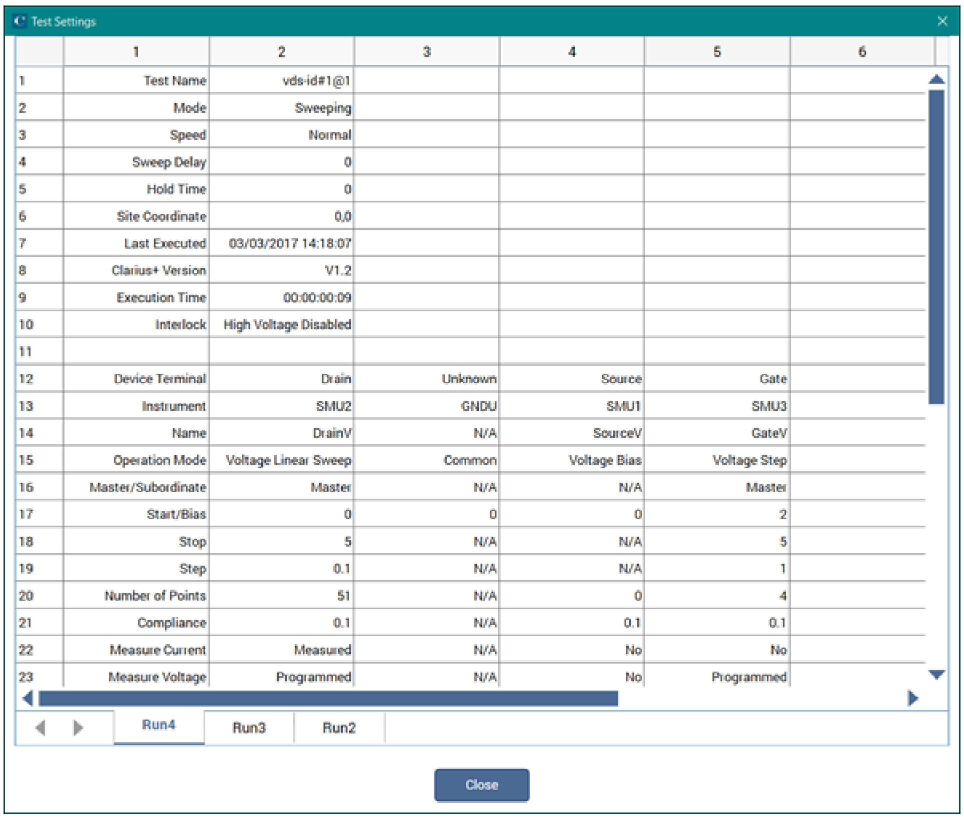
Run Settings

The Run Settings button displays the test configuration information from the Configure pane for the test runs. If multiple test runs are selected in the Run History pane, each run is available in a separate tab in the Test Settings dialog.

The information in the Test Settings dialog is read-only. You can select data and right-click to copy it.

An example is shown in the following figure.

Figure 49: Run Settings example



	1	2	3	4	5	6
1	Test Name	vds-id#1@1				
2	Mode	Sweeping				
3	Speed	Normal				
4	Sweep Delay	0				
5	Hold Time	0				
6	Site Coordinate	0,0				
7	Last Executed	03/03/2017 14:18:07				
8	Clarius+ Version	V1.2				
9	Execution Time	00:00:00:09				
10	Interlock	High Voltage Disabled				
11						
12	Device Terminal	Drain	Unknown	Source	Gate	
13	Instrument	SMU2	GNDU	SMU1	SMU3	
14	Name	DrainV	N/A	SourceV	GateV	
15	Operation Mode	Voltage Linear Sweep	Common	Voltage Bias	Voltage Step	
16	Master/Subordinate	Master	N/A	N/A	Master	
17	Start/Bias	0	0	0	2	
18	Stop	5	N/A	N/A	5	
19	Step	0.1	N/A	N/A	1	
20	Number of Points	51	N/A	0	4	
21	Compliance	0.1	N/A	0.1	0.1	
22	Measure Current	Measured	N/A	No	No	
23	Measure Voltage	Programmed	N/A	No	Programmed	

Run sheet

There is an Analyze Run sheet for every run of every test. Each column contains the results for one test parameter or for a Formulator calculation.

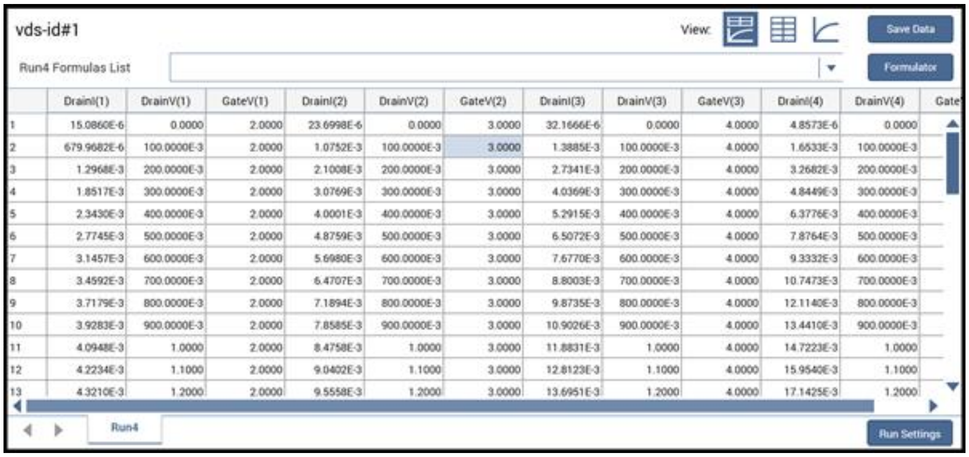
Each column heading identifies the data in that column. Headings are assigned by the data source:

- For ITMs, Clarius assigns headings
- For UTMS, the headings are assigned by the user modules
- For Formulator calculations, the names of the formulas in the Formulator are used as the headings

If a project contains multiple instances of a test under the same name, each instance generates its own data. Tests are numbered in the order in which they are added to the project. Ensure that you select the correct instance of the test in the project tree.

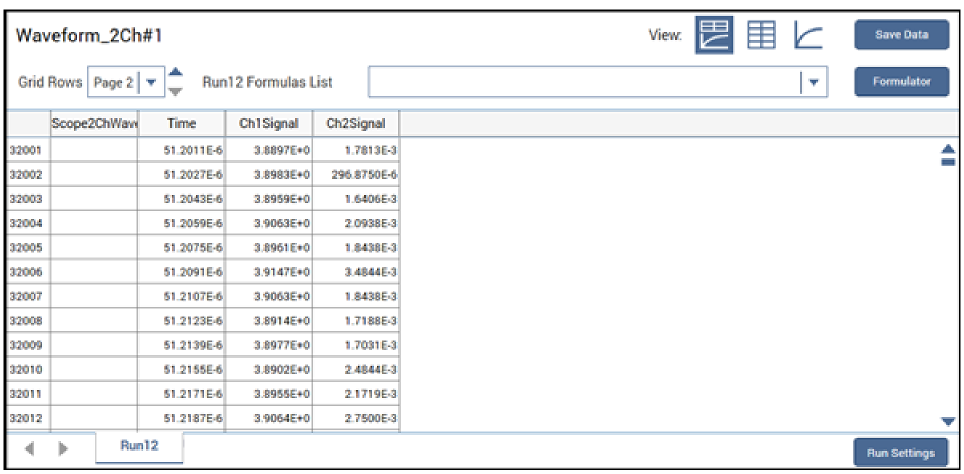
The following figure shows a Run sheet that contains data generated by the `vds-id` test.

Figure 50: Analyze Run sheet



If you have more than 32,000 data rows, the data is separated into multiple pages and the Grid Rows list is displayed, as shown in the following figure. You can select from the list or use the arrows to move between the pages.

Figure 51: Analyze Sheet with more than 32,000 rows of data



The maximum number of columns in a sheet is 256. An error is generated if a test results in more than 256 columns.

The maximum number of rows in a sheet depends on the type of test:

- ITM maximum is 4096.
- UTM maximum is 65,535 for UTMs that use `PostDataDouble` if data is sent point by point.

For a test or action, you can hide columns in the graph. Right-click the column and select **Hide**. To display the columns, right-click and select **Unhide** to choose from a list of columns, or select **Unhide All** to display all columns.

If cell contains `#REF`, there is a problem with the data in the formulas, such a divide by zero error or the log of a negative number. If `NaN` is displayed, there is a problem with the data that was acquired during the test.

All data in the Analyze spreadsheets can be saved to an external file. Refer to Save results and graphs (on page 3-33). To export all data from a project, test, or device, use the Data Export tool. Refer to Data Export tool (on page 5-19).

To access the Analyze Run sheet for a project, test, or action:

1. In the project tree, select the project, test, or action.
2. Select **Analyze**. The sheet and graph are displayed.
3. If data is from a CVU, right-click a cell to select the **CVU Data Type**.

Formulas List of the Run spreadsheet

If a column in the Run spreadsheet contains the results of Formulator calculations, you can display the formula that was used to get the results.

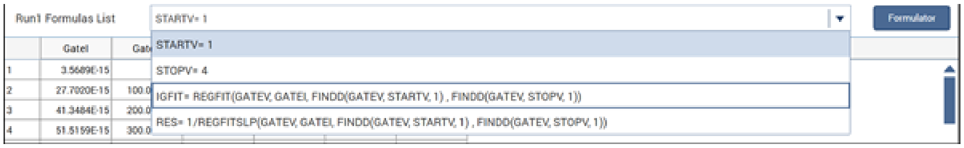
The `#REF` notation in a cell indicates that the Formulator could not calculate a valid value. This can occur if a formula needs multiple rows as arguments, if a calculated value is out of range, or if a divide by zero is attempted. For example, if the `MAVG` function is using five points to calculate a moving average of a column that contains five values, the first two and last two cells contain `#REF`.

For more information on working with the Formulator, refer to Formulator (on page 8-1).

To display a formula:

1. Select the **Formulas List**. A list of formulas is available.
2. Select the formula you want to display.
3. If you need to make changes to the formula, select **Formulator** to open the formula in the Formulator.

Figure 52: Displaying a Formulator equation using the Formulas list



Terminal Settings pane (Analyze)

When Analyze is selected, the Terminal Settings pane displays the settings for each terminal used in the presently selected test.

The options that are available depend on the instrument that is selected. For descriptions of parameters, refer to:

- "SMU Test Settings" in the *Model 4200A-SCS Source-Measure Unit (SMU) User's Manual*
- "CVU Test Settings" in the *Model 4200A-SCS Capacitance-Voltage Unit (CVU) User's Manual*
- "PMU Test Settings" in the *Model 4200A-SCS Pulse Card (PGU and PMU) User's Manual*

Module Settings pane (Analyze)

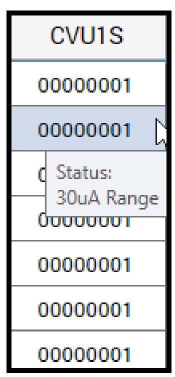
When Analyze is selected for a test that is based on a user module, the Module Settings pane is available. The options in this pane are the same as the options in Configure for the module. Changes made in the Module Settings pane are also changed in Configure.

Descriptions of the options are available in the Help pane.

Measurement status

Many tests provide status information for the measurements in the spreadsheet of the Analyze pane. For example, for the 4210-CVU or 4215-CVU, the data column for the status codes is labeled CVU1S. CVU status code indicates the current measure range for each impedance measurement and flags any errors. You can hover on a cell to see information about the measurement.

Figure 53: Measurement status column



When a measurement error occurs, the data values in the flagged data row are color-coded to identify the fault type as follows:

- Red: Measurement timeout.
- Magenta: Measurement overflow.
- Orange: Auto Balance Bridge (ABB) not locked.

An example of the color codes is shown in the following figure. In the status column (CVU1S), you can hover over the cell to see information about the measurement.

Figure 54: Spreadsheet indicating a measurement error

	Cp_AB	Gp_AB	DCV_AB	F_AB	CVU1S
2	-158.6073E-9	181.4187E-3	-4.8000E+0	1.0000E+6	00080002
3	-158.5251E-9	181.2184E-3	-4.6000E+0	1.0000E+6	00080002
40	-282.2661E-12	3.7564E-3	2.8000E+0	1.0000E+6	03010002
41	-283.2877E-12	3.7541E-3	3.0000E+0	1.0000E+6	03010002

Status codes

The 16 basic codes used for CVU measurement status are listed in following table. Each code is represented as a 32-bit hexadecimal value (0x).

CVU measurement status codes (CVU1S)

#	Code	Description
0	000000*yy*	No faults
1	8*xxxxxyy*	Measurement timeout occurred
2	01*xxxxyy*	CVH1 current measurement overflow
3	02*xxxxyy*	CVH1 voltage measurement overflow
4	03*xxxxyy*	CVH1 current and voltage measurement overflow
5	08*xxxxyy*	CVH1 ABB not locked
6	09*xxxxyy*	CVH1 ABB not locked, current measurement overflow
7	0A*xxxxyy*	CVH1 ABB not locked, voltage measurement overflow
8	0B*xxxxyy*	CVH1 ABB not locked, current and voltage measurement overflow
9	*xx*01*xxyy*	CVL1 current measurement overflow
10	*xx*02*xxyy*	CVL1 voltage measurement overflow
11	*xx*03*xxyy*	CVL1 current and voltage measurement overflow
12	*xx*08*xxyy*	CVL1 ABB not locked
13	*xx*09*xxyy*	CVL1 ABB not locked, current measurement overflow
14	*xx*0A*xxyy*	CVL1 ABB not locked, voltage measurement overflow
15	*xx*0B*xxyy*	CVL1 ABB not locked, current and voltage measurement overflow

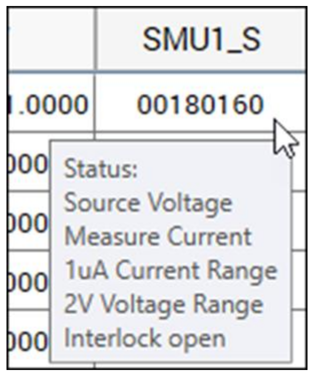
The *yy* value indicates the range used, as shown in the following table.

yy value	00	Lowest range (1 µA) used for the impedance measurement
	01	Middle range (30 µA) used for the impedance measurement
	02	Highest range (1 mA) used for the impedance measurement

Enable status codes for SMUs

Status codes are automatically available for CVUs, PGUs, and PMUs. For SMUs, you must choose to display the status codes. When this option is selected, Clarius records measurement status information when the test executes. A column of the Analyze spreadsheet displays this information. Hover over a cell to review the information. An example of the status information is displayed in the following figure.

Figure 55: SMU Report Status column in the Analyze sheet



To display status codes for a SMU:

1. In the project tree, select the test.
2. Select **Configure**.
3. Select **Terminal Settings** in the right pane.
4. Select Advanced.
5. Select **Report Status**.

Measurement status notes

NOTE

Whenever a fault occurs, run the Confidence Check utility before performing any other troubleshooting actions (see CVU Confidence Check (on page 5-14) for details).

Measurement timeout: Indicates that the measurement was not received after a set time (total aperture). This timeout error may indicate that there is an issue with the CVU card. Try resetting the hardware and running the project test again. If this error reoccurs, contact Keithley.

To reset the hardware:

1. Select **Start**.
2. Type `resethw`.
3. Select the instruments that need to be reset.
4. Select **Reset**.

Current measurement overflow:

- Try a higher current measure range (or Auto).
- Try a lower AC drive voltage.

Voltage measurement overflow: Try a lower DC bias voltage.

ABB unbalance errors

The CVU uses the autobalancing bridge (ABB) technique to achieve accurate impedance measurements. ABB creates a virtual ground at the DUT to minimize measurement error. Every CVU measurement is made with ABB active. The ABB always attempts to lock the low side of the DUT to virtual ground.

If the ABB fails to lock, the measurement is made, but may be out of specification. If this occurs, the returned data is flagged and shown in yellow on a blue background on the Analyze sheet.

The most common reasons that ABB fails to lock are:

- The cable lengths on the CVU terminals are not the same
- HPOT or LPOT terminals were disconnected
- Excessive noise on the LPOT terminal
- High frequency sources
- Physical cable lengths do not match the cable length set in Clarius
- Improperly torqued SMA cables
- Sub-optimal I_{RANGE} setting
- Too much parasitic load on the low side of the DUT

You can use CVU Confidence Check to help troubleshoot ABB errors. Refer to Run an open check and short check (on page 5-14) for instructions on performing a confidence check.

Run History

When a test is selected in the project tree and Analyze is selected, the right pane includes a Run History tab. If the test is one of the pre-built tests, Run History contains reference data and the Analyze pane displays a sample graph. If you created a new test, there is no information in the Run History pane or graph until the test is run.

If you ran the test, the latest test run is displayed at the top of the Run History pane. The data and graph from this test run are displayed in the Analyze pane.

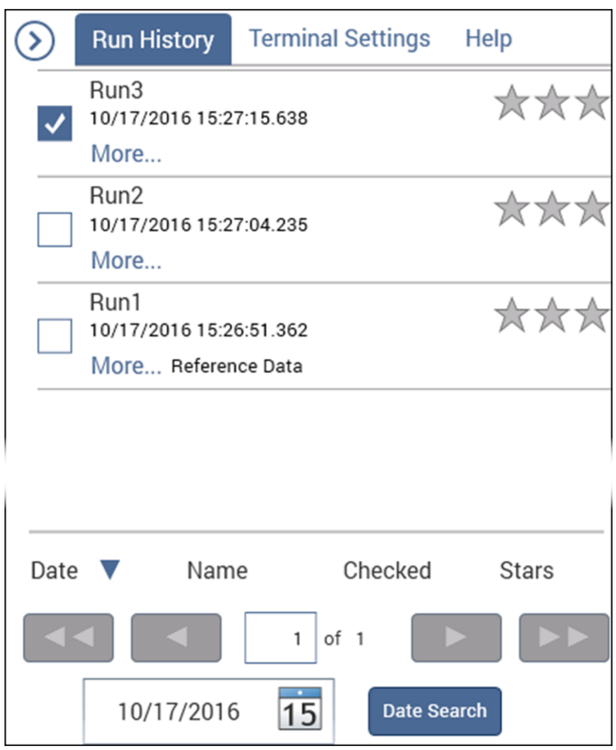
You can have up to 10,000 runs for each test and you can select up to 128 test runs from the Run History to be displayed in the Analyze pane. The data for each selected test run is shown in a separate tab in the Analyze sheet. All the selected test runs are graphed on the Analyze graph. By default, the latest test run data and graph are displayed in the Analyze sheet and graph. Select a different test run to view that information in the Analyze sheet and graph.

When you run stress subsite operations, only data from the latest loop is saved to the Run History data file in a test. The oldest run is removed first. This optimizes disk space usage.

Each Run History entry includes:

- A timestamp that shows the date and time when the test was run.
- The execution time.
- Rating stars that you can use to flag specific tests.
- Notes.
- The indicator "Monitor" if the data was generated using the Monitor option.

Figure 56: Run History pane



NOTE

If you imported data from a 4200 project, each set of appended data is shown as a separate test run in the Run History.

Change the name of a test run

You can change the name of the test run. If you change the name, the new name is displayed in the following locations:

- The data tab name in the Analyze sheet.
- In the Formulator in the Formula Set list.
- In the test library if you add the test to the test library.

The names for test runs:

- Must be less than 19 characters.
- Must be unique; multiple test runs cannot have the same name. The name comparison is not case-sensitive. For example, `Best` and `best` are considered to be the same name.
- Must not use Clarius reserved names, including `Data`, `Calc`, `Settings`, and `Run1` to `Run9999`.

To change the name of a test run:

1. In the Run History pane, select **More**.
2. In the Run number text box, type the new name.
3. Select **Enter**. The name you entered is displayed, followed by the run number.

NOTE

To revert to the default name, clear the information in the Run number text box and select **Enter**.

Delete test runs from Run History

You can delete test runs from the Run History, including reference data. This deletes the data from Clarius.

NOTE

If you are deleting multiple test runs, only the highlighted test runs are deleted. The status of the checkbox does not affect what is deleted.

To delete a test run from the Run History:

1. Right-click the Run History and select **Delete**.
2. Select **Yes**.

To delete multiple test runs:

1. Highlight the test runs to be deleted. You can use Ctrl+click and Shift+click to highlight them.
2. Right-click and select **Delete Selected**.

To delete all test runs, including reference data:

Right-click in the Run History pane and select **Delete All**.

Work with the Run History pane

You can change the sort order of the test runs in the Run History tab using the options at the bottom of the Run History tab. Select **Date**, **Name**, **Checked**, or **Stars** to sort the test runs in ascending or descending order using that option.

If you have more than one pane of test runs, use the arrow buttons to move between the panes.

To highlight a group of test runs, right-click and select **Begin Select Range**. To select a range, highlight another test run and right-click and select **End Select Range**. To clear the selections, right-click and select **Clear Selected**.

Search for a test run

You can search for a test run by date.

To search for a test run by date:

1. At the bottom of the Run History pane, enter the date.
2. Select **Date Search**.

Add notes to a test run

To add notes to a test run:

1. Select **More**.
2. Enter the note into the text box below Less.
3. Select **Enter** when the notes are complete.

Copy the settings of a test run to the Configure screen

You can copy the test and terminal settings that were used to generate a test run to the Configure pane. All settings in the Configure pane are replaced by the settings from the selected test run. This allows you to quickly restore the settings that were used to generate a test run.

To copy the settings:

1. In the Run History pane, right-click the test run that contains the settings you want to use.
2. Select **Load Configuration**. The changes are made to the Configure pane.

Set the Run History size

You can set the maximum number of test runs that are stored and displayed in the Run History pane.

When you run a test, the number of test runs stored in the Run History is limited to the number that is set. To maintain the number of runs, Clarius deletes the oldest unselected run. If there is existing data that exceeds the limit, only the oldest unselected run is deleted. If you have all runs selected, the most recent run is deleted when a new run is created. The runs continue to be numbered sequentially.

For example, if you have 200 existing runs and change the maximum run history size to 5, the number of runs in the Run History remains at 200. The next test run triggers the deletion of the oldest unselected run. The new run is numbered 201.

If you are running a project with subsite cycling or stressing, all existing runs are automatically unselected at the start of execution. Only data collected during the last run of the project is selected and displayed on the graphs. To save data between project runs, increase the run history size to be more than the number of cycles or stresses in your project.

If the run history size setting is less than the number of cycles or stresses in your project, a run is still generated for each cycle or stress.

To set the number of test runs stored in Run History:

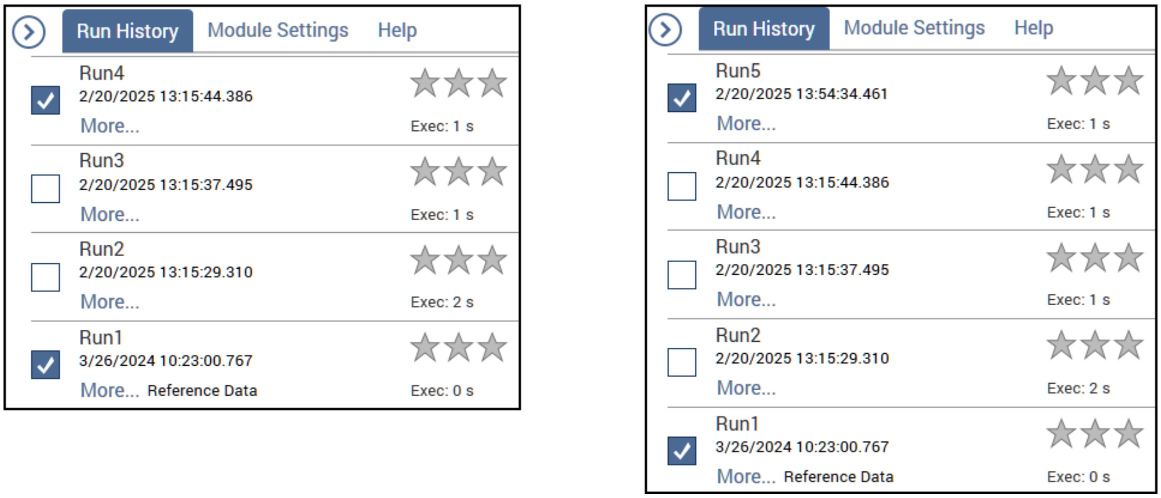
1. In Clarius, select **My Settings**.
2. Select **Run Settings**.
3. In **Run History Size**, enter the number of runs (1 to 10,000).
4. Select **OK**.

Run History selections when running a test

You can select whether or not to clear previous run history selections when a test is run.

By default, when a test is run, the selection of the previous run history is cleared and the most recent run history is selected. The selections of any other runs are maintained. For example, in the following figure, the selection of Run4 is automatically cleared when Run5 runs, but Run1 remains selected.

Figure 57: Run History selections



You can choose to keep the previous run history selected when the test is run again. If you choose to keep previous run histories, you also set the maximum number of selected runs. You can have up to 128 runs selected. If there are more than 128 run history selections, the oldest run history is cleared automatically when the test is run.

To keep the previous Run History selection:

1. Select **My Settings**.
2. Select **Run Settings**.
3. Select or clear **Keep selected runs while running the test**.
4. If you selected the option, set the maximum number of selected runs to maintain in the Run History list.
5. Select **OK**.

Analyze test data for projects

You can select test data to display at the project level. This allows you to compare data from different tests in your project in the sheet and graph.

You can also use the Formulator at the project level. The Formulator allows you to make data calculations on test data and on the results of other Formulator calculations. Refer to The Formulator (on page 8-1) for information on creating formulas.

When you look at a graph at the project level, you cannot view multiple test runs from the same test and family curves functionality is not available. Other graph settings are available as described in Analyze data for test runs (on page 3-1).

Save the project to preserve the Formulator and graph settings with the project.

Select data for the project-level Analyze pane

You can select up to 255 columns of data to display in the project-level Analyze pane.

To select test data to display at the project level:

1. Select **Analyze**.
2. In the project tree, select the project.
3. In the right pane, select **Data Series**.
4. From **Device**, select the device that contains the data.
5. From **Test**, select the test that contains the data.
6. From **Run History**, select the test run that contains the data. To display the most recent data, select **LatestRun**.
7. From **Data Series**, select All to display all available data or select a specific set of data.
8. Select **Add**.

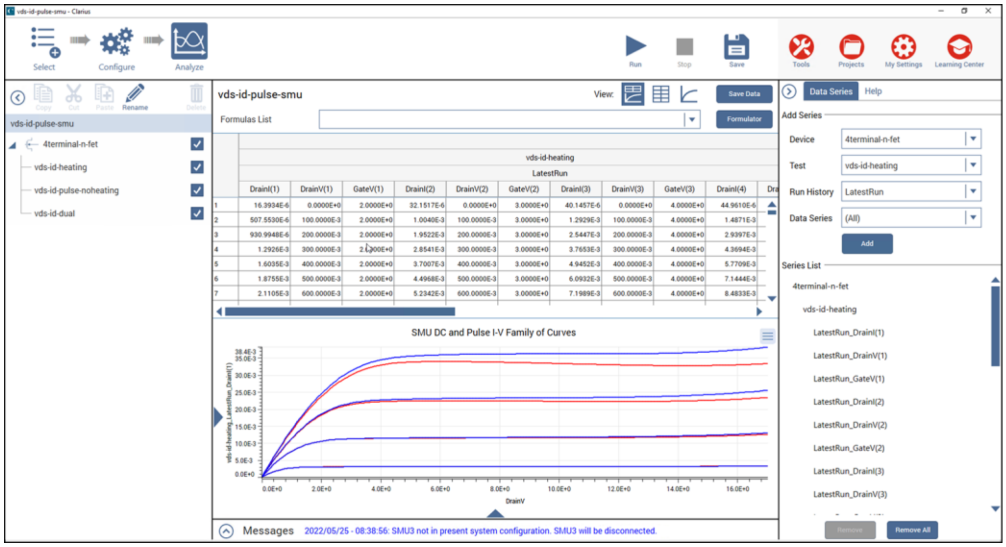
The data is displayed in the Analyze pane. The selected data series is also listed in the Series List at the bottom of the right pane.

The data for each selected device, test, and run is grouped in the Sheet. Scroll to the right to view the data.

You can select any of the data for graphing. See Define data to be graphed (on page 3-19) for details.

An example of a project with data and a graph set up is shown in the following figure.

Figure 58: Analyze for a project



Remove data from the project-level Analyze pane

You can remove data from the project-level Analyze sheet and graph. The original data remains intact.

To remove data from the project-level Analyze display:

1. In the Series List in the right pane, select the device, test, or run. You can use Ctrl+click to select multiple individual items or Shift+click to select a series of items.
2. Select **Remove**.

To remove all data from the project-level Analyze display:

In the Series List in the right pane, select **Remove All**.

Graph the data for test runs

The Analyze graph allows you to create graphs of test results. The graph provides you with flexible plot-data selection, formatting, annotation, and numerical coordinate display using precision cursors.

For test runs, the graph displays the data from the Run sheet tabs for the test runs selected from Run History. Each test run updates the graph to display the latest set of data. See Run History (on page 3-9) for details on selecting the runs that are displayed on the graph.

You can change the format of the graph using the options in the Graph Definition Menu.

NOTE

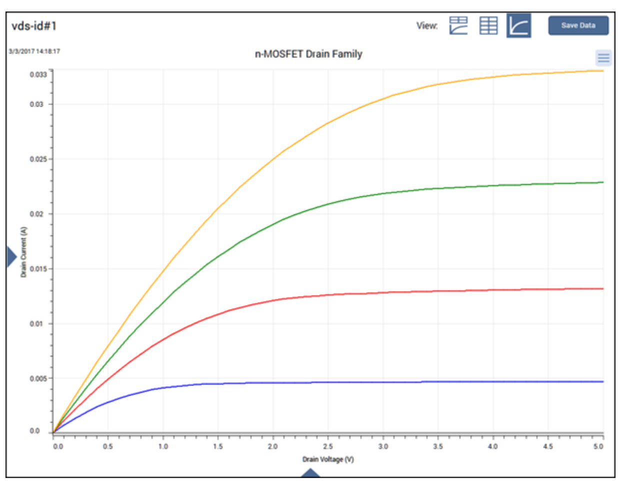
If you are viewing data for a subsite, refer to Analyze data for subsites (on page 9-36) for information on the graphing options for the subsite.

Open a graph

To open a graph:

1. In the project tree, select a test.
2. Select **Analyze**. The graph is displayed at the bottom of the center pane. The time and date when the data was generated are displayed in the upper left corner.
3. To enlarge the graph, select the Graph only option:

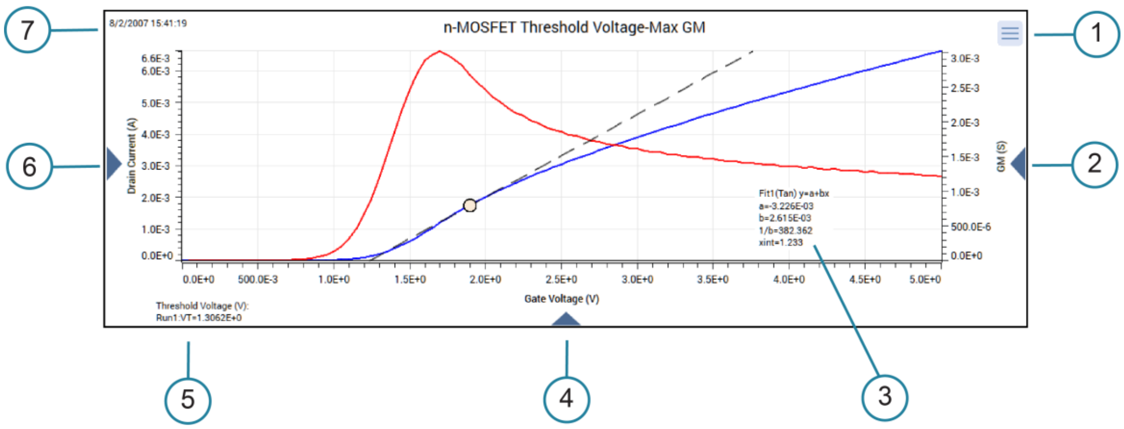
Figure 59: Example of n-MOSFET drain family of curves graph



Options on the graph

The following table provides brief descriptions of the options on the Analyze graph.

Figure 60: Example of an Analyze graph



1	Graph Definition Menu. Accesses options that you can use to customize the graph. You can also access this menu by right-clicking the graph. See Customize the graph (on page 3-19).
2	Right Y-axis settings. Only displayed if the Right Y Axis Definition is set up. See Define graph axes (on page 3-21).
3	Line Fit information. Only displayed if Line Fits are selected. See Line fits (on page 3-29).
4	X-axis settings. See Define graph axes (on page 3-21) to customize the axis.
5	Text added to the graph from the Graph Definition menu Add Text option. See Add text to the graph (on page 3-25).
6	Left Y-axis settings. See Define graph axes (on page 3-21) to customize the axis.
7	The date and time when the data was generated.

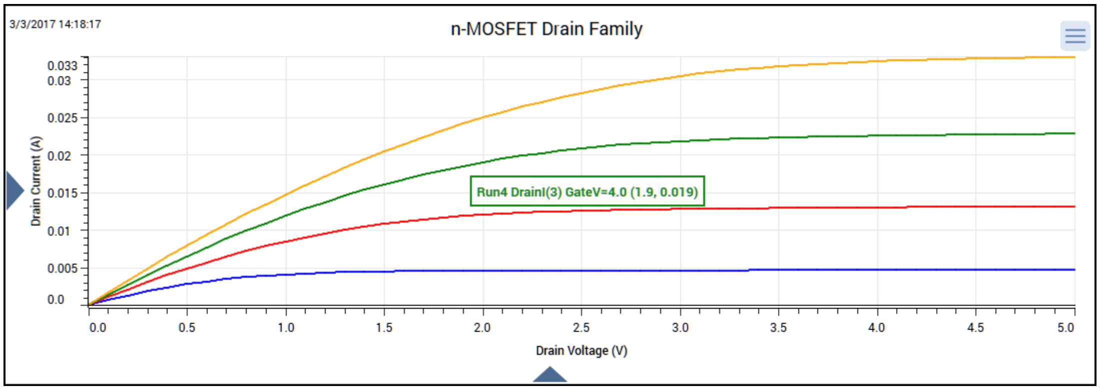
View details on a data point

When you select a point on any graph, Clarius displays the following information about the point:

- Run name, such as Run 4.
- Legend name, which includes the column name, such as DrainI(4). If a family of curves is graphed, it also includes the family curve name, such as GateV=5.0.
- Graph coordinates

This feature allows you to check information about any point on the graph. An example is shown in the following figure.

Figure 61: Data point display

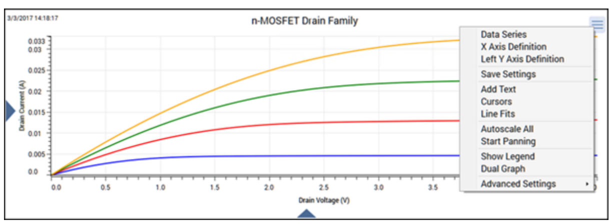


To zoom in on data, on a touch screen, pinch. If you are using a mouse, select a portion of the graph. To return to the full graph view, double tap or double-click.

Customize the graph

You can customize the graph using options in the Graph Definition Menu. You can also access these settings by right-clicking the graph.

Figure 62: Graph Definition Menu



Define data to be graphed

You can add or change the data series that is graphed and which axis it is displayed on.

When you set the style for a test run, the next test run uses the same style. If you compare multiple test runs, the line pattern changes for each run to make it easier to distinguish the runs in the graph.

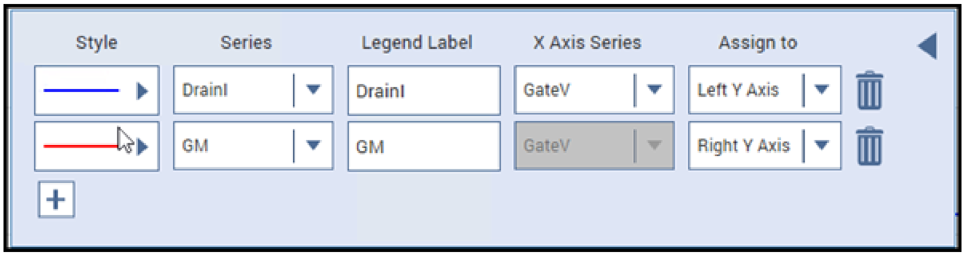
NOTE

If no test runs are selected in Run History, you can define the data to be graphed as described in the following procedure. The settings you make are applied to the next run.

To select the data series for the graph:

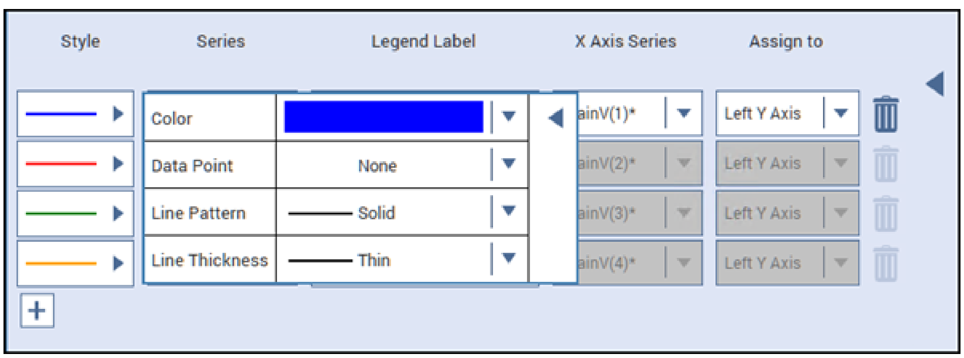
1. From the Graph Definition Menu, select **Data Series**.

Figure 63: Data Series dialog



2. To add a data series to the graph, select **+**. The next column of data from the sheet is added.
3. For **Series**, select the data series to be displayed on the y-axis.
4. For **Legend Label**, enter the text that is displayed in the legend when it is selected.
5. For **X Axis Series**, select the data series to be displayed on the x-axis. If you are at the project level or if Enable Multiple X Series is selected, you can use a different series for each x-axis.
6. For **Assign To**, select which the y-axis to display this data series on.
7. For **Style**, select ►.

Figure 64: Data Series Style options



8. Select the ▼ next to the Color, Data Point, Line Pattern, and Line Thickness to set each of the style options.
9. When changes are complete, select ◄ to close the dialog. The settings are applied immediately.
10. To save these settings, from the Graph Definition Menu, select **Save Settings**.

To remove a data series from the graph:

1. Select the trash can icon for that series.
2. If you select a new data series, the Legend Label is reset to the defaults.

Define a family of curves

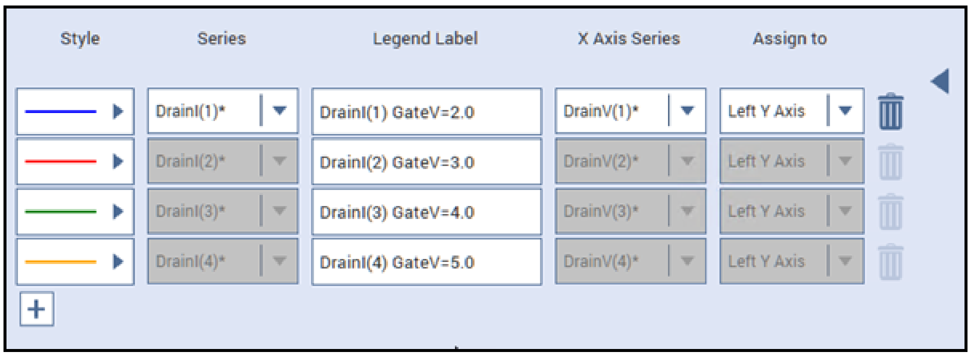
For tests that generate a family of curves, the family of curves is treated as one data series. Deleting or adding one series deletes or adds them all. You can change the style and legend for an individual data series in a family of curves.

NOTE

If there are multiple parameters with the same name, an * is displayed after the parameter name in the data series dialog.

An example of the Data Series dialog for a family of curves is shown in the following figure.

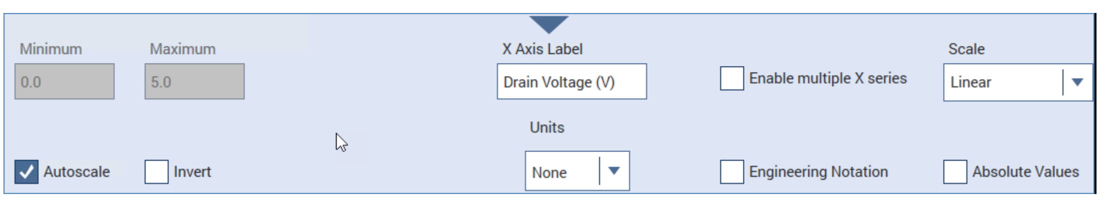
Figure 65: Data Series for a family of curves



Define graph axes

You can change the settings for the graph axes using the Axis Definition options in the Graph Definition Menu or by selecting the triangles on the bottom and sides of the graph. An example of the options for the X-axis is shown in the following figure.

Figure 66: Graph X-axis options



Minimum and **Maximum** set the graph to display only data between these two values. These options are not available if Autoscale is selected.

Autoscale automatically scales the axes to best fit the data. When you select Autoscale, the Minimum and Maximum settings are set to the autoscale values.

Invert allows you to flip the orientation of the graph so that the lowest value is at the top of the Y axis or the right side of the X axis.

The **Label** is the text that identifies the axis.

The **Units** option selects the unit of measure (volts, amps, seconds, farads, siemens, or hertz) that is displayed on the axis. When a unit of measure is selected, you can also set how the value is displayed, such as in A, mA, or µA. When Engineering Notation is selected, the Auto option is available. In this case, the graph automatically adjusts the units of measure for the best presentation of the data.

Engineering Notation displays the scale values and cursor data points in engineering notation. When this option is cleared, the scale values and data points are displayed in either standard form or scientific notation, depending on the value.

Scale allows you to display the data on a linear or logarithmic scale. If Log is selected, Absolute Values is automatically set. If you return to Linear, you may need to clear Absolute Values if you want to graph negative values.

Absolute Values graphs the data as unsigned values when selected.

The X-axis also includes **Enable Multiple X Series** for items at the test level of the project tree. This allows you to select additional X Axis Series data in the Data Series dialog. Refer to Define data to be graphed (on page 3-19) for additional information. When you clear Enable Multiple X Series, the X Axis Series changes to the value of the first series shown in the Data Series dialog.

NOTE

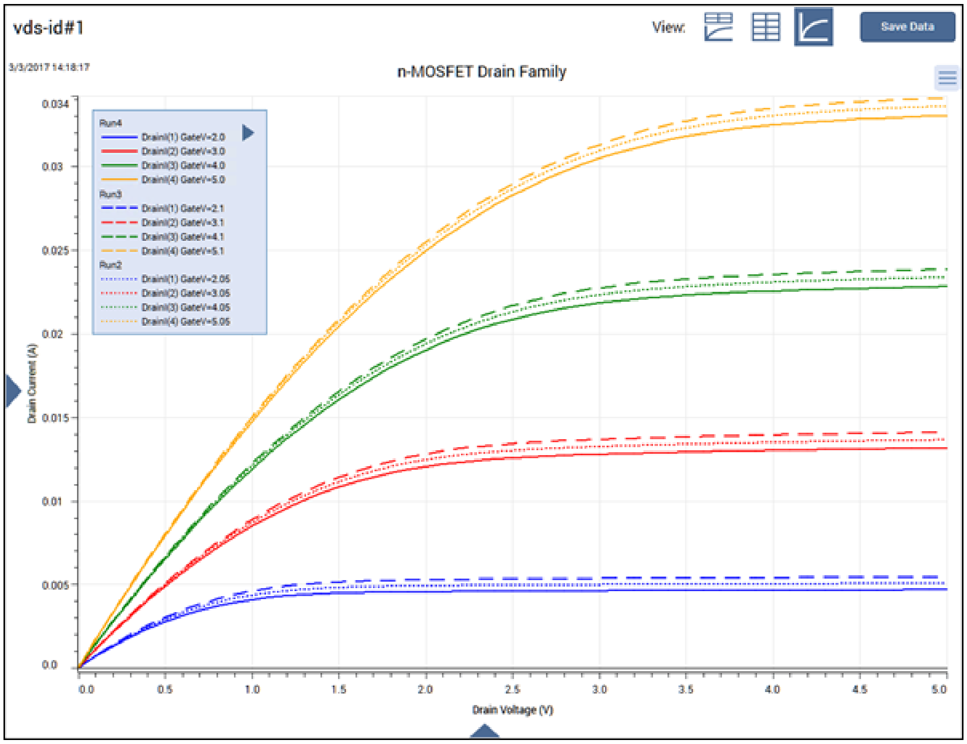
Enable Multiple X Series is always enabled at the project level.

Display multiple runs

You can display multiple run histories in the Analyze graph and select how to differentiate the data for appended run histories. Appended run histories are run histories for a graph that already has a selected run history. Options for differentiating the data from different runs include using combinations of line styles, line colors, and data point markers. You can set the differentiation for each graph in each test independently, as described in the following, or use the default method described in Set the Clarius graph defaults (on page 3-24). If you set up defaults, you can later change the individual data series without affecting the graphs of the other run histories.

In the example in the following figure, Run3 and Run2 are appended runs. The series from appended runs use the same colors as the first selected run (in this example, Run4), but are differentiated by line styles. In this figure, the Run3 series uses the long dash line style and the Run2 series uses the dotted line style.

Figure 67: Example of multiple runs



Change the append style for multiple run histories

NOTE

If the append style is not changed for the graph in the selected test, it is set to the default described in Set the Clarius graph defaults (on page 3-24).

To change the line style when multiple run histories are graphed:

1. From the Graph Definition Menu, select **Advanced Settings**.
2. Select the line or point option:
   - **Append by Line Style:** Use a different line style, such as line, dotted line, or dashed line, for the data from each run history.
   - **Append by Line Color:** Use a different line color for the data from each run history.
   - **Append by Point Markers:** Use different point markers for the data from each run history.
3. To save these settings, from the Graph Definition Menu, select **Save Settings**.

Return to graph defaults for selected run histories

You can reset the data series for the selected run histories. The default settings for the most recent run are a blue line, no data point marker, solid line pattern, and thin line thickness.

If you select more than one data series in the run, the following series are plotted in different colors.

If more than one run history is selected, the appended run histories are returned to the default append styles for this graph in the test.

To return to the graph defaults for the selected run histories:

1. From the Graph Definition Menu, select **Advanced Settings**.
2. Select **Set Data Series to Defaults**.

Set appended run history graphed data to match the most recent run history

You can set the graphed data of appended run histories to use the settings of the first selected run history as a template. This does not change settings for the first (not appended) run history.

You can use this feature to customize the series of the first run and synchronize all appended runs to its settings.

To set the display of appended run histories to match the most recent selected run history:

1. From the Graph Definition Menu, select **Advanced Settings**.
2. Select **Reset Appended Data Series**.

Set the Clarius graph defaults

When you select multiple run histories for display on the Analyze graph, the lines for each run are differentiated by the line style, line color, or point markers.

If you do not define the append style for a specific test using the Graph Definition Menu, Clarius uses the style selected for the Default Append Style.

To set the append style defaults:

1. On the top pane of Clarius, select **My Settings**.
2. Select **Graph Defaults**.
3. Set the **Default Append Style** to the default style.
4. Select **OK**.

Save graph style settings for a test

You can use the same style settings for all runs of a test. By default, the color, data point, line pattern, and line thickness may change depending on the order in which run histories are selected.

Saving the settings for a test takes precedence over the selection made for the Graph Defaults in My Settings.

To use the same style settings for all run histories in a test:

1. Select a run history.
2. Set the styles in the graph.
3. Select the **Graph Definition Menu**.
4. Select **Save Settings**.

Add text to the graph

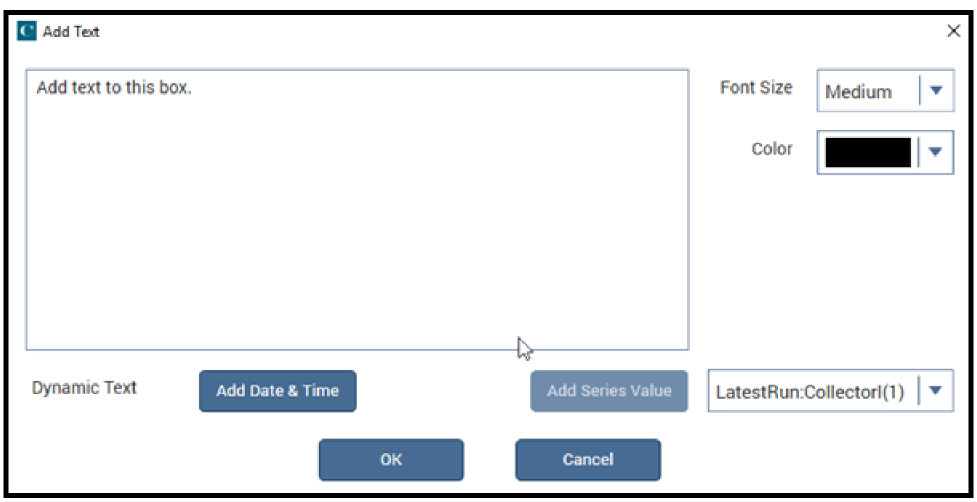
You can add up to four text boxes to the graph. You can use Latin character sets in the text.

Each text box can contain up to 512 characters in up to 12 lines of text.

To add text:

1. Select the **Graph Definition Menu**.
2. Select **Add Text**. The Add Text dialog is displayed.

Figure 68: Graph Add Text dialog

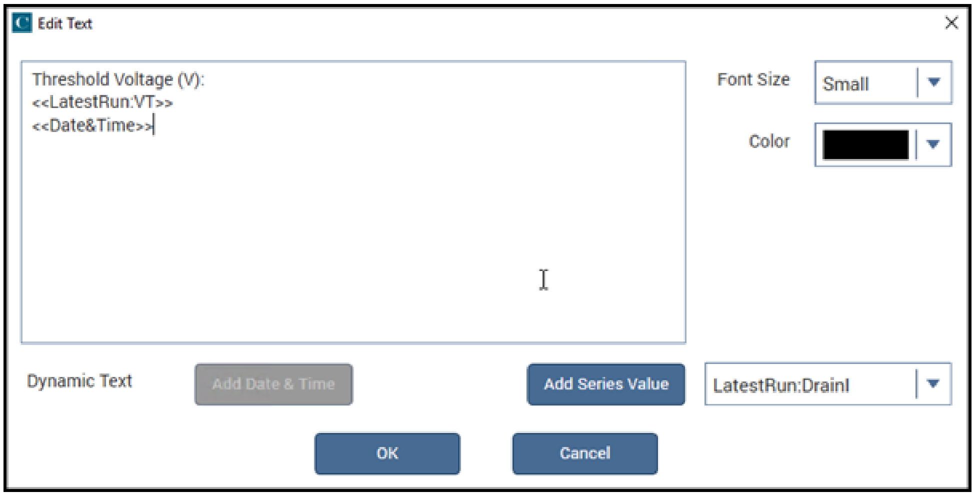


3. In the Text box, enter the text.
4. Select the **Font Size**.
5. Select the **Color**.
6. To include the date and time when the data was generated, select **Add Date & Time**.

7. To add the value from a data series, select the data series from the list, then select **Add Series Value**.
8. Select **OK**.
9. Drag the text to the appropriate location on the graph.

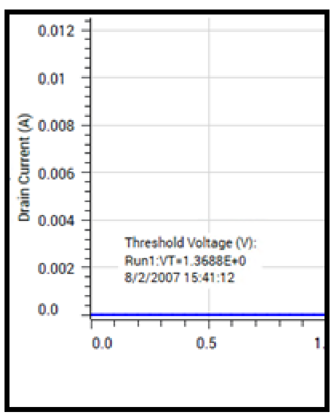
The following figure shows an example with a series (LatestRun:VT) and the date and time (Date&Time) options added.

Figure 69: Add Text dialog with date, time, and series values included



The following figure shows how the Edit Text settings in the previous figure are displayed on the graph.

Figure 70: Example of a text box on the graph



To edit text:

Double-click the text to open the Edit Text dialog.

To remove text:

Right-click the text and select **Remove Text**. The text is removed from the graph and deleted from the Edit Text dialog.

Add cursors to the graph

You can add up to two sets of cursors to the graph.

When cursors are displayed, the following information is displayed next to the cursor:

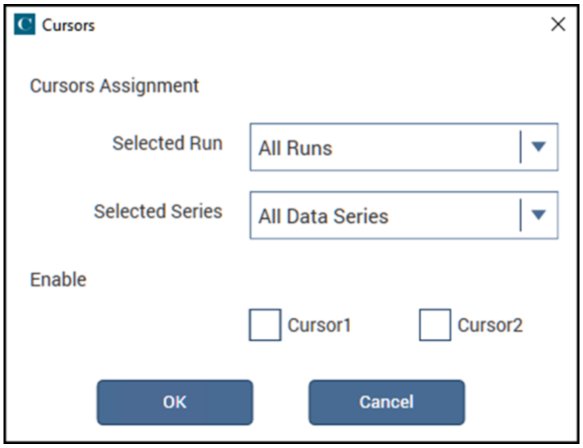
- The data series.
- The cursor coordinates.

Cursor selections are saved if you leave the graph and return.

To add cursors to the graph:

1. If you want to select cursors for specific runs, select Run History in the right pane and select the runs.
2. From the Graph Definition Menu, select **Cursors**. The Cursors dialog is displayed.

Figure 71: Cursors dialog

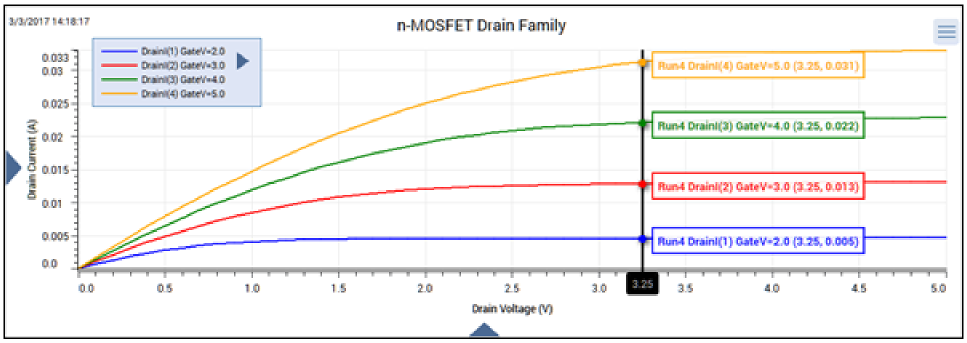


3. From the **Selected Run** list, select one of the following:
   - **All Runs**
   - **Latest Run**
   - A specific run (available if runs are selected in Run History
4. From the **Selected Series** list, select **All Data Series** or a specific series.
5. Under **Enable**, select which cursors to display.
6. Select **OK**. The cursors are added to the graph.

To use cursors in the graph:

Select the line for the cursor and drag it as needed to view the data. An example is shown in the following figure.

Figure 72: Graph cursors



To remove cursors from the graph:

1. Right-click the line for the cursor.
2. Select **Remove Cursor1** or **Remove Cursor2**.

You can also remove a cursor by clearing the Cursor1 or Cursor2 selection in the Cursors dialog.

Display the graph legend

You can display a legend that describes each series of data.

To display the legend, right-click the graph and select **Show Legend**. You can drag the legend to a new location on the graph.

To change the labels of the legend, select the arrow on the legend. This opens the Data Series dialog, which you can use to change the line style, data series, label, and axes. The legend label can be up to 32 characters.

To hide the legend, right-click the legend and select **Hide Legend**.

To change the size of the text on the legend:

1. From the Graph Definition Menu, select **Advanced Settings**.
2. Select the **Legend Font Size** (small, medium, or large).

Display two graphs

You can display two graphs with different data series.

You can also hide one of the graphs. The settings for the hidden graph are retained.

To display two graphs:

1. In the Graph Definition Menu, select Dual Graph.
2. Use the options in the Graph Definition Menu for the second graph to set up the graph.

To hide one of the graphs:

1. Select the Graph Definition Menu for the graph you want to hide.
2. Select **Hide Graph**.
3. To redisplay the graph, select **Dual Graph**.

Line fits

You can use the graph line fit feature to create lines that have the best fit to a series of data points. Line fits are used to study the nature of the relation between multiple variables.

You can fit lines to test result graphs for one or two line fits between existing cursors. When the line is fitted, the graph displays:

- The fitted line.
- The fit parameters in a text box on the graph.
- The point at which a tangent line is fitted to the plot or the starting and ending points (data range).
- The data-point coordinates.

Add line fits

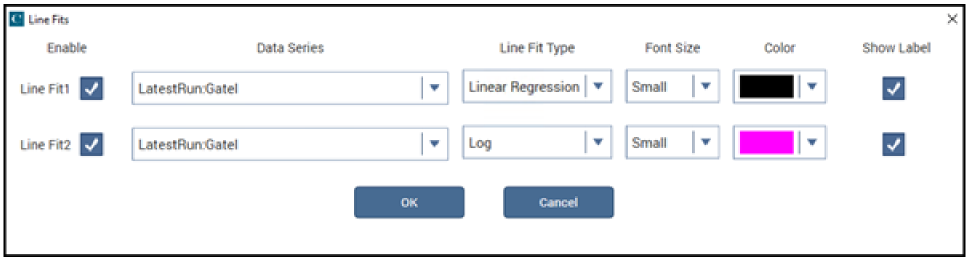
When you add a line fit, a dashed line is displayed, along with cursors that you can drag to a new location on the line. If you select the Free Linear line fit type, you can drag the cursors to any point on the graph.

A text box is displayed for each line fit. You can move this text box as needed. The text box contains the fit parameters and cursor coordinates.

To add a line fit:

1. Select **Analyze**.
2. Select the Graph Definition Menu and select **Line Fits**.

Figure 73: Line Fits dialog

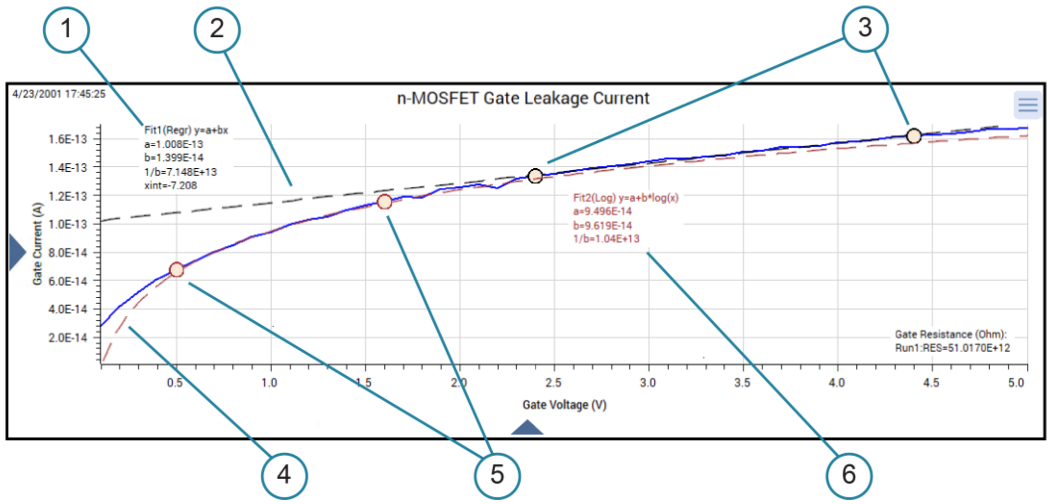


3. Select **Enable** for the line fits you want to display.
4. Select the **Data Series** for each line fit. If you are selecting the Free Linear line fit type, you select the Left Y Axis or, if it is defined, the Right Y Axis as the data series.
5. Select the **Line Fit Type**. You can select:
   - **Linear:** Chord line of the form y = a + bx.
   - **Linear Regression:** Regression line of the form y = a + bx.
   - **Exponential:** Regression line of the form y = a * e^bx.
   - **Log:** Regression line of the form y = a + b * log10(x).
   - **Tangent:** Tangent to the plot. The tangent line has the form y = a + bx.
   - **Free Linear:** Add a line fit that is not constrained to a specific data series.
6. Select the **Font Size**.
7. Select the **Color** of the line and text box.
8. Select **Show Label** to display a text box with the line fit coordinates.
9. Select **OK**. The line fits are displayed on the graph.
10. If you selected Show Label, you can drag the text box with the coordinates to a new location.

Line fit examples

The following figure illustrates the linear regression and log line fit types.

Figure 74: Linear fit example



1	Text box that contains the parameters of the linear regression fit line and the cursor coordinates. You can move this box.
2	Linear regression fit line.
3	Cursors for the data points that define the linear regression fit line. You can move the cursors along the line.
4	Log fit line.
5	Cursors for the data points that define the log fit line. You can move the cursors along the line.
6	Text box that contains the parameters of the log fit line and the cursor coordinates. You can move this box.

Formulator line fits

The Formulator also includes options to generate line fits. The Formulator fit results can be used in other calculations. The results of the graph line fits are similar to the results with the corresponding Formulator functions, as shown in the following table.

Correspondence between graph and Formulator line fits

Graph line fit	Formulator fits that return the corresponding fit line and fit parameters			
	Fit line	**Fit parameter "a"**	**Fit parameter "b"**	**Fit parameter "xint"**
Linear	LINFIT	LINFITYINT	LINFITSLP	LINFITXINT
Linear Regression	REGFIT	REGFITYINT	REGFITSLP	REGFITXINT
Exponential	EXPFIT	EXPFITA	EXPFITB	Not applicable
Log	LOGFIT	LOGFITA	LOGFITB	Not applicable
Tangent	TANFIT	TANFITYINT	TANFITSLP	TANFITXINT

Zoom, pan, and autoscale

If you are using the touch screen:

- To zoom in on the graph, pinch the area of the graph to view.
- To zoom out, move your fingers apart.
- To return to the full graph view, double-tap the graph.

If you are using a mouse:

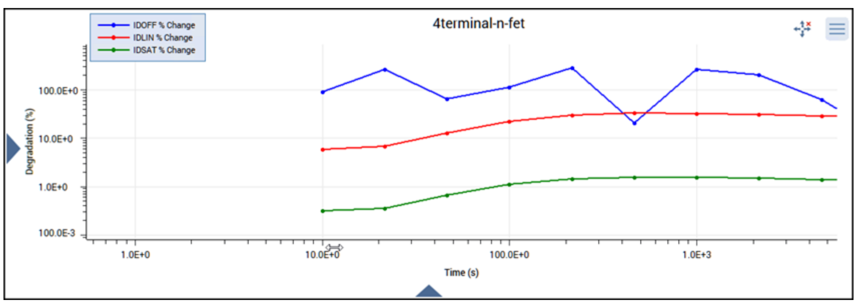
- To zoom, select the area on the graph with the left mouse button.
- To return to the full graph view, double-click the graph.

To use the mouse to pan along the axis (left to right on x-axis or up-down on the y-axis):

1. Place the mouse over the axis in the graph. The mouse cursor changes to the scrolling icon.
2. Drag the axis to the new location.

An example of the panned axis is shown in the following figure.

Figure 75: Panning along the x-axis



To pan in all directions:

1. From the Graph Definition Menu, select **Start Panning**.
2. Drag the graph in any direction.

When panning is active, the following icon is displayed to the left of the Graph Definition Menu:

Select the icon to stop panning. You can also stop panning by selecting another option from the Graph Definition Menu.

Rescale the graph:

From the Graph Definition Menu, select **Autoscale All**.

This option automatically scales all graph axes one time. You can use this to reset the graph after panning or zooming.

When you pan or zoom, Autoscale All is disabled. The minimum and maximum values of each axis show the values set by zoom.

Save results and graphs

You can save the sheet data and graphs for the entire project, a subsite, or for individual tests or actions.

The sheet data can be saved to the `.xls`, `.xlsx`, `.csv`, or `.txt` format. The `.xls` and `.xlsx` formats are compatible with the Microsoft™ Excel™ application.

When you save test data to the `.xls` or `.xlsx` format, the saved data includes data for all selected runs and the settings for the runs. If you select the `.csv` or `.txt` format, only the data from the most recent run is saved.

You can save graphs to `.jpeg`, `.bmp`, `.gif`, or `.png` format.

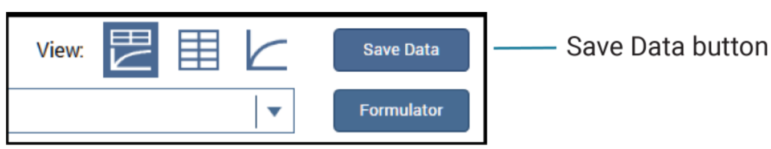
NOTE

To export all data from a project, test, or device, use the Data Export tool. Refer to Data Export tool (on page 5-19) for more information.

To save the information in the Run sheet:

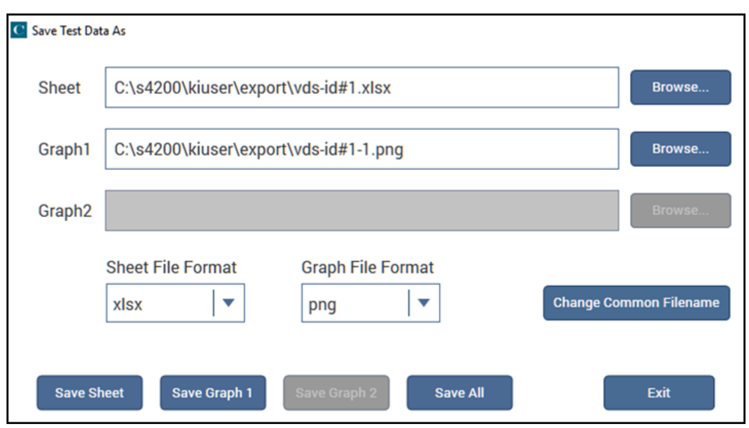
1. Select the project, subsite, test, or action.
2. Select **Analyze**.
3. Select **Save Data**.

Figure 76: Save Data button



The Save Test Data As dialog is displayed.

Figure 77: Save Test Data As dialog



4. In the **Sheet** field, select the file location and name.
5. Select the **Sheet File Format**.
6. Select **Save Sheet**.

To save the graph:

1. Select the project, test, or action.
2. Select **Analyze**.
3. Select **Save Data**. The Save Test Data As dialog is displayed.
4. In the **Graph1** and **Graph2** fields, select the file locations and graph names.
5. Select the **Graph File Format**.
6. Select **Save Graph 1** or **Save Graph 2**.

To save graph and Run sheet information:

1. Select the project, test, or action.
2. Select **Analyze**.
3. Select **Save Data**. The Save Test Data As dialog is displayed.
4. If you would like to use the same name for all graphs and sheets, enter the name in the **Change Common Filename** field. The file names are changed. No change is made to the file locations.
5. In the **Sheet** field, select the file location and name.
6. In the **Graph1** and **Graph2** fields, select the file locations and graph names.
7. Select the **Sheet File Format**.
8. Select the **Graph File Format**.
9. Select **Save All**.

Section 4

Customize Clarius

In this section:

Customize Clarius

To customize Clarius, you can:

- Add your own tests, actions, and projects to the Clarius library.
- Adjust the options in My Settings to set up your working environment and modify project execution. You can also set up custom GPIB abort settings and view information about Clarius.
- Hide or display the project tree, right pane, and Messages areas of the Clarius window.

Add objects to the library

You can add tests, devices, actions, and projects to the library. The new version of the object includes the settings you made to the object in the Configure pane. Once an object has been added, you can use it to create new objects in the project tree.

You cannot add sites and subsites to the library.

When you copy a project, it includes all test definitions, formulas, graph settings, and selected test runs.

NOTE

When you add objects to the library, you have the option to enter keywords. You can use these keywords to label the object with information that you can use to search for the object, such as including your name or a project name as part of the project.

Add a test to the library

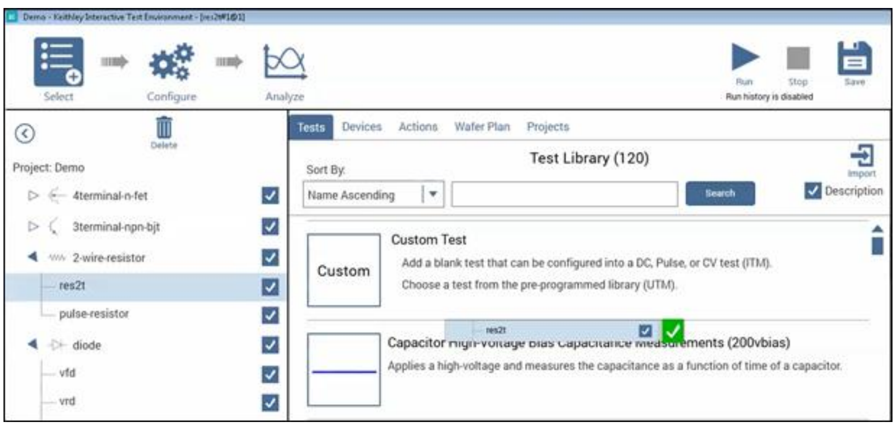
The following steps provide specifics on how to add a test to the library. You can follow the same basic procedure to add devices, actions, and projects. The primary difference is the type of object and which library the object is added to.

When you add a test to the library, the test runs that are selected in the Run History tab of the Analyze pane are included with the new test. However, notes and aliases that were assigned to the test runs are not included.

To add a test to the library:

1. In Clarius, set up the test so that it contains the settings you want the new library object to have.
2. Select **Analyze**.
3. In the Run History tab, select the test runs that you want to include in the new test.
4. Open **Select**.
5. Drag the test from the project tree to the library. You see a copy of the test and a checkmark, as shown in the following figure. The test is automatically added to the Tests library, regardless of the tab that is open. When you drop the test, a confirmation dialog is displayed.

Figure 78: Add a test to the Tests Library



6. Select **OK**. The Library Information Editor is displayed. Refer to Edit a library object you added (on page 4-3) to complete the Library Information Editor.

Add a device to the Device Library

You can copy a device from the project tree to the library to create a new device.

To submit a device to a library:

1. In Clarius, choose **Select**.
2. In the project tree, drag the device into the library. The test is automatically added to the Devices library, regardless of the tab that is open. A confirmation message is displayed.
3. Select **OK**. The Library Information Editor dialog is displayed. Refer to Edit a library object you added (on page 4-3) to complete the Library Information Editor.

Add an action to the library

The following example provides specifics on how to add an action to the library.

To add an action to the library:

1. In Clarius, set up the action so that it contains the settings you want the new library object to have.
2. Open the **Select** pane.
3. Drag the action from the project tree to the library. You see a copy of the action and a checkmark. The action is automatically added to the Actions library, regardless of the tab that is open. When you drop the action, a confirmation dialog is displayed.
4. Select **OK**. The Library Information Editor is displayed. Refer to Edit a library object you added (on page 4-3).

Add a project to the library

The following example provides specifics on how to add a project to the library.

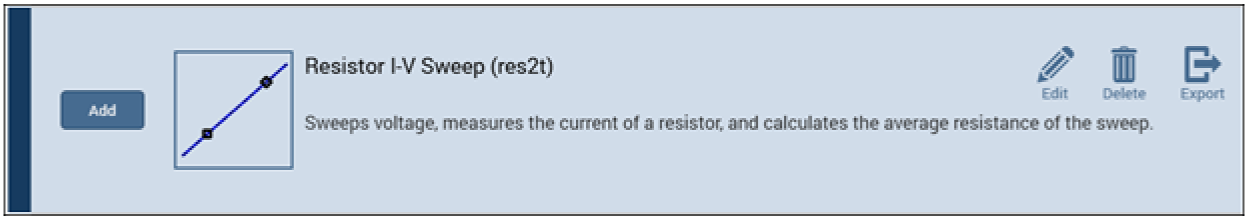
To add a project to the library:

1. In Clarius, set up the project so that it contains the settings you want the new library object to have.
2. Open the **Select** pane.
3. Drag the project from the project tree to the library. You see a copy of the project and a check mark. The project is added to the Projects library regardless of the tab that is open. When you drop the project, a confirmation dialog is displayed.
4. Select **OK**. The Library Information Editor is displayed. Refer to Edit a library object you added (on page 4-3).

Edit a library object you added

You can edit items that you added to a library. Items that can be edited have Edit, Delete, and Export options, as shown in the following figure.

Figure 79: Edit, Delete, and Export options for a library item



To edit an item:

1. Select the item in the library.
2. Select **Edit**.
3. Refer to Edit an object in the library (on page 4-4) for the options.

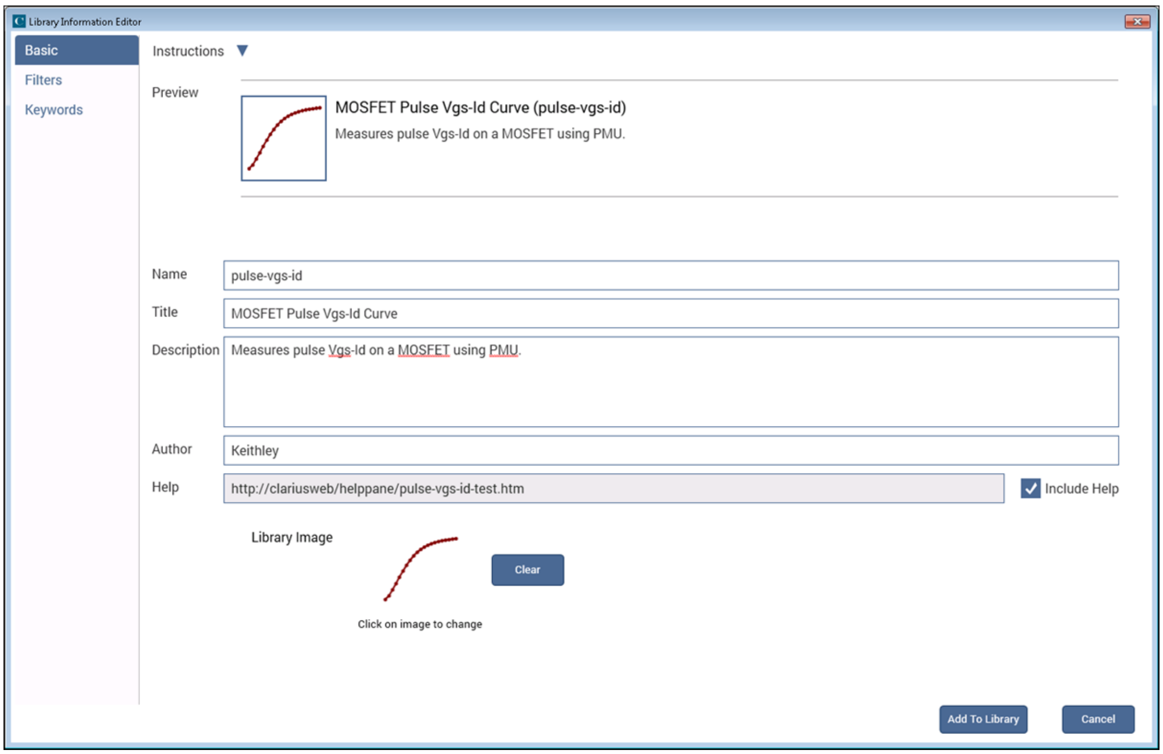
Edit an object in the library

The Library Information Editor allows you to change the information for a new library object. You can change information such as the name, description, graphic, help information, filters, and keywords.

The Library Information Editor is displayed when you drag an object from the project tree to the library. For objects that you created, you can also access it using Edit in the library.

As you make changes in the Basic tab, the Preview area displays the changes as they will appear in the library.

Figure 80: Library Information Editor



To change information for a library object:

1. In the Basic tab, complete the information as needed. Refer to the following table for the options.
2. Select the **Filters** tab. These options set the filters that will cause this item to appear in the library when you select the right-pane filters.
3. Select the filters that help a user find this item in the library.
4. Select the **Keywords** tab. These options determine what you can type in the library Search field to locate this item. You can use the Sort By options at the bottom of the lists to change the order of the entries in the Information Editor. It does not affect the order in the library.
5. Drag a keyword from the left to the right to add a keyword.

6. To remove a keyword, select the keyword and select **Delete**. This does not remove the keyword from the Global Keywords list.
7. To add a keyword, select **New** and type the keyword.

Options in the Information Editor	
Preview	Displays the changes you make as they will appear in the library.
Name	Type the new name. This is the name that is used in the library and the project tree.
Title	Type the title. This is used in the library.
Description	Type a brief description of the item. This is displayed in the library.
Author	Type information that identifies who created this item. This is available only through the Project Information Editor.
Help	Only editable when adding an object from the project tree to the library. This is the help file that is displayed in the right pane when Configure or Analyze is selected. It is also displayed when the item is selected in the library.
Include Help	Only displayed when you add an item from the project tree to the library. Determines if the existing help is included with the new item. If you want to include the help that was associated with the original object, select **Include Help**. Clear **Include Help** to keep the help from displaying (the Help pane will be blank). You cannot change the help link; you can only include or hide it.
Library Image	The image that is displayed in the library. Select the image to select a different image. Images should be 400x400 pixels in `png` format. Larger images display, but anything larger than 400x400 is cut off in the library display. To re-use an image from an older project, you may need to save the existing `bmp` image to `png` format. You can use a tool such as Microsoft™ Paint to convert the image. To leave the image area blank, select **Clear**.
Icon Image	Devices only. The image that is displayed in the project tree. Select the image to choose a different image. Images should be 80×80 pixels in `png` format. Larger images display, but anything larger than 80×is cut off in the project tree. To re-use an image from an older project, you may need to save the existing `bmp` image to `png` format. You can use a tool such as Microsoft Paint to convert the image. To leave the image area blank, select **Clear**.

Project tree display options

You can select whether or not Clarius displays the project tree if you need more workspace. To hide the project tree, select **<** at the top of the tree. Redisplay the project tree by selecting **>**.

Messages display options

You can hide the Messages area at the bottom of the Clarius window. To hide Messages, select v. To display the messages, select ^. To display the messages in the entire center pane, select the double ^ on the right.

My Settings

The options in My Settings allow you to set environment settings, run settings, GPIB abort, and logging preferences. You can also view revision and copyright information regarding Clarius.

Specify environment settings

The options in Environment Settings allow you to make 4200A system settings.

You can set:

- GPIB devices to reset on startup.
- PMUs to allow unsettled measurements.
- Site tracking for stress testing.
- Save as functionality.
- Where your projects are stored.
- Which project loads at startup.

Reset GPIB devices when Clarius starts

Select this option to reset GPIB devices in the system to their default settings when Clarius starts up.

PMU: Allow unsettled measurements

When this option is selected, all 4225-PMU instrument cards ignore the minimum timing versus measure range relationship.

This option is only recommended for advanced users. It causes spot mean results to be unsettled, which may cause a variety of operational issues, such as:

- Inconsistent current measure range-changing on the 4225-PMU or 4225-RPM.
- Lack of proper load-line effect compensation (LLEC) for the PMU.
- Lack of correlation between PMU or PMU and RPM results and SMU results.

For additional information, see “PMU minimum settling times versus current measure range” in the *Model 4200A-SCS Pulse Card (PGU and PMU) User's Manual*.

Track site during stress test

Select this option to track sites during a stress test.

When this option is selected, during the subsite stress test:

- The site identification box in the project tree is automatically updated to the presently started site. For example, if the run site was set to 3, after Run is selected, it is automatically reset to 1. It changes to 2 as soon as testing switches to site 2.
- All open subsite views are automatically switched to the presently running site. The Configure and Analyze panes will match the presently running site.
- If the subsite is opened, that also automatically converts to the present site during execution.

You can still change other sites manually during execution.

Enable the Save As button

You can enable Save As functionality, which allows you to save the open project to a different name into the same folder or a different folder. The folder location can be on the 4200A hard drive, a USB drive, or a mapped network drive.

To enable the Save As functionality:

1. Select **My Settings**.
2. Select **Environment Settings**.
3. Select **Enable Save As button**.

The Save As button is now available to the right of the Save button in the Clarius header.

When you select the Save As button, Clarius displays a file tree dialog. The name and path of the presently selected project is selected when the dialog is opened.

To save the project that is open in Clarius to a new name, enter the new file name in the Project name box. If the name already exists in the selected folder, you can cancel or overwrite the existing project. The newly created project opens after the Save As operation completes.

NOTE

Clarius automatically saves any unsaved changes to the open project before completing the Save As operation. The original and new projects will have identical configurations and data after completion of Save As. If you cancel the Save As operation, changes are not saved to the presently open project.

My Projects Directory

Use this option to change the directory where your projects are stored. Changing the project directory does not affect the open project or previously created projects. If you save the open project, it is saved to its original location.

Existing projects remain in the directory in which they were previously stored. The default location is `C:\s4200\kiuser\projects`.

In the My Projects Directory dialog, you can right-click to add a new folder, rename an existing folder, or delete a folder.

You can move projects to the new location using Windows Explorer. When moving projects, move the project folder and all its contents.

Project to load at startup

You can choose which project opens when you start Clarius.

If you select **Load last open project at startup**, the project that was open when you last closed Clarius is opened at startup.

If you select **Load default project at startup**, you can select a project to be the default project that opens at startup. Use Browse to select the default project.

Specify run settings

The options in the Run Settings dialog allow you to customize Clarius behavior when runs occur. The behaviors you can adjust are:

- If a project continues to run after an error.
- If interlock states are ignored.
- If autoscroll is active on the Analyze sheet during a test.
- If hardware is reinitialized after the run.
- If a test can be monitored while repeating continuously.
- The size of run history.

Continue to run after an error

This option determines if Clarius continues running a project sequence when it encounters errors in a test. When this option is selected, if Clarius encounters an error, it displays an error message in the message area at the bottom of the Clarius window and continues execution on the next test in the sequence.

Some examples of the types of errors that are ignored are:

- Tests for which no SMUs have been specified.
- Tests that are not configured or are improperly configured.

Ignore interlock state

The "Ignore interlock state" option allows you to choose if tests should continue if the interlock circuit is open.

When "Ignore interlock state" is selected and the 4200A-SCS interlock circuit is open, Clarius continues to execute tests. However, Clarius automatically limits the output voltage to a safe level, even if a test specifies a higher level.

If "Ignore interlock state" is cleared and the 4200A-SCS interlock circuit is open, Clarius displays a warning message and disables the execution of all tests.

Autoscroll the Analyze sheet during test

The autoscroll option determines whether or not the Analyze sheet scrolls during test execution.

When autoscroll is selected, the sheet scrolls so that new data is always displayed during test execution.

When autoscroll is cleared, the sheet does not display data until the test is complete.

Reinitialize hardware after run

When this option is selected, all instruments in the system return to their default settings after the test completes. The SMU output remains at the last programmed value for a brief time before being reinitialized.

When this option is cleared, all instruments in the system remain at their last programmed settings after the test completes.

WARNING

If "Reinitialize hardware after run" is cleared, all outputs remain at their last programmed levels after the test is completed. To prevent electrical shock that could cause injury or death, never make or break connections to the 4200A-SCS while the output is on.

Enable Monitor button

The "Enable Monitor button" option allows you to set a test to run continuously until stopped. When this option is selected, the Monitor icon is displayed to the left of the Run icon. Monitor allows you to run the test continuously until Stop is selected.

For more detail on using the monitor feature, refer to Monitor a test (on page 2-40).

Enable post run configuration of subsite data

If you change the output values that are included in the Analyze view of a subsite, the subsite data is rebuilt by default to include the changes to the output values. If you have a large data set, this can take a significant amount of time.

To turn off the automatic rebuild of subsite data, clear "Enable post run configuration of subsite data."

To enable the automatic rebuild of subsite data, select "Enable post run configuration of subsite data." When this option is selected, the maximum number of cycles for the subsite is limited to 128 and the maximum number of stresses is limited to 127.

Keep selected runs while running the test

By default, when a test is run, the previous run history is cleared and the most recent run history is selected.

The option **Keep selected runs while running the test** leaves the previous run history selected, in addition to the latest run, when the test is run again. This displays both runs on the graph.

If you choose to keep previous run histories, you also set the maximum number of selected runs. You can have up to 128 runs selected. When the run history reaches 128 run history selections, the oldest run history is cleared automatically when the test is run.

For more detail, refer to Run History selections when running a test (on page 3-14).

Run History Size

This setting allows you to control the number of test runs that are stored and displayed in the Run History pane.

When you run a test, the number of test runs stored in the Run History is limited to the number that is set. To maintain the number of runs, Clarius deletes the oldest unselected run. If there is existing data that exceeds the limit, only the oldest unselected run is deleted. If you have all runs selected, the most recent run is deleted when a new run is created. The runs continue to be numbered sequentially.

For example, if you have 200 existing runs and change the maximum run history size to 5, the number of runs in the Run History remains at 200. The next test run triggers the deletion of the oldest unselected run. The new run is numbered 201.

If you are running a project with subsite cycling or stressing, all existing runs are automatically unselected at the start of execution. Only data collected during the last run of the project is selected and displayed on the graphs. To save data between project runs, increase the run history size to be more than the number of cycles or stresses in your project.

If the run history size setting is less than the number of cycles or stresses in your project, a run is still generated for each cycle or stress.

Set graph defaults

For the Analyze graph, you can set the default line style when multiple run histories are selected. The lines for each run are differentiated by the line style, line color, or point markers. The Default Append Style selects the default that is used.

The default line style is applied to new tests. The changes do not affect existing tests.

Refer to Set the graph defaults (on page 3-24) for more information on setting these options.

Custom GPIB Abort Options

These options allow you to set the operations that occur when a GPIB abort is sent.

You can select that a `*RST` and `DCL` occur when an abort occurs.

If you enable Custom String, you can send a user-defined GPIB command string to the instrument.

NOTE

While most instruments will respond to the `DCL` command, if an instrument is not SCPI compliant, it will not respond. Erratic operation may result. Refer to the instruction manual for the instrument to determine its capabilities.

Instruments are added to this list when they are added to the system through KCon. Instruments added as General Purpose Test Instruments are shown. Refer to “Use KCon to add equipment to the 4200A-SCS” in *Model 4200A-SCS Parameter Analyzer Setup and Maintenance* for instruction.

To add a custom string, select the box to the left of the Custom String box for the instrument.

Logging

The Logging dialog allows you to select which messages are logged. It also allows you to opt in or opt out of sharing data with Keithley.

The Logging Level options selects the which error messages are logged based on severity. The logged error messages include messages at the selected logging level and messages at higher severity levels. The levels are listed from highest to lowest severe. For example, if you select **Informational**, the 4200A-SCS also logs error and warning messages, but not debug messages. Select **Debug** to log all messages.

The location of the logging directory is displayed in this dialog. It cannot be changed. If you contact Keithley for help troubleshooting a problem, you can copy the information in this directory and send it to Keithley to help diagnose the problem.

The "Share anonymous usage data with Keithley Instruments" allows you to opt in or out of sharing usage analytic data with Keithley.

NOTE

These options are also available in the Keithley Logging Client Control option in the Microsoft Windows notification area.

Section 5

Clarius Tools

In this section:

Tools

The Tools menu includes tools specific to the SMU, CVU, and PMU instruments that are installed in your 4200A-SCS. It also includes a tool that allows you to export data to a Microsoft Excel spreadsheet.

Instrument Tools

The Instrument Tools tab in the Tools dialog includes tools specific to the SMU, CVU, and PMU instruments that are installed in your 4200A-SCS.

The options include:

- **SMU Auto Calibration:** Recalibrates the current and voltage offsets for all source and measurement functions of all SMUs in the system. To maintain SMU performance specifications, you must autocalibrate the 4200A-SCS every 24 hours or any time after the ambient temperature has changed more than ±1 °C. Refer to Autocalibrate the SMUs (on page 5-2) for instructions.
- **CVU Connection Compensation:** Corrects offset and gain errors caused by the connections between the CVU and the device under test (DUT). Refer to Connection compensation (on page 5-3) for instructions.
- **CVU Real-Time Measure Mode:** Provides a direct real-time user interface to the CVU to help you set up and debug your system. For example, you can use it to confirm that contact has been made with the pads on a wafer. Refer to CVU Real-Time Measurement (on page 5-12) for instructions.
- **CVU Confidence Check:** CVU Confidence Check is a diagnostic tool that allows you to check the integrity of open and short connections and connections to a device under test (DUT). Refer to CVU Confidence Check (on page 5-14) for instructions.
- **PMU Connection Compensation:** Corrects errors caused by the connections between the 4225-PMU and the DUT. Refer to PMU connection compensation (on page 5-15) for instructions.

Autocalibrate the SMUs

To maintain SMU performance specifications, you must autocalibrate the 4200A-SCS every 24 hours or any time after the ambient temperature has changed more than ±1 °C.

The autocalibration routine recalibrates the current and voltage offsets for all source and measurement functions of all SMUs in the system.

NOTE

Before initiating a calibration, allow the system to warm up for at least 30 minutes. Clarius prevents autocalibration if the system is not sufficiently warmed up.

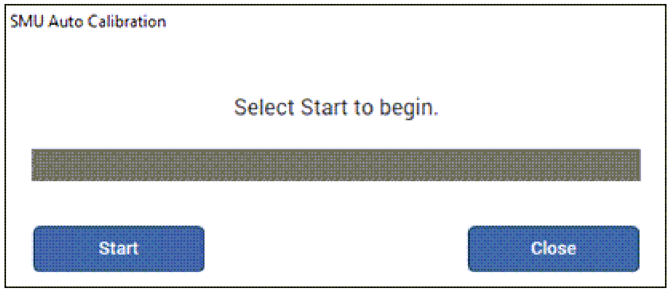
To autocalibrate:

1. Allow the system to warm up for at least 30 minutes.
2. Remove connections to all SMUs in the 4200A-SCS.
3. Open Clarius.
4. Select **Tools**.
5. Select **Instrument Tools**.

Figure 81: Clarius Tools dialog

6. Select **SMU Auto Calibration**. A warning is displayed.
7. Select **OK.** The SMU Auto Calibration dialog opens.

Figure 82: SMU Auto Calibration dialog



8. Select **Start**. A progress bar is displayed.
   When autocalibration is complete, the message "Auto calibration successfully completed" is displayed.
9. Select **Close**. Autocalibration is complete.

Connection compensation

You can correct offset and gain errors caused by the connections between the CVU and the device under test (DUT) by using connection compensation. To use correction, you:

- Generate connection compensation data for open, short, and load conditions.
- Enable CVU connection compensation.

When a test is run, the enabled compensation values are factored in by each measurement.

If open, short, or load compensation is disabled, those compensation values are not used by the test.

Once compensation values are stored, they are available to any project that uses a CVU.

NOTE

Update connection compensation any time the connection setup is changed or disturbed. Changes in temperature or humidity do not affect connection compensation.

NOTE

If the CVU is connected to a 4200A-CVIV Multi-Switch, run the `cvu-cviv-comp-collect` action. Refer to the *Model 4200A-CVIV Multi-Switch User's Manual* for details.

Use the following general guidelines to determine which correction needs to be done:

- **Open** correction: Offset correction for small capacitances (>1 MΩ, large impedance).
- **Short** correction: Offset correction for large capacitances (<10 Ω, small impedance).
- **Load** correction: Resistive load correction for gain error. Keithley recommends a load that is as close in impedance to the cabling system (100 Ω) as possible. The load must have an impedance versus frequency characteristic that is purely resistive over the frequency range of subsequent measurements.

Generate open connection compensation data

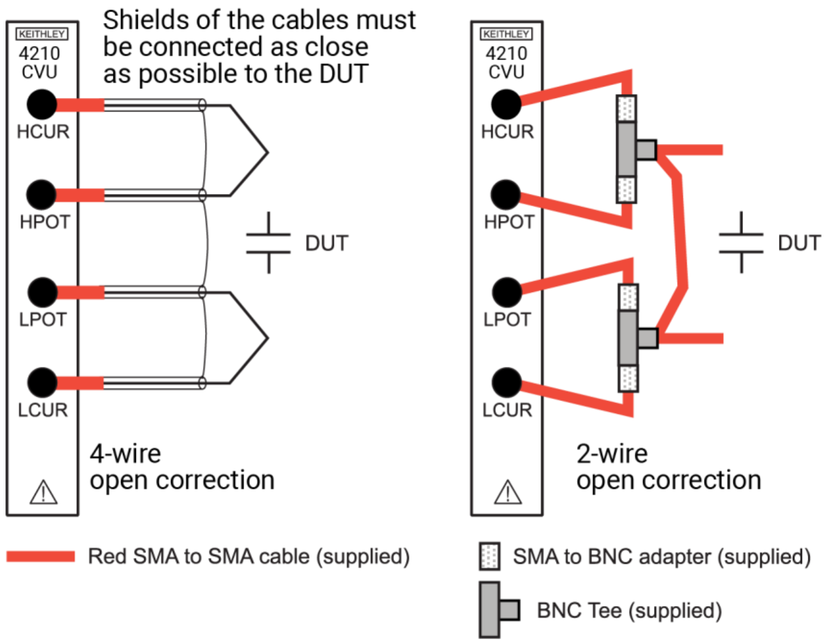
Open connection compensation is usually done to offset correction for small capacitances.

Open compensation is done with all the cables, adapters, switch matrices, and other hardware connections connected to the device under test in the test circuit. The probes must be lifted up or the device must be removed from the test fixture.

To generate open connection compensation data:

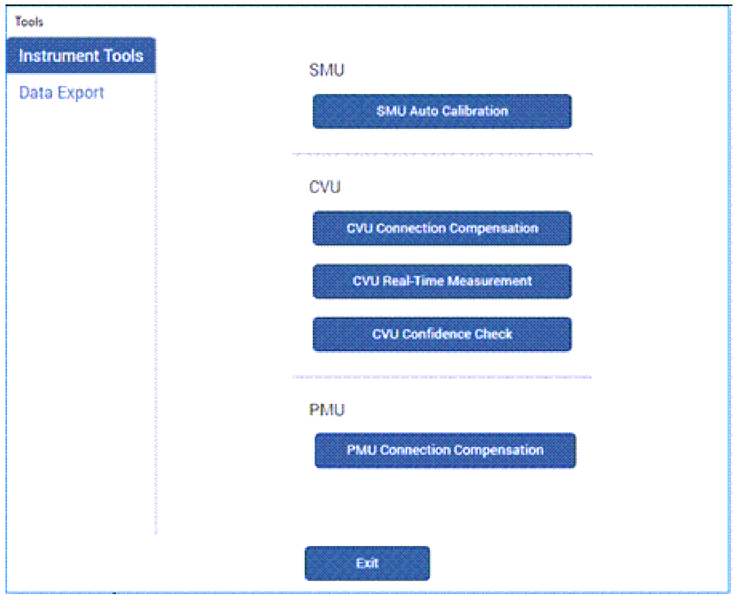
1. Make the connections to the CVU, as shown in the following figure. For remote (4-wire) sensing, the shields of the four SMA cables must be connected as close as possible.

Figure 83: Connections for open connection compensation CVU



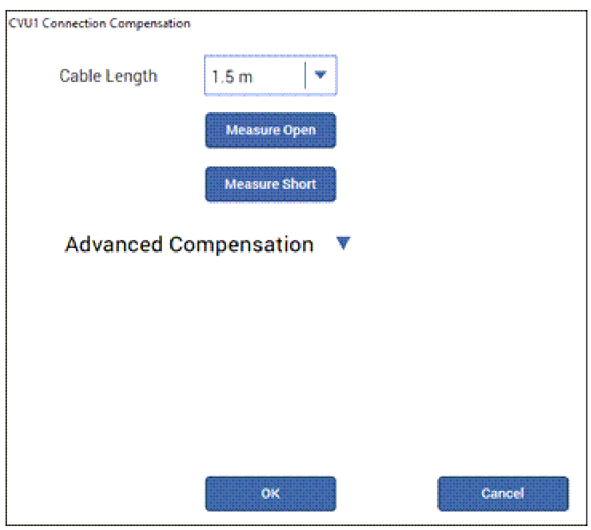
2. In Clarius, select **Tools**. The Clarius Tools dialog opens.

Figure 84: Clarius Tools dialog



3. Select **CVU Connection Compensation**.

Figure 85: CVU Connection Compensation dialog



4. Select the **cable length**. You can select:
   - **0 m:** Use if measurements are made at the terminals of the CVU (no cables).
   - **1.5 m:** Use with the standard red SMA cables (part number CA-447B) that are supplied with the CVU.
   - **3 m:** Use with the red SMA cables (part number CA-446A or part number CA-621A). CA-446A cables are supplied with the 4200-CVU-PROBER-KIT. You can also use this setting if you are using a switching matrix.
   - **Custom:** Cable length coefficients are measured by the user using the Measure Custom Cable Length option under Advanced Compensation.
5. If you selected Custom cable length, select **Advanced Compensation** and select **Measure Custom Cable Length**. Follow the on-screen instructions.
6. If you are using a switching matrix, close the matrix switches that connect the CVU to the open. Refer to “Using Switching Matrices” in *Model 4200A-SCS Prober and External Instrument Control*.
7. In the Clarius CVU Connection Compensation dialog, select **Measure Open**.
8. Follow the instructions.
9. Select **OK**.

Generate short connection compensation data

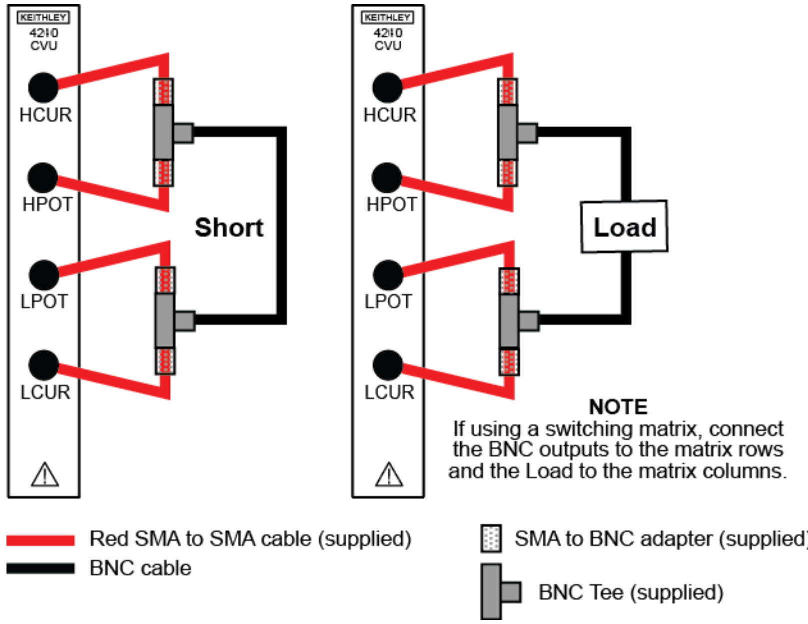
Short connection compensation is usually done to offset correction for large capacitances.

Short compensation is done by connecting all the CVU terminals directly together. A known short is connected to the CVU terminals through all the cables, adapters, and probes that may be in the test circuit. You can make a short at the wafer level by shorting all probes together.

To generate short connection compensation data:

1. Make the connections to the CVU, as shown in the following figure. For remote (4-wire) sensing, the shields of the four SMA cables must be connected.

Figure 86: Connections for short and load connection compensation

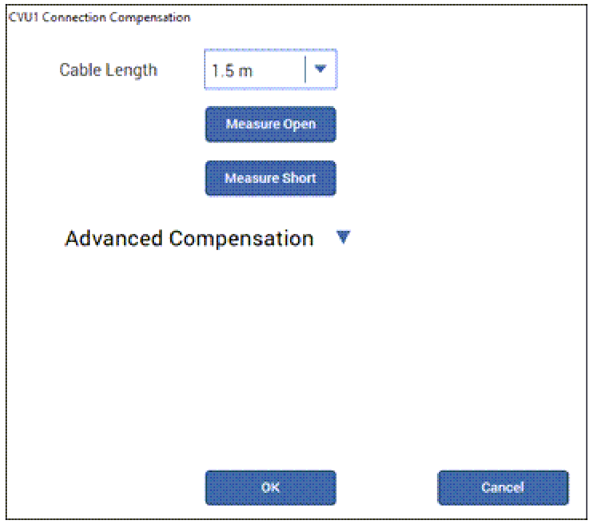


2. In Clarius, select **Tools**. The Clarius Tools dialog opens.

Figure 87: Clarius Tools dialog

3. Select **CVU Connection Compensation**.

Figure 88: CVU Connection Compensation dialog



4. Select the **cable length**. You can select:
   - **0 m:** Use if measurements are made at the terminals of the CVU (no cables).
   - **1.5 m:** Use with the standard red SMA cables (part number CA-447B) that are supplied with the CVU.
   - **3 m:** Use with the red SMA cables (part number CA-446A or part number CA-621A). CA-446A cables are supplied with the 4200-CVU-PROBER-KIT. You can also use this setting if you are using a switching matrix.
   - **Custom:** Cable length coefficients are measured by the user using the Measure Custom Cable Length option under Advanced Compensation.
5. If you selected Custom cable length, select **Advanced Compensation** and select **Measure Custom Cable Length**. Follow the on-screen instructions.
6. If you are using a switching matrix, close the matrix switches that connect the CVU to the open. Refer to “Using Switching Matrices” in *Model 4200A-SCS Prober and External Instrument Control*.
7. In the Clarius CVU Connection Compensation dialog, select **Measure Short**.
8. Follow the instructions.
9. Select **OK**.

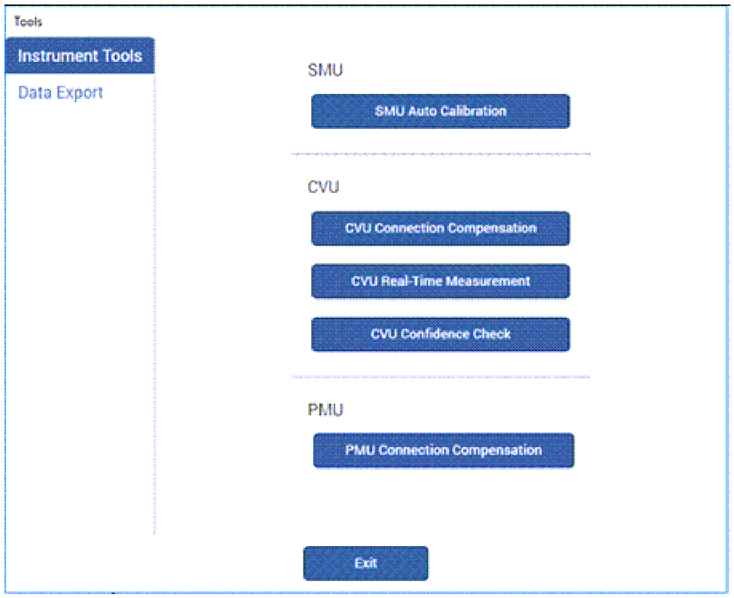
Generate load connection compensation data

Loads are reference resistors, typically 50 or 100 Ω or less, that must be resistive and constant over the entire frequency range (1 kHz to 10 MHz). A load is connected to the output terminals using all the cables, adapters, probes, and other hardware in the test circuit.

To generate load correction data:

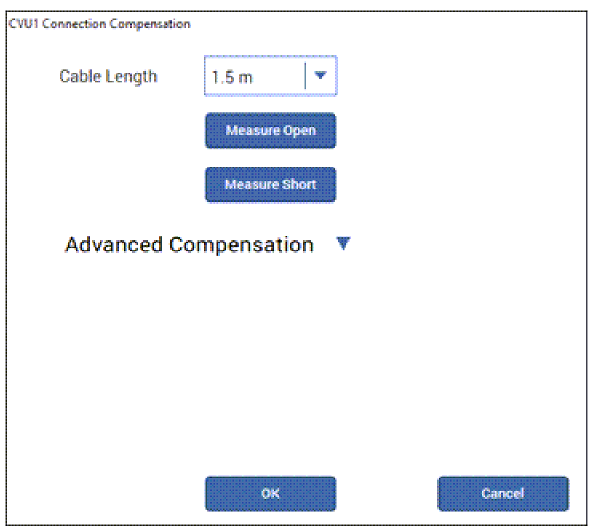
1. Make the connections to the CVU. See "Test connections for a switching matrix" in the *Model 4200A-SCS Capacitance-Voltage Unit (CVU) User's Manual*.
2. In Clarius, select **Tools**. The Clarius Tools dialog opens.

Figure 89: Clarius Tools dialog



3. Select **CVU Connection Compensation**.

Figure 90: CVU Connection Compensation dialog



4. Select the **cable length**. You can select:
   - **0 m:** Use if measurements are made at the terminals of the CVU (no cables).
   - **1.5 m:** Use with the standard red SMA cables (part number CA-447B) that are supplied with the CVU.
   - **3 m:** Use with the red SMA cables (part number CA-446A or part number CA-621A). CA-446A cables are supplied with the 4200-CVU-PROBER-KIT. You can also use this setting if you are using a switching matrix.
   - **Custom:** Cable length coefficients are measured by the user using the Measure Custom Cable Length option under Advanced Compensation.
5. If you selected Custom cable length, select **Advanced Compensation** and select **Measure Custom Cable Length**. Follow the on-screen instructions.
6. If you are using a switching matrix, close the matrix switches that connect the CVU to the open. Refer to “Using Switching Matrices” in *Model 4200A-SCS Prober and External Instrument Control*.
7. If it is not open, select **Advanced Compensation**.
8. In Measure Load, enter the value of the load in ohms.
9. Select **Measure Load**.
10. Follow the instructions.
11. Select **OK**.

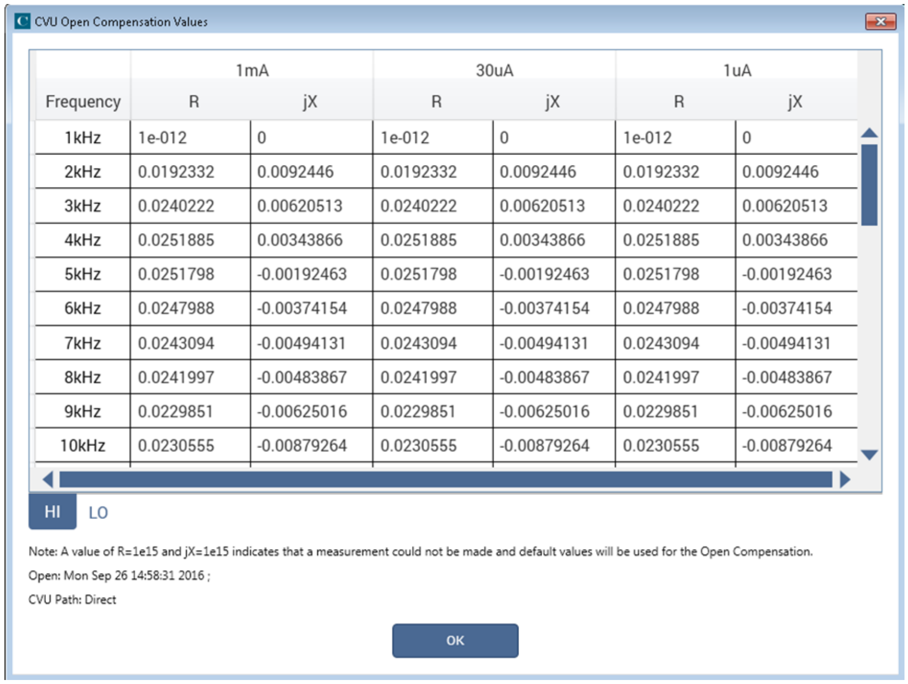
Compensation data

You can view the compensation data. Clarius lists R and jX compensation values for every test frequency and measurement range for open, short, and load.

To view the data generated by connection compensation:

1. In Clarius, select **Tools**.
2. Select **CVU Connection Compensation**.
3. Select **Advanced Compensation**.
4. Next to View Compensation Data, select the data you would like to display: **Open**, **Short**, or **Load**.
5. Select **View Compensation Data**.
6. Select the **HI** tab to review the high side values.
7. Select the **LO** tab to review the low side values.

Figure 91: Open compensation values example


CVU Open Compensation Values

	1mA		30uA		1uA	
Frequency	R	jX	R	jX	R	jX
1kHz	1e-012	0	1e-012	0	1e-012	0
2kHz	0.0192332	0.0092446	0.0192332	0.0092446	0.0192332	0.0092446
3kHz	0.0240222	0.00620513	0.0240222	0.00620513	0.0240222	0.00620513
4kHz	0.0251885	0.00343866	0.0251885	0.00343866	0.0251885	0.00343866
5kHz	0.0251798	-0.00192463	0.0251798	-0.00192463	0.0251798	-0.00192463
6kHz	0.0247988	-0.00374154	0.0247988	-0.00374154	0.0247988	-0.00374154
7kHz	0.0243094	-0.00494131	0.0243094	-0.00494131	0.0243094	-0.00494131
8kHz	0.0241997	-0.00483867	0.0241997	-0.00483867	0.0241997	-0.00483867
9kHz	0.0229851	-0.00625016	0.0229851	-0.00625016	0.0229851	-0.00625016
10kHz	0.0230555	-0.00879264	0.0230555	-0.00879264	0.0230555	-0.00879264

HI LO

Note: A value of R=1e15 and jX=1e15 indicates that a measurement could not be made and default values will be used for the Open Compensation.

Open: Mon Sep 26 14:58:31 2016 ;

CVU Path: Direct

OK

Enable compensation

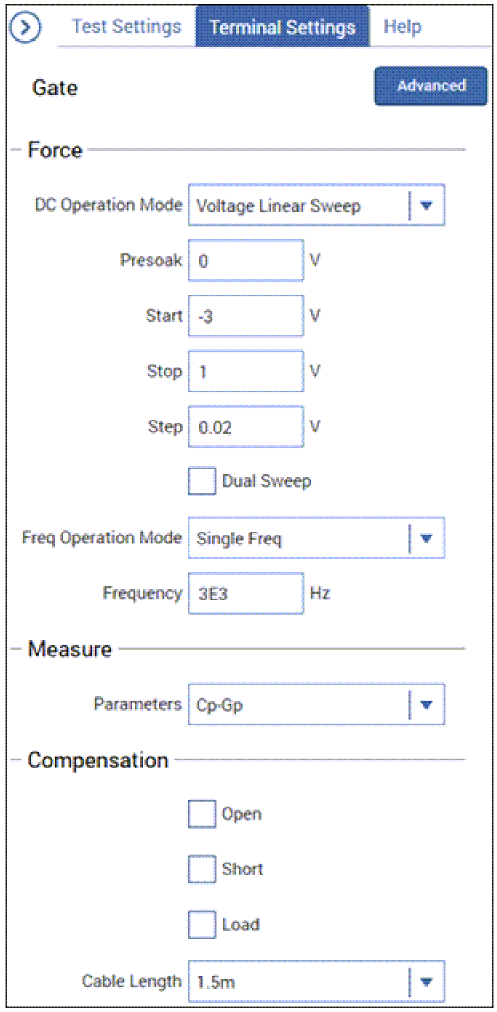
To use the values generated by connection compensation, you need to enable compensation for each test.

When compensation is enabled, the most recently acquired CVU compensation data is applied. Compensation values can be gathered using the CVU Connection Compensation option in Tools or through actions and user modules.

To enable compensation:

1. Select the test from the project tree.
2. Select **Configure**.
3. Select the terminal in the center pane.
4. In the right pane, select **Terminal Settings**.
5. Under Compensation, select the types of compensation as needed.
6. Make sure **Cable Length** is the same as the setting that was used in the Tools > CVU Connection Compensation dialog to generate connection compensation data.

Figure 92: Enable connection compensation



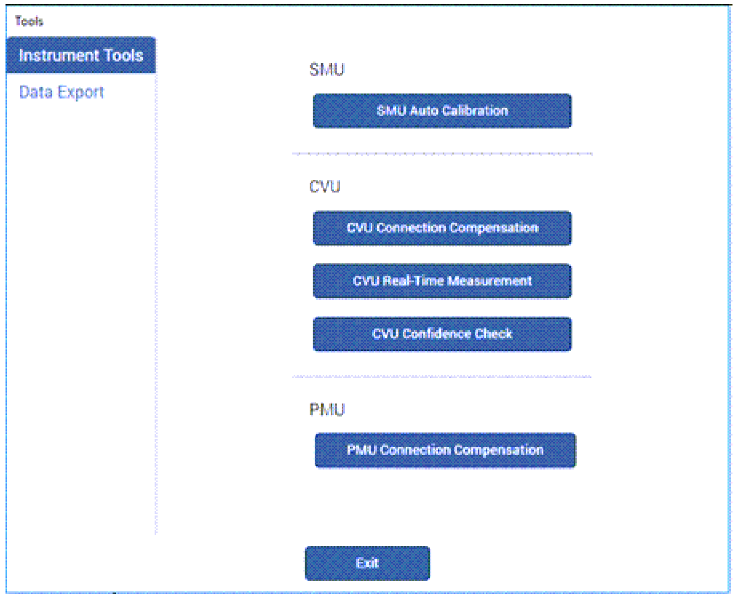
CVU Real-Time Measurement

The CVU Real-Time Measurement provides a direct real-time user interface to the CVU to help you set up and debug your system. For example, you can use it to confirm that contact has been made with the pads on a wafer. The measurements are independent of the open and short confidence checks.

To make real-time measurements:

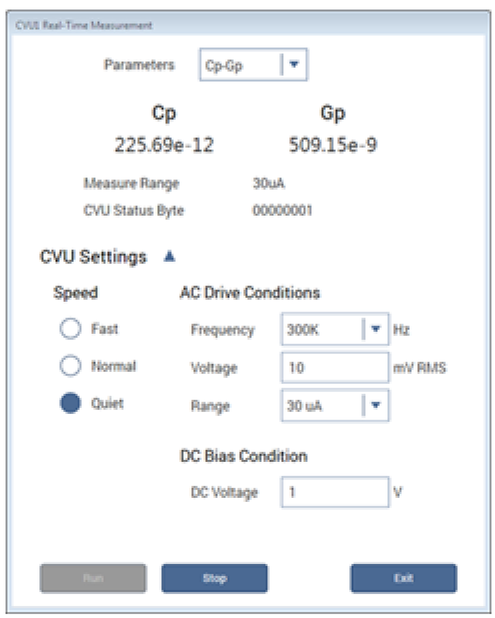
1. In Clarius, select **Tools**. The Clarius Tools dialog opens.

Figure 93: Clarius Tools dialog



2. Select **Instrument Tools**.
3. Select **CVU Real-Time Measurement**.

Figure 94: Real-Time Measurement dialog



4. Select the parameters for which you want to return results.
5. Set the Speed, AC Drive Conditions, and DC Bias Conditions for the conditions you want to test.
6. Select **Run**. The results for the selected parameters are displayed at the top of the dialog.

CVU Confidence Check

CVU Confidence Check is a diagnostic tool that allows you to check the integrity of open and short connections and connections to a device under test (DUT). When the CVU is connected to the DUT, the Confidence Check displays the measured readings in real time in the Messages area of Clarius.

An open or short confidence check makes a measurement on the high and the low sides of the test circuit.

The open check is not compatible with the 4200A-CVIV Bias Tee configurations.

To get the best results from a confidence check:

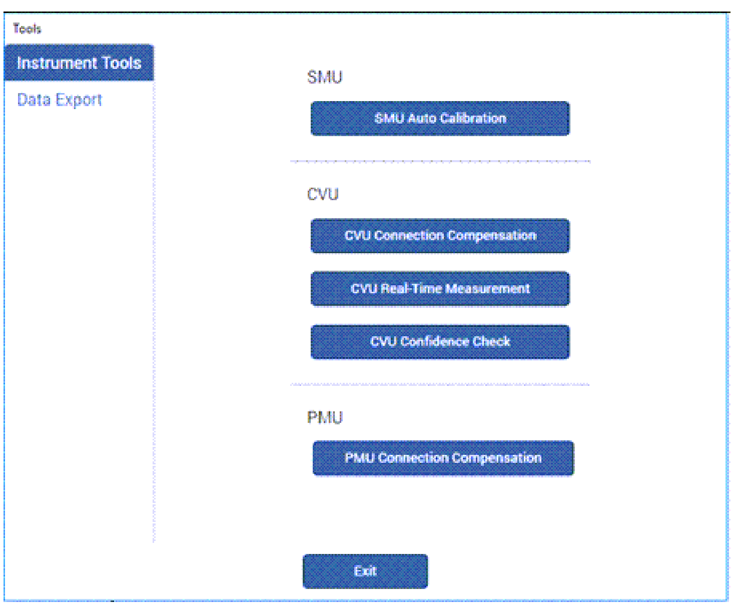
- Use the red CA-446A, CA-621A, or CA-447B cables or equivalent.
- If applicable, make sure the prober chuck is connected.
- If you are using a switching matrix, make sure all channels are closed.

Run an open check and short check

To run a CVU confidence check:

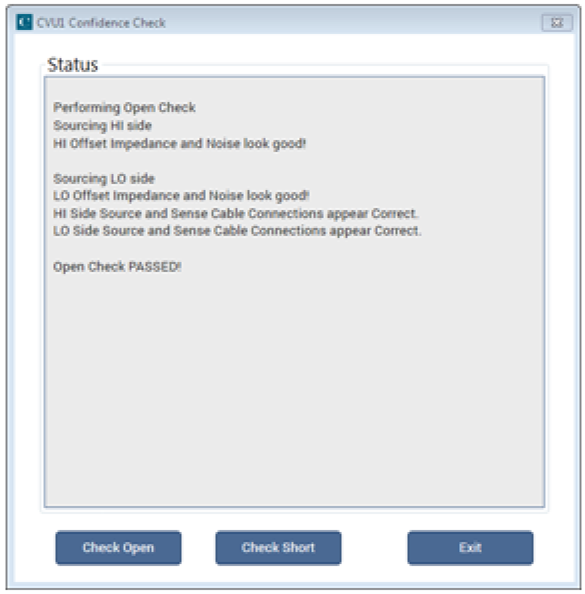
1. If you are using a switching matrix, connect the switching matrix to the CVU and DUT or the short as explained in “Test connections for a switching matrix” in the *Model 4200A-SCS Capacitance-Voltage Unit (CVU) User's Manual*.
2. For the short check, close the matrix switches to connect the CVU to the DUT or short. For the open check, also close the matrix switches, but lift the probes or disconnect the DUT.
3. In Clarius, select **Tools**. The Clarius Tools dialog opens.

Figure 95: Clarius Tools dialog



4. Select **CVU Confidence Check**.
5. Select **Check Open** or **Check Short**.

Figure 96: CVU Confidence Check dialog



6. Follow the instructions and select **OK**.

When the check is complete, the dialog displays the results of the test. If the test failed, the results include suggestions for troubleshooting.

PMU connection compensation

You can correct errors caused by connections and cable length between the 4225-PMU and the device under test (DUT) by using connection compensation. When connection compensation is enabled, the default or measured compensation values are factored into each DUT measurement.

Connection compensation includes short and offset current compensation options.

You have the option to use either default connection compensation values (PMU or RPM) or custom connection compensation values. The default values can be used for typical connection setups that use the supplied cables. The custom connection compensated values are generated when connection compensation is done from the Clarius software. The custom values provide optimum compensation. Custom connection compensation data is generated for offset current and short conditions. The custom connection compensation values can be enabled or disabled from a test in Clarius.

If connection compensation is disabled, the compensation values will not be applied to the measurements.

NOTE

For optimum performance, you should do connection compensation any time the connection setup is changed or disturbed. Changes in temperature or humidity do not affect connection compensation.

Short compensation

NOTE

For UTMs, the default connection compensation values for short can only be enabled using the LPT function `pulse_conncomp`.

You can perform short compensation to remove measurement errors due to stray resistance in your test configuration. When you run short connection compensation, the following status messages are generated:

- `Starting PMU Cable Compensation...`
- `R = % Ohms`
- `PMU Cable Compensation complete.`
- `% = value (V and I measured, Ohms calculated).`

Offset current compensation

Error currents can be introduced into your pulsed measurements setup by PMUs. PMU offset compensation reduces error currents by subtracting measurements taken at 0 V from all subsequent readings.

Perform connection compensation

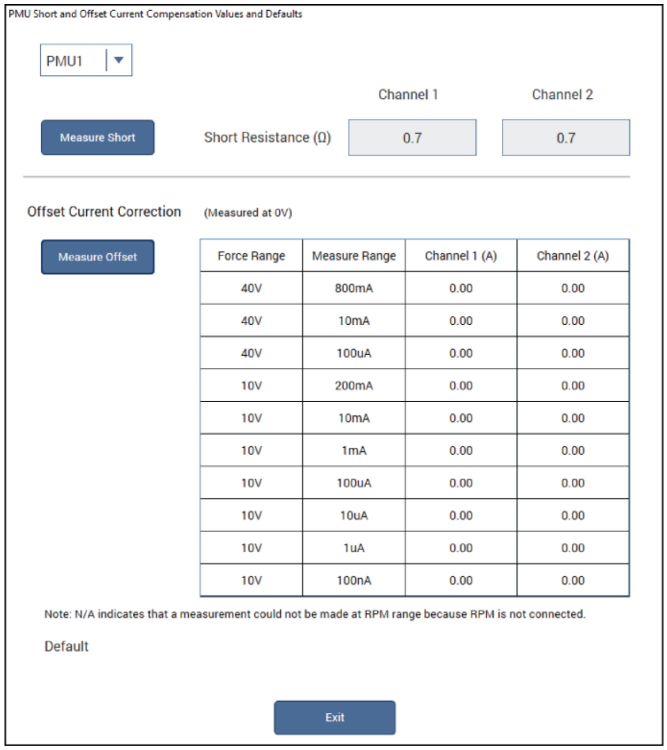
To compensate for connections:

1. In Clarius, select **Tools**. The Clarius Tools dialog opens.

Figure 97: Clarius Tools dialog

2. Select **PMU Connection Compensation**. The Short and Offset Current Connection Compensation Values and Defaults dialog opens.
3. From the PMU list, select the PMU that you want to perform compensation for.

Figure 98: PMU Connection Compensation dialog



4. To perform short connection compensation, select **Measure Short,** then follow the on-screen instructions or replace the DUT in the test fixture with a short.
5. To perform offset current compensation, select **Measure Offset**, then follow the on-screen instructions.

Compensation results are displayed when compensation is complete. If an error occurred, it is displayed in the Clarius Messages area. The compensation data is displayed in the Short and Offset Current Connection Compensation Values and Defaults dialog.

If your test setup uses both PMU channels, you will have new custom data for both channels.

Enabling connection compensation

NOTE

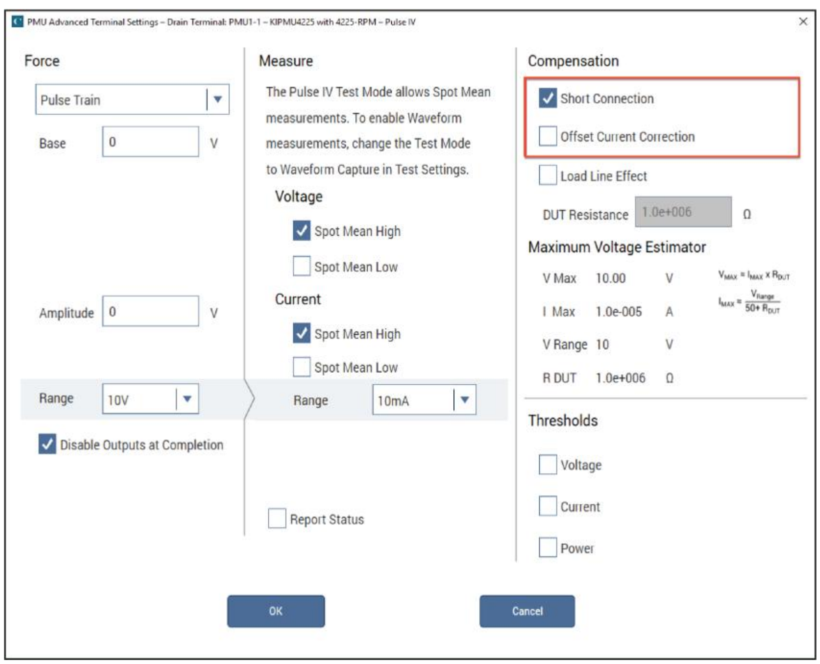
This procedure is for ITMs. For UTMs, you need to enable connection compensation data using the LPT command `pulse_conncomp` or `setmode` functions.

To apply the connection compensation data to DUT measurements, you must enable connection compensation for the test.

To enable connection compensation:

1. Select the test.
2. Select **Configure**.
3. Select the terminal to be compensated.
4. In the right pane, select **Terminal Settings**.
5. Select **Advanced**. The PMU Advanced Terminal Settings dialog is displayed.
6. Select either **Short Connection** or **Offset Current Correction**.

Figure 99: Enabling connection compensation



7. Select **OK**.
8. To disable connection compensation, clear either **Short Connection** or **Offset Current Correction**, then select **OK**. When disabled, connection compensation values are not applied to DUT measurements.

Data Export tool

You can export test data from Clarius to a file in Microsoft Excel `.xlsx` format.

You can export all or specific data for devices, tests, and runs in a project. If you have a subsite, you can also export data at the subsite level.

These options are available from the Data Export option in the Tools dialog.

Select specific runs to export

When you export data, you can choose to export only selected runs. You must define these runs in Analyze before opening the Data Export dialog.

NOTE

Refer to Run History (on page 3-9) for more information on finding and selecting test runs.

To define the selected runs:

1. Select **Analyze**.
2. In the Run History tab, select the runs for each test that you want to export.

Export data

NOTE

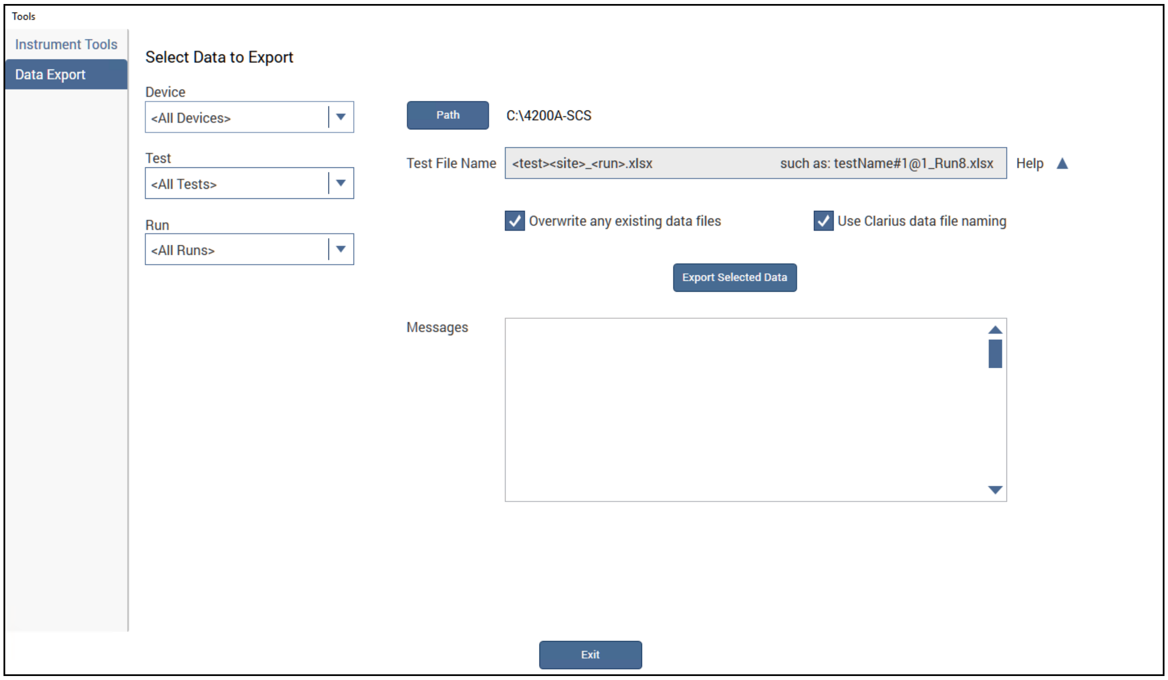
If you are exporting data from a project with a subsite, see Export subsite data (on page 5-21).

You can export all data or a subset of data.

To export data:

1. Select **Tools**.
2. From the left pane, select **Data Export**. The Select Data to Export dialog is displayed.

Figure 100: Select Data to Export dialog



3. From the Device list, select **<All Devices>** or a specific device.
4. From the Test list, select **<All Tests>** or a specific test.
5. From the Run list, select one of the following:
   - **<All Runs>:** Exports data for all runs for the selected devices and tests.
   - **<Latest Runs>:** Exports the data in the most recent run for each of the selected devices and tests.
   - **<Selected Runs>:** Exports the data in the runs that are selected in the Run History for each of the selected devices and tests. Refer to Select specific runs to export (on page 5-19) to select the runs.
6. If you chose <Selected Runs>, you can also select an option to combine the runs into one file.
7. Select **Path** to set the location of the exported file.
8. To use Clarius naming conventions for the file names, select **Use Clarius data file naming**.
9. If you did not select Clarius file naming, enter text and patterns to define a custom file name for the **Test File Name**. Select **Help** for the available patterns, character limitations, and examples.
10. To overwrite existing files, select **Overwrite any existing data files**.

NOTE

If you clear the Overwrite field, you must use a new name for each export. If you selected “Use Clarius data file naming,” errors occur if you save the same export multiple times.

11. Select **Export Selected Data**. Data is exported to a file in the selected path.

A progress bar is displayed during the export and the results of the export are listed in Messages.

Export data options for subsites

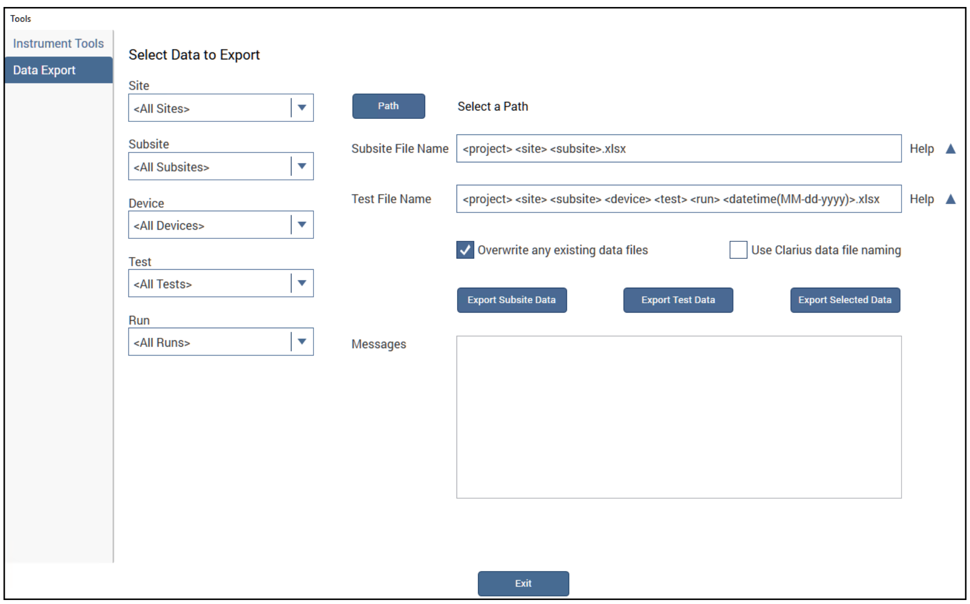
If the project includes a subsite, you have additional export options.

You can export:

- **Subsite Data:** Exports data for the selected sites and subsites.
- **Test Data:** Exports test data for the selected devices, tests, and runs.
- **Selected Data:** Allows you to specify which data is exported. If you select all options, the entire data set is exported.

An example of the Select Data to Export dialog when the project contains a subsite is shown in the following figure.

Figure 101: Select Data to Export dialog for subsites



Export subsite data only

When you export subsite data, any selections you make for devices, tests, and runs are ignored. The Test File Name is also ignored.

To export data for a subsite:

1. Select **Tools**.
2. From the left pane, select **Data Export**. The Select Data to Export dialog is displayed.
3. From the Site list, select **<All Sites>** or a specific site.
4. From the Subsite list, select **<All Subsites>** or a specific subsite.
5. Select **Path** to set the location of the exported file.
6. To use Clarius naming conventions for the file names, select **Use Clarius data file naming**.

7. If you did not select Clarius file naming, in **Subsite File Name**, enter text and patterns to define a custom file name. Select **Help** for the available patterns, character limitations, and examples.
8. To overwrite existing files, select **Overwrite any existing data files**.

NOTE

If you clear the Overwrite field, you must use a new name for each export. If you selected “Use Clarius data file naming,” errors occur if you save the same export multiple times.

9. Select **Export Subsite Data**. Data is exported to a file in the selected path.

A progress bar is displayed during the export and the results of the export are listed in the Messages box.

Export test data only

When you export test data, any selections you make for sites and subsites are ignored. The Subsite File Name is also ignored.

To export data for tests:

1. Select **Tools**.
2. From the left pane, select **Data Export**. The Select Data to Export dialog is displayed.
3. From the Device list, select **<All Devices>** or a specific device.
4. From the Test list, select **<All Tests>** or a specific test.
5. From the Run list, select one of the following:
   - **<All Runs>:** Exports data for all runs for the selected devices and tests.
   - **<Latest Runs>:** Exports the data in the most recent run for each of the selected devices and tests.
   - **<Selected Runs>:** Exports the data in the runs that are selected in the Run History for each of the selected devices and tests. Refer to Select specific runs to export (on page 5-19) to select the runs.
6. Select **Path** to set the location of the exported file.
7. To use Clarius naming conventions for the file names, select **Use Clarius data file naming**.
8. If you did not select Clarius file naming, in **Test File Name**, enter text and patterns to define a custom file name. Select **Help** for the available patterns, character limitations, and examples.
9. To overwrite existing files, select **Overwrite any existing data files**.

NOTE

If you clear the Overwrite field, you must use a new name for each export. If you selected “Use Clarius data file naming,” errors occur if you save the same export multiple times.

10. Select **Export Test Data**. Data is exported to a file or files in the selected path.

A progress bar is displayed during the export and the results of the export are listed in the Messages box.

Export selected or all data

You can export selected data for a project. You can also use this option to export all subsite and test data.

To export selected data:

1. Select **Tools**.
2. From the left navigation pane, select **Data Export**. The Select Data to Export dialog is displayed.
3. From the Site list, select **<All Sites>** or a specific site.
4. From the Subsite list, select **<All Subsites>** or a specific subsite.
5. From the Device list, select **<All Devices>** or a specific device.
6. From the Test list, select **<All Tests>** or a specific test.
7. From the Run list, select one of the following:
   - **<All Runs>:** Exports data for all runs for the selected sites, subsites, devices, and tests.
   - **<Latest Runs>:** Exports the data in the most recent run for the selections.
   - **<Selected Runs>:** Exports the data in the runs that are selected in the Run History for each of the selections. Refer to Select specific runs to export (on page 5-19) to select the runs.
8. If you chose **<Selected Runs>**, you can also choose to combine the runs into one file.
9. Select **Path** to set the location of the exported file.
10. To use Clarius naming conventions for the file names, select **Use Clarius data file naming**.
11. If you did not select Clarius file naming, enter text and patterns to define custom file names for the **Test Files Name** and **Subsite File Name**. Select **Help** for the available patterns, character limitations, and examples.
12. To overwrite existing files, select **Overwrite any existing data files**.

NOTE

If you clear the Overwrite field, you must use a new name for each export. If you selected “Use Clarius data file naming,” errors occur if you save the same export multiple times.

13. Select **Export Selected Data**. Data is exported to a file or files in the selected path.

A progress bar is displayed during the export and the results of the export are listed in the Messages box.

Data compression

In this section:

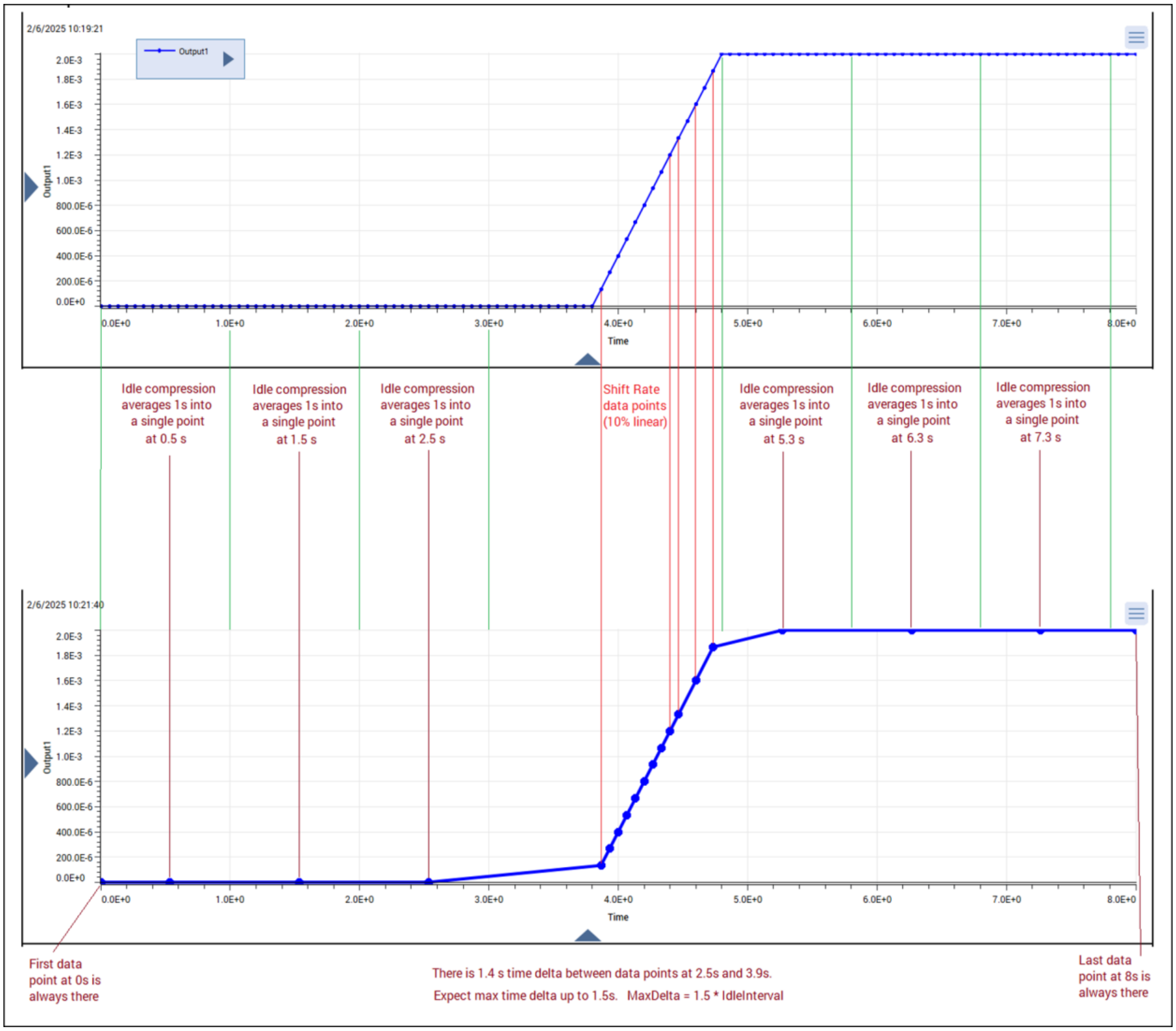
Data compression

The data compression feature of the 4200A-SCS minimizes data that is not of interest. It allows you to collect uncompressed data from areas that are of interest while compressing data from other areas of the data acquisition. For example, in a breakdown test, you may want to store uncompressed data points near a breakdown transition, but compress other data. In the following figure, data is compressed by the Idle Compression rule while the test is idle. When the data shifts at a rate that matches the definition of the shift rate rule, data points are compared to previously returned data. If the change ratio of one point exceeds the defined percentage when compared to previously returned data point, the data is returned.

The following figure demonstrates the differences in the Analyze graphs for compressed and uncompressed data. In the figure, the top graph includes all data points for the test with no data compression. The bottom graph contains the compressed version of the same test. In this example, the following data compression rules were defined:

- **Shift Rate rule:** Configured to post a data point whenever a value of the data sample changed 10% or more compared to the previously posted data point.
- **Idle Compression rule:** Configured to post a single data point as an average of 1 second of data samples if the Shift Rate compression rule did not post a point in that 1 second interval.

Figure 102: Data compression graph comparison



In addition to simplifying the display of data, data compression allows you to execute long-running tests without exceeding the maximum number of points allowed in Clarius.

Averaging for compression can be set up to be linear or logarithmic. When the compression type is set to linear, Clarius averages a number of samples. When the compression type is set to log, the average number of samples per decade are returned.

Data compression is available for all 4200A-SCS cards (SMU, CVU, PGU, and PMU) and external instruments. All measurement types are supported. The first and last data points in a test are always returned, regardless of the settings for the compression rules. These points are not included in the data compression calculations.

Readings from a compressed test may be delayed because of the time required to calculate the compression rules.

This feature is available in Clarius+ versions 1.14 and later. Clarius version 1.14 projects that are configured to use data compression cannot be used in previous versions of the Clarius+ software.

In addition to the information in this section, for information that demonstrates how to use data compression, refer to Application Note 1KW-74148, *Long-Term Data Collection and Breakdown Testing Using Data Compression in the Keithley 4200A-SCS Parameter Analyzer*.

Compression rule types

You can define up to ten compression rules. Depending on the type of rule, you can enable multiple rules. The rule types are:

- **Shift Rate:** Compress data based on a change in the output rate.
- **Idle Compression:** Compress data until a defined number of seconds or steps has occurred when all other data compression rules are idle.
- **Beginning of the Test:** Compress data at the beginning of a test until a defined number of seconds or steps has occurred.
- **End of the Test:** Compress data at the end of a test until a defined number of seconds or steps has occurred.
- **Time Interval:** Compress data in a defined period.

For the rules that use time, you must have a data series with a time data type. The duration values for time-based rules (especially the beginning of test and time interval) are offset from the first timestamped sample received. For example, if a beginning of test rule is defined for the first 10 seconds of a test and the first timestamped sample is 2.0 seconds, the rule is in effect until the timestamped samples reach 12.0 seconds.

When multiple rules are defined and enabled, the order of rule hierarchy is end of the test, beginning of the test, time interval, shift rate, then idle compression.

Shift Rate

When the Shift Rate compression rule is selected, data points are compared to previously returned data. If the change ratio of one point exceeds the defined percentage when compared to the previously returned data point, the data is returned. Shift rate stops showing complete data when the data points no longer meet the shift rate rule criteria. This will override the Idle Compression rule, so that data is compressed until a shift occurs, then uncompressed data is gathered until the data no longer exceeds the defined percentage.

You can set multiple shift rate rules for different data series in a test.

To set the shift rate, select the Data Series to compare against the percentage change of the output. Set the % Change to Output to define the absolute value percentage change. The values of the specified Data Series are used to determine if the shift rate has occurred. However, when a shift occurs, the data is returned for all data series.

You can also set the type of data return to be Linear or Log. When the compression type is set to linear, Clarius returns linear data. When the type is set to log, the samples per decade are returned.

The Lowest Line setting prevents zero noise value changes. When Lowest Line is selected, absolute values below the defined value are excluded from the Shift Rate evaluation. The limits for the lowest line are from 1E−15 to 1.0. For example, if the current value changes from 0 µA to 10 µA, an attempt to verify the shift rate at values close to 0 does not make sense, so those values are ignored.

This rule is superseded by the Beginning of the Test, End of the Test, and Time Interval rules. The shift rate is not checked for the duration or number of samples defined for the beginning or end of the test.

Beginning of the Test

The Beginning of the Test rule determines what kind of compression, if any, occurs during the initial part of the test. You can set this rule to compress or not compress data. Only one Beginning of the Test rule can be enabled in a test.

You can set the rule to be applied for a number of seconds (Duration) or samples (Samples). If Condition to Apply is set to Duration, you set the Number of Seconds. The rule is in effect for that time. If Compression is selected, data is averaged during this time. When the test is run, data is compressed until the time elapses. To use the duration option, you must have a data series with a time data type and the time must increase sequentially.

If Condition to Apply is set to Samples, data is collected until the number of samples is reached. If Compression is selected, the samples are averaged.

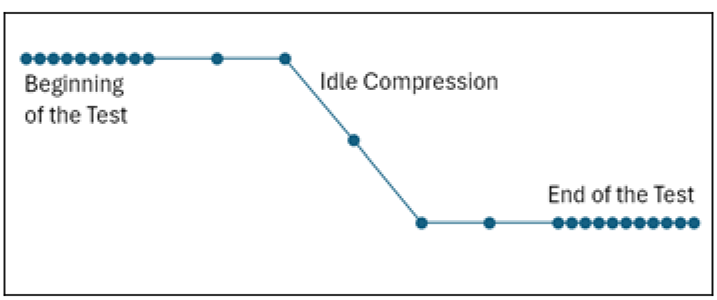
When Compression is selected, you can set data compression to Linear or Log. When Linear is selected, the average of the selected number of samples is returned. If you set the Linear setting to a value of 1, data compression is effectively disabled. When Log is selected, the average of the selected samples per decade is returned.

You may want to clear Compression so that uncompressed data is collected at the beginning of a test before Idle Compression begins.

This rule supersedes all other compression rules except the End of the Test rule. For example, if you have a Shift Rate rule applied at the initial part of the test, the Beginning of the Test rule settings will override the Shift Rate rule settings.

A simplified example of the Beginning and End of Test rules is shown in the following figure. In this figure, the data is not compressed during the beginning and end of the test, but is compressed during the rest of the test.

Figure 103: Beginning and End of the Test compression rules



End of the Test

The End of the Test rule determines what kind of compression, if any, occurs at the end of the test. You can set this rule to compress or not compress data. Only one End of the Test rule can be enabled in a test.

You can set the rule to be applied for a number of seconds (Duration) or samples (Samples). When Condition to Apply is set to Duration, you also set the Number of Seconds. The rule is in effect for that time. If Compression is selected, the data is averaged during this time. You can set the duration from 0 seconds to 3600 seconds. To use the duration option, you must have a data series with a time data type and the time must increase sequentially.

If Condition to Apply is set to Samples, data is collected using this rule until the number of samples is reached. If Compression is selected, the samples are averaged.

When Compression is selected, you can set data compression to Linear or Log. When Linear is selected, the average of the selected number of samples is returned. If you set the Linear Setting to a value of 1, data compression is effectively disabled. When Log is selected, the average of the selected samples per decade is returned.

You may want to clear Compression so that uncompressed data is collected at the end of a test, even if the data would normally be compressed by another rule..

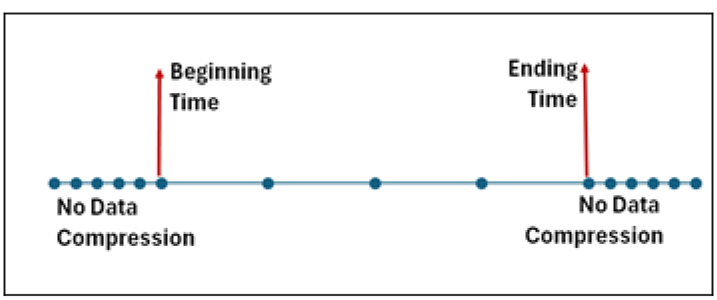
To determine when the end of test occurs, Clarius buffers data when this rule is enabled. This results in a delay in data population in the Analyze sheet and graph. No data points are returned to Clarius while the data is buffered.

This rule supersedes all other compression rules. For example, if you have a Shift Rate rule applied at the end of the test, the End of the Test rule settings will override the Shift Rate rule settings.

Time Interval

When the Time Interval rule is set, data in a defined period have this rule applied. At the completion of the set time, the data points collected during the period are returned. The returned data can be averaged data points or all data. An example of data collected using the Time Interval rule is shown in the following figure.

Figure 104: Time Interval rule data compression example



You can use multiple time interval rules in a test. Time Interval rules supersede the Shift Rate and Idle Compression rules. This rule is superseded by the Beginning and End of the Test rules.

For each time interval rule, set the Beginning Time and Ending Time. The start time must be less than the stop time. If you define multiple time interval rules, the time intervals cannot overlap. Set the time in the format `HH:MM:SS`, defined as:

- `HH`: Hours, 0 to 99
- `MM`: Minutes, 0 to 59
- `SS`: Seconds, 0 to 59

You can select or clear Compression during the time interval. When Compression is selected, you can set data compression to Linear or Log. When Linear is selected, the average of the selected number of samples is returned. When Log is selected, the average of the selected samples per decade is returned.

Idle Compression

The Idle Compression rule compresses data when all other data compression rules are idle. For example, there may be idle time outside of the areas defined for the beginning and end of the test rules, such as when the data series defined for the shift rate rules are steady and do not have changes significant enough to trigger return data.

You can base the idle compression on samples or duration. When Samples is selected, you define the number of idle samples to average. The averaged samples are shown in the Analyze sheet and graph. When Duration is selected, you configure the time intervals of the sample. The average result of the samples within the interval are returned. To return all samples, which effectively disables this rule, select 1 sample or 0 seconds.

This is the lowest priority data compression rule. It is superseded by the Beginning of the Test, End of the Test, and Time Interval rules. If Shift Rate rules are defined, Idle Compression can occur when all Shift Rate rules did not output any samples for the defined idle number of samples or time interval.

You can only enable one Idle Compression rule for a test.

Define data compression

You define data compression using a user test module (UTM) from the user library. Each UTM can have up to ten rules that define the data to be compressed.

To apply data compression, you can use the prebuilt tests provided in the project `data-compression-examples` or the UTMs from the user library `CompressedAcquisitionULib`.to define data compression, as described in the following. You can also create a UTM to compress data. For more information, refer to Create a user test module with data compression (on page 6-11).

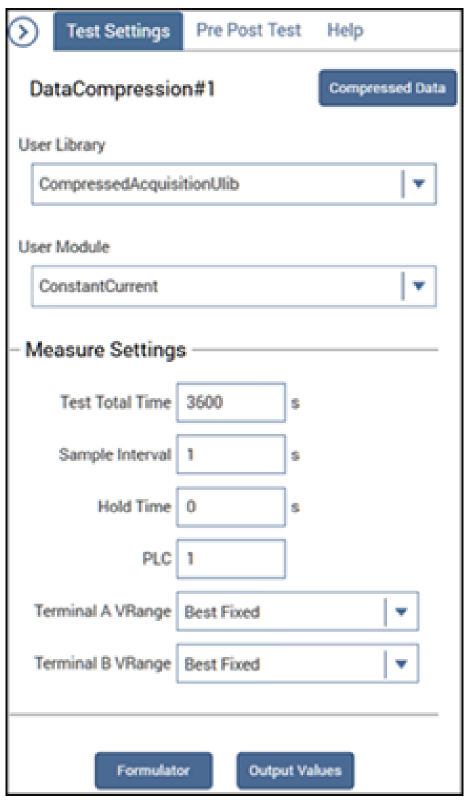
NOTE

To change the display of the data compression UTM or the default values, use the UTM UI Editor. Refer to Define the UTM user interface (on page 7-1) for information.

To define a data compression UTM using `CompressedAcquisitionULib`:

1. Choose **Select**.
2. Select the **Tests** tab.
3. For the Custom Test, select **Choose a test from the preprogrammed library (UTM)**.
4. Drag **Custom Test** to the project tree. The test has a red triangle next to it to indicate that it is not configured.
5. Select **Rename**.
6. Type a name for the test and press Enter.
7. Select **Configure**.
8. In the right pane, from the User Libraries list, select **CompressedAcquisitionULib**.
9. From the User Module list, select **ConstantCurrent** or **ConstantVoltage**. The center pane is populated and the Compressed Data button is added to the Test Settings pane.

Figure 105: Test Settings for a test with data compression



10. Select **Compressed Data** to open the Compress Data Acquisition Settings dialog. Refer to Select the data compression rules (on page 6-8) to define the rules.

Select the data compression rules

Each data compression rule has a tab in the Compressed Data Acquisition dialog. The tabs are numbered sequentially. If a rule is removed, subsequent tabs are renumbered.

Rules can be enabled or disabled. This allows you to turn rules on or off for test runs while maintaining the settings for future use. If all rules are disabled, no data compression occurs.

NOTE

To open information about the selected rule type in the Learning Center, select the Data Compression Rule, then select .

To add a data compression rule:

1. Select **Add Rule**. A new tab is added.
2. Select the Data Compression Rule. An example of the dialog with the Shift Rate rule applied is shown in the following figure.

Figure 106: Compressed Data Acquisition Settings dialog

3. Refer to Compression rule types (on page 6-3) for detail on the options available for each rule.

To remove a data compression rule:

1. Select the tab that contains the rule to be removed.
2. Select **Remove Rule**.

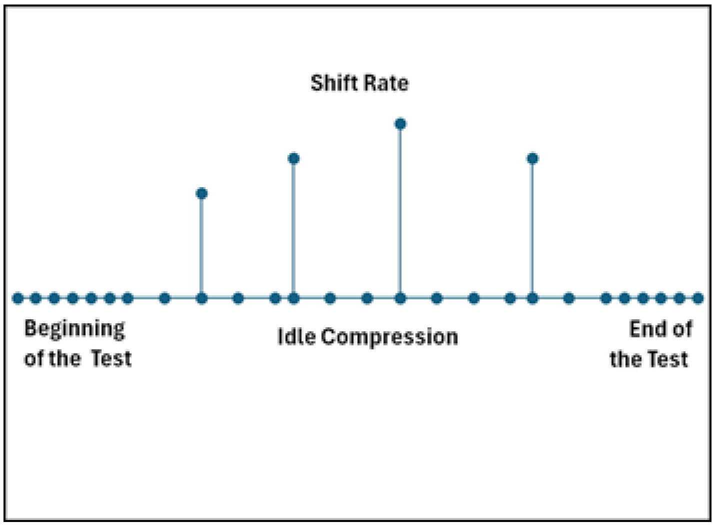
To enable or disable a compression rule:

From the State list, select **Enable** or **Disable**.

Combine data compression rules

The following figure shows simplified test results when multiple compression rules are used when collecting long-term data. It uses the Beginning of the Test and End of the Test rules to collect all data at a defined time at the beginning and end of the test. The Idle Compression rule is then used to compress data until a Shift Rate occurs. At that point, data points are compared to previously returned data. If the change ratio of one point exceeds the defined percentage when compared to previously returned data point, the data is returned.

Figure 107: Example of combined data compression rules



When multiple rules are defined and enabled, the rule hierarchy is as follows:

1. End of the test
2. Beginning of the test
3. Time interval
4. Shift rate
5. Idle compression

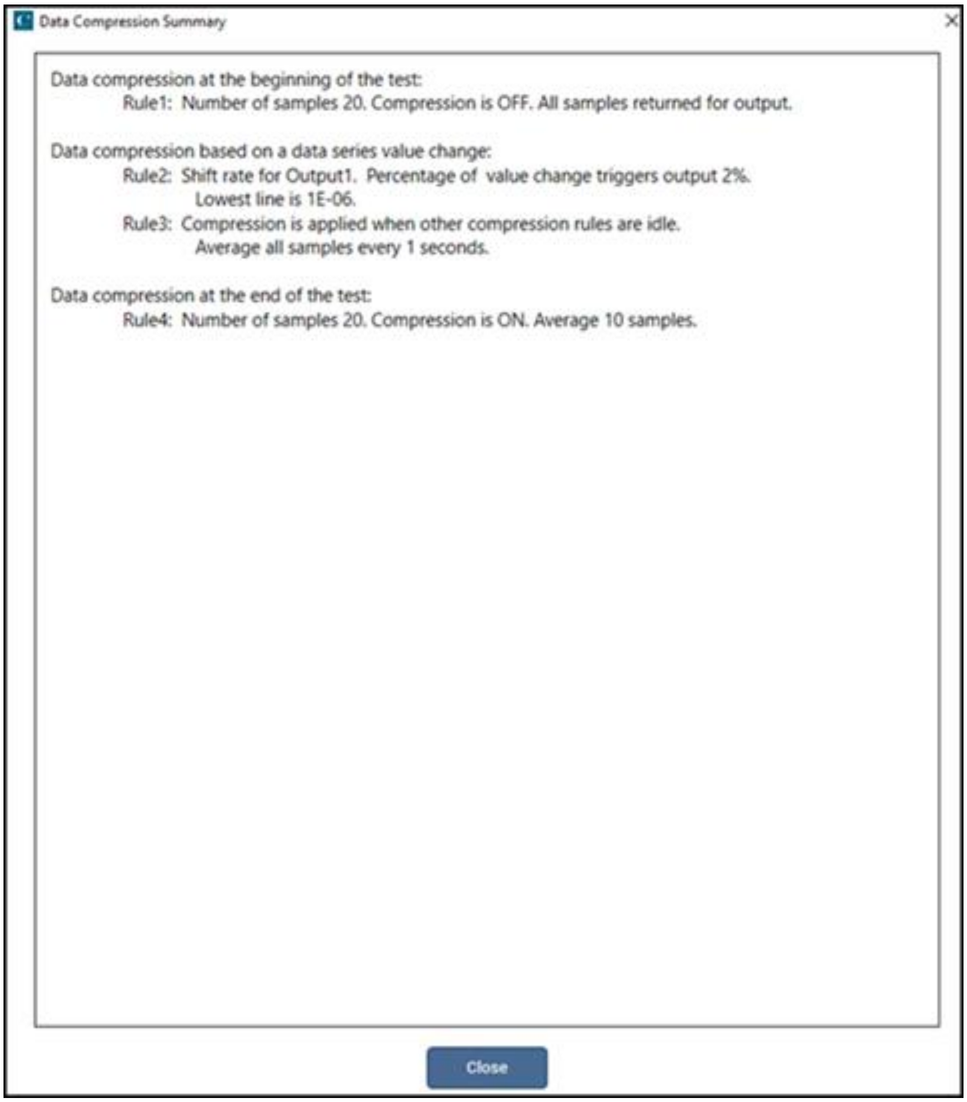
Data compression summary

Select **Summary** to see an overview of the settings for the compression rules that are enabled. The summary also lists the rules that are disabled.

The Data Compression Summary is stored in the Run Settings.

An example of the Data Compression Summary is shown in the following figure.

Figure 108: Data Compression Summary dialog



Tutorial: Create a user test module with data compression

The following tutorial describes how to use the KULT Extension in Visual Studio Code to create a user library and user module that includes data compression. When this tutorial is complete, Clarius and the UTM UI Editor display the Compressed Data button in the Test Settings pane for this user module, which allows you to define data compression rules.

You can also use the `ConstantCurrent` and `ConstantVoltage` modules in the user library `CompressedAcquisitionULib` as examples.

NOTE

Some basic steps that were detailed in "Tutorial: Creating a new user library and user module" in *Model 4200A-SCS KULT and KULT Extension Programming* (4200A-KULT-907-01) are abbreviated in this tutorial.

Start Visual Studio Code

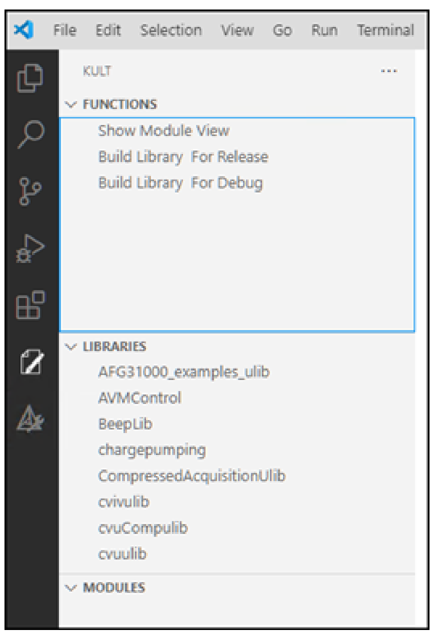
NOTE

Complete Visual Studio Code Installation and setup before using this tutorial. To install Visual Studio Code, refer to "Installation." To set it up, refer to "Setting up Visual Studio Code for library development." Both are in *Model 4200A-SCS KULT and KULT Extension Programming* (4200A-KULT-907-01).

To start Visual Studio Code:

1. In the Windows Start menu, select **Visual Studio Code**.
2. Select the KULT icon to open the KULT side bar.

Figure 109: Opening the KULT side bar in Visual Studio Code



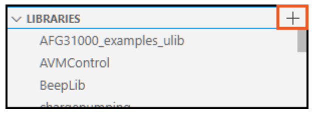
Create a user library and user module with data compression

All user modules must have unique names to avoid conflicts in library dependencies.

To create a new user library and module:

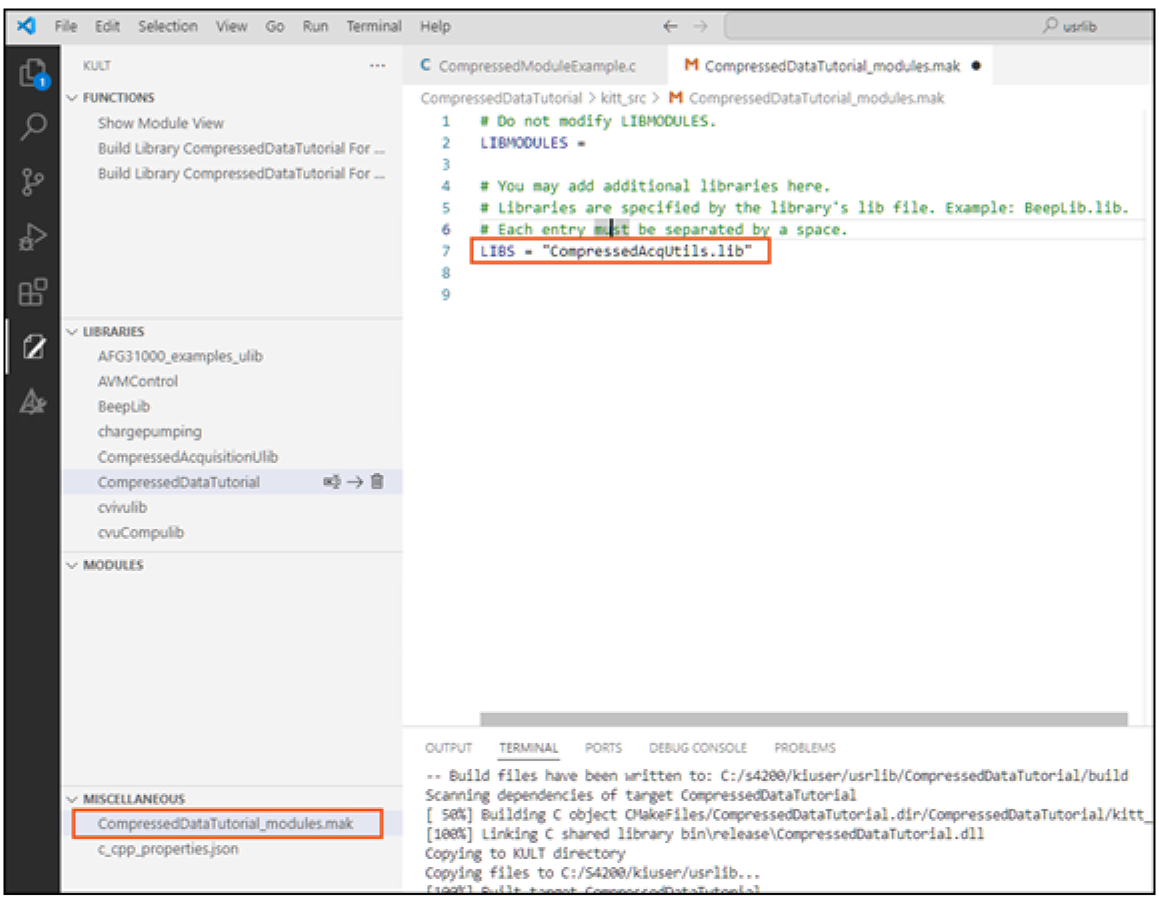
1. In the KULT side bar, in Libraries, select **+**.

Figure 110: Add a new user library



2. Enter `CompressedDataTutorial` as the new user library name.
3. Select **Enter**. The library is displayed in the list of libraries. The build files are automatically created.
4. In the KULT side bar, under Miscellaneous, select `CompressedDataTutorial_modules.mak` to open it in the editor.
5. Place your cursor in the quotes next to the `LIBS=` variable.
6. Type `CompressedAcqUtils.lib`.

Figure 111: Select the .mak file and the library



7. Select **File > Save**.
8. In Libraries, select the new library `CompressedDataTutorial`.
9. In Modules, select **+**.

Figure 112: Create a new module

10. Type `CompressedModuleExample` as the name for the new module.
11. Select **Enter**.
12. Select the `CompressedModuleExample` module to open it in the editor.
13. Set the Return Type to **int**.
14. Select **Apply**.
15. In the `/* USRLIB MODULE PARAMETER LIST */`, add the header file `CompressedAcqUtils.h` after the `keithley.h` header file, as shown here:

```
#include "keithley.h"
#include "CompressedAcqUtils.h"
```

16. Select **File > Save**.

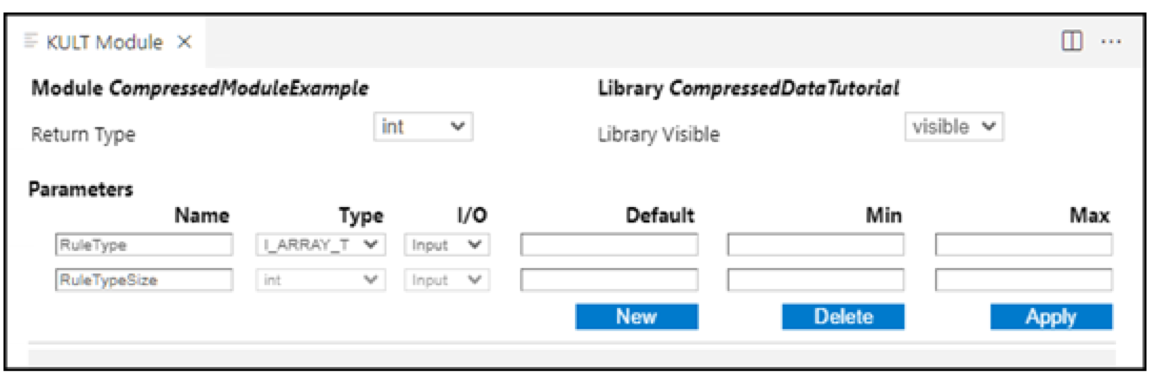
Enter the user module parameters

This example uses the double-precision `D_ARRAY_T` and `I_ARRAY_T` array types. The `D_ARRAY_T`, `I_ARRAY_T`, and `F_ARRAY_T` are special array types that are unique to Keithley User Libraries. For each of these array types, you cannot enter values in the Default, Min, and Max fields. An extra parameter is automatically created to indicate the array size.

To define the user module parameters:

1. In the KULT module, select **New** to create a new parameter.
2. Set the Name to **RuleType** and select the Type to **I_ARRAY_T**. The size parameter is automatically created.
3. Change the name of the size parameter to **RuleTypeSize**. The Type and I/O are automatically set to `int` and `Input`.

Figure 113: Compressed data KULT Module RuleType parameter

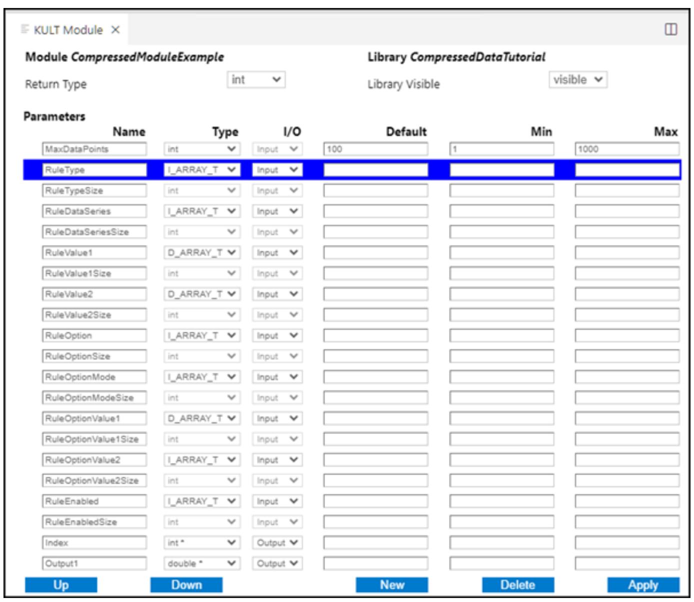


4. Repeat these steps to create the parameters shown in the following table.

Parameter Name	**Type**	**I/O**
`RuleDataSeries`	I_ARRAY_T	Input
`RuleDataSeriesSize`	int	Input
`RuleValue1`	D_ARRAY_T	Input
`RuleValue1Size`	int	Input
`RuleValue2`	D_ARRAY_T	Input
`RuleValue2Size`	int	Input
`RuleOption`	I_ARRAY_T	Input
`RuleOptionSize`	int	Input
`RuleOptionMode`	I_ARRAY_T	Input
`RuleOptionModeSize`	int	Input
`RuleOptionValue1`	D_ARRAY_T	Input
`RuleOptionValue1Size`	int	Input
`RuleOptionValue2`	I_ARRAY_T	Input
`RuleOptionValue2Size`	int	Input
`RuleEnabled`	I_ARRAY_T	Input
`RuleEnabledSize`	int	Input
`Index`	int*	Output
`Output1`	double*	Output

The parameters should look like the following figure.

Figure 114: Parameters for a Data Compression User Module



Enter the code

Add the following code after `/* USRLIB MODULE CODE */`.

```
char ErrMsg[255];
   int status = 0;
   // Data Series parameters.
   int seriesCount = 2; //number of outputs is 2: Index and Output1.
   char* dataSeries = "Index,Output1";
   int dataType[2] = { 1, 0 }; //Data types will be double=0, integer=1 and time=2.

   // Initialize the Compression library.
   status = InitializeCompression();
   if (status != 0)
   {
      return status;
   }

   // Call SetCompressionRules().
   status = SetCompressionRules(RuleType, RuleDataSeries, RuleValue1, RuleValue2,
      RuleOption, RuleOptionMode, RuleOptionValue1, RuleOptionValue2, RuleEnabled,
   RuleEnabledSize);
   if (status != 0)
   {
      return status;
   }

   status = SetDataSeriesForCompression(dataSeries, dataType, seriesCount);
   if (status != 0)
   {
      return status;
   }

   // Call ProcessSampleOfData with simulated data.
   BOOL postData = TRUE; // ProcessSampleOfData will post data if needed to Clarius using
   PostDataDouble().

   double dataSample[1] = { 0.0 };
   for (int i = 0; i < MaxDataPoints; i++)
   {
      dataSample[0] = 1e-3*(i+1);
      status = ProcessSampleOfData(dataSample, seriesCount, postData);
      if (status != 0)
      {
         // Continue the loop even if errors occurred.
          // Log errors if needed.
      }
   }

   // Notify the Compression library that the "test" is complete.
   // This is required to flush any remaining data and allow the end of test rules to
   be processed.
   status = FinalizeCompression(postData);

   return status;
```

Document the module

Module descriptions are entered between the comment lines `USRLIB MODULE HELP DESCRIPTION` and `END USRLIB MODULE HELP DESCRIPTION`. Code entered here in Markdown format will appear in the Clarius help pane. To format the code, use Markdown, a web markup language. See markdownguide.org for information on using Markdown.

The following code shows a sample description.

```
<link rel="stylesheet" type="text/css"
   href="http://clariusweb/HelpPane/stylesheet.css">

CompressedModuleExample module
--------------
This example demonstrates how to set up a user test module that uses data compression
   with simulated data.
```

Save and build the user module

To save and build the module:

1. From the **File** menu, select **Save**.
2. Under Functions, select **Build Library CompressedDataTutorial for Release**.
3. Select the **Run** icon.
4. Check the status of the build output in the Terminal tab at the bottom of the Visual Studio Code window.
5. Correct any errors and rebuild the user module.

Check the data compression user module in Clarius

You can check the user module by adding it to a user test module (UTM) in Clarius and executing the UTM.

With these settings, the UTM produces 11 data points instead of 100. Data compression always outputs the first and last data points, regardless of the type of compression.

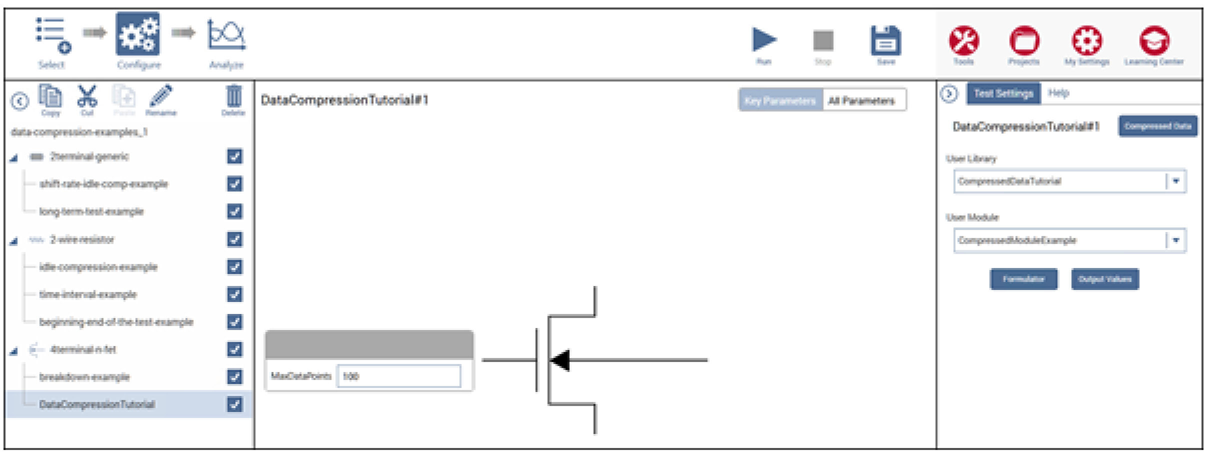
NOTE

To change the display of the data compression UTM or the default values, use the UTM UI Editor. Refer to Define the UTM user interface (on page 7-1) for information.

To check the user module:

1. In Clarius, choose **Select**.
2. Select **Projects**, then select **New Project**.
3. Select **Devices** and select an appropriate device.
4. Select the **Tests** tab.
5. Select **Custom Test**.
6. Select **Choose a test from the preprogrammed library (UTM)**.
7. Select **Add**. The test has a red triangle next to it to indicate that it is not configured.
8. Select **Rename**.
9. Enter the name `DataCompressionTutorial` and select **Enter**.
10. Select **Configure**.
11. In the right pane, from the User Libraries list, select the `CompressedDataTutorial` library.
12. From the User Modules list, select `CompressedModuleExample`. A default schematic and group of parameters are displayed for the UTM.

Figure 115: Data Compression tutorial UTM in Clarius

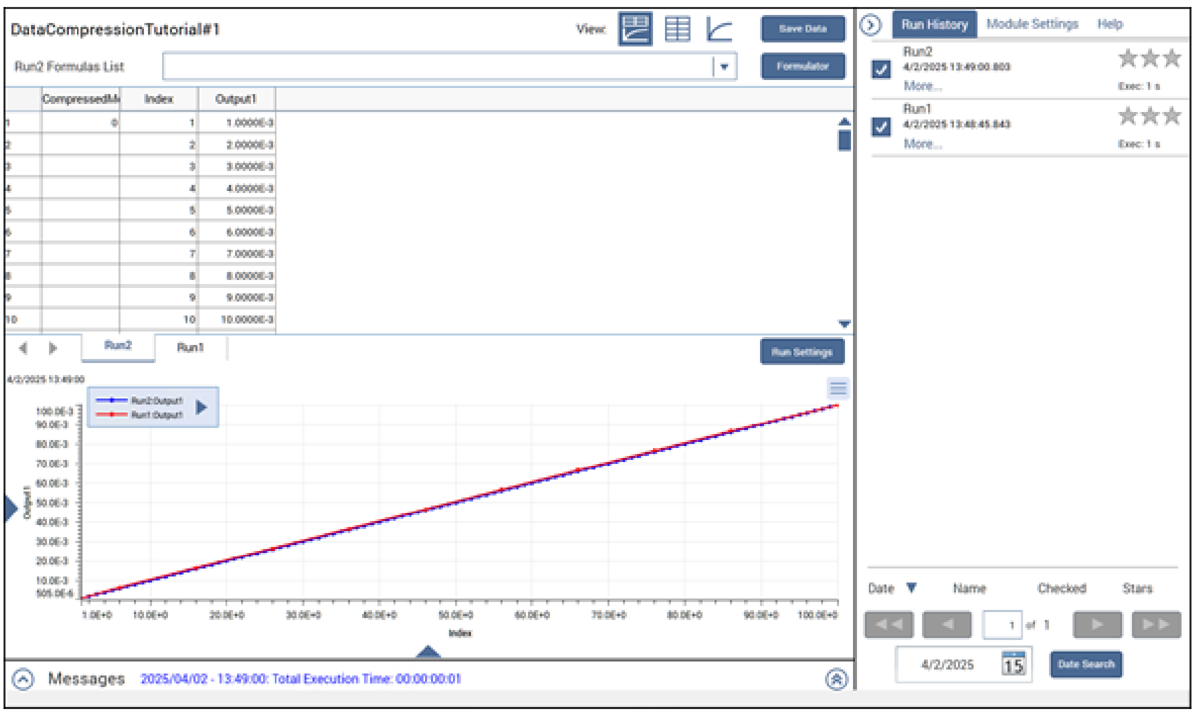


13. Leave the **MaxDatapoints** at 100 for this example. It can be set to a value between 1 to 1000.
14. Select **Compressed Data**.
15. Select **Add Rule**.
16. For Data Compression Rule, select **Idle Compression**.
17. For Condition to Apply, select **Samples**.
18. For Number of Samples, enter **10**.
19. Select **OK**.
20. Select **Run**.
21. Select **Analyze** to view the graph.

To compare the runs with and without data compression:

1. Select **Configure**.
2. Select **Compressed Data**.
3. Set the state to **Disable**.
4. Select **OK**.
5. Select **Run**.
6. Select **Analyze**.
7. From Run History, select both runs. The results should look similar to the following figure.

Figure 116: Data Compression Tutorial Analyze results



Section 7

Define the UTM user interface

In this section:

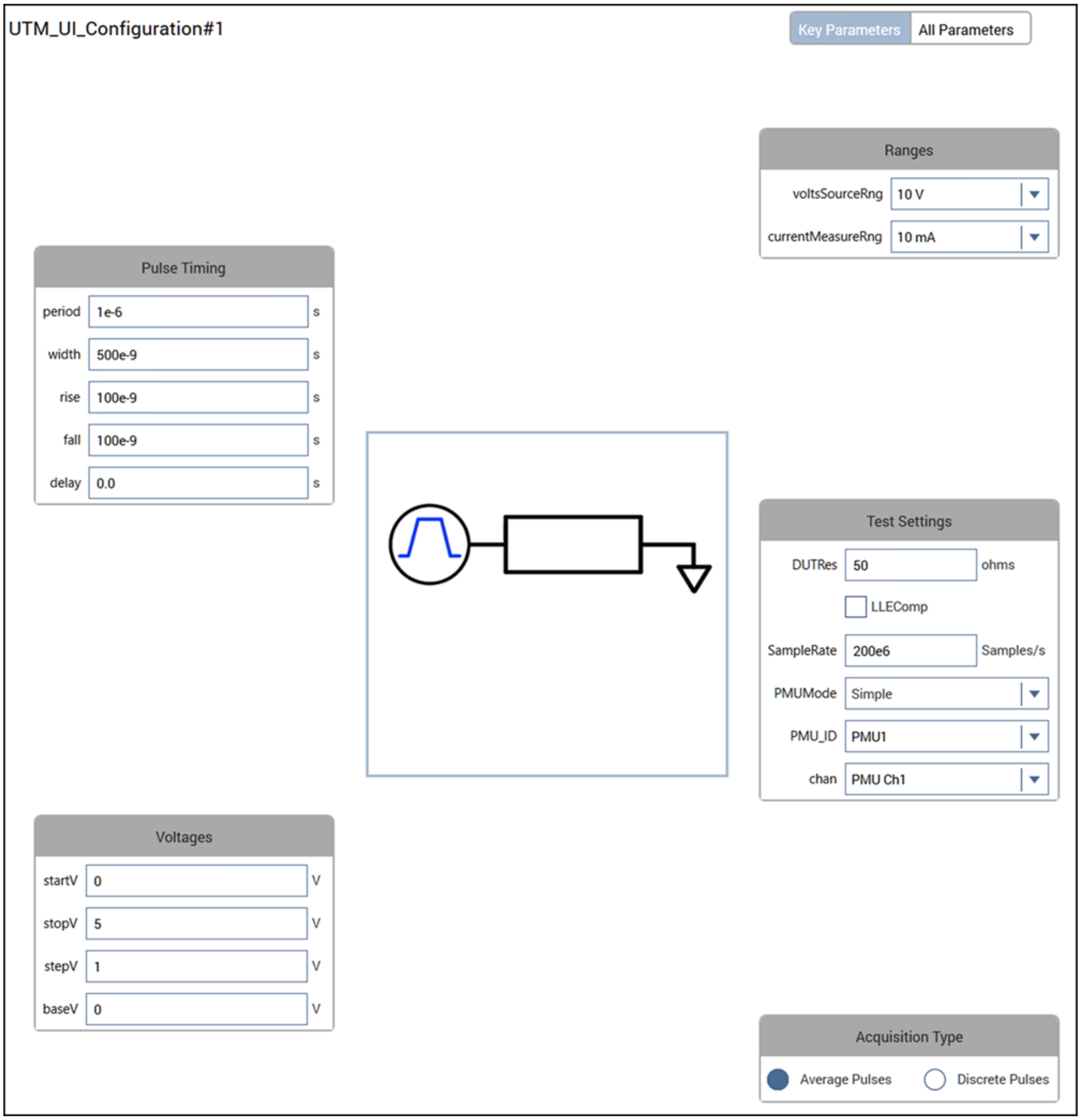
Define the user interface for a user test module (UTM)

When a user test module (UTM) is created, Clarius automatically creates the user interface (UI) that is displayed on the Configure pane when Key Parameters is selected. The parameters are placed in groups around a default image of the device under test, as shown in the following figure.

You can change the display and add verification checking of the parameters using the UTM UI Editor. The UTM UI Editor is a Clarius+ utility that runs separately from Clarius. You can use it to:

- Add or change the image that illustrates the test
- Change the grouping of UTM parameters
- Set up stepping or sweeping
- Add verification rules for input and output parameters
- Add visibility rules for parameters
- Add tooltips for parameters
- Determine if selected parameters are displayed in the center pane or the right pane

Figure 117: Key Parameters view of Configure pane



NOTE

If you change a UTM parameter name using KULT after defining the user interface, make sure you update the previously created UI definition. This ensures that the new parameter name has a group and is displayed. If you do not update it, the new parameter name is not assigned to a group and is not displayed.

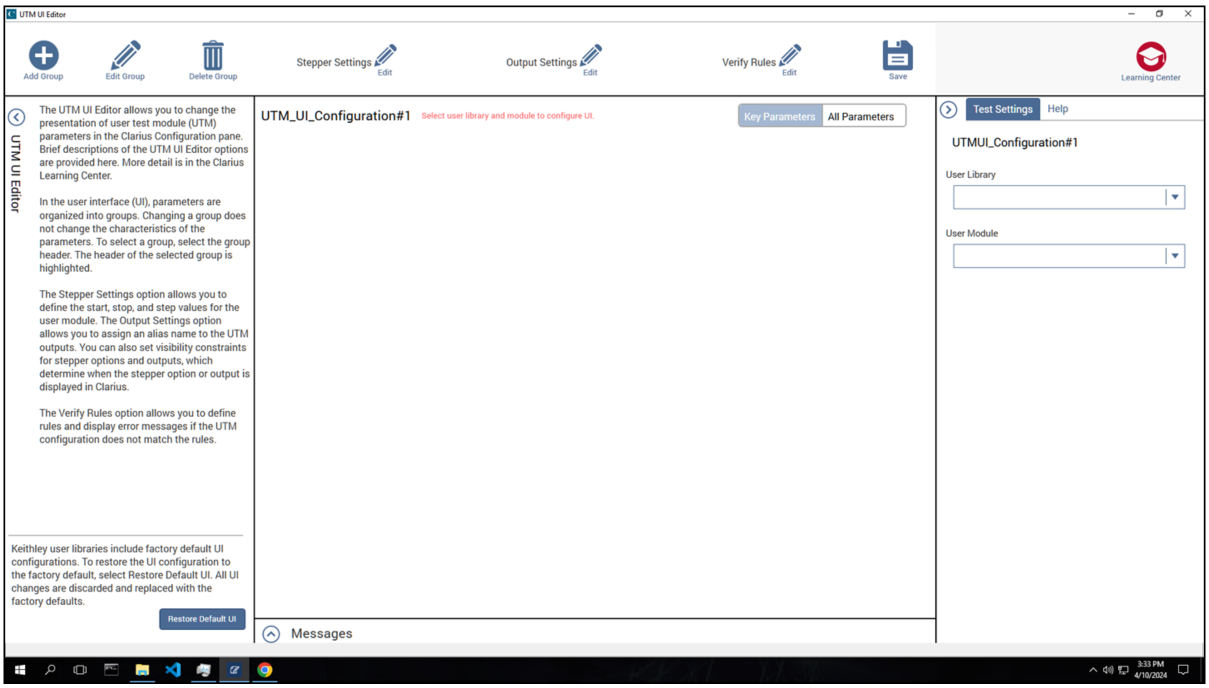
Open the UTM UI Editor

The UTM UI Editor is a Clarius+ application that allows you to set up the UTM UI. It cannot be run at the same time as Clarius.

To open the UTM UI Editor:

1. Close Clarius.
2. From the Windows Start menu, select **Keithley Instruments > UTM UI Editor**. The UTM UI Editor application opens, as shown in the following figure.

Figure 118: UTM UI Editor application



Select the user library and user module

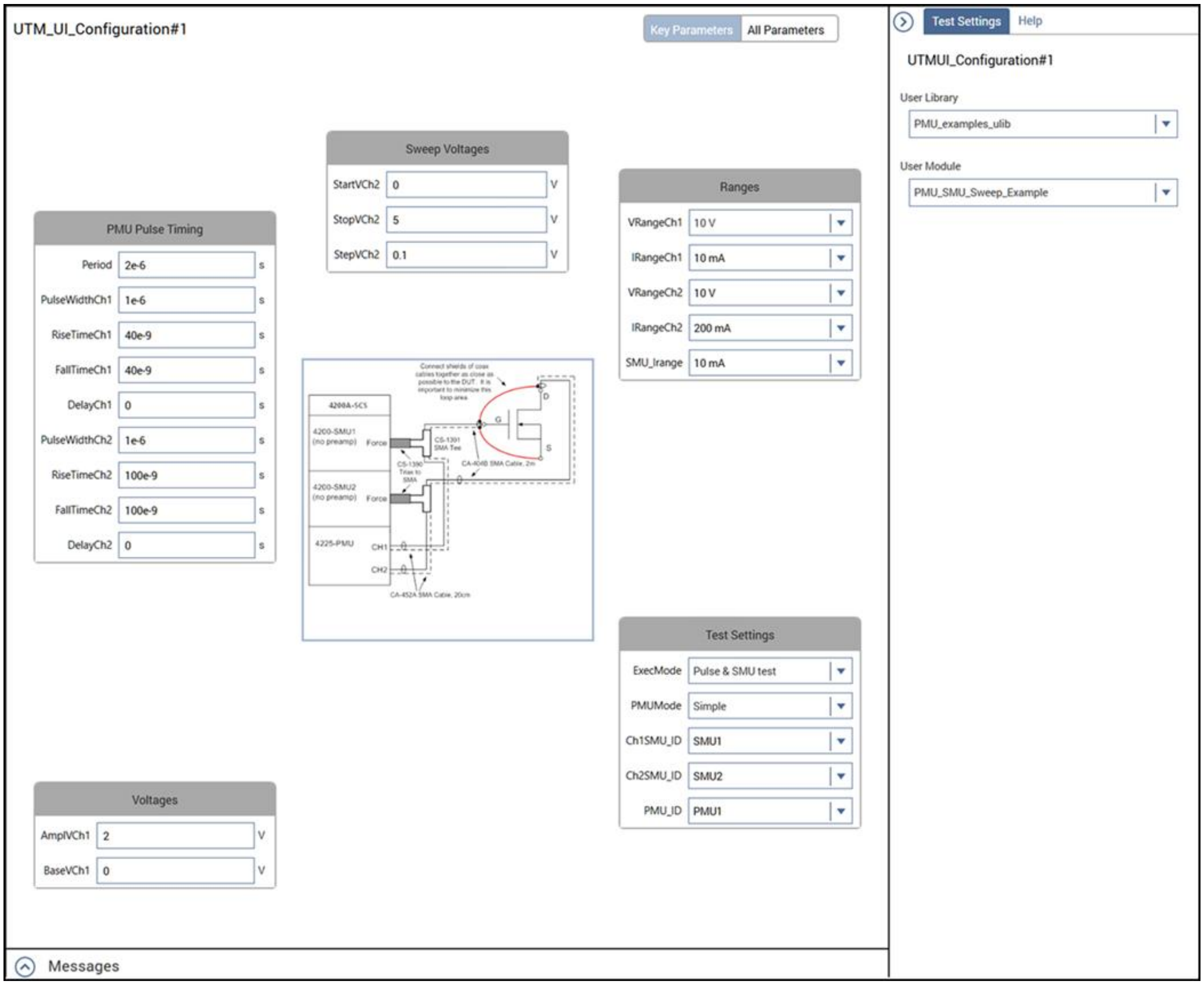
To work on a UI, you need to open the user module.

To open the user module:

1. In the Test Settings pane, select a **User Library**.
2. Select a **User Module**. The center pane displays the existing UI configuration for this module.

An example of the `PMU_SMU_Sweep_Example` user module from the `PMU_examples_ulib` is shown in the following figure.

Figure 119: User Library and Use Module selected in the UTM UI Editor



Define an image for the user interface

Each UTM has an image displayed in the Configure pane for the UTM.

By default, the image is the device under test and is centered in the Configure pane. You can change the image in the UTM UI Editor. Each UTM can have only one image. You can use `.bmp` or `.png` images.

Factory images are stored in the source directory of the user library. For example, an image for a UTM in the `VLowFreqCV` user library is stored in:

```
C:\s4200\kiuser\usrlib\VLowFreqCV\src
```

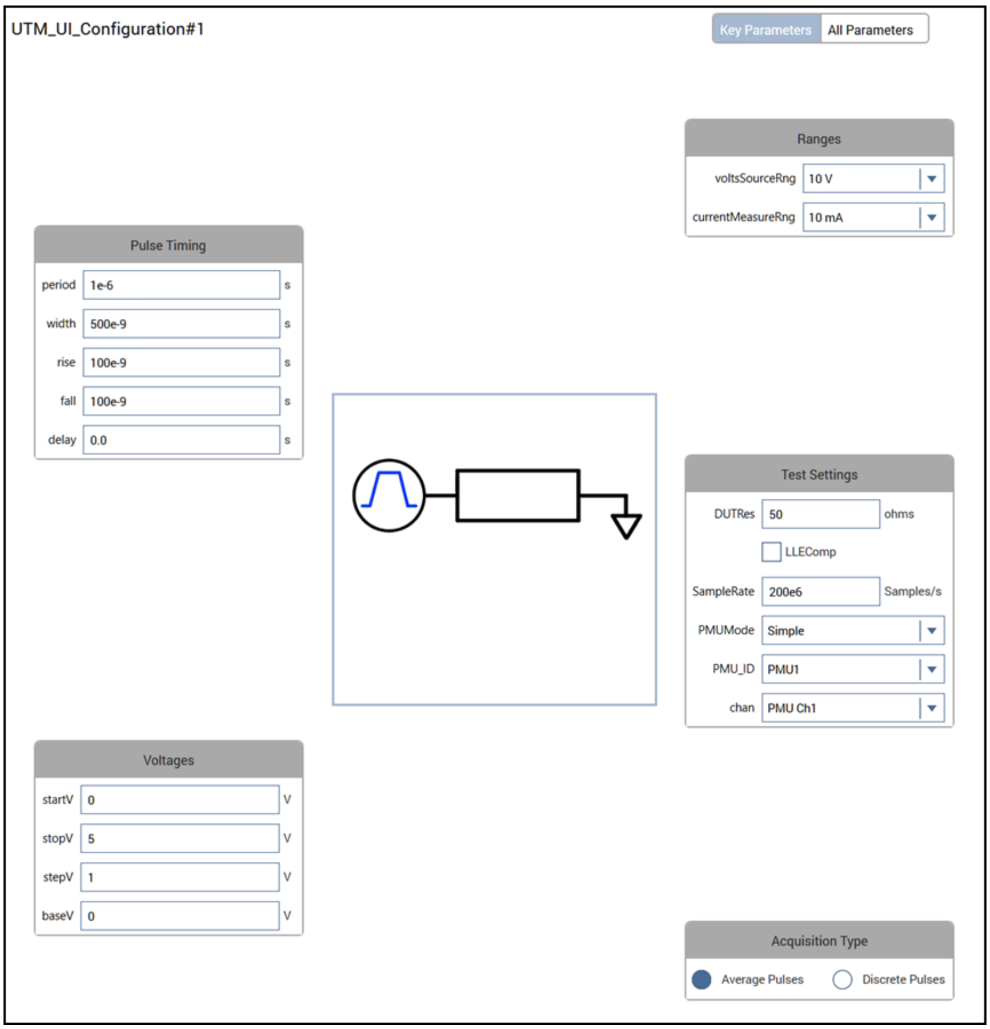
To add an image:

1. In the UTM UI Editor, double-click the existing image. If there is no image, double-click the center of the UTM UI Configure pane. The Edit UTM Image dialog is displayed.
2. From UTM Image list, select the image to display:
   - **None:** Do not display an image.
   - **Device Default:** Use the image of the device that the UTM is under in the Clarius project tree.
   - **File:** Select a file to use as the device image. Select **Browse** to locate the file or enter a file path and name.
3. If you selected **File**, you can also change the size and position of the image. The options are:
   - **Expand to Top:** Choose **None** to keep the image centered or select a percentage to determine how far the image expands toward the top of the pane.
   - **Expand to Bottom:** Choose **None** to keep the image centered or select a percentage to determine how far the image expands toward the bottom of the pane.
   - **Maintain Aspect Ratio:** Select to keep the ratio of height and width the same when the image is expanded. Clear this option to have the image expand to fill the space defined by the **Expand to** options. This may distort the image.
4. Select **OK**.

Organize parameters into groups

You can use groups to organize input and output parameters on the Clarius Configure pane. In the groups, users can choose parameters from lists, enter specific values, or select checkboxes. In the following figure, the parameters are organized into the groups Pulse Timing, Voltages, Ranges, Test Settings, and Acquisition Type.

Figure 120: Example of groups



You can have a maximum of 20 groups. Up to eight groups can be in the center pane. In the right pane, you can have up to six in tab 1 and six in tab 2.

You do not need to place all parameters in a group. For example, for the majority of tests, the size values of the output arrays can be left at the default values, so you do not need to display them.

To view parameters that have not been assigned to a group, select **All Parameters**. The parameters are listed in the Unassigned Parameters list.

Add a group

To add a group:

1. Select **Add Group**. The Add Group dialog is displayed.
2. Select the options for the group. Refer to Group options (on page 7-7) for descriptions.
3. Add and configure the parameters. Refer to Edit the attributes for a test parameter (on page 7-12) for details.
4. Select **OK**.

Edit a group

To modify a group, you can:

- Double-click the group heading.
- Select the heading for the group, then select **Edit Group**.

From the Edit Group dialog, you can select **Change Group** to edit a different group without exiting the dialog.

Refer to Group options (on page 7-7) for descriptions of the options for the group.

Refer to Edit parameters (on page 7-12) for details on changing the parameters.

Delete a group

Deleting a group removes the group from the UTM UI. The parameters that were in the group are moved to the Unassigned Parameters list, which you can view using All Parameters.

To delete a group:

1. Select the group heading.
2. Select **Delete Group**.
3. On the confirmation dialog, select **Yes**.

Group options

The options for groups are described in the following.

Group Title: The name of the group, which is displayed above the group in the Configuration pane.

Group Position: Where the group is displayed. You can select:

- **Clarius Center Pane:** The group is placed in the center pane.
- **Clarius Right Pane Tab1:** The group is placed in the Test Setting tab under the user library and user module selections.
- **Clarius Right Pane Tab2:** The group is placed in the Extra Settings tab to the right of the Test Settings tab.

Orientation or tab position: The location of the group in the center pane or right pane tab:

- **Clarius Center Pane options:** You can select where the group is oriented around the UTM UI image, using compass orientation.
- **Right Pane Tab1 or Tab2:** You can set the position of the group on the tab. The #1 position is the top group. On Right Tab1, the #1 position is below the user library and user module lists.

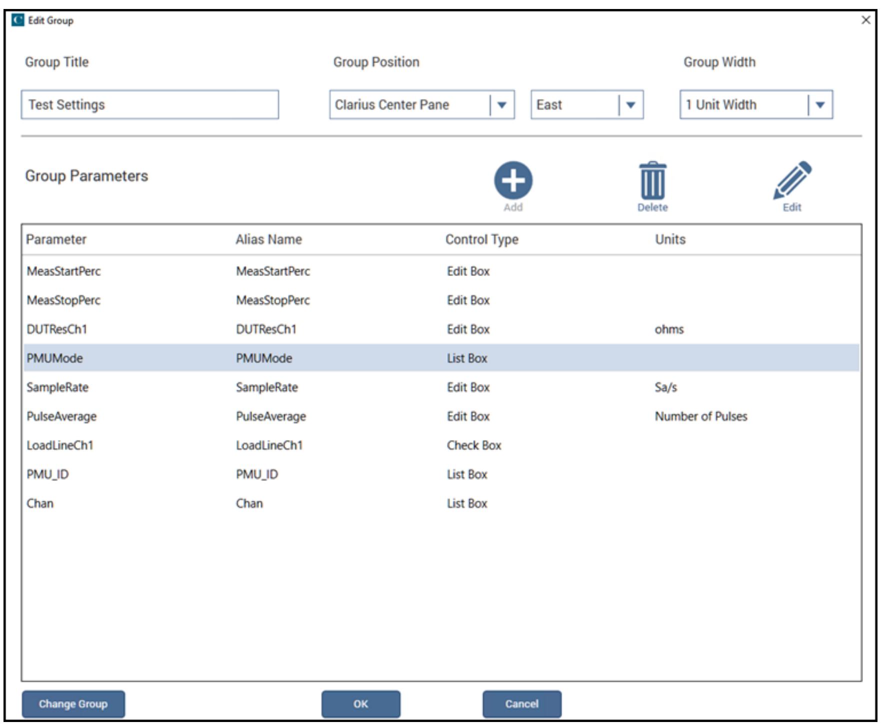
Group Width: Sets the maximum possible width of the group when it is in the center pane. The group width is dynamically adjusted to the selected width based on the length of the static text and test entry fields. The group is center-aligned to the width. For example, if the width is set to 3 unit widths but the group content fits in 1 unit width, the group is 1 unit wide in the center of the pane. If the text for a group exceeds the selected width, a scrollbar is displayed. You cannot set the width for groups in the right pane tabs and for groups set to North-East, East, and South-East in the center pane.

Add: Add parameters to the group. Unassigned parameters are added to the group. You can delete the parameters you do not want to include in the group.

Parameter list: The list of parameters in the group. You can drag the parameters to reposition them. For information on adding and configuring the parameters, refer to Edit the attributes for a test parameter (on page 7-12).

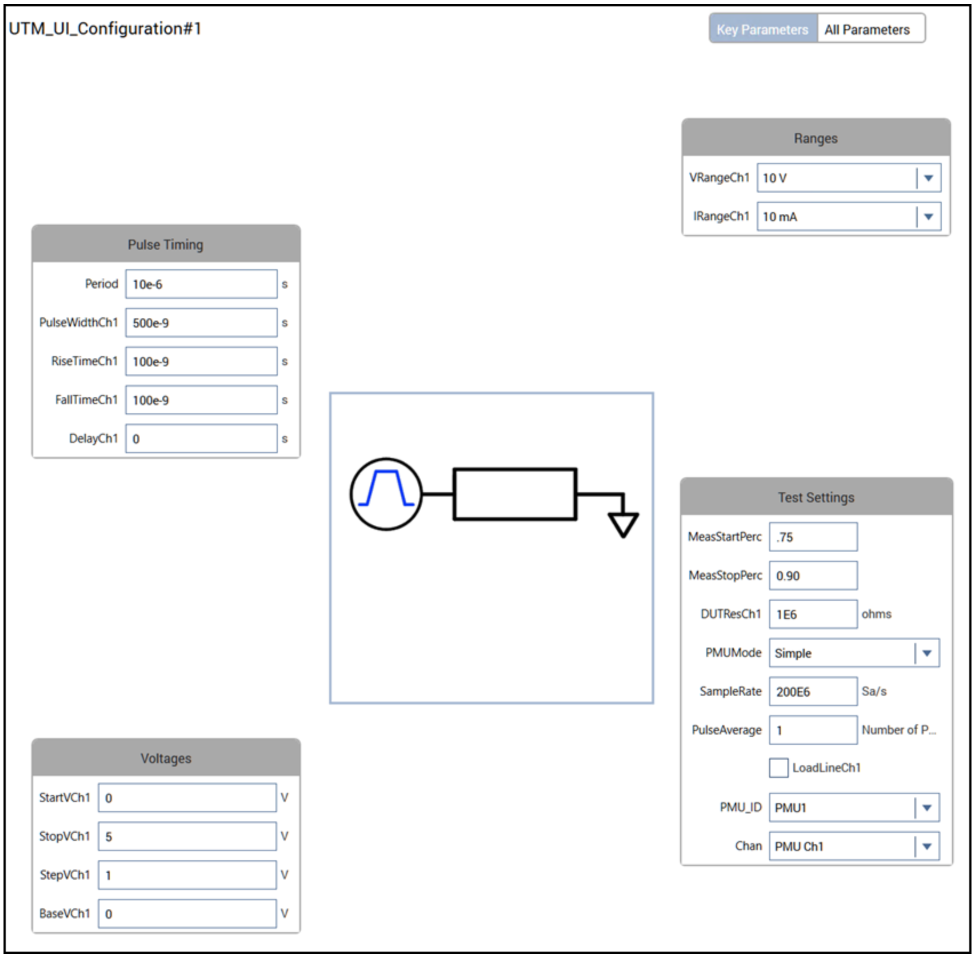
An example of the Edit Group dialog is shown in the following figure.

Figure 121: Example of the Add Group dialog



An example of the UTM UI that includes the group defined in the previous figure is shown in the following figure.

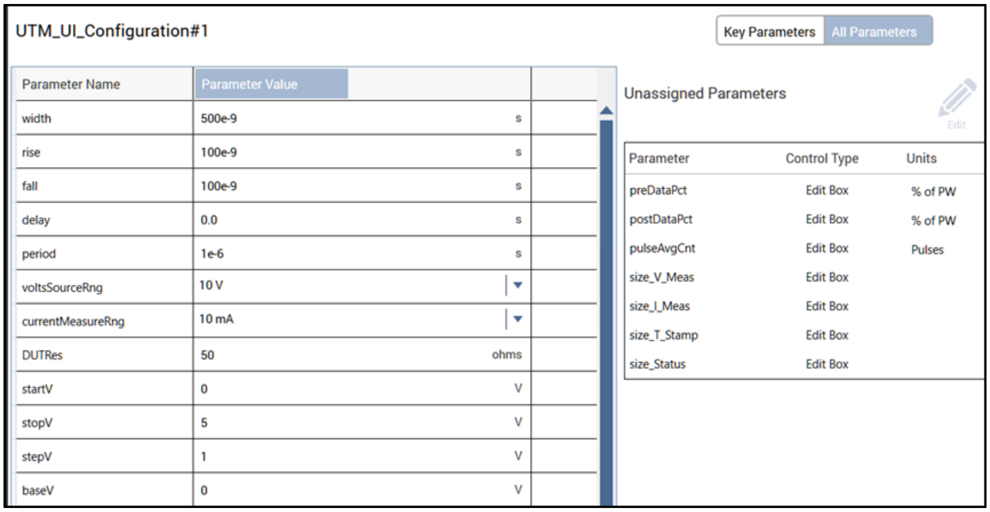
Figure 122: Group added to Configure pane



Editing parameters that are not assigned to a group

You can view and edit parameters that are not assigned to a group using the All Parameters view in the Unassigned Parameters list. The example of the unassigned parameters in the following figure are from the user library `PMU_examples_ulib` user module `PMU_1Chan_Waveform_Example`.

Figure 123: Unassigned parameters



To edit these parameters, select **Edit**. Refer to Edit parameters (on page 7-12) for details on the editing options.

Settings to step voltage, frequency, or current

You can use the UTM UI Editor to set up UTMs that step through voltage, frequency, or current values.

NOTE

If you completed the KULT tutorials, you can use the user modules in user library `my_2nd_lib` as an example of UTMs that include stepping and sweeping. The tutorials are in "Tutorial: Creating a user module for stepping or sweeping" in *Model 4200A-SCS KULT and KULT Extension Programming* (4200A-KULT-907-01).

The UTM GUI Editor Stepper Settings dialog allows you to select parameters for step and sweep UTMs. This information is required for Clarius to manage data coming from a user module that is set up for stepping or sweeping.

In the Analyze sheet, when step or sweep data is returned, the first column contains the return value of user module execution. This is normally `0` or `1` to indicate the success or failure of the execution. The actual return code is set in the user module code. In addition, the output values for each step are displayed.

You can set up the steppers to use different parameters depending on the values of other parameters. In the example shown in the following figure, the UTM test uses the StartV, StopV, and StepV parameters as a stepper if DCOperationMode parameter is set to `Voltage Step`. It uses the StartFrequency, StopFrequency, and StepFrequency parameters as the stepper values if the FreqOperationMode parameter is set to `Freq Step`.

Figure 124: Stepper example

In this example, the Visibility Constraints for a Voltage Step test are set to:

```
DCOperationMode={Voltage Step}
```

The Visibility Constraints for a Frequency Step test are set to:

```
FreqOperationMode={Freq Step}
```

To set up a stepper and sweeper in the UTM UI Editor:

1. Select a UTM that includes stepping and sweeping.
2. Select the group that contains the stepper or sweeper.
3. In the Group Parameters list, add the **Start**, **Stop**, and **Step** parameters. Refer to Options when editing parameters (on page 7-13) for descriptions of the parameters.
4. Select **OK** to save the group.
5. In the UTM UI Editor, select **Stepper Settings**.
6. Add the **Start**, **Stop**, and **Step Parameters**.
7. If needed, set up Visibility Constraints. Refer to Visibility constraints (on page 7-13) for details.
8. Select **OK**.

Verification rules for UTM configuration

You can add verification rules to prevent mistakes and constrain values when UTM modules are configured. When you create a rule, you also create an error message that is displayed to notify the user of the constraint. Clarius checks the rule before saving the UTM configuration and before running a test based on the UTM. You can save a UTM that violates the rule, but you cannot run it.

Refer to Visibility constraints (on page 7-13) for descriptions of the expressions that are available.

NOTE

For an example of configuration verify rules, refer to the User Library `PMU_examples_ulib` user module `PMU_IV_sweep_step_Example`.

To add verification rules:

1. Select **Verify Rules**.
2. Select **Add** to add a rule.
3. In **Verify Rule**, enter the rule.
4. In **Error Message**, enter the error message to display to the user.
5. Repeat these steps to create additional rules as needed.
6. Select **OK**.

For example, to prevent a user from creating a UTM with a stepV parameter value of `0`, set the Verify Rule to:

```
StepV!=0
```

With an error message of:

```
Voltage Step cannot be 0
```

Edit parameters

You can change some of the attributes of a test parameter using the UTM UI Editor.

NOTE

Some values, such as Minimum and Maximum Values, are defined in the KULT user module. If values are dimmed, you must use KULT to change the values.

To edit parameters:

1. Open a group.
2. Select a parameter.
3. Select **Edit**. The Edit Parameter dialog is displayed.
4. Make changes as needed. The options that may be available are described in Options when editing parameters (on page 7-13).
5. Select **OK**.

Options when editing parameters

The following table lists the options for parameters that you can set up for the UTM UI. Some values, such as Minimum and Maximum Values, are defined in the KULT user module and cannot be changed in the Edit Parameter dialog. If values are dimmed, you must use KULT to change the values.

The options that you can modify may differ depending on the control type of the parameter. See the descriptions of each control type for specific information.

Edit Parameter options

Parameter Name	Select the parameter that you want to change. Note that if you select another parameter while in this dialog, any changes to the previous parameter are lost. Select OK to save your changes.
Alias Name	The name that is displayed on the Configure pane. If you do not define a name, the parameter name defined in the UTM is used.
Control Type	The control type of the parameter. Refer to Control types for details on the options.
Minimum Value	An integer or double type value that is the lowest value the user can enter in this field.
Default Value	The value that is automatically selected when the user opens the Configure pane. Typically, this value is defined in KULT, but can be changed here.
Maximum Value	An integer or double type value that is the highest value the user can enter in this field.
Displayed Units	The units of measure for the value, such as voltage, amperes, or seconds. No conversions are made, so these must be the same units as the applicable command.
Displayed Tooltips	Text that is displayed when users hover over the parameter field or long press the parameter field on the touchscreen. Guidelines for tooltips: ▪ Use tooltips to help explain the selection. ▪ Line feeds and carriage returns are not available. ▪ Keep tooltips short. ▪ Use simple present tense. ▪ Use clear and consistent language. ▪ Check your spelling.
Parameter Visibility Constraints	Rules that define which options are displayed on the UTM UI. Refer to Visibility constraints (on page 7-13) for information.

Visibility constraints

In the UTM UI Editor, you can use expressions to set up parameters so that options are only displayed when certain conditions are met. You can use expressions to:

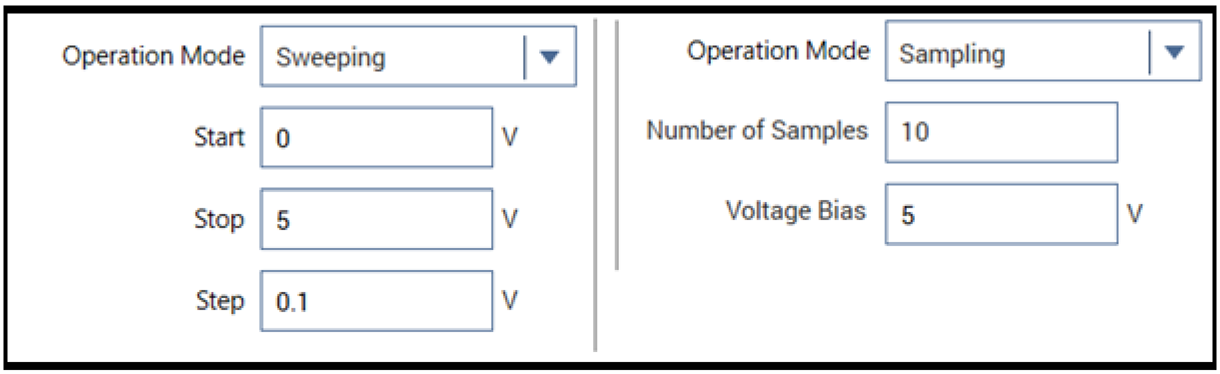
- Determine which parameters are visible in the Configure pane.
- Determine which items are visible in a list for a parameter in the Configure pane.
- Determine which outputs are visible in the Analyze pane.
- Establish verification rules when users configure UTMs.
- Determine stepper usage when users configure stepper UTMs.

Constrain parameters

You can use the visibility constraints to limit the options that are available through the UI for each parameter.

For example, there are different parameters that are appropriate if the Operation Mode is set to Sweeping or Sampling. When the Operation Mode is set to Sweeping, the parameters are Start, Stop, and Step. When the Operation Mode is set to Sampling, the parameters are Number of Samples and Voltage Bias. The visibility constraints hide the options that are not appropriate. Examples of the UI display for these Operation Modes are shown in the following figure.

Figure 125: Operation Mode visibility constraint example



To set these constraints, set the `OperationMode` constraint for the parameters. Set the Start, Stop and Step parameters to:

```
OperationMode=Sweeping
```

Set the Number of Samples and Voltage Bias parameters to:

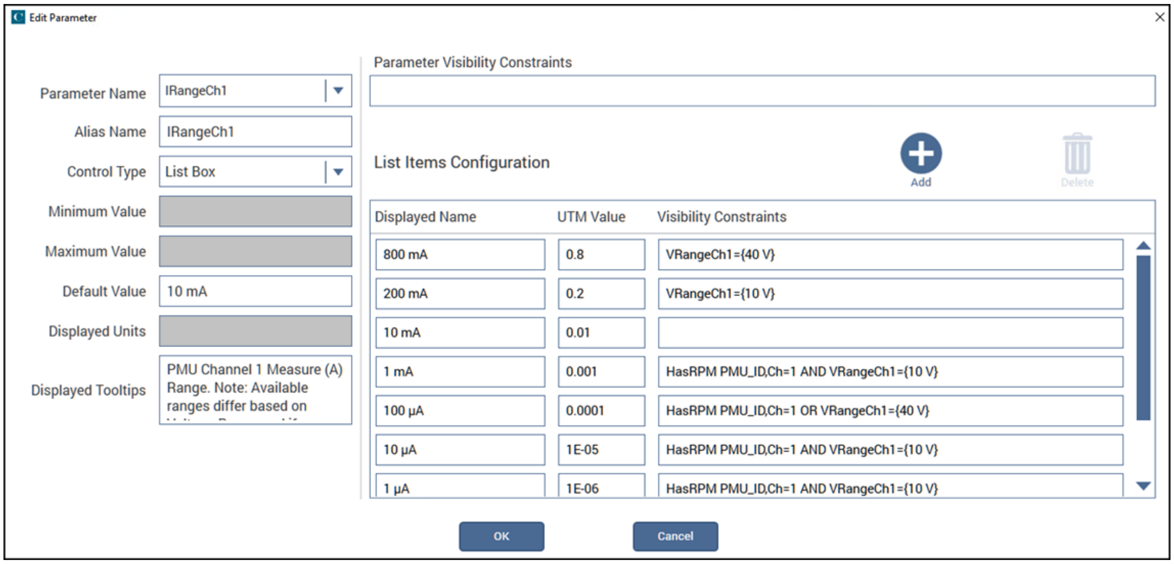
```
OperationMode=Sampling
```

Constrain items in lists

Lists are often used in the UTM UI layout to allow users to select one of several options. You can use visibility constraints to limit the options to be appropriate based on the settings of other parameters.

For example, if the voltage range (VRange) is set to `10 V`, the allowed current ranges (IRange) are `200 mA` and `10 mA`. If the Vrange is set to `40 V`, the allowed IRanges are `800 mA` and `10 mA`. In the `PMU_examples_ulib` User Library, the user module `PMU_SMU_Sweep_Example` uses these settings, as shown in the following figure.

Figure 126: Example of constraining items in lists



In this example, the Visibility Constraint for 800 mA is set to:

```
VRangeCh1={40 V}
```

The Visibility Constraint for 200 mA is set to:

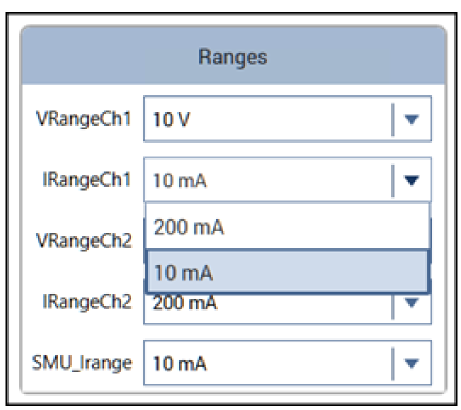
```
VRangeCh1={10 V}
```

NOTE

The brackets `{}` in the expression are used for a parameter value that contains a space in it, such as the `40 V` for the Vrange parameter.

When this UTM is displayed on the user interface, the user can only select options that meet these constraints. An example of the current range options when the voltage range is set to 10 V is shown in the following figure.

Figure 127: Configure view with constrained parameters



Constrain the outputs available in Analyze

In the Clarius Analyze view, users can select the outputs to graph. You can limit the outputs that are available for selection based on value of other parameters. For example, to use the state of the Timestamp checkbox to determine if the Timestamp output option is shown in Analyze view after running the UTM, set the Visibility Constraints to:

```
Timestamp=1
```

NOTE

Checkbox parameter values are `0` for cleared and `1` for selected.

Constrain choices based on instrumentation

You can limit the choices that are presented to the user by using the following keywords:

- `PresentInSystem`: Only instruments that are presently in the system are shown.
- `HasPA` *`smuid`*: Only SMUs with preamplifiers are displayed. *`smuid`* is the identification number of the SMU.
- `HasPA SubSMU`: This is used to populate a SMU current range list when the SMU is defined using another parameter (SubSMU). Only SMUs with preamplifiers can have preamplifier-specific current ranges.
- `HasRPM PMU_ID,Ch=`*`n`*: This returns true if the PMU defined in the `PMU_ID` parameter has an RPM connected to the channel defined by *`n`*.

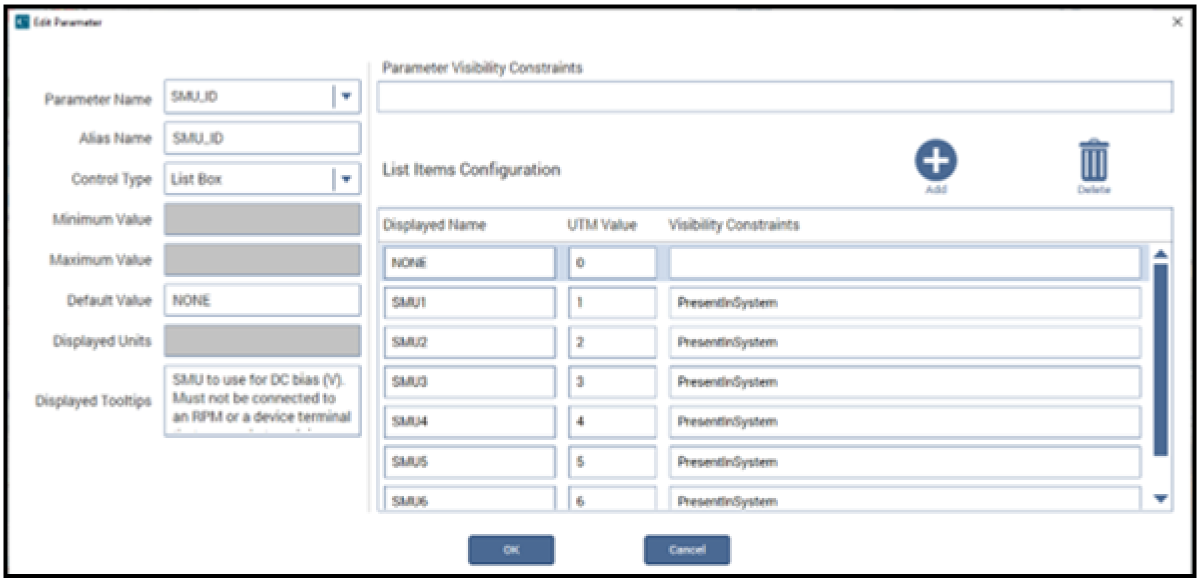
The keywords must be at the beginning of the expression.

You can use logical operators (AND or OR) with the keywords, as shown in the following example. This example returns true if the PMU defined in the `PMU_ID` parameter has an RPM connected to its channel 2 and the VRangeGate parameter is set to `10 V`.

```
HasRPM PMU_ID,Ch=2 AND VRangeGate={10 V}
```

The following figure shows an example that constrains the SMU options that are displayed. The `SMU_ID` list allows for up to nine SMUs. However, the list in the Clarius UTM configuration only shows the SMUs that are presently available in the system. This prevents the user from accidentally selecting a SMU that is not available.

Figure 128: PresentInSystem example



The following figure shows an example that constrains display of SMU current ranges so that they are only displayed when the SMU has a PA.

Figure 129: HasPA SubSMU example

Logical and comparison operators

You can constrain parameters using logical operators and comparison operators.

`AND` and `OR` are available as logical operators. When using logical operators, separate conditions with a space before and after each operator. You can have a maximum of two operators for each row.

You can use comparison operators to compare constants or parameter values. The available comparison operators are:

- `=`
- `==` (same as `=`)
- `!=`
- `<`
- `>`
- `>=`
- `<=`

For example, the following expression returns true if SpeedCVU is set to `Custom` and FilterTypeCVU is set to `Point Averaging`:

```
SpeedCVU==Custom AND FilterTypeCVU=={Point Averaging}
```

The following expression returns true if the number of points generated between VdStop and VdStart with a step defined in VdStep is less or equal to 65000. Otherwise, it returns false. This example can be used in the verification rules for a UTM.

```
(ABS((VdStop-VdStart)/VdStep)+1)<=65000
```

The following expression returns true if StepV is not 0.

```
StepV!=0
```

The following expression returns true if VrangeCh1 is set to `10 V`. VrangeCh1 is a list box with multiple items, one of which is `10 V`.

```
VrangeCh1 = {10 V}
```

The following expression returns true if Source power (`SourceVoltage*SourceCurrent`) is less than 3.5 kW on 3.5 kV source range or the source range is not 3.5 kV.

```
((ABS(SourceVoltage*SourceCurrent)<3.5E4) AND (SourceRange=={3.5 kV})) OR
    (SourceRange!={3.5 kV})
```

The following expression returns true if the DCOperationMode value is set to `Voltage Linear Sweep` or `Voltage Step`. It uses braces to present the list of multiple-choice values.

```
DCOperationMode=={Voltage Linear Sweep,Voltage Step}
```

Summary of operations for UTM UI Editor expressions

The UTM UI Editor operations described in the previous topics are summarized in the following table.

Reminders:

- Use brackets `{}` in the expressions for parameter values that contain a space in the parameter name.
- Use brackets `{}` for lists of multiple-choice values.
- You can include parameter names that are defined as input parameters to the UTM module.
- Numeric numbers are allowed.
- Engineering units and scientific notation are allowed.
- Parentheses `( )` are allowed. You can use up to 10 levels of nested parentheses.
- The maximum length of an expression is 256 characters.
- Instrument keywords ((`PresentInSystem`, `HasPA` *`smuid`*, `HasPA SubSMU`, and `HasRPM PMU_ID`) must be at the beginning of the expression.

Keyword or operator	Description
`PresentInSystem`	Limits lists to only include SMU, CVU, PMU, and PGU instruments that are presently installed in the 4200A-SCS.
`HasPA` *`smuid`*	True if a SMU preamplifier is connected to the specified SMU. *`smuid`* can be defined as a constant, such as `SMU1`, or as a parameter, such as `SubSMU`.
`HasRPM` *`pmuid`*`, Ch=`*`chanid`*	True if an RPM is connected to the specified channel of the specified PMU. *`pmuid`* can be defined as a constant, such as `PMU1`, or as a parameter, such as `PMU_ID`. *`chanid`* can be defined as a constant, such as `1`, or as a parameter, such as `GateCh`.
`OperationMode`	Set to Sweeping or Sampling.
`ABS(`*`expression`*`)`	Use the absolute value of the expression.
`AND` `OR`	Logical operators.

`+`
`-`
`/`
`*`
	Arithmetic operators.

`<`
`<=`
`>`
`>=`
`=`
`==`
`!=`
	Comparison operators.

Examples of Visibility Constraints for low current ranges

The following table provides constraint settings for the lower current ranges. The availability of these ranges depends on whether the voltage range is set to 10 V or 40 V and the presence of a 4225-RPM, which adds the lower current measure ranges to the 10 V range.

Displayed Name	UTM Value	Parameter Visibility Constraint	Comment
800 mA	0.8	`VRangeGate={40 V}`	Display this option when the selected voltage range is 40 V.
200 mA	0.2	`VRangeGate={10 V}`	Display this name when the selected voltage range is 10 V.
10 mA	0.01		Always display this option.
1 mA	0.001	`HasRPM PMU_ID,Ch=1 AND VRangeGate={10 V}`	Display this name only when the voltage range is 10 V and there is an RPM on channel 1 of the PMU defined by the `PMU_ID` parameter.
100 µA	0.0001	`HasRPM PMU_ID,Ch=1 OR VRangeGate={40 V}`	Display this name in two cases: ▪ If there is an RPM on channel 1 of the PMU defined by the `PMU_ID` parameter. ▪ If the voltage range is 40 V.
10 µA	1E-05	`HasRPM PMU_ID,Ch=1 AND VRangeGate={10 V}`	Display this name only when the voltage range is 10 V and there is an RPM on the channel 1 of the PMU defined by the `PMU_ID` parameter.
1 µA	1E-06	`HasRPM PMU_ID,Ch=1 AND VRangeGate={10 V}`	Display this name only when the voltage range is 10 V and there is an RPM on the channel 1 of the PMU defined by the `PMU_ID` parameter.
100 nA	1E-007	`HasRPM PMU_ID,Ch=1 AND VRangeGate={10 V}`	Display this name only when the voltage range is 10 V and there is an RPM on the channel 1 of the PMU defined by the `PMU_ID` parameter.

Control types

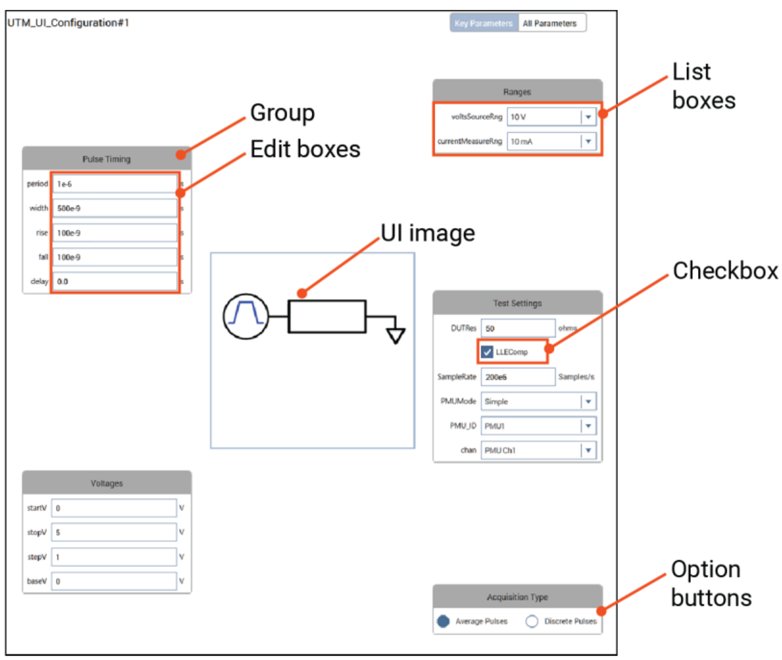
You can set up the parameters for the following control types for entry and display in the Key Parameters pane:

- Edit Box
- List Box
- Check Box
- Option Button
- Input Array
- Static Text
- SegArb Config (only available when there are input array parameters in the user module)

Descriptions of the control types are provided in the following topics.

The following figure illustrates some of the available controls for the UTM UI view, including groups, edit boxes, list boxes, check boxes, and option buttons.

Figure 130: UTM UI control type examples



Edit Box

The Edit Box Control type displays a data entry box on the UI. This is the simplest option for allowing users to change a parameter value. You can use this control type for source values (such as voltage or current), pulse timing parameters, or any other parameter that has a wide range of continuous values.

This control type may be used for all nonarray inputs. It is the default control type for all nonarray inputs in dynamically generated UTM UI views.

List Box

Use a list box to specify a set of values that the user can select from, such as an instrument, a measure range, or a source range.

For each list item, define the following characteristics:

- **Displayed Name:** The name that is displayed in the list. Create short displayed names (one or two words are best).
- **UTM Value:** Enter the user module value that correlates to this setting.
- **Visibility Constraints:** Optional. Set up rules that determine when this option is displayed. Refer to Visibility Constraints (on page 7-13) for details.

In the List Items Configuration, you can use Ctrl+Tab to move from field to field. Use Ctrl+Shift+Tab to move in the opposite direction.

An example of the List Items Configuration and the resulting list in the Configure pane is shown in the following figure. In this example, the system has two SMUs, so there are only two SMUs displayed in the list on the Configure pane, even though six are defined for the list.

Figure 131: ListBox UTM UI Parameter Configuration example

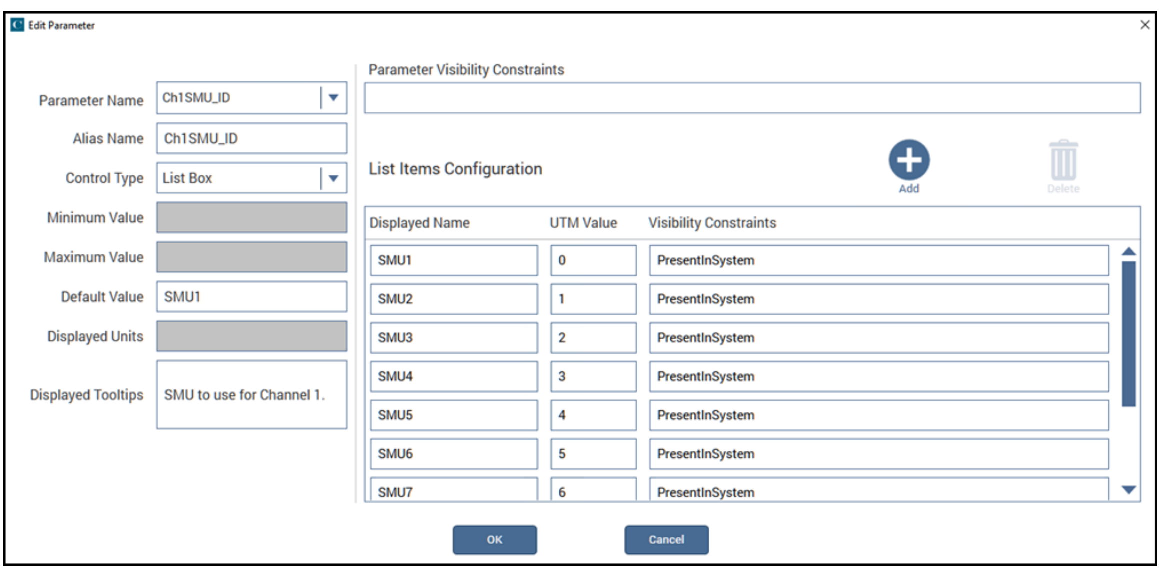
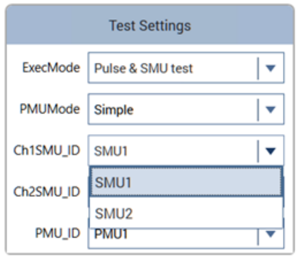


Figure 132: List constrained to two SMUs



Check Box

The Check Box control is used for parameters that have two states. This control returns a zero (`0`) to indicate the box is not selected or a one (`1`) to indicate that it is selected.

For example, in `PMU_examples_ulib`, the LLEComp parameter from the user module `PMU_1Chan_Waveform_Example` can be disabled or enabled. These values refer to the state of the LLEC compensation and are used in the LPT command `pulse_meas_wfm`. The Edit Parameter dialog for the LLEComp UTM parameter is shown in the following figure.

Figure 133: CheckBox UTM parameter UI configuration

Option Button

Use the Option Button control type when only one item in a group can be selected. Using an option button permits different values to be returned for each choice, similar to a list box.

NOTE

If you need to save space in the Key Parameters pane, use a List Box instead of an Option Button. List boxes display only one state, with a list displayed only when the use is making a selection. The Option Button control type always shows all choices.

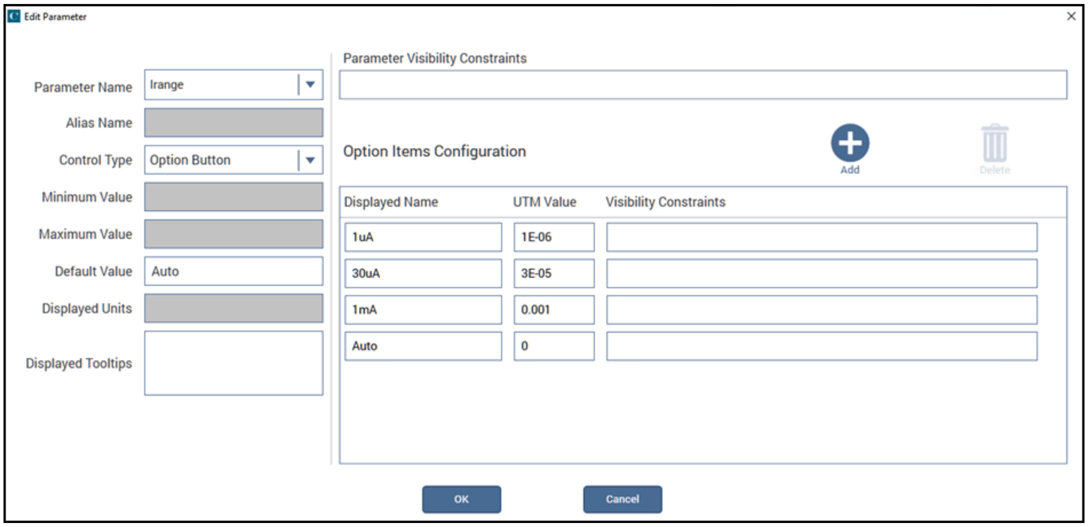
For each option button, you define:

- **Displayed Name:** The name of the option button.
- **UTM Value:** The value that is set when the option is selected.
- **Visibility Constraints:** Optional. Set up rules that determine when this option is displayed. Refer to Visibility Constraints (on page 7-13) for details.

You can only have five option buttons. If more than five are defined, the option buttons are automatically converted to a list box.

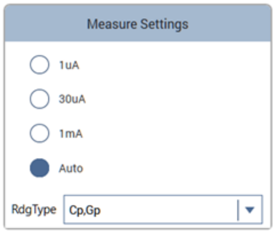
In the example in the following figure, the values correspond to the available current ranges.

Figure 134: Option button UTM UI Edit Parameter dialog



This example is displayed on the configuration pane.

Figure 135: Option Button control type example



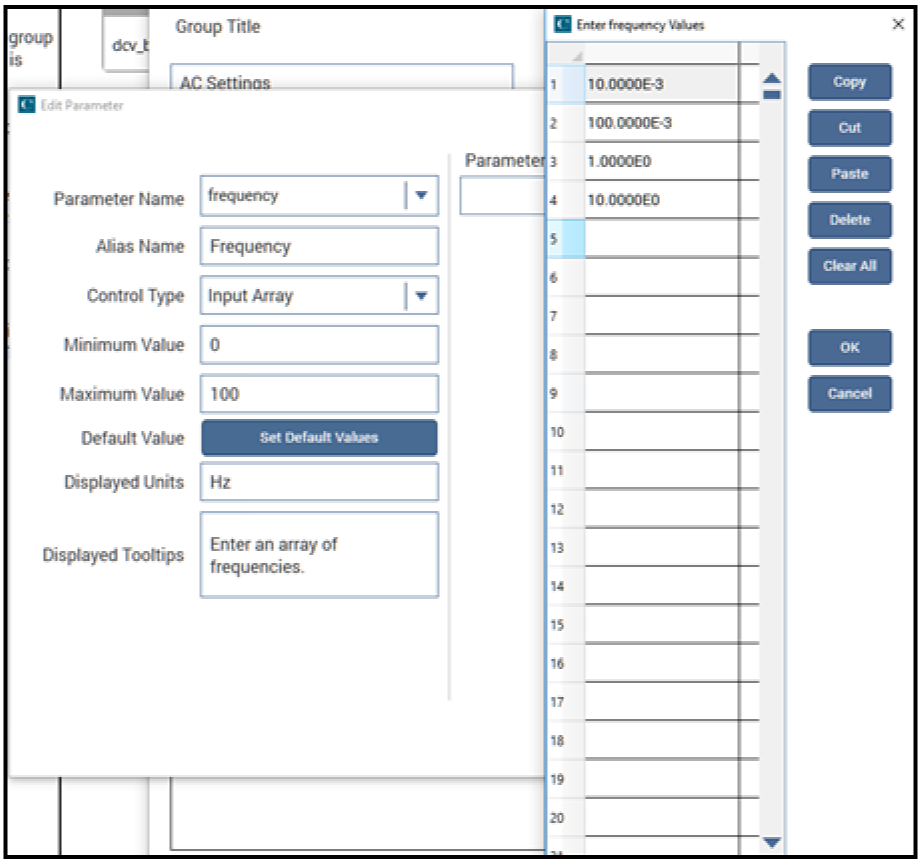
Input Array

The Input Array control type is automatically assigned to parameters that are assigned to an array data type in KULT. You cannot change the Input Array control type in the UTM UI Editor.

The following example uses the `vlfcv_measure_sweep_freq` user module in the `VLowFreqCV` user library as an example.

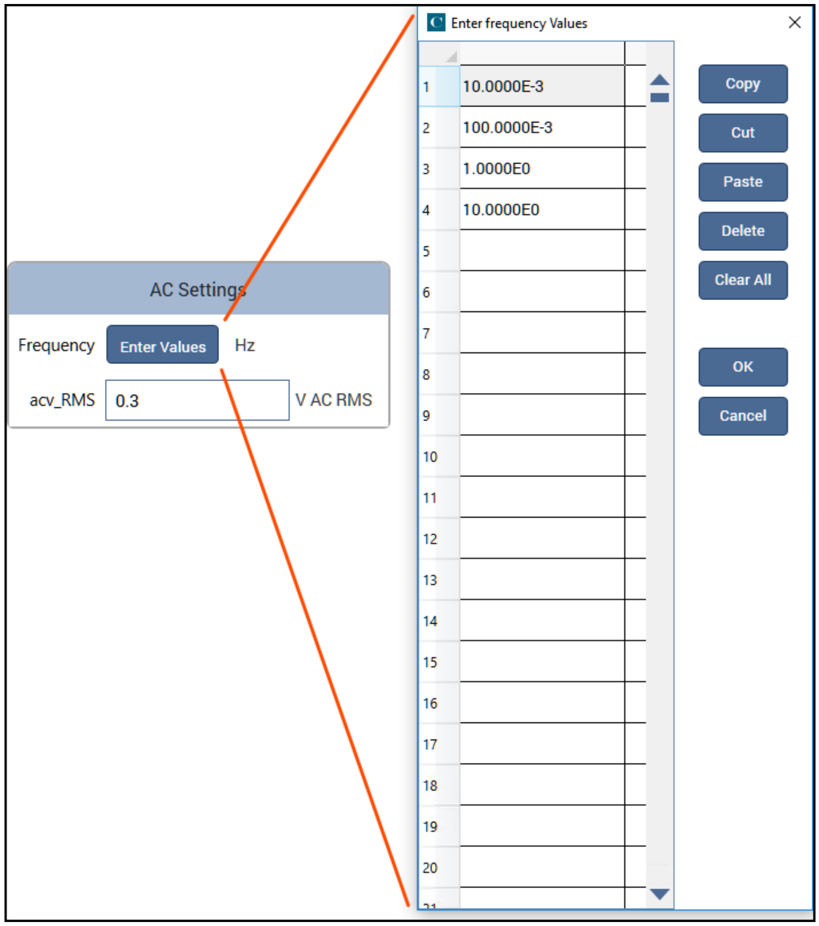
You can add default values to the list by enter the values using the Default list. An example is shown in the following figure.

Figure 136: Input Array UTM UI configuration



When the frequency parameter is placed in the UTM UI, the user can select the Enter Values button and enter array values. The list of values is presented as shown in the following figure on the Configure pane.

Figure 137: UTM Key Parameters Input Array displayed



SegARB Config

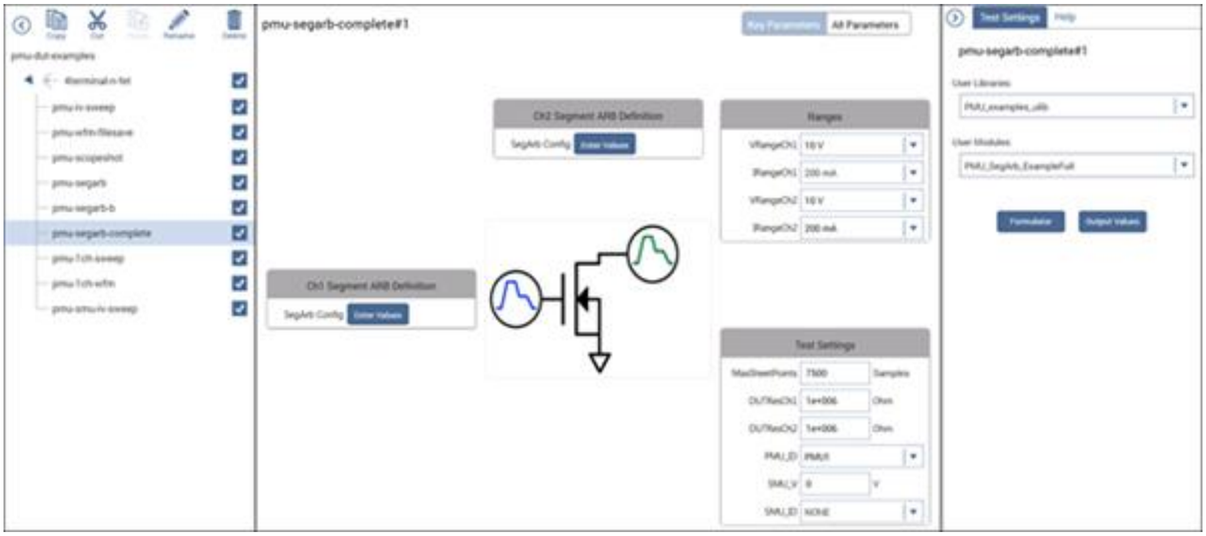
SegARB Config is the most complex control type available for the UTM UI. Segment Arb™ waveforms have many parameters and most of them are in arrays. This control type provides the interface to user modules that use the LPT commands `seg_arb_sequence` and `seg_arb_waveform`.

Setting up a Segment Arb control type

The arbitrary waveform generator Segment Arb™ features are available in UTMs using the SegARB Config control type. To work with a Segment Arb control type, assign a Segment Arb parameter to a group as described in Organize parameters into groups (on page 7-6).

The following topics use the user library `PMU_examples_ulib` and the user module `PMU_SegArb_ExampleFull` as examples. In Clarius, this module underlies the UTM `pmu-segarb-complete` test in the `pmu-dut-examples` project. The Configure pane for this test is shown in the following figure.

Figure 138: UTM Configure pane for PMU_SegArb_ExampleFull

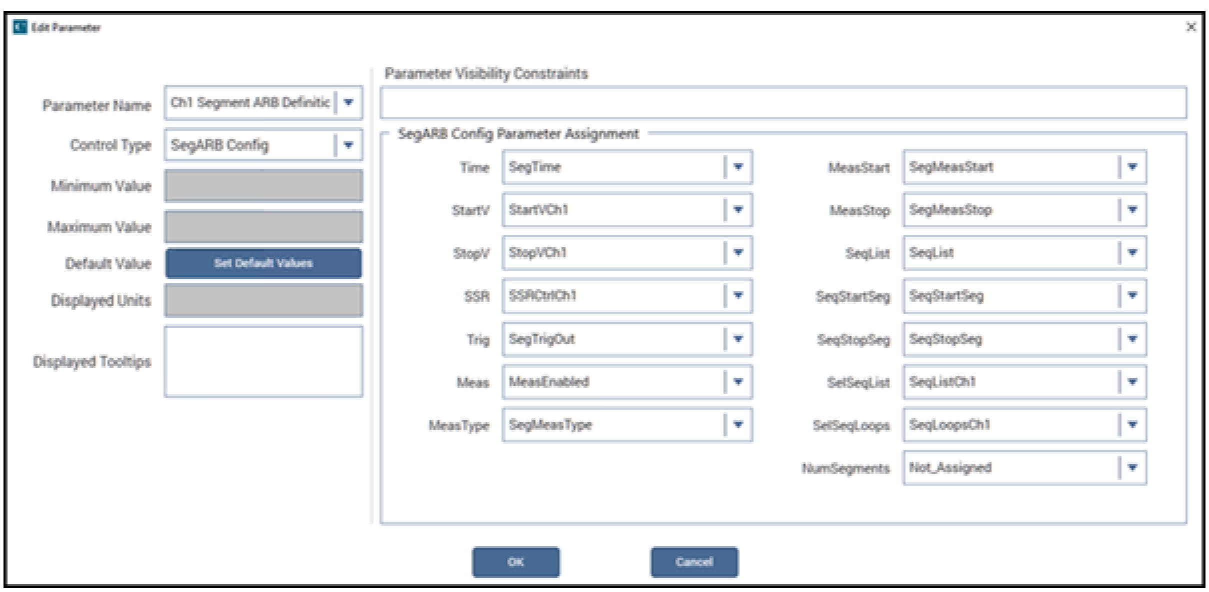


When you save the changed parameter listing, the UTM UI Editor performs some error checks, such as identifying duplicate parameter assignments. If any errors are found, an error message is displayed. You must correct the errors before saving.

To set up the Segment Arb control type:

1. Select the group that contains the SegARB Config control type and select **Edit Group**.
2. In the Group Parameters area, select the SegARB Config parameter and select **Edit** to open the Edit Parameter dialog.

Figure 139: SegARB Config UTM UI Parameter Configuration dialog



3. In SegARB Config Parameter Assignment, select the user module variable name from the list for each Segment Arb argument (such as Time).

4. Select **OK** to accept the changes and return to the Edit Group dialog.
5. Select **OK** to close the Edit Group dialog.

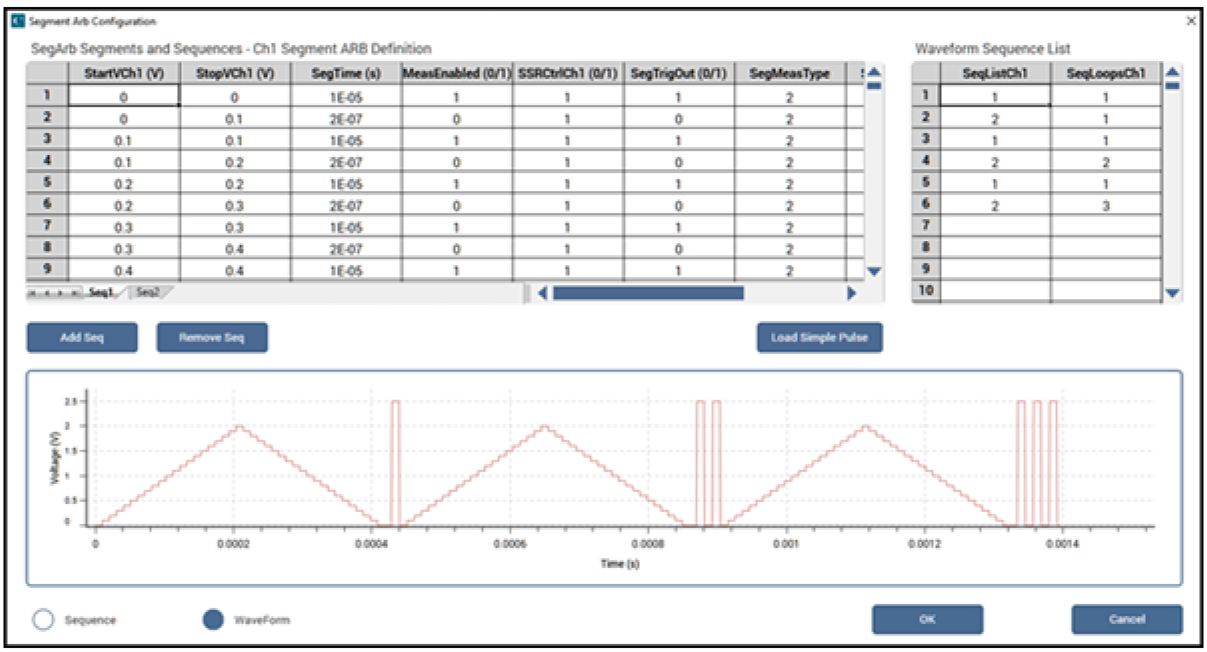
Setting up default waveforms

You can set up default waveforms for Segment Arb configurations. These are the values that are displayed when you open the Segment Arb Configuration dialog.

To set up a default waveform:

1. Select the group that contains the Segment Arb definition.
2. Select **Edit Group**.
3. Select the Segment Arb parameter.
4. Select **Edit**. The Edit Parameter dialog is displayed.
5. In the Default Value field, select **Set Default Values** to open the Segment Arb Configuration dialog.

Figure 140: Segment Arb default configuration



6. Change the values as needed. You can copy and paste using right-click options.
7. Select **OK** to save the new default values.

NOTE

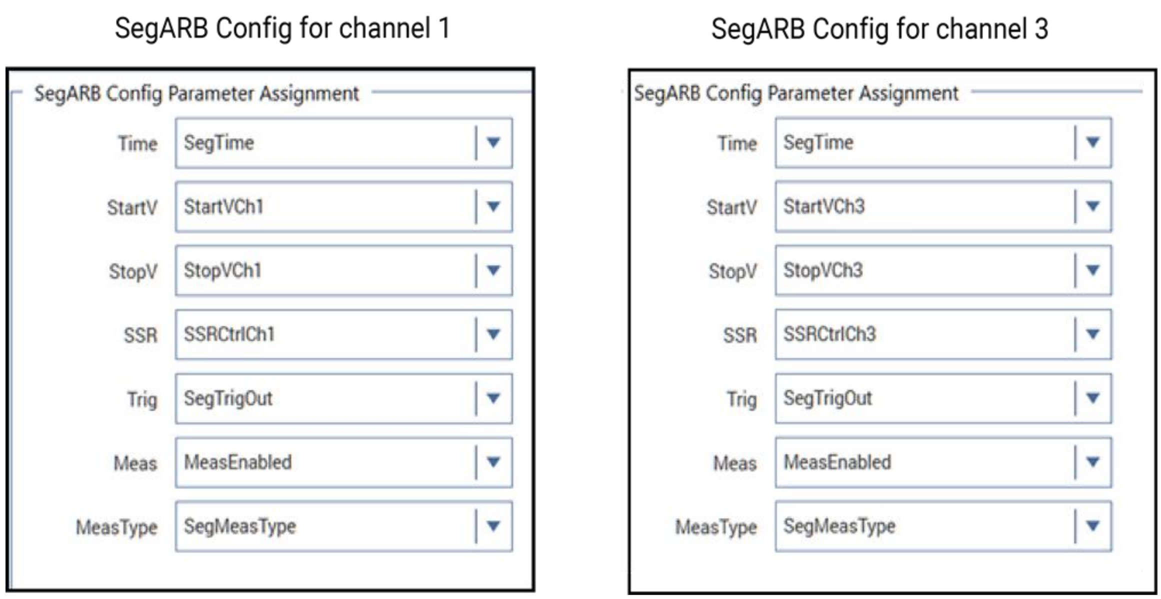
To move up and down in the table, you can use keyboard Page Up and Page Down keys. To select multiple rows, you can use Shift+Page Up or Shift+Page Down. To decrement or increment the selection, use Shift+ the up, down, left, and right arrow keys.

Using multiple SegARB Config controls

Each UTM UI can contain up to 10 Segment Arb type configurations. Each Segment Arb configuration must use unique module parameters for the `StartV`, `StopV`, `SSR`, `SelSeqList`, and `SelSeqLoops` arguments. All other Segment Arb configuration arguments, such as `Time`, `Trig`, and `Meas`, should be set to a common single parameter in the module.

For example, in the following figure, `SegTime` is a common parameter that is used in the configuration for channel 1 and channel 3. `StartV` is assigned to different parameters: `StartVCh1` for channel 1 and `StartVCh3` for channel 3.

Figure 141: Example of multiple Segment Arb configurations



Example of using the UTM UI Editor

The KULT tutorial “Creating a user module for stepping or sweeping” includes an example of setting up the user interface using the UTM UI Editor. This tutorial is in *Model 4200A-SCS KULT and KULT Extension Programming*.

UTM UI definition file information

The UTM UI Definition is stored as an `.xml` file in the source directory of the user library. The definition can be a factory or user file.

The factory file stores UTM UI definitions for all user modules in the user library. The factory files are named *user_module_name*`_GUI_Config.xml`. If you modify a UTM UI for a user module that has a factory UTM UI, Clarius automatically creates a user `.xml` file named *user_module_name*`_User_GUI_Config.xml`. If a user definition exists for a user module, it is used for the UTM UI.

If a user module does not have a factory UTM UI definition, creating a definition creates a user file.

CAUTION

Do not modify the `.xml` files outside of the UTM UI Editor. Errors or nonfunctional UTM UI definitions may result.

Both files are in the `src` directory of the user module. For example, for the factory PMU example user library:

- The user library file is named `PMU_examples_ulib_GUI_Config.xml`
- The `.xml` files are stored in the directory `C:\s4200\kiuser\usrlib\PMU_examples_ulib\src`
- User-updated UI changes are saved in the file named `PMU_examples_ulib_User_GUI_Config.xml`

Reset defaults

You can reset the UTM to the defaults from the underlying user module definition file. In the UTM UI Editor dialog, on the lower left, select **Restore Default UI**.

Section 8

Formulator

In this section:

Introduction

The Formulator allows you to make 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. Some of these may be used for real-time, in-test calculations for test data. Others can be used only for post-test data computations. Clarius automatically inserts the Formulator calculation results into the Run sheet, in addition to the raw data.

A formula created by the Formulator is an equation that is made from a series of functions, operators, constants, and arguments.

A formula created using the Formulator calculates any combination of the following:

- Test data.
- Secondary data created by other Formulator formulas.
- Standard constants from the list of constants.

Formulator functions may be limited to specific sets of data. For example:

- Some of the functions operate on only on Run tab columns of values (vectors) only.
- Some operate on single values (scalars) only.
- Some operate on both single values (scalars) and columns of values (vectors).

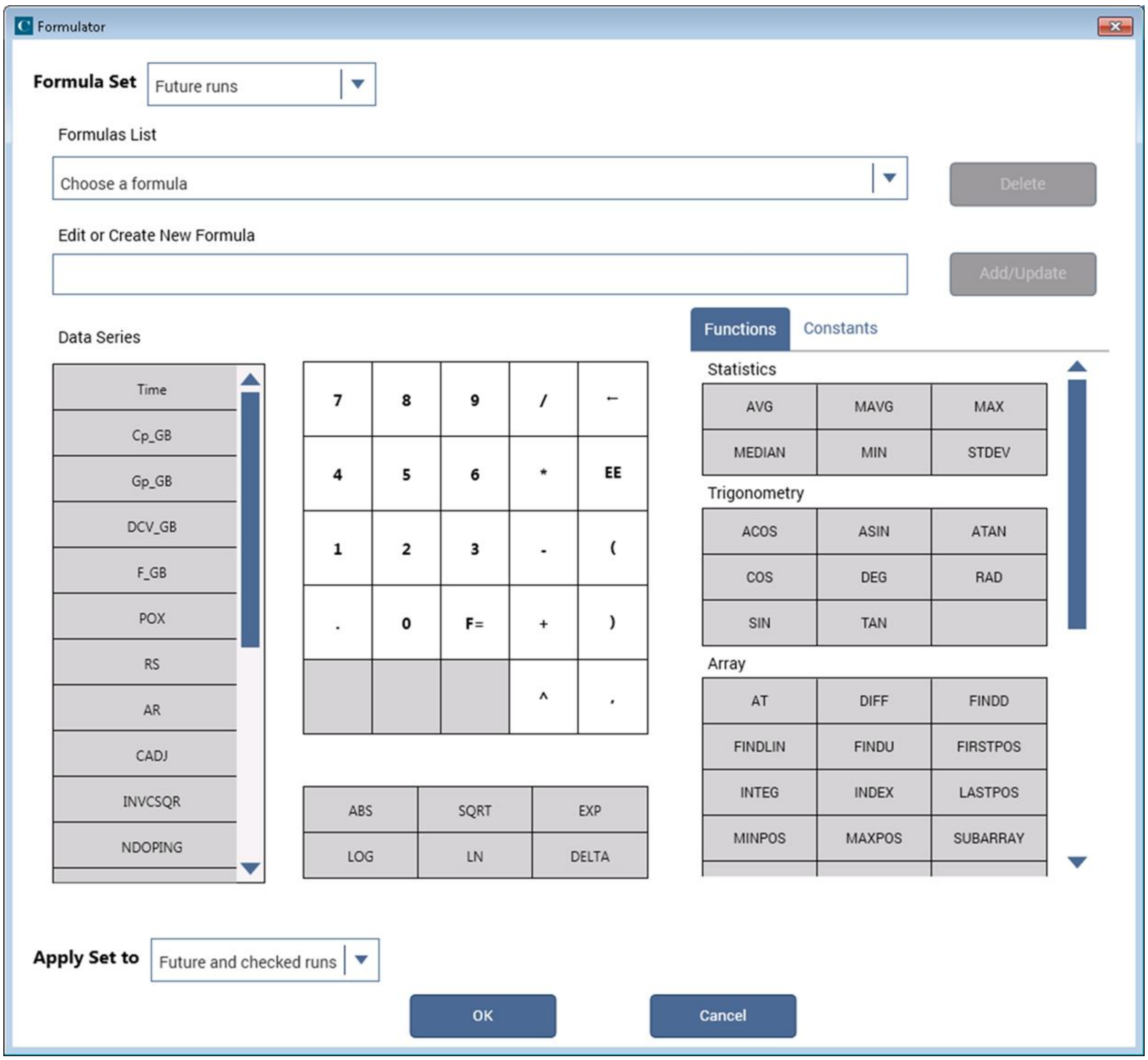
The results of some calculations may be a column of values (vector) in the Analyze Run sheet or a column that contains only a single value (scalar).

Open the Formulator

To open the Formulator:

1. In the project tree, select a test.
2. Select **Configure**.
3. In the Test Settings pane, select **Formulator**. The Formulator dialog opens, as shown below.

Figure 142: Formulator with no entries



Configure Formulator calculations

The Formulator allows you to do simple in-test calculations on test data and complex post-test data calculations. You can use the following operators and functions for in-test, real-time calculations on test data:

- Operators: +, -, *, /, ^
- General functions: ABS, SQRT, EXP, LOG, LN, DELTA

A variety of other functions may be used for post-test calculations. Refer to Formulator function reference (on page 8-13).

Formulator dialog

This section summarizes how you can use each Formulator feature.

Formula area

The top area of the Formulator dialog allows you to manage formulas.

From the **Formula Set** list, select whether you want to apply the formula to the test runs selected in Run History in the Analyze pane or to future runs.

If formulas exist, use the **Formulas List** to open a formula. When a formula is selected, the formula is displayed in the Edit or Create New Formula box.

Use the **Edit or Create New Formula** box to view, edit, or create formulas.

After adding or editing a formula, select **Add/Update** to add the calculation to the Data Series list and the Formulas List. To clear a formula that you do not want to add or update, use the backspace or delete key on the keyboard.

The **Delete** button deletes the formula that is selected in the Formulas List and removes it from the Data Series list.

Data Series

Lists the names of all columns in the Run tab of the Analyze sheet and any existing formulas. When you select a data series, the data series is added to the Edit or Create New Formula box.

When you add a formula, it is added to this list and to the Analyze sheet.

Number pad

The number pad displays number keys, mathematical operators, and `F=`. When you select an option from the number pad, the option you selected is added to the Edit or Create a New Formula box at the cursor position.

You can use `F=` in place of a variable name to complete an equation. When you add an equation that uses `F=`, Clarius adds a numeric suffix to the `F`, for example, `F1`, `F2`, `F3`. This is the heading that is used in the Analyze Run sheet for the formula.

For details about the available functions, refer to Using the Formulator functions (on page 8-5).

Functions

You can select functions to include in your formula from the Functions tab to the right of the number pad and the table below the number pad. In the Function tab, use the scroll bar to view all options.

For descriptions of each of the functions, refer to Formulator function reference (on page 8-13).

Constants

The Constants tab provides constants that you can use in the formula. Select the symbol of the constant to add it to the formula.

The definitions of the default constants are:

- **PI** π
- **K** Boltzmann constant
- **Q** Charge on an electron
- **M0** Electron mass
- **EV** Electron volt
- **U0** Permeability
- **E0** Permittivity of a vacuum
- **H** Planck’s constant
- **C** Speed of light
- **KTQ** Thermal voltage

You can edit the values and units of constants in the constants list. Place your cursor in the cell to edit and make changes as needed. Changes are automatically saved for all tests.

To add a new constant to the constants list, select **Add**. Enter the name, value, and unit for the new constant.

To delete a constant, select **Delete**. A list of constants is displayed that you can select from.

Apply Set to

This option determines which Analyze sheets this set of formulas apply to.

- **Future runs:** Only apply this set of formulas to future test runs.
- **Currently checked runs:** Only apply this set of formulas to test runs that are presently selected in the Run History pane of the Analyze sheet. The formulas are not applied to any future runs.
- **Future and currently checked runs:** Apply this set of formulas to future test runs and the runs that are presently selected in the Run History pane.

Using the Formulator options

You can use the Formulator functions, operators, and constants in combination to create simple or complex analysis equations.

You can nest multiple functions. For example, in one equation you can:

- Calculate a series of moving averages for a column of data (vector) in the Analyze sheet, using the `MAVG` function.
- Find the maximum value of the `MAVG` averages, using the `MAX` function.
- Multiply the `MAX` found value by a constant.

The equation below illustrates this use of nested Formulator functions.

```
MAXDIFF = 10*MAX(MAVG(COLUMNA))
```

The degree (number of levels) of nesting is unlimited.

The purpose, format, and arguments for the above functions and other functions available in the Formulator are described in the following topics.

Keithley recommends using the function `FIRSTPOS` as the argument for the first value in a vector:

```
[format: FIRSTPOS(DataWorksheetColumn)]
```

Similarly, use the function `LASTPOS` for the last value in the vector:

```
[format: LASTPOS(DataWorksheetColumn)]
```

In Graph tab graphs, you can directly perform composite line fits that are equivalent to the following groups of individual Formulator line fits:

- EXPFIT, EXPFITA, and EXPFITB
- LINFIT, LINFITSLP, LINFITXINT, and LINFITYINT
- LOGFIT, LOGFITA, and LOGFITB
- REGFIT, REGFITSLP, REGFITXINT, and REGFITYINT
- TANFIT, TANFITSLP, TANFITXINT, and TANFITYINT

The fit lines and parameters only display in the graphs. They are not available for use in calculations.

Correspondence between Graph tab and Formulator line fits

Formulator fit functions*				Corresponding Graph tab line fit
LINFIT	LINFITYINT	LINFITSLP	LINFITXINT	Linear
REGFIT	REGFITYINT	REGFITSLP	REGFITXINT	Regression
EXPFIT	EXPFITA	EXPFITB	———	Exponential
LOGFIT	LOGFITA	LOGFITB	———	Logarithmic
TANFIT	TANFITYINT	TANFITSLP	TANFITXINT	Tangent

* These functions calculate individual fit lines and parameters that may be used in other calculations. By contrast, the Graph tab calculates and displays only the fit line and all fit parameters.

Real-time functions, operators, and formulas

A formula that contains only real-time operators and functions is a real-time formula. If a real-time formula is specified as part of a test, it executes for each point generated by the test immediately after it is generated. The results of a real-time formula may be viewed in the Analyze sheet or plotted during the test in the same way as test data.

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

- Operators: +, −, *, /, EE, ^ (exponentiation)
- Functions: ABS, SQRT, EXP, LOG, LN, DELTA, DIFF, INTEG

The formula below is a real-time formula:

- RESULT1 = ABS(DELTA(GATECURRENT))

Real-time formulas execute as follows:

- If a real-time formula is created before the test is run, the formula executes automatically during each run.
- If a real-time formula is created after a test has been run, the formula executes initially upon adding it to the test and automatically during each subsequent run.

Post-test-only functions and formulas

Some Formulator functions are post-test-only. Post-test-only functions execute only at the end of each run of the test in which the formula is defined. The results of a post-test-only formula may be viewed in the Analyze Run sheet or plotted at the end of a test.

The post-test-only functions are listed in the following table.

AT (on page 8-26)	IFFT_I (on page 8-53)	MIN (on page 8-21)
AVG (on page 8-19)	IFFT_R (on page 8-52)	MINPOS (on page 8-31)
COND (on page 8-54)	LASTPOS (on page 8-31)	REGFIT (on page 8-42)
EXPFIT (on page 8-33)	LINFIT (on page 8-35)	REGFITSLP (on page 8-43)
EXPFITA (on page 8-34)	LINFITSLP (on page 8-36)	REGFITXINT (on page 8-44)
EXPFITB (on page 8-35)	LINFITXINT (on page 8-37)	REGFITYINT (on page 8-45)
FFT_FREQ (on page 8-50)	LINFITYINT (on page 8-37)	SMOOTH (on page 8-53)
FFT_FREQ_P (on page 8-51)	LOGFIT (on page 8-38)	SUBARRAY (on page 8-32)
FFT_I (on page 8-49)	LOGFITA (on page 8-39)	SUMMV (on page 8-32)
FFT_R (on page 8-48)	LOGFITB (on page 8-40)	TANFIT (on page 8-45)
FINDD (on page 8-27)	MAVG (on page 8-20)	TANFITSLP (on page 8-46)
FINDLIN (on page 8-27)	MAX (on page 8-20)	TANFITXINT (on page 8-47)
FINDU (on page 8-28)	MAXPOS (on page 8-31)	TANFITYINT (on page 8-47)
FIRSTPOS (on page 8-28)		

For example, the formula below is a post-test only formula, because `MAVG` is a post-test-only function:

```
RESULT2 = MAVG(ABS(DELTA(GATECURRENT)),3)
```

Post-test-only formulas execute as follows:

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

Editing Formulator formulas and constants

To edit a Formulator formula:

1. From the **Formulas List**, choose a formula. It is displayed in the Edit or Create New Formula box.
2. Edit the formula as needed.
3. Select **Add/Update**.
   - If you renamed the result variable on the left side of the formula, the Formulator adds the edited formula to the Formulas List as a new formula.
   - If you did not rename the variable on the left side of the formula, a confirmation dialog is displayed.
4. Select:
   - **No** if you edited the formula to create a new formula. Nothing happens to either of the formula boxes. Edit the name of the result variable, then select **Add/Update** again.
   - **Yes** if you edited the formula to update it. The replacement formula appears in the Formulas List.

Deleting Formulator formulas and constants

To delete a Formulator formula:

1. From the **Formulas List**, select the formula.
2. Select **Delete**.

Identify data analysis requirements

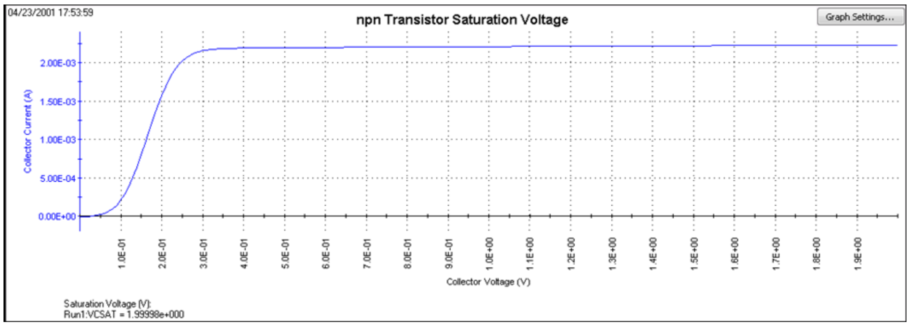
In many cases, you may already know a needed analysis formula, even before running a test. In fact, for a test, you may create a real-time formula in advance so that you can monitor its output during a test, either in the Analyze sheet or graph.

However, in other cases, you may decide to perform an analysis after a review of existing test data. The Formulator functions to use and the data to be included in the Formulator calculations must be evaluated to meet your requirements. The following topics illustrate one such evaluation.

Determining the type of calculation: an example

For example, after looking at the BJT saturation voltage plot below, you need to have a better look at the point where the slope of the saturation plateau becomes constant.

Figure 143: Formulator example data



You might decide to apply the function `REGFIT` to the CollectorV values (and corresponding CollectorI values) between 1 V and 3 V. The line generated by `REGFIT`, when co-plotted with the existing curve, should depart from the plateau at the point of curvature.

The following topics apply the `REGFIT` function to the data in the above figure to demonstrate use of the Formulator.

Determining range data for a calculation: an example

Many Formulator functions do not require you to specify row numbers (indices) as arguments. However, some Formulator functions, such as `REGFIT`, require you to specify the range of data to be included in the calculation. These are typically as the row numbers (indices) for the first and last values to be included (refer to Using the Formulator functions (on page 8-5)). This requirement allows you to apply a calculation to only a specific part of the data.

To find the corresponding row numbers (indices) for that specific part of the data, check the Run sheet. For example, referring to the following figure, you might decide to apply `REGFIT` only to the `CollectorV` values between 1 V and 2 V. Looking at the Analyze sheet for this data, you note that the `CollectorV` values between 1 V and 2 V are located between rows 101 and 201. This is the range information that you need to create the regression analysis equation using `REGFIT`.

Figure 144: Determining the starting and ending row numbers (indices) for the data to be analyzed

	CollectorI	CollectorV	BaseV
97	2.2098E-3	959.9553E-3	699.0722E-3
98	2.2100E-3	969.9862E-3	699.0660E-3
99	2.2102E-3	979.9798E-3	699.0726E-3
100	2.2105E-3	990.0086E-3	699.0747E-3
101	2.2107E-3	1.0000E+0	699.0748E-3
102	2.2110E-3	1.0100E+0	699.0801E-3
103	2.2112E-3	1.0200E+0	699.0759E-3
104	2.2114E-3	1.0300E+0	699.0726E-3
105	2.2117E-3	1.0400E+0	699.0829E-3
106	2.2119E-3	1.0500E+0	699.0737E-3
107	2.2122E-3	1.0600E+0	699.0722E-3
108	2.2124E-3	1.0701E+0	699.0723E-3
109	2.2126E-3	1.0800E+0	699.0801E-3
110	2.2129E-3	1.0901E+0	699.0789E-3
111	2.2131E-3	1.1001E+0	699.0743E-3
112	2.2133E-3	1.1101E+0	699.0740E-3
113	2.2136E-3	1.1200E+0	699.0748E-3
114	2.2138E-3	1.1301E+0	699.0600E-3
115	2.2140E-3	1.1400E+0	699.0608E-3
116	2.2143E-3	1.1500E+0	699.0635E-3
117	2.2145E-3	1.1600E+0	699.0668E-3
118	2.2147E-3	1.1700E+0	699.0674E-3
119	2.2149E-3	1.1800E+0	699.0572E-3
120	2.2152E-3	1.1900E+0	699.0560E-3
121	2.2154E-3	1.2000E+0	699.0518E-3
122	2.2156E-3	1.2099E+0	699.0555E-3
123	2.2158E-3	1.2199E+0	699.0443E-3

	CollectorI	CollectorV	BaseV
175	2.2267E-3	1.7399E+0	698.9456E-3
176	2.2269E-3	1.7499E+0	698.9495E-3
177	2.2271E-3	1.7599E+0	698.9557E-3
178	2.2273E-3	1.7699E+0	698.9453E-3
179	2.2275E-3	1.7799E+0	698.9540E-3
180	2.2277E-3	1.7899E+0	698.9564E-3
181	2.2278E-3	1.7999E+0	698.9256E-3
182	2.2280E-3	1.8099E+0	698.9381E-3
183	2.2282E-3	1.8199E+0	698.9393E-3
184	2.2284E-3	1.8299E+0	698.9346E-3
185	2.2286E-3	1.8399E+0	698.9323E-3
186	2.2288E-3	1.8499E+0	698.9418E-3
187	2.2290E-3	1.8599E+0	698.9480E-3
188	2.2292E-3	1.8700E+0	698.9329E-3
189	2.2294E-3	1.8800E+0	698.9381E-3
190	2.2296E-3	1.8900E+0	698.9332E-3
191	2.2298E-3	1.9000E+0	698.9386E-3
192	2.2300E-3	1.9100E+0	698.9304E-3
193	2.2301E-3	1.9200E+0	698.9313E-3
194	2.2303E-3	1.9300E+0	698.9384E-3
195	2.2305E-3	1.9400E+0	698.9293E-3
196	2.2307E-3	1.9500E+0	698.9235E-3
197	2.2309E-3	1.9600E+0	698.9282E-3
198	2.2311E-3	1.9700E+0	698.9354E-3
199	2.2313E-3	1.9799E+0	698.9368E-3
200	2.2315E-3	1.9900E+0	698.9073E-3
201	2.2317E-3	1.9999E+0	698.9119E-3

Creating an analysis formula

After you have identified the needed Formulator functions and data, create an analysis formula.

To create an analysis formula, complete the following steps:

1. Enter the left side of the equation. You can use the **F=** option (available from the center number pad) or type in a variable name that contains no spaces and is not the same as a Function name.

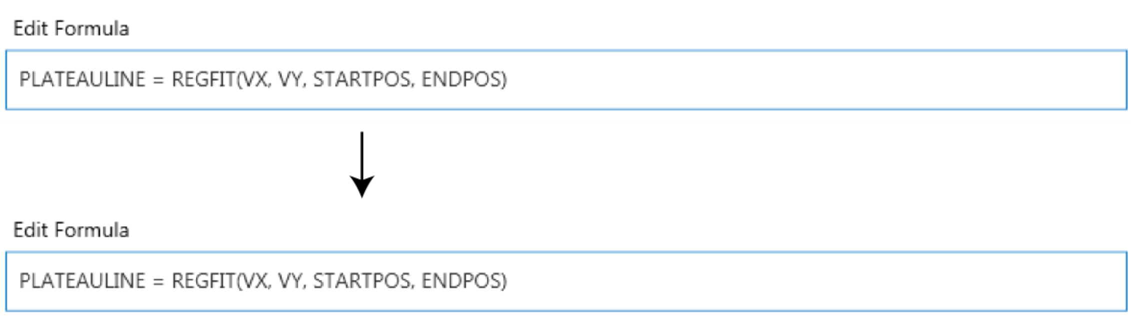
NOTE

Each time that you create an equation with **F=**, the Formulator adds a sequential numerical suffix to the F when you select **Add**. That is, the left side of the first equation is **F1 =**, the left side of the second is **F2 =**, and so on.

2. Enter the right side of the equation at using the function buttons, constant buttons, columns buttons, and keyboard, as appropriate.
   - To insert a function or operator, select a button in the Functions area.
   - To replace the format version of an argument in a function (for example, V1) with a column (vector) or value from the Data Series area, select the area in the formula and select the Data Series item.
   - To insert a constant from the Constants area, select the constant.

For example, to find the regression line for the plateau in the figure in Determining the type of calculation: an example (on page 8-9), enter the equation shown in the following figure.

Figure 145: Creating the regression formula for the data



Adding an analysis formula to the test

To display a formula in the Edit or Create New Formula, box, select it.

After editing an existing formula or creating a new one, select **Add/Update**. You are given the option to replace the same-named formula in the lower box or to rename and add it to the collection of formulas. Refer also to Editing formulas and constants (on page 8-8).

Executing an analysis formula

If you specify future project data as arguments of a formula (for example, you create the formula when you configure the associated test before running it) the following occurs:

- If you compose the formula using exclusively real-time functions, it executes in real time during each run of a test.
- If you compose a formula containing one or more post test only functions, it executes at the end of each run of the test.

If you specify existing project data as the arguments of a formula, the formula immediately executes and acts on the existing data when you select **Add/Update**. Thereafter, it executes as listed above.

Viewing analysis results in the Analyze sheet

After executing a new formula, a new column of data containing the results is added to the Analyze sheet. If the formula in the Formulas List was edited to replace a previous version, the corresponding column in the sheet updates to reflect the changes.

In some cases, a results column contains only a single value.

NOTE

After some Formulator calculations, you may see one or more instances of `#REF` in a column, instead of a number. `#REF` in a cell indicates that a valid value could not be calculated. This occurs when a Formulator function needs multiple rows as arguments, when a calculated value is out of range, when a divide by zero is attempted, and so on.

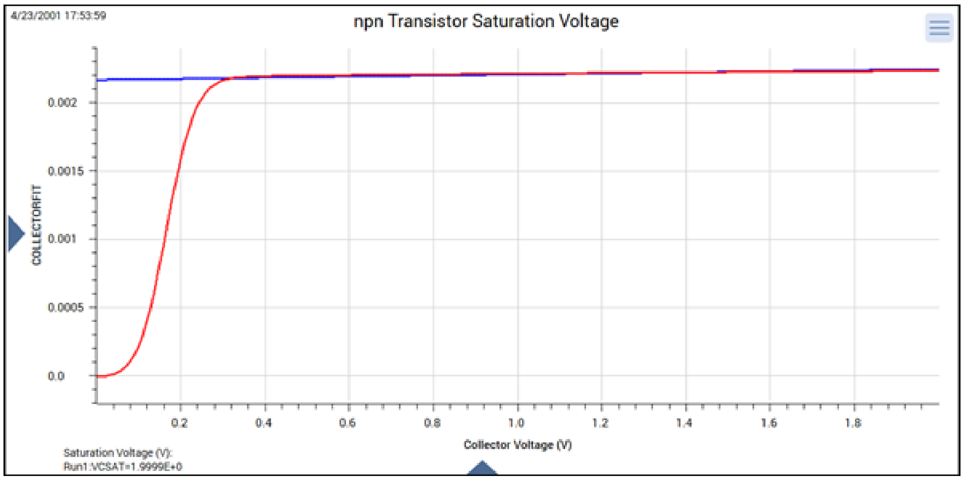
For example, each result of the `DIFF` function is a difference coefficient that is calculated as the ratio `DValues1/DValues2`, where `DValues1` and `DValues2` are differences between values in the present row and values in the previous row. Because no previous row exists before the first row, a valid calculation is not possible for the first row.
Therefore, the Formulator returns `#REF` in the first row.

A column contains multiple instances of `#REF` if the Formulator function requires multiple prior cells for the calculation. For example, if the `MAVG` function is using five points to calculate a moving average of a column containing five values, the first two and last two cells contain `#REF`.

Viewing analysis results in the Analyze graph

If a new column (vector) is added to the Analyze sheet after you create or change a formula, it can be plotted in the Analyze graph like any other column (vector). See the following figure for an example of the Formulator function REGFIT added to the graph.

Figure 146: Added linear regression line to the graph



NOTE

For information about using the Graph tab, refer to Analyze data (on page 3-1).

Formulator function reference

Each of the Formulator functions is described in the following topics.

General

Function	Brief description
ABS	Calculates the absolute value of each value in the designated column (vector) or the absolute value of any operand.
DELTA	Returns the differences between the adjacent values in a column (vector).
EXP	Returns the exponent, e^{value}, for each value in a column (vector) or for any operand.
LN	Returns the base-e (natural, Napierian) log of each value in a designated column (vector) or the Napierian log of any operand.
LOG	Returns the base-10 log of each value in a designated column (vector) or the base-10 log of any operand.
SQRT	Returns the square root of each value in a designated column (vector) or the square root of any operand.

Statistics

Function	Brief description
AVG	Returns the average of all values in the column (vector).
MAVG	Returns a new column (vector) consisting of the moving averages of successive groups of points from another column (vector).
MAX	Searches all values in a column (vector) and returns the maximum value.
MEDIAN	Searches all values in a column (vector), finds the middle point of that column used, and returns the value.
MIN	Searches all values in a column (vector) and returns the minimum value.
STDEV	Returns the standard deviation of all values in the column (vector).

Trigonometric

Function	Brief description
ACOS	Returns the arc cosine of each value in a designated column (vector) under Columns or any operand.
ASIN	Returns the arc sine of each value in a designated column (vector) under **Columns** or any operand.
ATAN	Returns the arc tangent of each value in a designated column (vector) under **Columns** or any operand.
COS	Returns the cosine of each value operand.
DEG	Converts an angle value in radians to degrees.
RAD	Converts an angle value in degrees to radians.
SIN	Returns the sine of each value in a designated column (vector) under **Columns** or any operand.
TAN	Returns the tangent of each value in a designated column (vector) under **Columns** or any operand.

Array

Function	Brief description
AT	Extracts and returns a single value from a column (vector).
DIFF	For all the values in two selected columns (vectors), returns a third column (vector) that contains the difference between the coefficients.
FINDD	Searches down the column until it finds a value that matches the user-specified value *X*. FINDD searches a column *V*, beginning at *START*. Then it returns the row number of that value.
FINDLIN	Searches down the column until it finds a value that is closest, but does not exceed, the user-specified value *X*.
FINDU	Searches up the column until it finds a value that matches the user-specified value *X*.
FIRSTPOS	Returns the row number of the first value in a column.
INTEG	From two columns (vectors) *VX* and *VY*, each one containing *N* values, the INTEG function returns a third column (vector) containing a series of numerical integrals A_n, where n = 1, 2, ..., N-1, N.
INDEX	Returns a specified number of points starting with a specified value and consecutive values incremented by one.
LASTPOS	Returns the row number of the last value in a column.
MAXPOS	Searches all values in a column, finds the maximum value, and returns the row number of the maximum value.
MINPOS	Searches all values in a column, finds the minimum value, and returns the row number of the minimum value.

Function	Brief description
SUBARRAY	Returns a new column containing a specified range of the values from an existing column.
SUMMV	Returns a column (vector) *VY* that consists of moving summation of a column (vector) V.

Line fits

Function	Brief description
EXPFIT	Performs an exponential fit and returns a new column of Y values.
EXPFITA	Returns the value of the constant EXPFITA as part of the formula to perform an exponential fit.
EXPFITB	Returns the value of the constant EXPFITB as part of the formula to perform an exponential fit.
LINFIT	Finds a linear equation and returns a new column.
LINFITSLP	Finds a linear equation and returns the slope.
LINFITXINT	Finds a linear equation and returns the X intercept.
LINFITYINT	Finds a linear equation and returns the Y intercept.
LOGFIT	Performs a base-10 log-linear fit.
LOGFITA	Performs a base-10 log-linear fit.
LOGFITB	Performs a base-10 log-linear fit.
POLY2FIT	Enables quadratic regression line fitting.
POLY2COEFF	Enables quadratic regression line fitting.
POLYNFIT	POLYNFIT (n^{th} order) does polynomial approximation from the 1st order to the 9th order.
REGFIT	Performs a linear regression fit.
REGFITSLP	Performs a linear regression fit.
REGFITXINT	Performs a linear regression fit.
REGFITYINT	Performs a linear regression fit.
TANFIT	Finds a linear equation of the form Y = a + bX from two columns, *VX* and *VY*.
TANFITSLP	Finds a linear equation of the form Y = a + bX from two columns, *VX* and *VY*.
TANFITXINT	Finds a linear equation of the form Y = a + bX from two columns, *VX* and *VY*.
TANFITYINT	Finds a linear equation of the form Y = a + bX from two columns, *VX* and *VY*.

FFT

Function	Brief description
FFT_R	Performs an FFT on the provided input arrays and then returns the real parts.
FFT_I	Performs an FFT on the provided input arrays and then returns the imaginary parts.
FFT_FREQ	Returns an array of positive and negative frequencies that correspond to the frequencies of an FFT output.
FFT_FREQ_P	This formula returns an array of the positive frequencies that correspond to the frequencies of an FFT output.
IFFT_R	Performs an inverse FFT on the provided input arrays and then returns the real parts scaled by 1/*N*, where *N* is the number of samples.
IFFT_I	Performs an inverse FFT on the provided input arrays and then returns the imaginary parts scaled by 1/*N*, where *N* is the number of samples.
SMOOTH	Performs digital filtering on an input array by zeroing out high frequency components.

Misc

Function	Brief description
`COND`	Returns one of two user-defined expressions, depending on the comparison of two other user-defined expressions.

General

The following Formulator functions provide general operations.

ABS Formulator function

Calculates the absolute value of each value in the designated column (vector) or the absolute value of any operand.

Usage

`ABS(`*`Value`*`)`

`Value`	The name of any column (vector) in the Data Series list or any operand

Details

You can use this function to do calculations in real time (while a test is executing).

Example

`F2 = ABS(GateI)`	Returns the absolute value of the gate current.

Also see

None

DELTA Formulator function

This command returns the differences between the adjacent values in a column (vector). That is, for column V, `DELTA` returns (V2 − V1), (V3 − V2), and so on.

Usage

`DELTA(Value)`

Value	The name of any column (vector) in the Data Series list

Details

You can use this function to do calculations in real time (while a test is executing).

Example

```
GM = DELTA(DRAINI)/DELTA(GATEV)
```

Also see

None

EXP Formulator function

Returns the exponent, e^{value}, for each value in a column (vector) or for any operand.

Usage

`EXP(Value)`

Value	The name of any column (vector) in the Data Series list or any operand

Details

You can use this function to do calculations in real time (while a test is executing).

Example

```
NEWCURRENT = CURRENT*EXP(ANODEV)
```

Also see

LN Formulator function (on page 8-18)

LN Formulator function

This command returns the base-e (natural, Napierian) log of each value in a designated column (vector) or the Napierian log of any operand.

Usage

`LN(Value)`

Value	The name of any column (vector) in the Data Series list or any operand

Details

You can use this function to do calculations in real time (while a test is executing).

Example

```
DIODEV = LN(ANODEI)*0.026
```

Also see

EXP Formulator function (on page 8-17)

LOG Formulator function

Returns the base-10 log of each value in a designated column (vector) or the base-10 log of any operand.

Usage

`LOG(Value)`

Value	The name of any column (vector) in the Data Series list or any operand

Details

You can use this function to do calculations in real time (while a test is executing).

Example

```
F1 = LOG(DRAINI)
```

Also see

None

SQRT Formulator function

Returns the square root of each value in a designated column (vector) or the square root of any operand.

Usage

`SQRT(Value)`

Value	The name of any column (vector) in the Data Series list or any operand

Details

A negative value of `X` returns `#REF` in the spreadsheet.

You can use this function to do calculations in real time (while a test is executing).

Example

```
TWO = SQRT(4)
```

Also see

None

Statistics

The following Formulator functions provide statistics operations.

AVG Formulator function

Returns the average of all values in the column (vector).

Usage

`AVG(Value)`

Value	The name of any column (vector) in the Data Series list

Example

```
LEAKAGE = AVG(GATEI)
```

Also see

MAVG (on page 8-20)

MAVG Formulator function

Returns a new column (vector) consisting of the moving averages of successive groups of points from another column (vector).

Usage

`MAVG(V, N)`

V	The name of any column (vector) in the Data Series list or any operand
N	The number of points to be averaged in each group

Details

You can configure the number of points in a group.

If *N* = 3 and *V* contains the 12 values `X1`, `X2`, `X3`, `X4`, `X5`, `. . .` , `X10`, `X11`, `X12`, then `MAVG` returns a column (vector) that contains the following values:

`#REF, (X1 + X2 + X3)/3, (X2 + X3+ X4)/3, (X3 + X4+ X5)/3, ... (X10 + X11+ X12)/3, #REF`

The new column's values in the new column may contain instances of `#REF` (as shown above) because `MAVG` uses cells from both sides of the target cell for its calculation.

Example

```
FILTER = MAVG(GATEI, 3)
```

Also see

None

MAX Formulator function

Searches all values in a column (vector) and returns the maximum value.

Usage

`MAX(Value)`

Value	The name of any column (vector) in the Data Series list

Example

```
MAXGM = MAX(DIFF(DRAINI, GATEV))
```

Also see

None

MEDIAN Formulator function

Searches all values in a column (vector), finds the middle point of that column used, and returns the value.

Usage

`MEDIAN(Value)`

Value	The name of any column (vector) in the Data Series list

Also see

None

MIN Formulator function

Searches all values in a column (vector) and returns the minimum value.

Usage

`MIN(Value)`

Value	The name of any column (vector) in the Data Series list

Example

```
SMALLESTI = MIN(DRAINI)
```

Also see

None

STDEV Formulator function

Returns the standard deviation of all values in the column (vector).

Usage

`STDEV(Value)`

Value	The name of any column (vector) in the Data Series list or any operand

Details

Returns the standard deviation.

Example

```
LEAKAGE = STDEV(GATEI)
```

Also see

None

Trigonometry

The following Formulator functions provide trigonometric operations.

ACOS Formulator function

Returns the arc cosine of each value in a designated column (vector) under Columns or any operand.

Usage

`ACOS(`*`Value`*`)`

Value	The name of any column (vector) in the Data Series list or any operand

Details

Returns the value in radians.

Example

```
F1 = ACOS(DRAINI)
```

Also see

None

ASIN Formulator function

Returns the arc sine of each value in a designated column (vector) under **Columns** or any operand.

Usage

`ASIN(`*`Value`*`)`

Value	The name of any column (vector) in the Data Series list or any operand

Details

Returns the value in radians.

Example

```
F1 = ASIN(DRAINI)
```

Also see

None

ATAN Formulator function

Returns the arc tangent of each value in a designated column (vector) under **Columns** or any operand.

Usage

`ATAN(Value)`

Value	The name of any column (vector) in the Data Series list or any operand

Details

Returns the value in radians.

Example

```
F1 = ATAN(DRAINI)
```

Also see

None

COS Formulator function

Returns the cosine of each value operand.

Usage

`COS(Value)`

Value	The name of any column (vector) in the Data Series list or any operand

Details

Returns the value in radians.

Example

```
F1 = COS(DRAINI)
```

Also see

None

DEG Formulator function

Converts an angle value in radians to degrees.

Usage

`DEG(Value)`

Value	The name of any column (vector) in the Data Series list or any operand

Details

Returns the value in degrees.

Example

```
F1 = DEG(ANGLE)
```

Also see

None

RAD Formulator function

Converts an angle value in degrees to radians.

Usage

`RAD(Value)`

Value	The name of any column (vector) in the Data Series list or any operand

Details

Returns the value in radians.

Example

```
F1 = RAD(ANGLE)
```

Also see

None

SIN Formulator function

Returns the sine of each value in a designated column (vector) under **Columns** or any operand.

Usage

`SIN(Value)`

Value	The name of any column (vector) in the Data Series list or any operand

Details

Returns the value in radians.

Example

```
F1 = SIN(DRAINI)
```

Also see

None

TAN Formulator function

Returns the tangent of each value in a designated column (vector) under **Columns** or any operand.

Usage

`TAN(Value)`

Value	The name of any column (vector) in the Data Series list or any operand

Details

Returns the value in radians.

Example

```
F1 = TAN(DRAINI)
```

Also see

None

Array

The following Formulator functions work with arrays.

AT Formulator function

Extracts and returns a single value from a column (vector).

Usage

`AT(Value, POS)`

Value	The name of any column (vector) in the Data Series list or any operand
POS	The row number of column *Value* where the single value is located

Example

```
IDSAT = AT(DRAINI, 36)
```

Also see

None

DIFF Formulator function

For all the values in two selected columns (vectors), returns a third column (vector) that contains the difference between the coefficients.

Usage

`DIFF(V1, V2)`

V1	The name of any column (vector) listed under **Columns**
V2	The name of any column (vector) listed under **Columns**

Details

Each coefficient is calculated as follows:

V1/*V2*

Where:

- *V1* = The difference between a pair of adjacent values in the first column.
- *V2* = The difference between the corresponding values in the second column.

That is, for columns *V1* and *V2*, `DIFF` returns the following:

$(V1_2 - V1_1)/(V2_2 - V2_1)$, $(V1_3 - V1_2)/(V2_3 - V2_2)$, and so on.

You can use this function to do calculations in real time (while a test is executing).

Example

```
GM = DIFF(DRAINI, GATEV)
```

Also see

None

FINDD Formulator function

The find down function searches down the column until it finds a value that matches the user-specified value *X*. `FINDD` searches a column *V*, beginning at *START*. Then it returns the row number of that value.

Usage

`FINDD(V, X, START)`

V	The name of any column (vector) listed under Columns
X	Any value, which may be the result of another calculations
START	The row number of the starting value for the search

Details

If `FINDD` does not find an exact match for *X*, it returns the row number of the *V* value that is closest to *X*.

Example

```
IF = AT(ANODEI, FINDD(ANODEV, 0.7, FIRSTPOS(ANODEV)))
```

Also see

FINDLIN (on page 8-27)
FINDU (on page 8-28)

FINDLIN Formulator function

Find using linear interpolation searches down the column until it finds a value that is closest, but does not exceed, the user-specified value *X*.

Usage

`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 of the starting value for the search

Details

`FINDLIN` searches a column *V*, beginning at *START*. Linear interpolation is then used to determine its decimal location between the found value and the next value in the column. The returned index number (in decimal format) indicates the position of the specified value.

Assume you want to use `FINDLIN` to locate value 6 in the following array:

```
(Index 1)0
(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) that 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.

Example

```
IF = AT(ANODEI, FINDLIN(ANODEV, 0.7, FIRSTPOS(ANODEV)))
```

Also see

FINDD (on page 8-27)
FINDU (on page 8-28)

FINDU Formulator function

The find up function searches up the column until it finds a value that matches the user-specified value *X*. It then returns the row number of that value. `FINDU` searches a column *V*, beginning at *START*. Then it returns the row number of that value.

Usage

`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
STARTPOS	The row number of the starting value for the search

Details

If `FINDU` does not find an exact match for *X*, it returns the row number of the *V* value that is closest to *X*.

Example

```
IF = AT(ANODEI, FINDU(ANODEV, 0.7, LASTPOS(ANODEV)))
```

Also see

FINDD (on page 8-27)
FINDLIN (on page 8-27)

FIRSTPOS Formulator function

Returns the row number of the first value in a column.

Usage

`FIRSTPOS(V)`

V	The name of any column (vector) listed under **Columns**

Example

```
STARTOFARRAY = FIRSTPOS(DRAINI)
```

Also see

LASTPOS (on page 8-31)

INTEG Formulator function

From two columns (vectors) *VX* and *VY*, each one containing *N* values, the `INTEG` function returns a third column (vector) containing a series of numerical integrals A_n, where `n` = 1, 2, ..., N-1, N.

Usage

`INTEG(VX, VY)`

VX	The name of any column (vector) listed under **Columns**
VY	The name of any column (vector) listed under **Columns**

Details

Each integral approximates the area under the parametric curve created by plotting the first `n` values in *VY* against the first `n` values in *VX*. For `n` = 1, A_n = 0. For all other values of `n`, each integral A_n corresponds to the following relationship:

$$A_n = \sum_{i=1}^{i=(n-1)} \frac{(X_{i+1} - X_i) \cdot (Y_{i+1} + Y_i)}{2}$$

For example, for the curve below, `INTEG` returns a column (vector) containing A_1 equal to 0 (zero area at the start of a curve, at X_1) and A_2, A_3, A_4, and A_5 equal to curve areas, as shown in the following graphics.

Figure 147: INTEG Formulator function

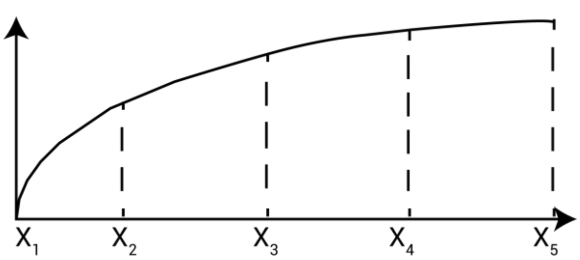


Figure 148: A5 curve area

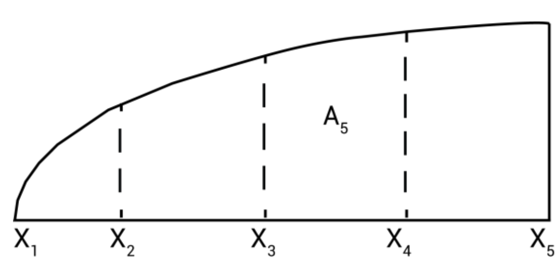


Figure 149: A2, A3, and A4 curve areas

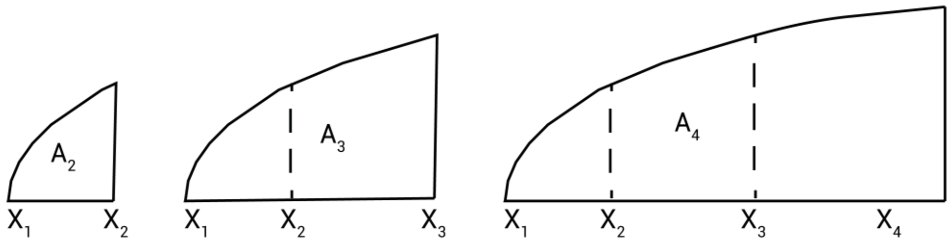
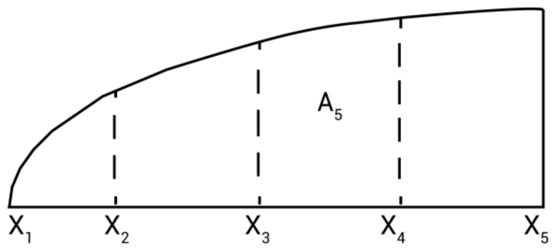


Figure 150: A5 curve area



You can use this function to do calculations in real time (while a test is executing).

Example

`QBD = INTEG(TIME, GATEI)`	

Also see

None

INDEX Formulator function

Returns a specified number of points starting with a specified value and consecutive values incremented by one.

Usage

`INDEX(START, N)`

START	The starting value
N	The number of points to be included

Example

`INDEX20 = INDEX(5, 20)`	Produces a new column labeled `INDEX20` that contains 20 values, starting with the value of 5 and ending with a value of 24.

Also see

None

LASTPOS Formulator function

Returns the row number of the last value in a column.

Usage

`LASTPOS(Value)`

Value	The name of any column (vector) listed under **Columns**

Example

```
NUMSWEEPPTS = LASTPOS(COLLECTORI)
```

Also see

FIRSTPOS (on page 8-28)

MAXPOS Formulator function

Searches all values in a column, finds the maximum value, and returns the row number of the maximum value.

Usage

`MAXPOS(V)`

V	The name of any column (vector) listed under Columns

Example

```
PEAKSTRESS = AT(GATEV, MAXPOS(SUBSTRATEI))
```

Also see

None

MINPOS Formulator function

Searches all values in a column, finds the minimum value, and returns the row number of the minimum value.

Usage

`MINPOS(V)`

V	The name of any column (vector) listed under Columns

Example

```
LOCATION = MINPOS(DRAINI)
```

Also see

None

SUBARRAY Formulator function

Returns a new column containing a specified range of the values from an existing column.

Usage

`SUBARRAY(V, STARTPOS, ENDPOS)`

V	The name of any column (vector) listed under Columns
STARTPOS	The row number of the existing value that you choose to become the first value in the new column (vector)
ENDPOS	The row number of the existing value that you choose to become the last value in the new column (vector)

Details

If *STARTPOS* and *ENDPOS* are invalid numbers, the function returns `#REF` as the result.

Example

`SUB1 = SUBARRAY(VEXIST, 10, 20)`	Given an existing column, VEXIST, containing values in rows 1 through 60, you can use `SUBARRAY` to return a new column, VNEW, containing only the values from rows 10 through 20 of VEXIST.

Also see

None

SUMMV Formulator function

Returns a column (vector) *VY* that consists of moving summation of a column (vector) V.

Usage

`SUMMV(V)`

V	The name of any column (vector) listed under Columns

Details

The n^{th} value in *VY* (Y_n) is the sum of the n^{th} and preceding values in *V*. This relationship may be expressed mathematically as follows:

$$Y_n = \sum_{i=1}^{i=n} X_i$$

Where X_i = the values in column (vector) V.

Example 1

```
F1 = SUMMV(BASEI)
```

Example 2

```
PSISPSIO = SUMMV((1-CQADJ/COX)*DELTA(VGS))*DOPETYPE
```

Example 3

The following example illustrates the SUMMV function numerically.

V	VY = SUMMV(V)
1.0000	1.0000
2.0000	3.0000
3.0000	6.0000
4.0000	10.0000
•	•
•	•
•	•

Also see

None

Line fits

The Line Fit Formulator functions allow you to set up distinct types of line fits.

EXPFIT Formulator function

Performs an exponential fit and returns a new column of Y values.

Usage

`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 of the starting values
ENDPOS	For the range of X and Y values to be exponentially fitted, the row number of the ending values

Details

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 this exponential relationship, returns a new column (vector) containing Y values calculated from all X values in column VX.

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is `#REF`), the function returns an invalid result.

Example

```
DIODEI = EXPFIT(ANODEV, ANODEI, 2, LASTPOS(ANODEV))
```

Also see

EXPFITA (on page 8-34)
EXPFITB (on page 8-35)

EXPFITA Formulator function

Returns the value of the constant `EXPFITA` as part of the formula to perform an exponential fit.

Usage

`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 of the starting values
ENDPOS	For the range of X and Y values to be exponentially fitted, the row number of the ending values

Details

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.

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is `#REF`), the function returns an invalid result.

Example

```
OFFSETI = EXPFITA(ANODEV, ANODEI, 2, LASTPOS(ANODEV))
```

Also see

EXPFIT (on page 8-33)
EXPFITB (on page 8-35)

EXPFITB Formulator function

Returns the value of the constant `EXPFITB` as part of the formula to perform an exponential fit.

Usage

`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 of the starting values
ENDPOS	For the range of X and Y values to be exponentially fitted, the row number of the ending values

Details

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.

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is `#REF`), the function returns an invalid result.

Example

```
DIODEIDEALITY = 1/(EXPFITB(ANODEV, ANODEI, 2, LASTPOS(ANODEV))*0.0257)
```

Also see

EXPFITA (on page 8-34)
EXPFITB (on page 8-35)

LINFIT Formulator function

Finds a linear equation and returns a new column.

Usage

`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 of the first set of `X` and `Y` values
ENDPOS	The row number of the second set of `X` and `Y` values

Details

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`.

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is `#REF`), the function returns an invalid result.

To return a linear regression fit for two columns (vectors), use the `REGFIT` function.

Example

```
RESISTORFIT = LINFIT (RESV, RESI, FIRSTPOS(RESV), LASTPOS(RESV))
```

Also see

REGFIT (on page 8-42)

LINFITSLP Formulator function

Finds a linear equation and returns the slope.

Usage

`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 of the first set of `X` and `Y` values
ENDPOS	The row number of the second set of `X` and `Y` values

Details

Finds a linear equation and returns the slope as follows:

- Finds a linear equation of the form `Y = a + bX` from two sets of `X` and `Y` values selected from two columns, *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`).

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is `#REF`), the function returns an invalid result.

To return the slope of a linear regression fit for two columns (vectors), use the `REGFITSLP` function.

Example

```
RESISTANCE = 1/LINFITSLP(RESV, RESI, FIRSTPOS(RESV), LASTPOS(RESV))
```

Also see

REGFITSLP (on page 8-43)

LINFITXINT Formulator function

Finds a linear equation and returns the X intercept.

Usage

`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 of the first set of X and Y values
ENDPOS	The row number of the second set of X and Y values

Details

Finds a linear equation and returns the X intercept as follows:

- 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`).

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is `#REF`), the function returns an invalid result.

To return the `X` intercept of a linear regression fit for two columns (vectors), use the `REGFITXINT` function.

Example

```
EARLYV = LINFITXINT(COLLECTORV, COLLECTORI, 56, 75)
```

Also see

REGFITXINT (on page 8-44)

LINFITYINT Formulator function

Finds a linear equation and returns the Y intercept.

Usage

`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 of the first set of X and Y values
ENDPOS	The row number of the second set of X and Y values

Details

Finds a linear equation and returns the Y intercept as follows:

- Finds a linear equation of the form `Y = a + bX` from two sets of `X` and `Y` values selected from two columns, *`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`).

If a *`VX`* or *`VY`* value at either *`STARTPOS`* or *`ENDPOS`* is an invalid number (that is, the value is `#REF`), the function returns an invalid result.

To return the `Y` intercept of a linear regression fit for two columns, use the `REGFITYINT` function.

Example

```
OFFSET = LINFITYINT(GATEV, GATEI, FIRSTPOS(GATEV), LASTPOS(GATEV))
```

Also see

REGFITYINT (on page 8-45)

LOGFIT Formulator function

Performs a base-10 log-linear fit.

Usage

`LOGFIT(`*`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 logarithmically fitted, the row number of the starting values
`ENDPOS`	For the range of *`X`* and *`Y`* values to be logarithmically fitted, the row number of the ending values

Details

Performs a base-10 log-linear fit as follows:

- Fits the following logarithmic relationship to a specified range of values in two columns (one column, *`VX`*, containing `X` values and the other column, *`VY`*, containing `Y` values):
  `Y = LOGFITA + LOGFITB * log(X)`

  where `LOGFITA` and `LOGFITB` are fit constants.
- Using the above logarithmic relationship, returns a new column containing `Y` values calculated from all `X` values in column *`VX`*.

If a *`VX`* or *`VY`* value at either *`STARTPOS`* or *`ENDPOS`* is an invalid number (that is, the value is `#REF`), the function returns an invalid result.

Example

```
GOODFIT = LOGFIT(GATEV, DRAINI, 30, 50)
```

Also see

LOGFITA (on page 8-39)
LOGFITB (on page 8-40)

LOGFITA Formulator function

Performs a base-10 log-linear fit.

Usage

`LOGFITA(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 logarithmically fitted, the row number of the starting values
ENDPOS	For the range of *X* and *Y* values to be logarithmically fitted, the row number of the ending values

Details

Performs a base-10 log-linear fit as follows:

- Fits the following logarithmic relationship to a specified range of values in two columns (one column, *VX*, containing X values and the other column, *VY*, containing Y values):
  `Y = LOGFITA + LOGFITB * log(X)`

  where LOGFITA and LOGFITB are fit constants.
- Using the above logarithmic relationship, returns the value of the constant LOGFITA.

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is #REF), the function returns an invalid result.

Example

```
OFFSET = LOGFITA(GATEV, DRAINI, 30, 50)
```

Also see

LOGFIT (on page 8-38)
LOGFITB (on page 8-40)

LOGFITB Formulator function

Performs a base-10 log-linear fit.

Usage

`LOGFITA(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 logarithmically fitted, the row number of the starting values
ENDPOS	For the range of *X* and *Y* values to be logarithmically fitted, the row number of the ending values

Details

Performs a base-10 log-linear fit as follows:

- Fits the following logarithmic relationship to a specified range of values in two columns (one column, *VX*, containing X values and the other column, *VY*, containing Y values):
  `Y = LOGFITA + LOGFITB * log(X)`

  where LOGFITA and LOGFITB are fit constants.
- Using the above logarithmic relationship, returns the value of the constant LOGFITB.

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is #REF), the function returns an invalid result.

Example

```
FACTOR = LOGFITB(GATEV, DRAINI, 30, 50)
```

Also see

LOGFIT (on page 8-38)
LOGFITA (on page 8-39)

POLY2FIT Formulator function

Enables quadratic regression line fitting.

Usage

`POLY2COEFF(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 of the first set of X and Y values
ENDPOS	The row number of the second set of X and Y values

Details

Enables quadratic regression line fitting. It allows a set of data to best fit an equation of the parabola $Y = aX^2 + bX + c$.

The a, b, and c values of the quadratic equation are returned.

The quadratic regression line fit functions are useful for deriving the defect density when you use the drive-level capacitance profiling (DLCP) technique.

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is #REF), the function returns an invalid result.

Also see

POLY2COEFF (on page 8-41)
POLYNFIT (on page 8-42)

POLY2COEFF Formulator function

Enables quadratic regression line fitting.

Usage

`POLY2COEFF(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 of the first set of X and Y values
ENDPOS	The row number of the second set of X and Y values

Details

Enables quadratic regression line fitting. It allows a set of data to best fit an equation of the parabola $Y = aX^2 + bX + c$.

The a, b, and c values of the quadratic equation are returned.

The quadratic regression line fit functions are useful for deriving the defect density when you use the drive-level capacitance profiling (DLCP) technique.

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is #REF), the function returns an invalid result.

Also see

POLY2FIT (on page 8-40)
POLYNFIT (on page 8-42)

POLYNFIT Formulator function

`POLYNFIT` (n^{th} order) does polynomial approximation from the 1st order to the 9th order.

Usage

`POLYNFIT(`*`VX`*`, `*`VY`*`, `*`ORDER`*`, `*`STARTPOS`*`, `*`ENDPOS`*`)`

VX	The name of any column (vector) listed under Columns
VY	The name of any column (vector) listed under Columns
ORDER	The order
STARTPOS	The row number of the first set of X and Y values
ENDPOS	The row number of the second set of X and Y values

Details

Enables quadratic regression line fitting. It allows a set of data to best fit an equation of the parabola $Y = aX^2 + bX + c$.

The `a`, `b`, and `c` values of the quadratic equation are returned.

The quadratic regression line fit functions are useful for deriving the defect density when you use the drive-level capacitance profiling (DLCP) technique.

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is `#REF`), the function returns an invalid result.

Also see

POLY2COEFF (on page 8-41)
POLY2FIT (on page 8-40)

REGFIT Formulator function

Performs a linear regression fit.

Usage

`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 of the starting values
ENDPOS	For the range of `X` and `Y` values to be fitted, the row number of the ending values

Details

Performs a linear regression fit as follows:

- Fits the following relationship, of the form `Y = a + bX`, to a specified range of values in two columns (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.

- Using the above linear relationship, returns a new column that contains Y values calculated from all X values in column *VX*.

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is #REF), the function returns an invalid result.

Example

```
COLLECTORFIT = REGFIT(COLLECTORV, COLLECTORI, 25, LASTPOS(COLLECTORV))
```

Also see

REGFITSLP (on page 8-43)
REGFITYINT (on page 8-45)

REGFITSLP Formulator function

Performs a linear regression fit.

Usage

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 of the starting values
ENDPOS	For the range of X and Y values to be fitted, the row number of the ending values

Details

Performs a linear regression fit as follows:

- Fits the following relationship, of the form `Y = a + bX`, to a specified range of values in two columns (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.

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is #REF), the function returns an invalid result.

Example

```
COLLECTORRES = 1/REGFITSLP(COLLECTORV, COLLECTORI, 25, LASTPOS(COLLECTORV))
```

Also see

REGFIT (on page 8-42)
REGFITYINT (on page 8-45)

REGFITXINT Formulator function

Performs a linear regression fit.

Usage

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 of the starting values
ENDPOS	For the range of X and Y values to be fitted, the row number of the ending values

Details

Performs a linear regression fit as follows:

- Fits the following relationship, of the form Y = a + bX, to a specified range of values in two columns (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 X intercept for relationship above.
  (–REGFITYINT/REGFITSLP).

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is #REF), the function returns an invalid result.

Example

```
EARLYV = REGFITXINT(COLLECTORV, COLLECTORI, 25, LASTPOS(COLLECTORV))
```

Also see

REGFIT (on page 8-42)
REGFITYINT (on page 8-45)

REGFITYINT Formulator function

Performs a linear regression fit.

Usage

`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 of the starting values
ENDPOS	For the range of X and Y values to be fitted, the row number of the ending values

Details

Performs a linear regression fit as follows:

- Fits the following relationship, of the form Y = a + bX, to a specified range of values in two columns (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 for relationship above (REGFITYINT).

If a *VX* or *VY* value at either *STARTPOS* or *ENDPOS* is an invalid number (that is, the value is #REF), the function returns an invalid result.

Example

```
OFFSET = REGFITYINT(COLLECTORV, COLLECTORI, 25, LASTPOS(COLLECTORV))
```

Also see

REGFIT (on page 8-42)
REGFITXINT (on page 8-44)

TANFIT Formulator function

Finds a linear equation of the form Y = a + bX from two columns, *VX* and *VY*.

Usage

`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 where the tangent is to be found

Details

Finds a linear equation of the form `Y = a + bX` from two columns, *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 containing `Y` values calculated from all `X` values in column *VX*.

If a *VX* or *VY* value at *POS* is an invalid number (that is, the value is `#REF`), the function returns an invalid result.

Example

```
VTFIT = TANFIT(GATEV, DRAINI, MAXPOS(GM))
```

Also see

TANFITSLP (on page 8-46)
TANFITXINT (on page 8-47)
TANFITYINT (on page 8-47)

TANFITSLP Formulator function

Finds a linear equation of the form `Y = a + bX` from two columns, *VX* and *VY*.

Usage

`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 where the tangent is to be found

Details

Finds a linear equation of the form `Y = a + bX` from two columns, *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 = a + bX`).

If a *VX* or *VY* value at *POS* is an invalid number (that is, the value is `#REF`), the function returns an invalid result.

Example

```
VTSLOPE = TANFITSLP(GATEV, DRAINI, MAXPOS(GM))
```

Also see

TANFIT (on page 8-45)
TANFITXINT (on page 8-47)
TANFITYINT (on page 8-47)

TANFITXINT Formulator function

Finds a linear equation of the form `Y = a + bX` from two columns, *VX* and *VY*.

Usage

`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 where the tangent is to be found

Details

Finds a linear equation of the form `Y = a + bX` from two columns, *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 = a + bX`).

If a *VX* or *VY* value at *POS* is an invalid number (that is, the value is `#REF`), the function returns an invalid result.

Example

```
VT = TANFITXINT(GATEV, DRAINI, MAXPOS(GM))
```

Also see

TANFIT (on page 8-45)
TANFITSLP (on page 8-46)
TANFITYINT (on page 8-47)

TANFITYINT Formulator function

Finds a linear equation of the form `Y = a + bX` from two columns, *VX* and *VY*.

Usage

`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 where the tangent is to be found

Details

Finds a linear equation of the form `Y = a + bX` from two columns, *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 = a + bX`).

If a *VX* or *VY* value at *POS* is an invalid number (that is, the value is `#REF`), the function returns an invalid result.

Example

```
OFFSET = TANFITYINT(GATEV, DRAINI, GMMAX)
```

Also see

TANFIT (on page 8-45)
TANFITSLP (on page 8-46)
TANFITXINT (on page 8-47)

FFT

The fast Fourier transform (FFT) formulas convert a signal or sequential group of measurements from the time domain to the frequency domain. They can also be used to convert from the frequency domain to the time domain.

For additional detail on using the FFT formulas, refer to the application note “Using the Built-In FFT Functions to Calculate the Spectral Density, 1/F Noise, AC Impedance and Other Parameters with the 4200A-SCS Parameter Analyzer” (1KW-73767).

FFT_R Formulator function

Performs an FFT on the provided input arrays and then returns the real parts.

Usage

`FFT_R(REAL, IMAG)`

REAL	The real portion of a complex number input array
IMAG	The imaginary portion of a complex number input array

Details

If *REAL* is set to `0` or any other constant, the real portion of all values for the input array are set to `0` or the constant.

If *IMAG* is set to `0` or any other constant, the imaginary portion of all values for the input array are set to `0` or the constant.

If either *REAL* or *IMAG* is not a power of 2, the input arrays are adjusted so that they are the same power of 2.

The output of an `FFT_R` formula is the real component of the calculated FFT, with an output size of a power of 2. The output is 0 Hz to −*Fs*/2, where *Fs* is the sampling frequency, such as the following:

```
... [0, 1, ..., n/2-1, -n/2, ..., -1] * Fs/n
```

If the input is invalid or the input size is less than 2, `#REF` is returned.

Also see

FFT_FREQ Formulator function (on page 8-50)
FFT_FREQ_P Formulator function (on page 8-51)
FFT_I Formulator function (on page 8-49)
IFFT_I Formulator function (on page 8-53)
IFFT_R Formulator function (on page 8-52)
SMOOTH Formulator function (on page 8-53)

FFT_I Formulator function

Performs an FFT on the provided input arrays and then returns the imaginary parts.

Usage

`FFT_I(REAL, IMAG)`

REAL	The real portion of a complex number input array
IMAG	The imaginary portion of a complex number input array

Details

If *REAL* is set to `0` or any other constant, the real portion of all values for the input array are set to `0` or the constant.

If *IMAG* is set to `0` or any other constant, the imaginary portion of all values for the input array are set to `0` or the constant.

If either *REAL* or *IMAG* is not a power of 2, the input arrays are adjusted so that they are the same power of 2. If possible, a message is displayed to indicate that the input arrays were adjusted.

The output of an `FFT_I` formula is the imaginary component of the calculated FFT, with an output size of a power of 2. The output is 0 Hz to −*Fs*/2, where *Fs* is the sampling frequency, such as the following:

```
... [0, 1, ..., n/2-1, -n/2, ..., -1] * Fs/n
```

If the input is invalid or the input size is less than 2, `#REF` is returned.

Also see

FFT_FREQ Formulator function (on page 8-50)
FFT_FREQ_P Formulator function (on page 8-51)
FFT_R Formulator function (on page 8-48)
IFFT_I Formulator function (on page 8-53)
IFFT_R Formulator function (on page 8-52)
SMOOTH Formulator function (on page 8-53)

FFT_FREQ Formulator function

Returns an array of positive and negative frequencies that correspond to the frequencies of an FFT output.

Usage

`FFT_FREQ(TIME, TOLERANCE)`

TIME	Input time array
TOLERANCE	The tolerance that is used to determine if the array is evenly spaced

Details

If *TIME* is not a power of 2, the input array is adjusted so that it is a power of 2. This is a single value, not an array.

The sampling period, or time step, is derived by computing the total period of the signal (after being adjusted if needed) and dividing by the number of samples. The samples are a power of 2.

The difference between two consecutive points is checked. If this difference is greater than the tolerance multiplied by the sampling period, an error is returned as `#REF`. This check happens for all points in the input array after the array has been adjusted to be a power of 2.

The output of an `FFT_FREQ` formula is 0 Hz to −*Fs*/2, where *Fs* is the sampling frequency and *n* is the size of the input array after being scaled to a power of 2, such as the following:

```
... [0, 1, ..., n/2-1, -n/2, ..., -1] * Fs/n
```

The output size is a power of 2.

`#REF` is returned if any one of the following are true:

- An input array delta between two points is greater than the tolerance value.
- The input size is less than 2.
- The average time step is less than or equal to zero.

Also see

FFT_FREQ_P Formulator function (on page 8-51)
FFT_I Formulator function (on page 8-49)
FFT_R Formulator function (on page 8-48)
IFFT_I Formulator function (on page 8-53)
IFFT_R Formulator function (on page 8-52)
SMOOTH Formulator function (on page 8-53)

FFT_FREQ_P Formulator function

This formula returns an array of the positive frequencies that correspond to the frequencies of an FFT output.

Usage

`FFT_FREQ_P(`*`TIME`*`, `*`TOLERANCE`*`)`

TIME	Input time array; adjusted to be a power of 2
TOLERANCE	The tolerance that is used to determine if the array is evenly spaced

Details

The output of this formula is a single value, not an array.

The sampling period, or time step, is derived by computing the total period of the signal, after being adjusted if needed, and dividing by the number of samples. The number of samples will be a power of 2.

The difference between two consecutive points is checked. If this difference is greater than the tolerance multiplied by the sampling period, an error is returned as `#REF`. This check happens for all points in the input array after the array has been adjusted to be a power of 2.

The output of an `FFT_FREQ_P` formula is 0 Hz to `(`*`n`*`/2-1) * `*`Fs`*`/`*`n`*, where *`Fs`* is the sampling frequency and *`n`* is the size of the input array after being scaled to a power of 2.

The output size is the input array size divided by two.

`#REF` is returned if any one of the following occurs:

- An input array delta between two points is greater than the tolerance value.
- The input size is less than 2.
- The average time step is less than or equal to zero.

Also see

FFT_FREQ (on page 8-50)
FFT_I Formulator function (on page 8-49)
FFT_R Formulator function (on page 8-48)
IFFT_I Formulator function (on page 8-53)
IFFT_R Formulator function (on page 8-52)
SMOOTH Formulator function (on page 8-53)

IFFT_R Formulator function

Performs an inverse FFT on the provided input arrays and then returns the real parts scaled by `1/`*`N`*, where *`N`* is the number of samples.

Usage

`IFFT_R(`*`REAL`*`, `*`IMAG`*`)`

REAL	The real portion of a complex number input array
IMAG	The imaginary portion of a complex number input array

Details

If *`REAL`* is set to `0` or any other constant, the real portion of all values for the input array are set to `0` or the constant.

If *`IMAG`* is set to `0` or any other constant, the imaginary portion of all values for the input array are set to `0` or the constant.

If either *`REAL`* or *`IMAG`* is not a power of 2, the input arrays are cut so that they are the same power of 2.

The output of an `IFFT_R` formula is the real component of the calculated inverse FFT after being scaled by `1/`*`N`*. The output size is a power of 2.

The output is 0 Hz to −*`Fs`*/2, where *`Fs`* is the sampling frequency, such as the following:

```
... [0, 1, ..., n/2-1, -n/2, ..., -1] * Fs/n
```

If the input is invalid or the input size is less than 2, `#REF` is returned.

Also see

FFT_FREQ Formulator function (on page 8-50)
FFT_FREQ_P Formulator function (on page 8-51)
FFT_I Formulator function (on page 8-49)
FFT_R Formulator function (on page 8-48)
IFFT_I Formulator function (on page 8-53)
SMOOTH Formulator function (on page 8-53)

IFFT_I Formulator function

Performs an inverse FFT on the provided input arrays and then returns the imaginary parts scaled by `1/N`, where *N* is the number of samples.

Usage

`IFFT_I(REAL, IMAG)`

REAL	The real portion of a complex number input array
IMAG	The imaginary portion of a complex number input array

Details

If *REAL* is set to `0` or any other constant, the real portion of all values for the input array are set to `0` or the constant.

If *IMAG* is set to `0` or any other constant, the imaginary portion of all values for the input array are set to `0` or the constant.

If either *REAL* or *IMAG* is not a power of 2, the input arrays are cut so that they are the same power of 2. If possible, a message is displayed to indicate that the input arrays were adjusted.

The output of an `IFFT_I` formula is the imaginary component of the calculated FFT after being scaled by `1/N`. The output size is a power of 2. The output is 0 Hz to −*Fs*/2, where *Fs* is the sampling frequency, such as the following:

```
... [0, 1, ..., n/2-1, -n/2, ..., -1] * Fs/n
```

If the input is invalid or the input size is less than 2, `#REF` is returned.

Also see

FFT_FREQ Formulator function (on page 8-50)
FFT_FREQ_P Formulator function (on page 8-51)
FFT_I Formulator function (on page 8-49)
FFT_R Formulator function (on page 8-48)
IFFT_R Formulator function (on page 8-52)
SMOOTH Formulator function (on page 8-53)

SMOOTH Formulator function

Performs digital filtering on an input array by zeroing out high frequency components.

Usage

`SMOOTH(X, PERCENT)`

X	The input data array; does not need to be a power of 2
PERCENT	The percent of high frequencies to smooth (0 to 100); this is a single value (not an array)

Details

The `SMOOTH` formula takes the FFT of the input signal, then uses the returned FFT array to zero out the frequency bin based on the percentage value. For example, if *PERCENT* is set to 25%, the top 25% frequency bins are zeroed. Note that the highest frequency bins are at the center of the array. After these bins are zeroed, this array is fed into the inverse FFT to return the value back to the time domain. The 0 Hz bin, which is the first item in the array, is never zeroed.

If *PERCENT* is set to 0%, no frequencies are zeroed and the original signal is returned. If *PERCENT* is set to 100%, all but the 0 Hz frequency are zeroed (a straight line is produced).

Also see

FFT_FREQ Formulator function (on page 8-50)
FFT_FREQ_P Formulator function (on page 8-51)
FFT_I Formulator function (on page 8-49)
FFT_R Formulator function (on page 8-48)
IFFT_I Formulator function (on page 8-53)
IFFT_R Formulator function (on page 8-52)

Misc

The Misc Formulator function allows you to compare user-defined expressions.

COND Formulator function

Returns one of two user-defined expressions, depending on the comparison of two other user-defined expressions.

Usage

`COND(EXP1, EXP2, EXP3, EXP4)`

EXP1, EXP2, EXP3, EXP4	Mathematical expressions created using valid Formulator functions, operators, and operands

Details

Returns one of two user-defined expressions (*EXP3* or *EXP4*), depending on the comparison of two other user-defined expressions (*EXP1* and *EXP2*).

If *EXP1* < *EXP2*, then *EXP3* is returned.

If *EXP1* ≥ *EXP2*, then *EXP4* is returned.

Example

```
CLIPCURRENT = COND(DRAINI, 1E-6, DRAINI, 1E-6)
```

Also see

None

Section 9

Site and subsite operation

In this section:

Introduction

This chapter describes how to configure sites and subsites. It also describes how to set up subsite cycling for stress and measure cycles.

For additional information on setting up prober movement between project sites and subsites, refer to *Model 4200A-SCS Prober and External Instrument Control*.

Sites

A site includes all of the subsites, devices, and tests in the project. If you set up multiple sites, all sites are identical. They will each have the same type and number of subsites and the sites are repeated across the wafer.

To add a site to the project tree:

1. Choose **Select**.
2. Select the **Wafer Plan** library.
3. Select **Site**.
4. Select **Add**. The site is added to the project tree.

Subsites

A subsite is a collection of devices and their associated tests. You can work with devices and tests as you do in a project that does not include a subsite.

You need to use actions to initiate prober movement between subsites and close matrix channels between devices.

NOTE

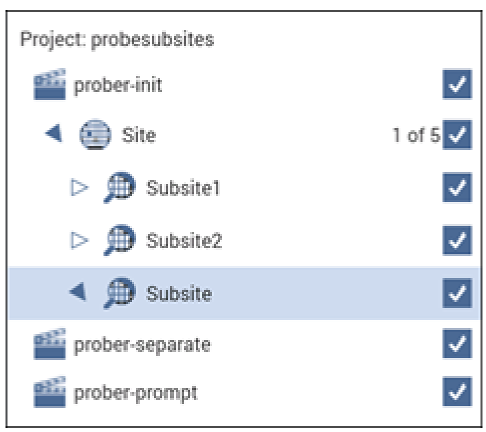
You can add subsites as shown below. If you are using a prober, you can also use `probesubsites` from the Project Library to start with a site and subsite template with prober actions.

To add a subsite:

1. Open a project or create a new one.
2. Choose **Select**.
3. From the Wafer Plan tab, drag **Subsite** to the project tree.
4. If needed, select **Rename**, enter a new name, and press **Enter**.

An example of a project tree with several subsites and prober actions are shown in the following figure.

Figure 151: Project tree with subsites example



Configure sites

For sites, you can set the following options:

- **Number of Sites:** The maximum number of sites that can be tested. This is typically set to the number of sites that have been programmed in a prober controller.
- **Start Execution at Site:** The site where project execution starts. This is normally the same as the prober starting site number.
- **Finish Execution at Site:** The site where project execution ends. This must be less than or equal to the number of sites.

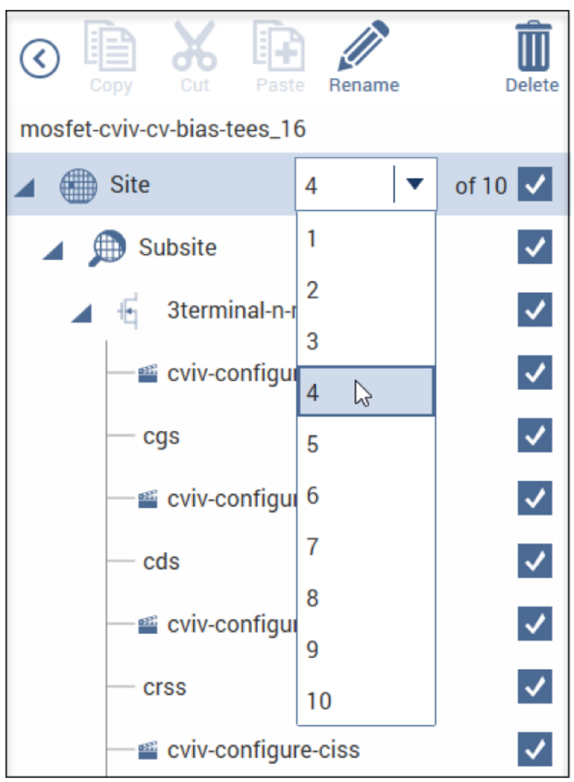
NOTE

If you use a semi-automatic prober, a Clarius probe action only triggers movements that are already programmed in the prober controller. Each execution of the action advances the prober to the next site in this programmed sequence. Site numbers are not communicated between the prober and Clarius. Therefore, if you evaluate multiple sites, the range of site numbers that you specify in the Clarius Project window must agree with the sequence of site numbers in the prober controller program.

To configure sites:

1. Select the site in the project tree.
2. Select **Configure**.
3. Set the **Number of Sites**. If you are configuring your sites for Segment Stress mode, the maximum number of sites is 999.
4. Select **Save**.
5. If there is more than one site, you can select the site where testing should start executing and the site where testing should stop executing.
6. If there is more than one site, you can display the project tree for a site. The site that is displayed is shown next to the site name in the project tree. For example, if the present site is set to 4 in a project with 10 sites, "4 of 10" is displayed next to the site name.

Figure 152: Selecting a site



The locations of sites to be visited are typically defined by the software of the prober. However, the commands that initiate prober movement are defined by one or more prober-movement actions.

Configure subsite cycling

You can use the 4200A-SCS to stress test DUTs using subsite cycling. A Clarius evaluation consists of pre-stress tests at a subsite, followed by alternate cycles of stressing and retesting. During the evaluation, Clarius can display intermediate numerical and graphical results and status information. Clarius ends the evaluation when the devices degrade beyond specified exit criteria (degradation targets) or when the total stressing time reaches a specified maximum, whichever comes first.

Subsite cycling allows you to cycle through the subsite tests up to 4096 times. Clarius can perform hot-carrier injection (HCI) tests, negative bias temperature instability (NBTI) tests, and similar wafer-level reliability (WLR) tests. The built-in software for stress testing is integrated with subsite cycling.

Data and graphs of the subsite cycles are available in Analyze for the subsite.

The measured readings listed in the Analyze sheet are output values. You define the output values to be included in the subsite Analyze sheet during test setup. See Define output values for Analyze (on page 9-36) for details.

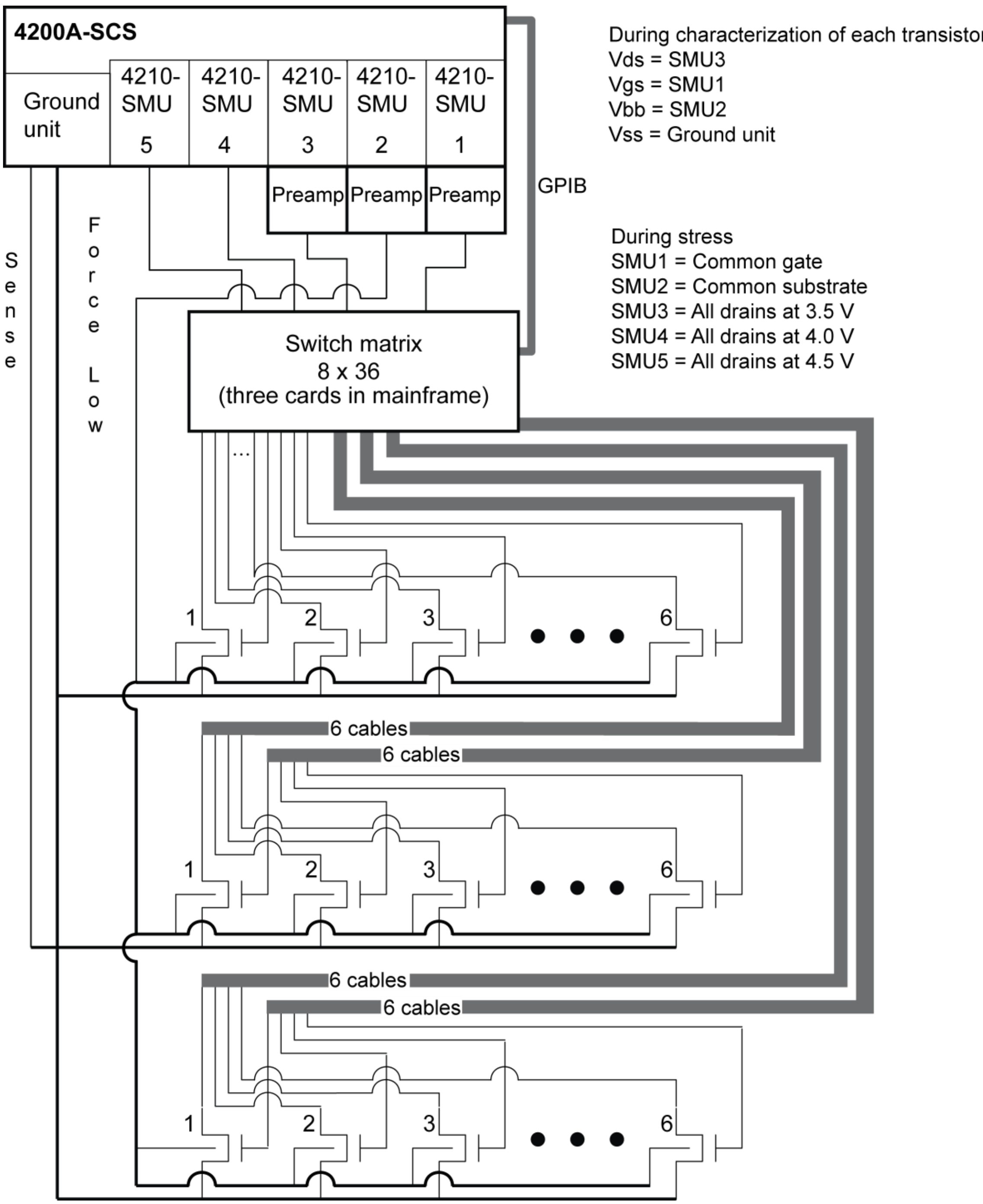
Stress mode integrates stressing with subsite cycling for testing. The first cycle is stress-free. For each subsequent cycle, the devices in the subsite are stressed with voltage or current for a specified period. After the stress period expires, the tests in the subsite are run. Device stressing includes DC voltage stress, DC current stress, AC voltage stress, and segment stressing. The DC stress is applied by one or more SMUs. Devices can be stressed individually, or they can be parallel-connected so that a single SMU can stress multiple devices. The SMUs can also be used to measure the DC stress. The AC stress is applied by pulse cards. Each pulse card has two pulse output channels, each of which can stress one device terminal.

Segment stress is similar to the standard Stress mode, but is done using Segment Arb™ pulse mode for stressing. Stress is provided by a Segment Arb waveform generated by a pulse card. Each channel of the pulse card can stress one device terminal. The DC bias voltage and current limit for the device can be provided by the SMUs in the system.

Connect devices for stress/measure cycling

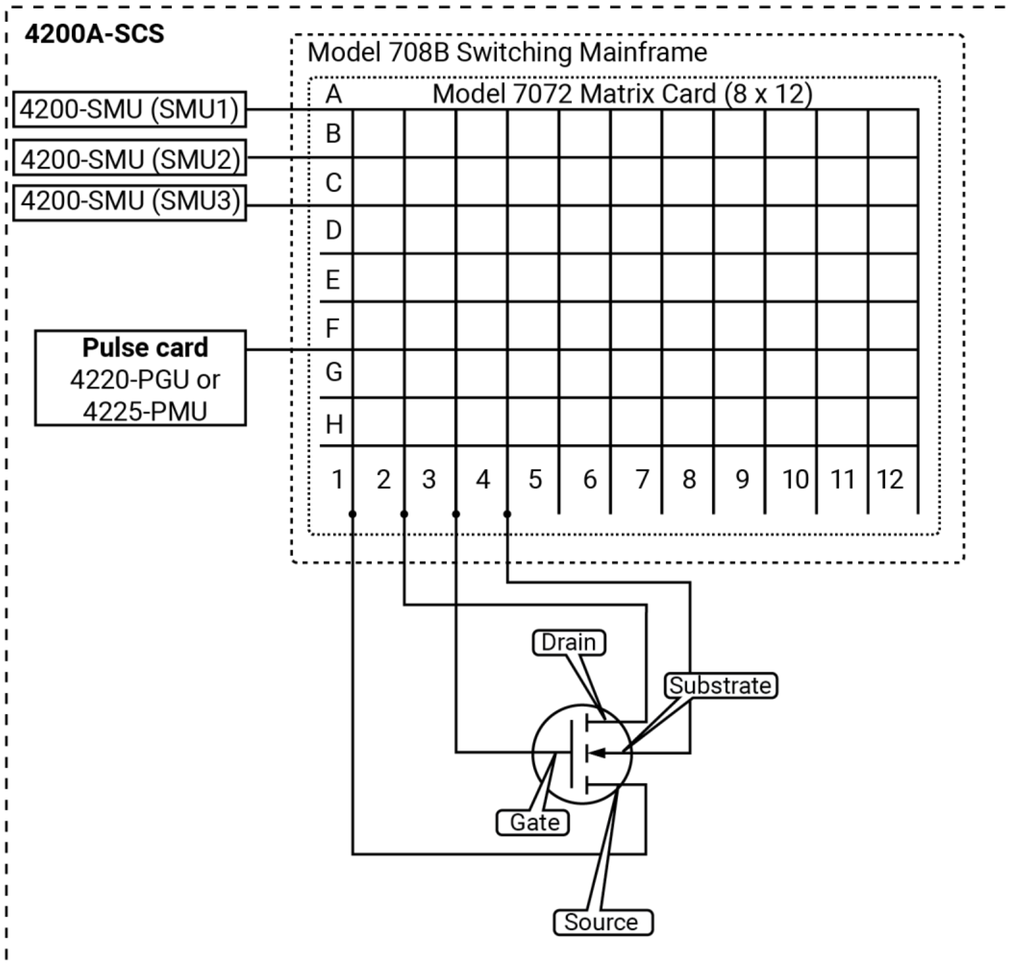
Devices that are stress/measure cycled in parallel are connected through a switching system. The following figure shows an example of connections for an HCI evaluation.

Figure 153: Stress/measure wiring example



Connections for matrix card

Figure 154: AC Pulse stress-measure — hardware matrix card simplified schematic



Connections for pulse card to device under test

Connect the pulse generator to the DUT during stress as shown in the following figures.

Figure 155: AC pulse stress-measure — hardware setup block diagram for stress

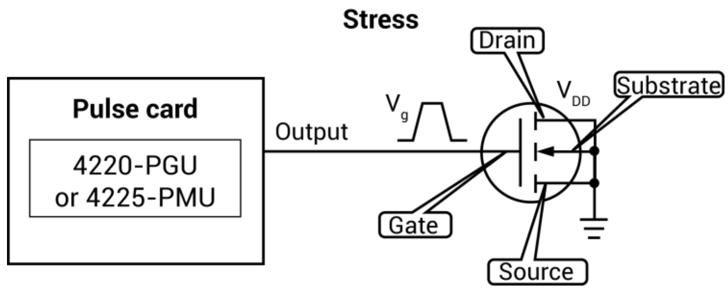
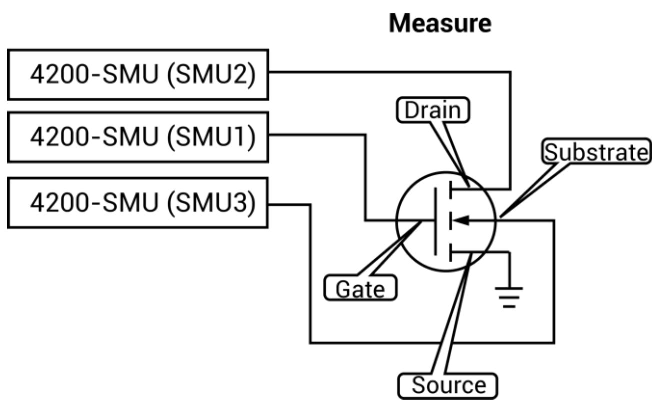


Figure 156: AC Pulse stress-measure — hardware setup block diagram for measure

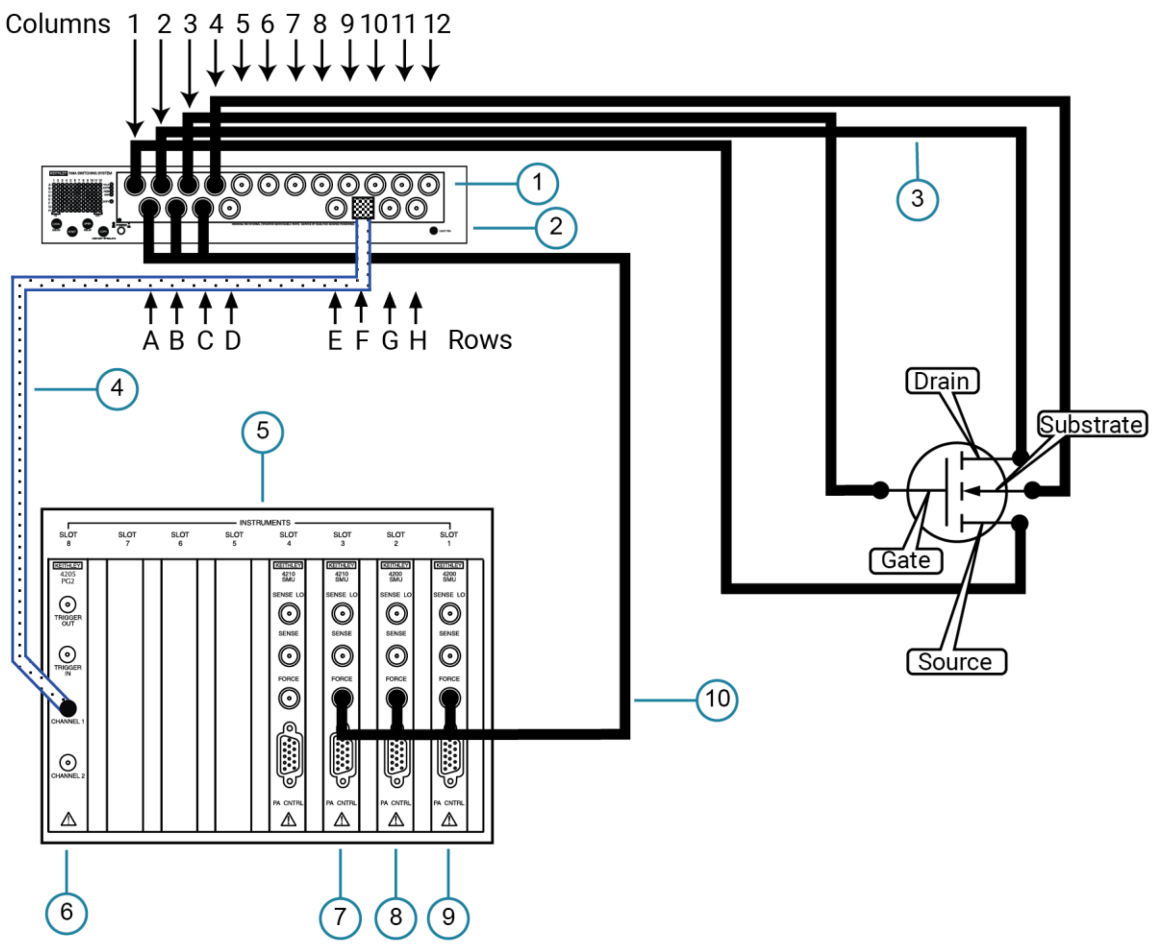


Connections for system hardware

NOTE

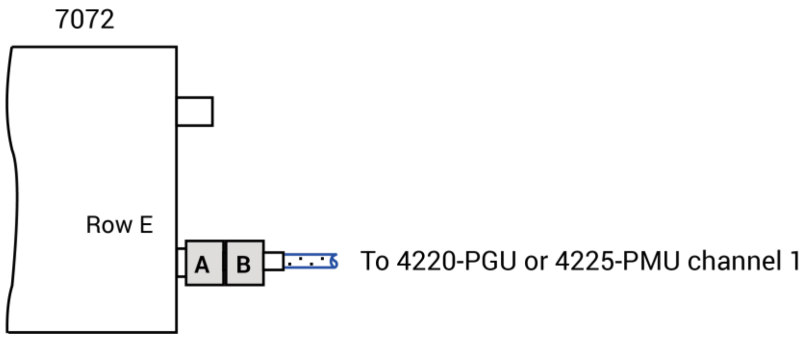
Use a torque wrench to tighten SMA connections to 8 in-lb.

Figure 157: AC Pulse stress-measure — hardware connections



1	Model 7072
2	Model 708B
3	Black BNC cable plug to plug (1 of 4)
4	White SMA cable plug to plug, 2 m (6 ft)
5	4200A-SCS instrument slots
6	4220-PGU or 4225-PMU
7	SMU3 (4200-SMU or 4210-SMU)
8	SMU2
9	SMU1
10	4200-MTRX-X cables, 2 m (6 ft), three

Figure 158: Model 7072 side view to show connector adapters



A	BNC plug to 3-lug triaxial plug adapter
B	SMA socket to BNC socket adapter

Set up the Subsite Operation

You can select one of the following Subsite Operations:

- **None:** No cycling or stressing operation is performed on the subsite.
- **Cycle:** Loops through the subsite tests without stressing the devices.
- **Stress:** Test-stress-test-stress cycles, such as hot-carrier injection (HCI) or negative bias temperature instability (NBTI) studies. You can use SMUs to provide bias voltage and current limit for the devices, but you cannot use them to measure stress. This operation can also include segment stress using the Segment Arb™ pulse mode.

When you run a subsite, subsite data is recorded in Analyze. You can display data in the graph for up to 128 runs. You can specify the number of runs that are displayed by defining the Max Displayed Runs value. When the number of displayed runs is reached, the oldest data is removed and the latest data is displayed. All data is available through the Run History.

To configure multiple subsites with the same settings, configure the first subsite, then select **Copy** in the project tree.

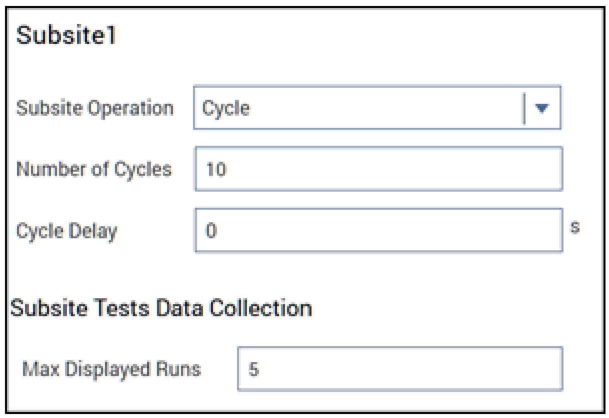
Set up Cycle operation

In the Cycle operation, the subsite test is repeated a specified number of times. There are only measure cycles, with no stressing. For each individual test in the subsite, data is acquired for each subsite cycle. For example, if the subsite is cycled five times, there are five sets of data and graphs for each test. You can execute up to 4096 cycles.

To set up the Cycle operation:

1. From the project tree, select the subsite.
2. Select **Configure**.
3. In the Subsite pane, select **Cycle** from the **Subsite Operation** menu.
4. Enter the **Number of Cycles**. This is the fixed number of times that you want the subsite to execute.
5. Enter the **Cycle Delay** in seconds.
6. Set the **Max Displayed Runs** to the number of Run Histories to display in the graph.
7. The setup is complete.

Figure 159: Stress Mode Setup pane



Set up Stress operation

If your project is set up to run on more than one subsite, you need to set the stress properties for each subsite separately. This allows you to have different levels of stress on each subsite. After you configure the first site, repeat the steps for the next subsite.

To set up Stress Subsite operation:

1. From the project tree, select the subsite.
2. Select **Configure**.
3. Select **Stress** from the **Subsite Operation** menu.
4. See the remaining topics in this section to configure the operation.

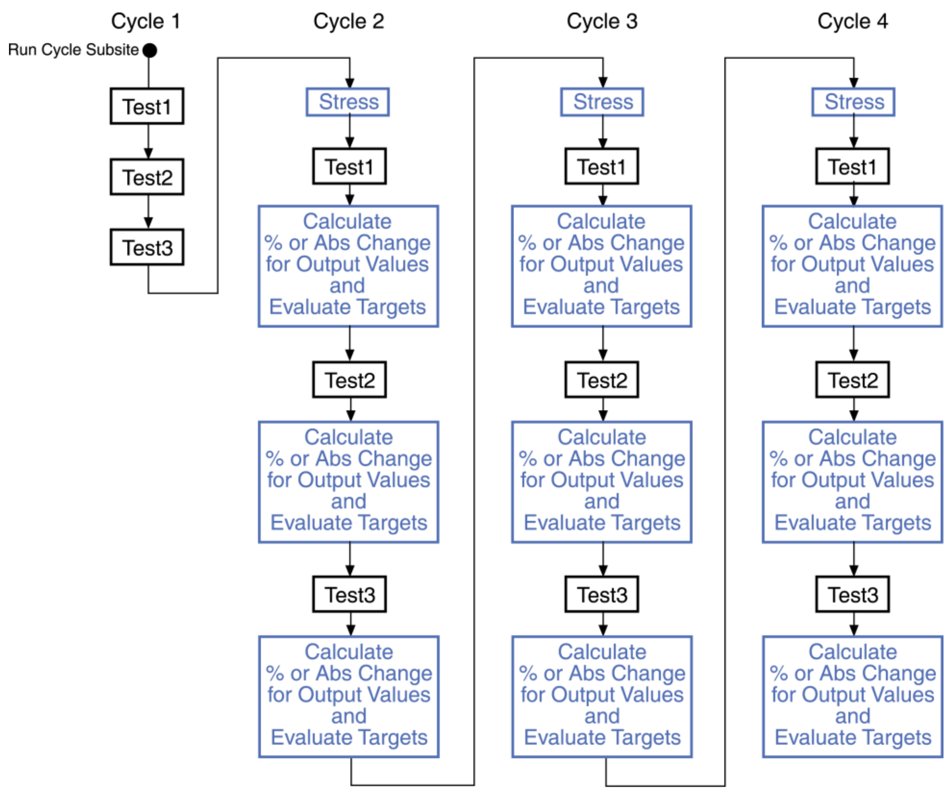
About the Stress operation

In the Stress mode, the subsite test is repeated a specified number of times. For each individual test in the subsite, data is acquired for each subsite cycle. For example, if the subsite is cycled five times, there are five sets of data and graphs for each test. You can execute up to 4096 cycles.

The test sequence includes components for stressing, percent change, and target evaluation. The following figure shows an example of a basic testing sequence. The components for stressing, percent change, and target evaluation are shown in blue.

When subsite cycling is started, the first pass through the subsite is a pre-stress cycle. Tests are run with no stressing. At the start of the next cycle, the configured stress (voltage or current) is applied to all devices. After the stress period expires, the stress is removed and enabled tests are run. Each additional stress cycle operates in the same manner. That is, the stress is applied for the specified stress time, then all the enabled tests are run. Notice that after each test is run, the percent absolute (Abs) change and targets are evaluated.

Figure 160: Example of the stress testing sequence (four cycles) for a single device



NOTE

The following information explains stress testing using the Stress mode. Stressing is provided by SMUs or Keithley pulse cards or both (using the standard pulse mode for AC stressing).

Stressing can also be provided by Keithley pulse cards using the Segment Arb pulse mode. Refer to Segment stressing (on page 9-17) for supplemental information on using Segment Arb for stress testing.

For stress testing, a Clarius evaluation consists of pre-stress tests at a particular subsite, followed by alternate cycles of stressing and retesting. Clarius performs these cycles automatically when you select Stress mode. During the evaluation, Clarius can display intermediate numerical and graphical results and status information. Clarius ends the evaluation when the devices degrade beyond specified exit criteria (target degradation) or when the total stressing time reaches a specified maximum, whichever comes first.

Combined stressing and testing

The following steps summarize an HCI evaluation for the stressing configurations shown in DC voltage stressing (on page 9-15) and AC voltage stressing (on page 9-16). Similar operations apply to other types of stress-measure studies.

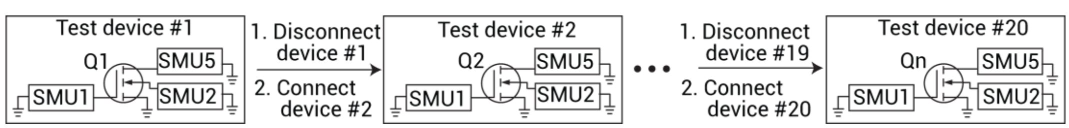
NOTE

For information about AC stress for wafer-level reliability, refer to Wafer-level reliability testing (on page 11-1).

Summary of an HCI evaluation:

1. Use the switching matrix to automatically connect the SMUs to device 1.
2. Run pre-stress parametric tests on each device individually in device-number sequence, as shown in the following figure.

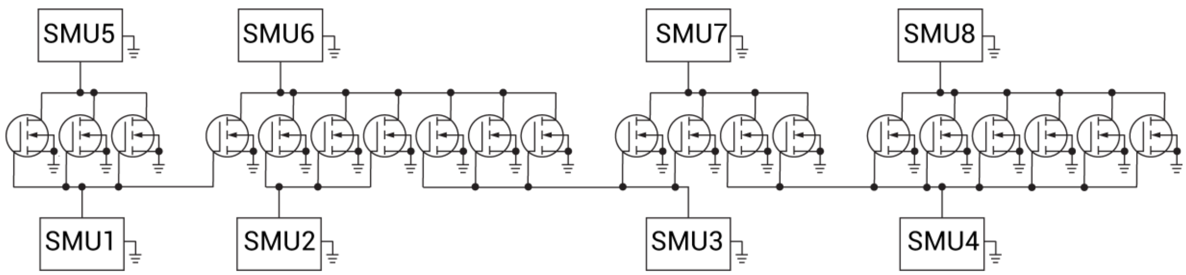
Figure 161: Pre-stress parametric tests



3. Disconnect all devices.
4. Use the switching matrix to automatically connect all devices to SMUs, as determined by the drain and gate voltages that were specified for each device.

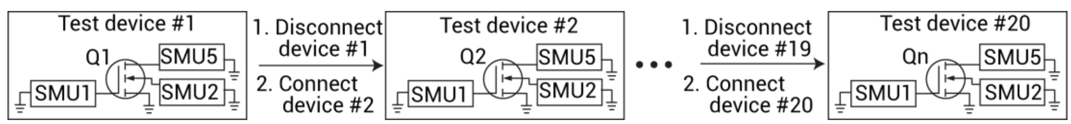
5. Run stress cycle 1, which stresses all the devices simultaneously, as shown in the following figure.

Figure 162: Stress all devices simultaneously



6. Disconnect all devices.
7. Use the switching matrix to automatically connect the SMUs to device 1.
8. Wait for a 10 s delay to promote uniform pre-test decay for all devices.
9. Run test cycle 1, running post-stress parametric tests on each device individually in device-number sequence, as shown in the following figure.

Figure 163: Post-stress parametric tests



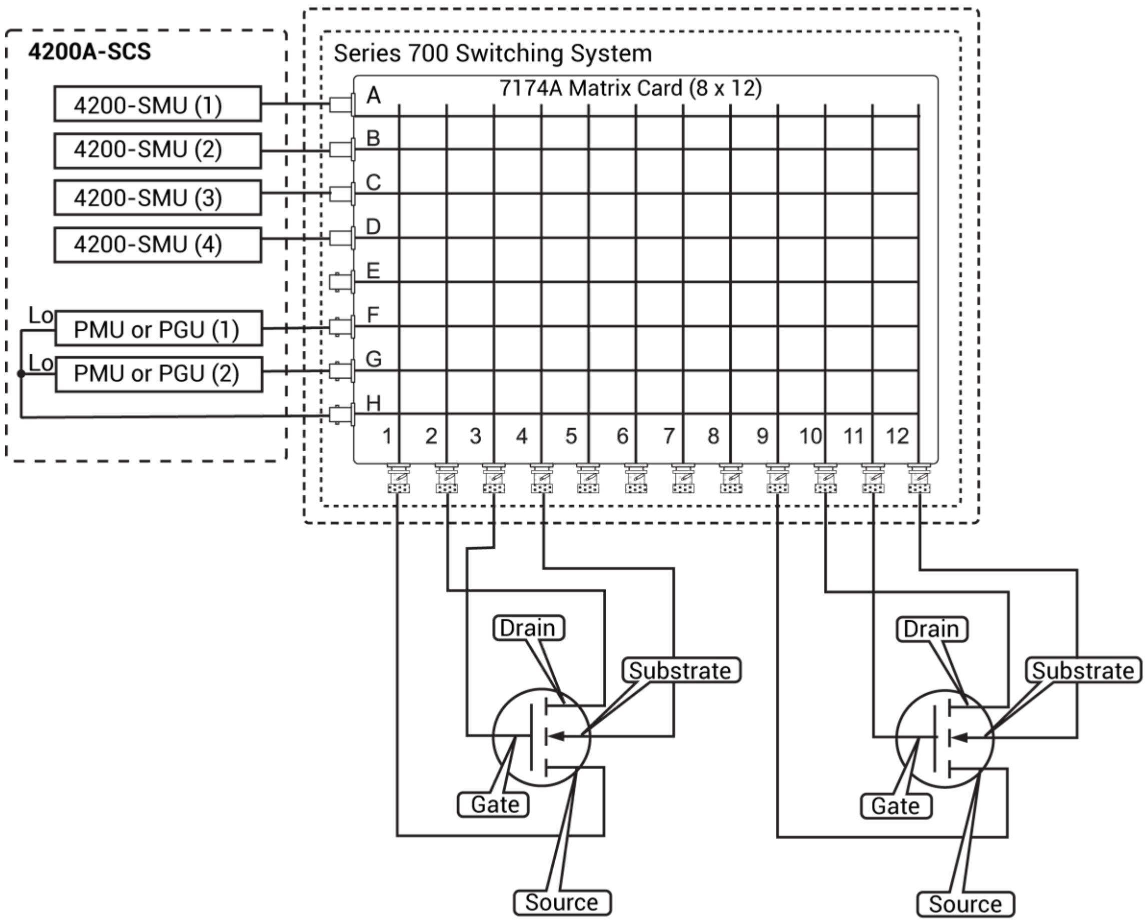
10. Disconnect all devices.
11. Reconnect all devices for stress cycle 2.
12. Run stress cycle 2.
13. Disconnect all devices.
14. Reconnect all devices for test cycle 2.
15. Wait for a 10 s to promote uniform pre-test decay for all devices.
16. Run test cycle 2.
17. Continue with stress and tests cycles (3, 4, … *n*) until a device degrades to all enabled target values or goes into compliance.
18. Stop testing this device but continue stress and test cycles until another device degrades to all target values or goes into compliance.
19. Stop testing this second device, but continue stress and test cycles until one of the following occurs:
    - All devices have either degraded to target values or have gone into compliance.
    - Total stress time reaches a user-specified value.

AC and DC voltage stress/measure system with a switching matrix

A switching matrix is supported for a pulse card AC voltage stress/measure system. In a system with a switching matrix, Clarius automatically assigns the SMUs, PGUs, and PMUs to the DUT terminals. For SMUs, Clarius attempts to use the same instrument to connect to multiple terminals. To connect to multiple terminals, the compliance current on both terminals must be less than the SMU current.

The following figure shows the use of a switching matrix for an AC and DC voltage stress/measure system. The recommended matrices for this system configuration are the Series 700 Switching Systems. To effectively transmit the higher frequency components of the typical pulse (Segment Arb or Standard), use a high bandwidth switching matrix card, such as the Keithley 7174A or 7173-50.

Figure 164: Switching matrix for AC and DC voltage stress/measure system



Select a Subsite Mode

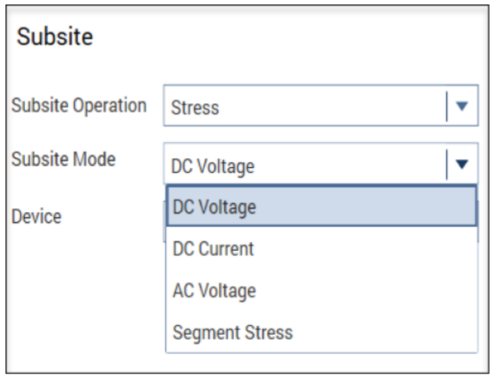
You can select four stress operation subsite modes. The modes are:

- DC Voltage
- DC Current
- AC Voltage
- Segment Stress

To select a Subsite Mode:

1. From the project tree, select the subsite.
2. Select **Configure**.
3. From the **Subsite Operation** menu select **Stress**.
4. Select a **Subsite Mode**.

Figure 165: Selecting a subsite mode



See the following topics in this section for a description of the subsite modes.

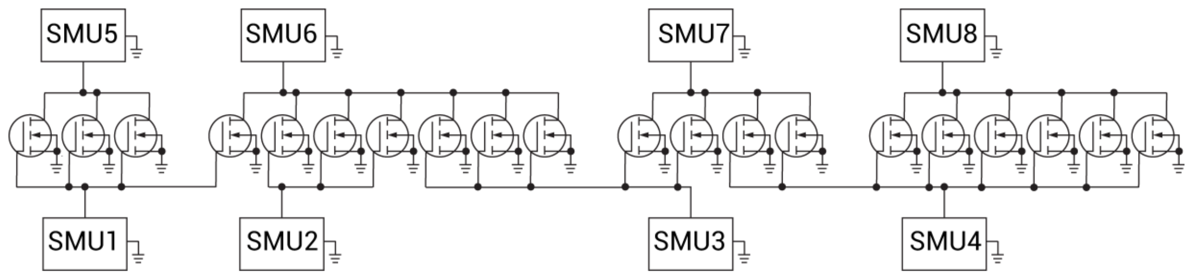
DC Voltage stressing

The stress algorithm built into Clarius uses SMUs to DC voltage stress multiple devices concurrently. The following capabilities apply to device stressing during hot-carrier injection (HCI) studies. Similar capabilities apply to other types of voltage stress-measure studies.

- A unique gate-stress bias voltage (Vg Stress) and a unique drain-stress bias voltage (Vd Stress) can be applied to each evaluated device, within the source limitations of the system. Each unique gate or drain stress bias condition requires a dedicated source. For example, if your 4200A-SCS system contains four SMUs, you can apply a maximum of four unique stress bias voltages (gate voltages plus drain voltages combined). If your 4200A-SCS system has eight SMUs, four medium power and four high power, you can apply a maximum of eight unique stress bias voltages.
- When some of the devices are connected in parallel, the program can voltage stress up to twenty devices at once, subject to system resource and matrix limitations. The following figure illustrates a voltage stressing configuration that uses the maximum software and system capabilities.
- If your voltage stress system is using a switching matrix, the 4200A-SCS tries to maximize the amount of SMU sharing in order to allow parallel testing. It determines which pins can share SMUs in the following fashion: If pins from different devices have the same name (for example, gate pin, drain pin) and the like-named pins are assigned the same voltage stress, then when the stress is applied, these pins are automatically connected to the same SMU through the switching matrix. That SMU supplies the voltage stress to all the pins simultaneously.

- Because parallel-connected devices share resources, Clarius monitors stressing resources when Stress Properties are configured. If the requirements exceed the resources, Clarius reports an error.

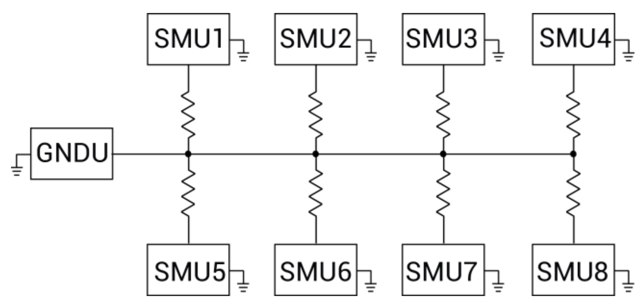
Figure 166: DC voltage stressing: 20 parallel-connected devices stressed at eight gate and drain voltages



DC Current stressing

For current stressing, the maximum number of devices depend on the number of SMUs in the system. Each SMU can current stress one device. For a system with eight SMUs, up to eight devices can be current stressed, as shown in the following figure.

Figure 167: EM test: Eight devices being current stressed by eight SMUs

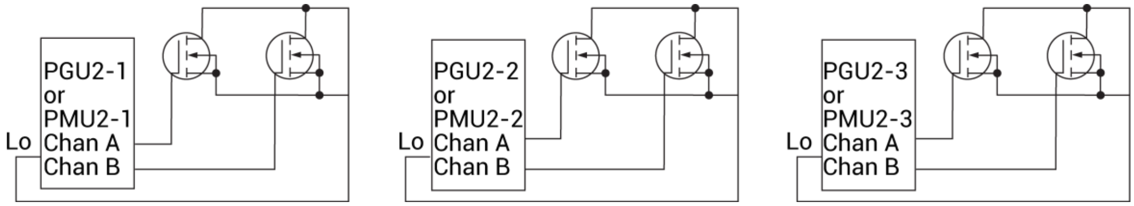


AC Voltage stressing

You can use four Keithley pulse cards to AC voltage stress eight devices (one device pin for each pulse channel). The following figure shows a Keithley pulse card providing AC voltage stress for six devices.

Parallel-connected devices cannot be AC voltage stressed using the pulse card. As shown in the figure, each pulse card channel can only stress one pin of one device.

Figure 168: AC Voltage stressing: Six devices stressed at the gates with six pulse outputs



Segment stressing

Segment stress testing consists of two phases:

- During a measure phase, the system makes measurements on the DUT.
- During a stress phase, the Keithley pulse card provides stress using Segment Arb waveforms, and the SMUs provide voltage bias and current limit. There are no measurements made during the stress phase.

For Segment Arb stressing, the waveform period is the fundamental unit of time for stressing. In the Stress Settings pane, the stress counts specify the number of times the Segment Arb waveform stresses the device. For example, assume the stress count is three, and the waveform period is four seconds. For that stress cycle, the Segment Arb waveform will stress the device three times for a total stress time of twelve seconds.

In a typical stress/measure test system that uses a switching matrix to automate the stress and measure phases of the test:

- During a measure phase, the switching matrix connects the instruments that will make the measurements on the DUT. The Keithley pulse card is disconnected from the DUT during a measure phase.
- During a stress phase, the switching matrix connects the pulse generator to the DUT. It also connects SMUs that will be used for device pin grounding or biasing.

NOTE

If your system contains 4225-RPMs, you cannot use SMUs during segment stresses. You must disconnect all RPMs from the 4200A-SCS and update the RPM configuration in KCon to enable DC biasing during subsite segment stress.

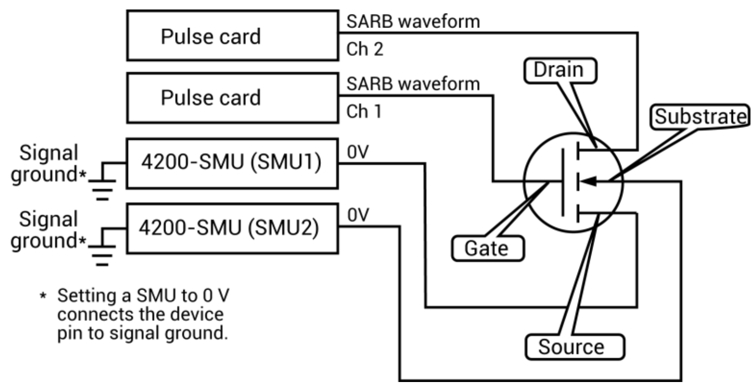
NOTE

To effectively transmit the higher frequency components of the typical pulse, a high-bandwidth switching matrix card should be used (for example, Keithley 7174A).

The stress phase example in the following figure shows an example of how a DUT can be stressed using Segment Arb waveforms. During a stress phase, the matrix connects the channels of the Keithley pulse card to the drain and gate of the DUT. The pulse generator stresses the drain and gate by outputting Segment Arb waveforms.

Two 4200-SMUs or 4201-SMUs (SMU1 and SMU2) are connected to the substrate and source terminals of the DUT. They are set to 0 V to effectively ground the terminals.

Figure 169: Segment stressing: Stress phase example



Select and configure a Device

From the Subsite pane, you can configure the devices in your subsite. You can assign the device terminals to an instrument or function and set the stress conditions.

Depending on the Subsite Mode and the instruments you have connected to your 4200A-SCS, you can assign the terminals of the device you have selected to one of the following types of instruments or functions:

- GNDU
- SMU
- PGU
- PMU
- PIN x (for test systems with a switching matrix card)
- NONE (not connected)

NOTE

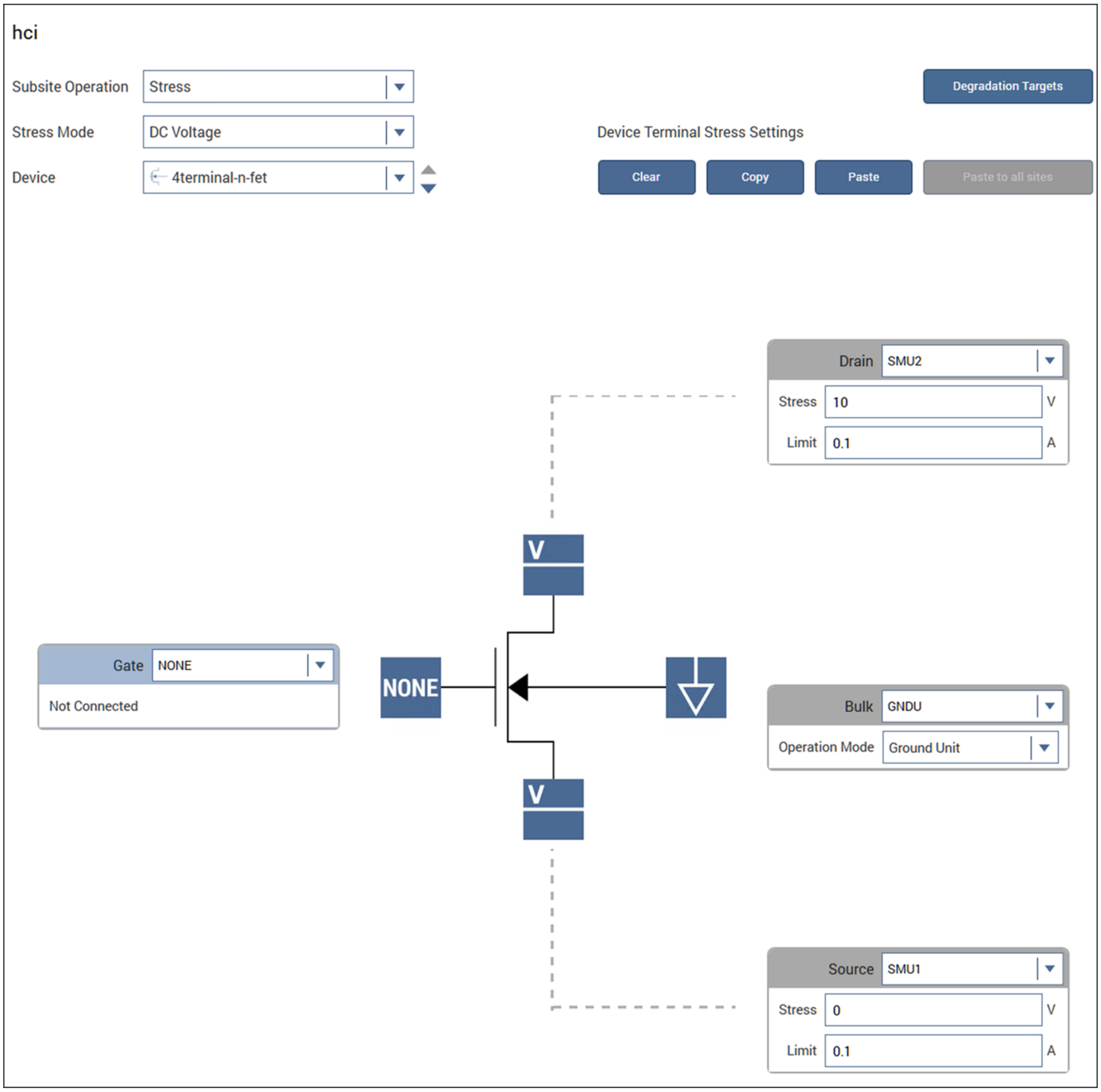
If a device terminal is set to GNDU, its Operation Mode is always Ground Unit.

To select and configure a device:

1. Select **Configure**.
2. Select a subsite.

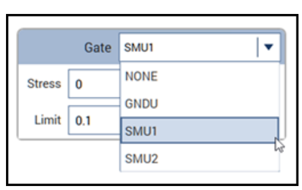
3. From the Device list, choose a device to configure for your subsite. A diagram of the device, its terminals, and its terminal connections is displayed, as shown in the following figure.

Figure 170: Device with terminals assigned to instruments



4. Assign an instrument or function to each terminal of the device, as needed.

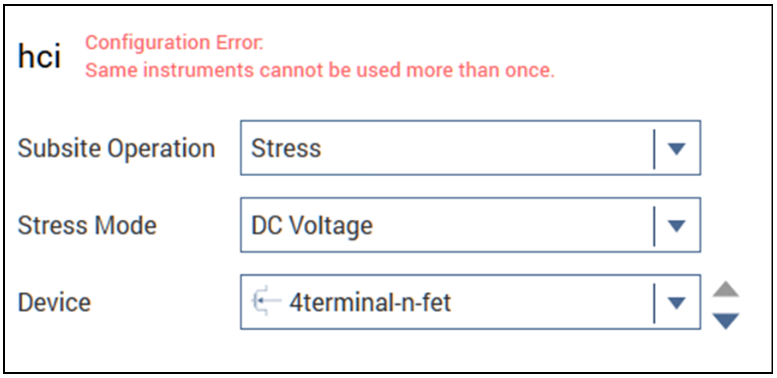
Figure 171: Selecting an instrument or function for the terminal



NOTE

You can only assign one device terminal to an instrument or function at a time. For example, you cannot assign the anode and cathode of a diode to SMU1. If you try to assign multiple terminals, the following error is displayed at the top of the Subsite pane.

Figure 172: Warning for one instrument or function connected to multiple device terminals



5. Specify the device terminal stress settings. The available settings depend on the subsite mode and the instrument or function assigned to the terminal. For more information, see Device terminal options for subsite modes (on page 9-20).

NOTE

For additional device configuration options, see Configure the Terminal Settings (on page 9-29).

Device terminal options for subsite modes

You see different terminal stress settings depending on the subsite mode and the instrument or function you assigned to a terminal. NONE and GNDU are always available regardless of the subsite mode.

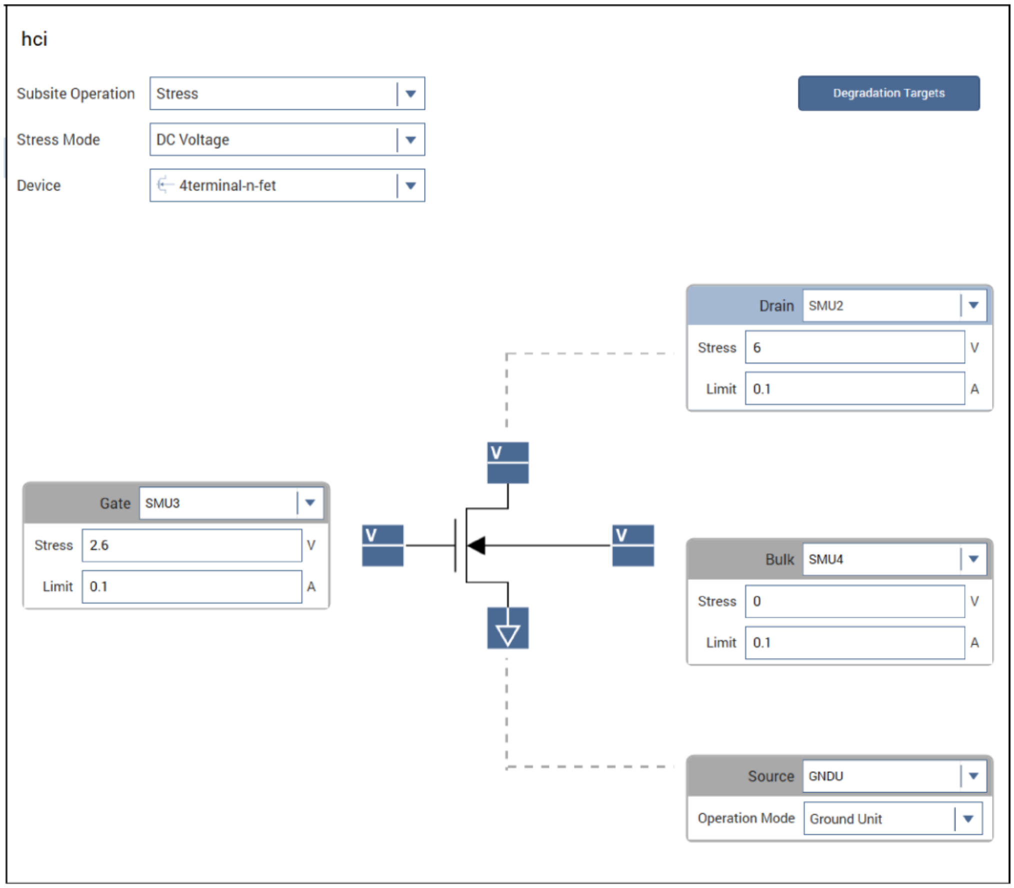
For example, if you select DC Voltage as a subsite mode and assign SMU1 to the Gate of a device terminal, you can adjust the stress in volts and limit in amperes. However, if you select the DC Current subsite mode, you specify the stress in amperes and limit in volts.

The following are device terminal fields you may see:

- **Operation Mode:** Applies to GNDU only. This field cannot be changed.
- **Stress:** The terminal voltage or current stress.
- **Duty Cycle:** The time, as a percentage of the pulse period, that the pulse is on (pulse width).
- **Instrument:** Only available when your test system includes a switching matrix.

The following figure shows an example of DC voltage stress operation using SMUs.

Figure 173: DC voltage stressing using SMUs



Device pin connections for matrix cards

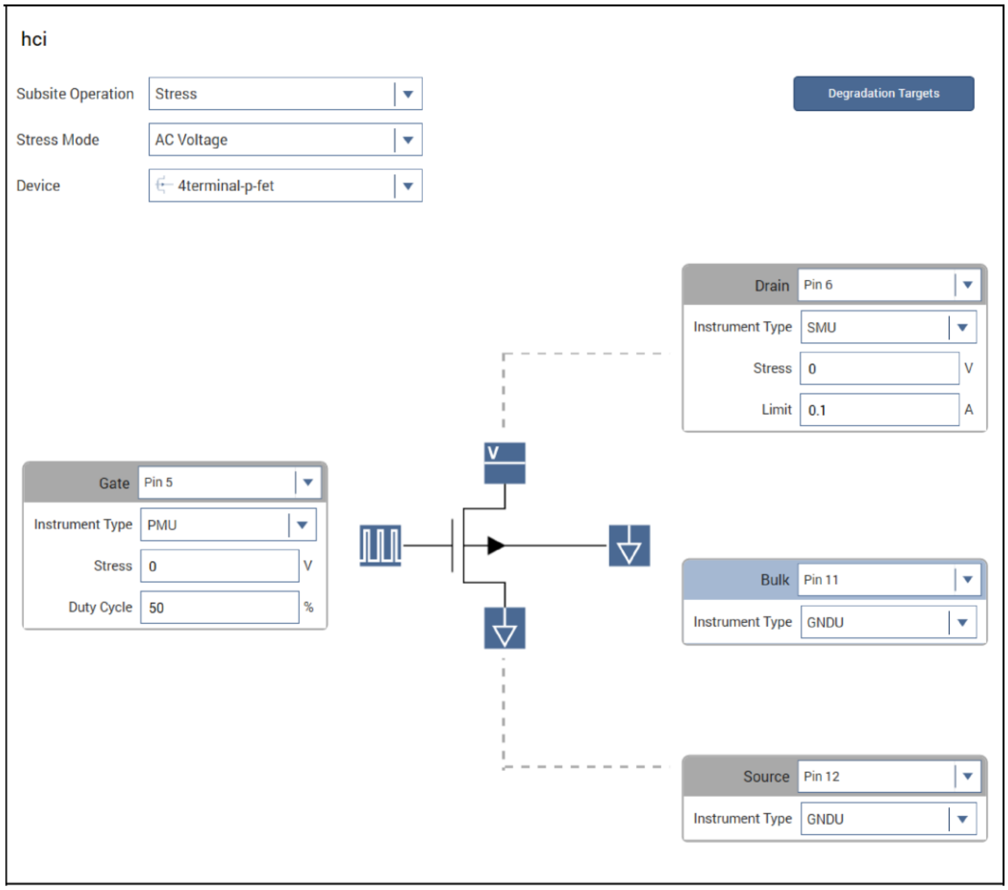
If you have a switching matrix connected to your test system, you will select the pin connection numbers and instrument type when you configure the DUT. The pin number assignments for the device must match the physical connections to the matrix card.

You can also specify the instrument for each terminal:

- **SMU** - This option is available if at least one SMU is connected to the switching matrix.
- **VPU** - Available when at least one PMU or PGU is connected to the matrix and the Stress Mode is AC Voltage or Segment Stress.
- **GNDU** - Available if the 4200A-SCS GNDU is connected to the matrix.
- **NONE** - No instrument is connected to the device terminal.

The following figure shows a subsite device terminal configuration in Clarius with a switching matrix connected to the test system.

Figure 174: AC stress operation using a PMU connected through a switching matrix



Copy device terminal settings to multiple sites

You can copy the device terminal stress settings to other sites for the DC Voltage, DC Current, and AC Voltage stress modes.

NOTE

If you use Paste to All Sites for a very large number of sites, it may take up to two minutes for the next save operation of the project to complete.

To copy device terminal settings to other sites:

1. Select **Configure**.
2. Select a subsite.
3. From the Device list in the center pane, select the device.
4. Set the terminal settings as needed.
5. From Device Terminal Stress Settings, select **Copy**.
6. From the Device list, select the device to which to copy the settings.
7. To copy the settings to the selected device:
   - In this site: Select **Paste**.
   - In all sites: Select **Paste to all sites**.

NOTE

After pasting, check pin assignments for each of the devices to verify that the changes are appropriate. To reset the device terminals to have no instrument selected and no connections, select **Clear**.

Import KPulse Segment Arb waveform files

If you exported a Segment Arb (SARB) waveform file from KPulse, you can import it when in Segment Stress subsite mode. You must assign a terminal of a device to use a PMU before you can select a SARB waveform file.

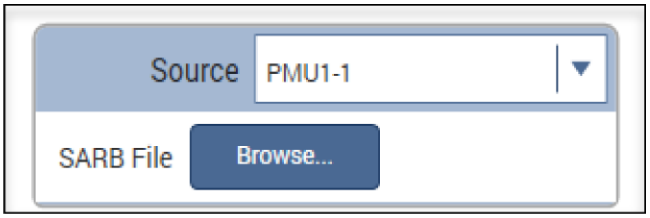
SARB waveform files have the extension `.ksf` and are normally stored at the following location:

`C:\s4200\kiuser\KPulse\SarbFiles`

To import a SARB file:

1. In the Subsite pane, select a device terminal.
2. Select a PMU.
3. Select **Browse**.

Figure 175: Selecting a SARB file



4. Select a file.
5. Select **Open**.

Configure the Stress Settings

To configure the Stress Settings for the subsite, select the Stress Settings pane. The Subsite Mode Stress Settings can include values for any of the following:

- Stress timing and counts
- Stress delays
- The terminal power on and power off sequence
- Pulse times
- The maximum number of displayed runs

The following topics describe the Stress Settings you may see when configuring your subsite.

Measurement Timing

When you select the Stress operation, you can configure the measurement timing for the Subsite Mode stress cycles. You can select:

- **Linear:** After the first stress cycle, all stress times or counts are identical.
- **Log:** After the first stress cycle, all stress times or counts increase logarithmically.
- **List:** You set the stress cycle times or counts.

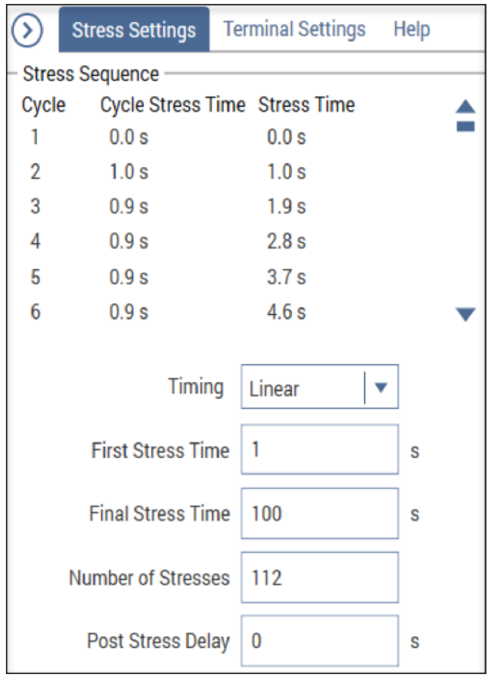
Set up Linear and Log timing modes for DC Voltage, DC Current, and AC Voltage stress modes

To set up the Linear and Log timing modes:

1. Select the Stress Settings pane.
2. From the Timing menu, select **Linear** or **Log**.
3. In **First Stress Time**, set the time in seconds that devices are stressed during the first cycle.
4. In **Final Stress Time**, set the time in seconds that devices are stressed during the last cycle.
5. Select a total number of stresses:
   - If you have selected **Linear** timing, set the **Number of Stresses.** This is the total number of stresses, up to 4095.
   - If you have selected **Log** timing, set the **# Stresses/Decade**. You can select a maximum of 10 per decade. There can be up to 4096 stresses for all decades combined.
6. If needed, enter the **Post Stress Delay** in seconds. This is the delay after each stress cycle. It allows the device to reach equilibrium before the next measurement

Clarius uses the values you entered to calculate the cumulative stress times for the cycles. The times are displayed in a table at the top of the Stress Settings pane.

Figure 176: Set timing for linear or log measure mode



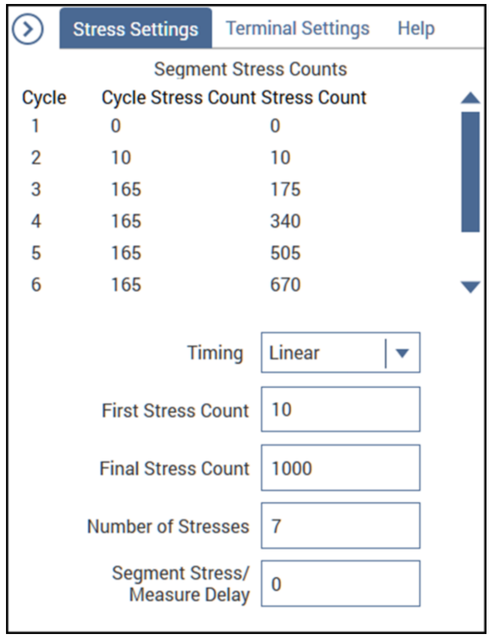
Set up Linear and Log timing modes for Segment Stress mode

To set up the Linear and Log timing modes for the Segment Stress Mode:

1. Select the Stress Settings pane.
2. From the Timing menu, select **Linear** or **Log**.
3. In **First Stress Count**, set the first stress count.
4. In **Final Stress Count**, set the final stress count.
5. Set the number of stresses:
   - If you have selected **Linear** timing, set the **Number of Stresses.** This is the total number of stresses, up to 4095.
   - If you have selected **Log** timing, set the **# Stresses/Decade**. You can select a maximum of 10 per decade.
6. If needed, enter the **Segment Stress/Measure Delay** in seconds. This allows the device to settle after stressing before performing the DC measurements.

Clarius uses the values you entered to calculate the stress counts for the linear or log cycles. The counts are displayed at the top of the Stress Settings pane, as shown in the following figure.

Figure 177: Set timing to linear for the Segment Stress Mode



Set up the List timing mode

When you select List, information that was entered if Linear or Log timing was selected is shown. You can use those settings as a starting point to set up your list. Clarius uses the List values to calculate the cumulative stress times or count for the cycles. The times are displayed in seconds.

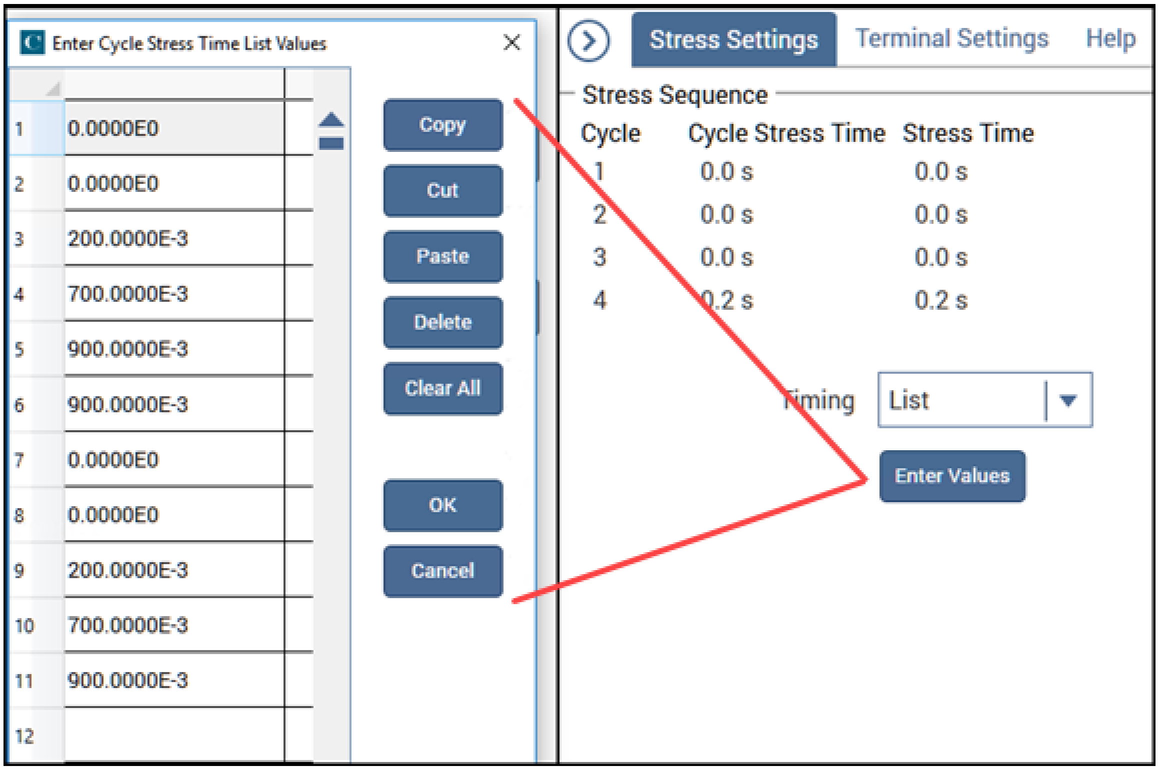
When entering values, you can copy and paste a list of values from another application, such as Microsoft Excel. Each value must be on a separate line. You can also use Shift+Click or Ctrl+Click to select multiple items in the list to copy and paste.

The first value in the Stress Sequence list is always 0 and cannot be changed.

To set up the List timing mode:

1. Select the **Stress Settings** pane.
2. From Timing, Select **List**.
3. Select **Enter Values**. The Enter Cycle Stress Time List Values dialog is displayed.

Figure 178: Set timing for list measure mode



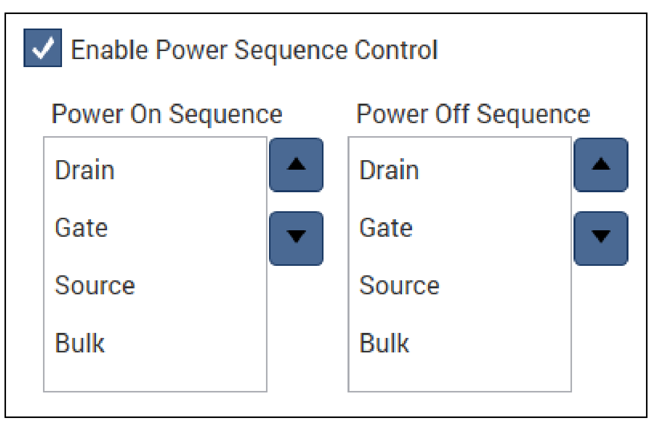
4. Enter the **Stress Time** (in seconds) or the **Stress Count** values.
5. Select **OK**.

Power On and Off Sequence

When Subsite Operation is set to Stress, you can set the sequence that the instruments follow for powering on and powering off your test device.

To set the sequence, select **Enable Power Sequence Control**.

Figure 179: Adjusting the power on and power off sequence for a four-terminal device



To change the sequence, select a terminal and use the arrows to the right of the box to move it up or down in the sequence.

NOTE

The devices must be connected to match the order selected.

Set the maximum displayed runs

When you run a subsite, subsite data is recorded in Analyze. You can display data in the graph for up to 128 runs. You can limit the number of runs that are displayed by defining the Max Displayed Runs value. When the number of displayed runs is reached, the oldest data is removed and the latest data is displayed. All data is available through the Run History.

To set up the displayed runs:

1. Select the **Stress Settings** tab.
2. Set the **Max Displayed Runs** to the number of Run Histories to display in the graph.

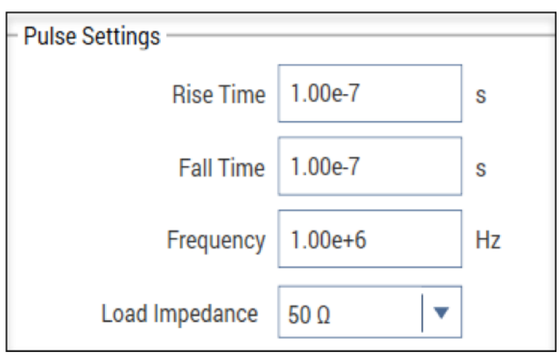
Pulse Settings

NOTE

Pulse Settings are only available when the Subsite Mode is AC Voltage and a PMU or PGU has been assigned to a terminal of a device in the Subsite pane.

To access Pulse Settings in AC Voltage Subsite Mode, select the **Stress Settings** pane. You can adjust the rise time, fall time, frequency, and the impedance of the load.

Figure 180: Pulse Settings dialog



Configure the Terminal Settings

The Terminal Settings pane lets you further configure the terminals of a devices. You can adjust the stress conditions specified when you selected a device, and, depending on the subsite mode and the instrument or function assigned to the terminals, perform the following:

- Turn the measurement on and off
- Specify the measurement range
- Specify the stress (voltage) low

NOTE

In the Segment Stress subsite mode, you can only upload a SARB file. See Import KPulse Segment Arb waveform files (on page 9-23) for more information.

See the following graphics for examples.

Figure 181: Terminal settings for a diode assigned to a PMU in DC Voltage subsite mode

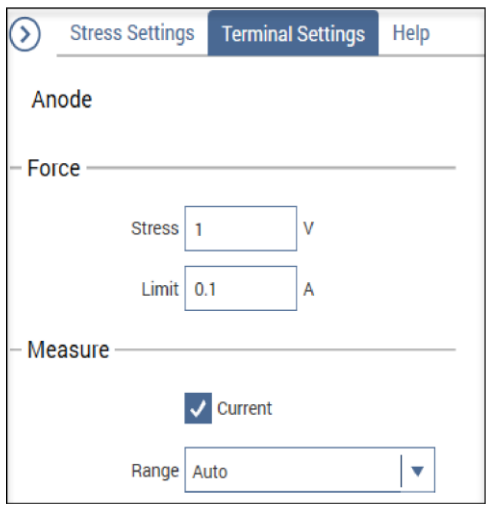
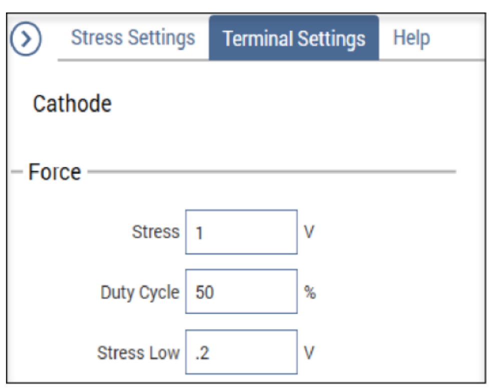


Figure 182: Terminal settings for a diode assigned to a PMU in AC Voltage subsite mode

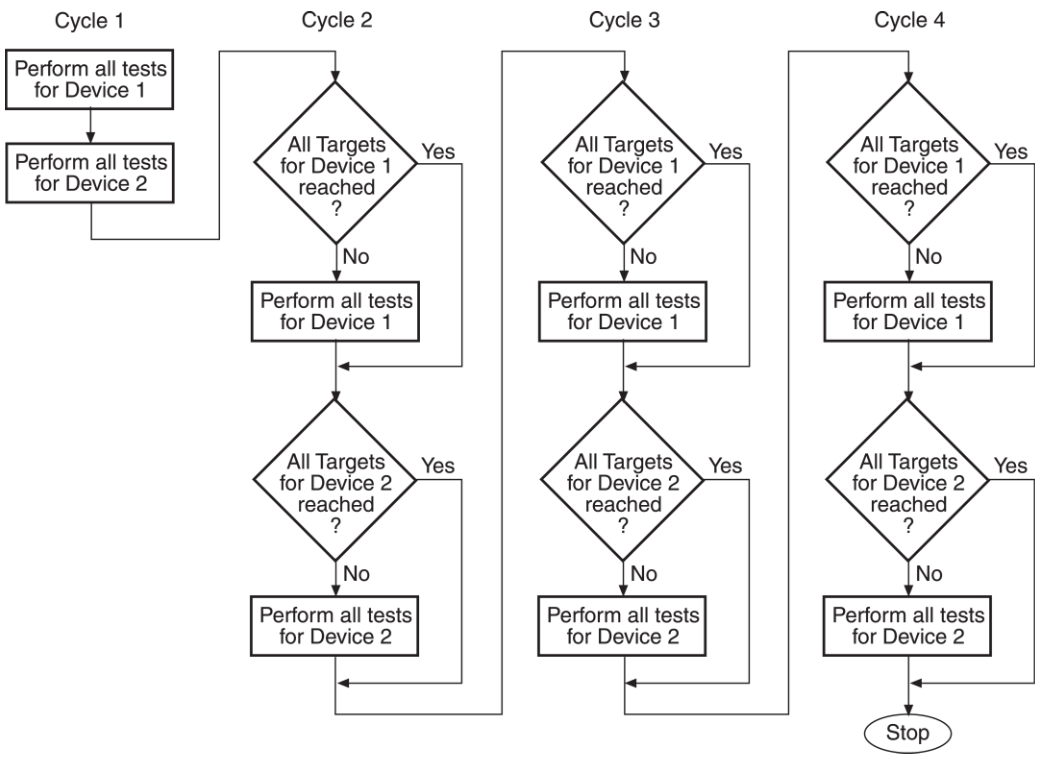


Degradation targets

You can enable an output value as a target and assign it a target value as a percent of change or absolute value. When all targets for a device are reached, that device is no longer tested. Subsequent cycles bypass the device tests that reached all their targets. The subsite stops when a target on each device is reached or the last subsite cycle is completed.

The testing process for target evaluation is shown in the following flowchart. As a simple example, assume all the targets for both devices are reached after the first cycle of the subsite test. Following the flowchart shows that the tests for cycles 2, 3 and 4 are not performed. The subsite test stops. The graph that is plotted is degradation versus stress time.

Figure 183: Target evaluation process (example for two devices, four cycles)

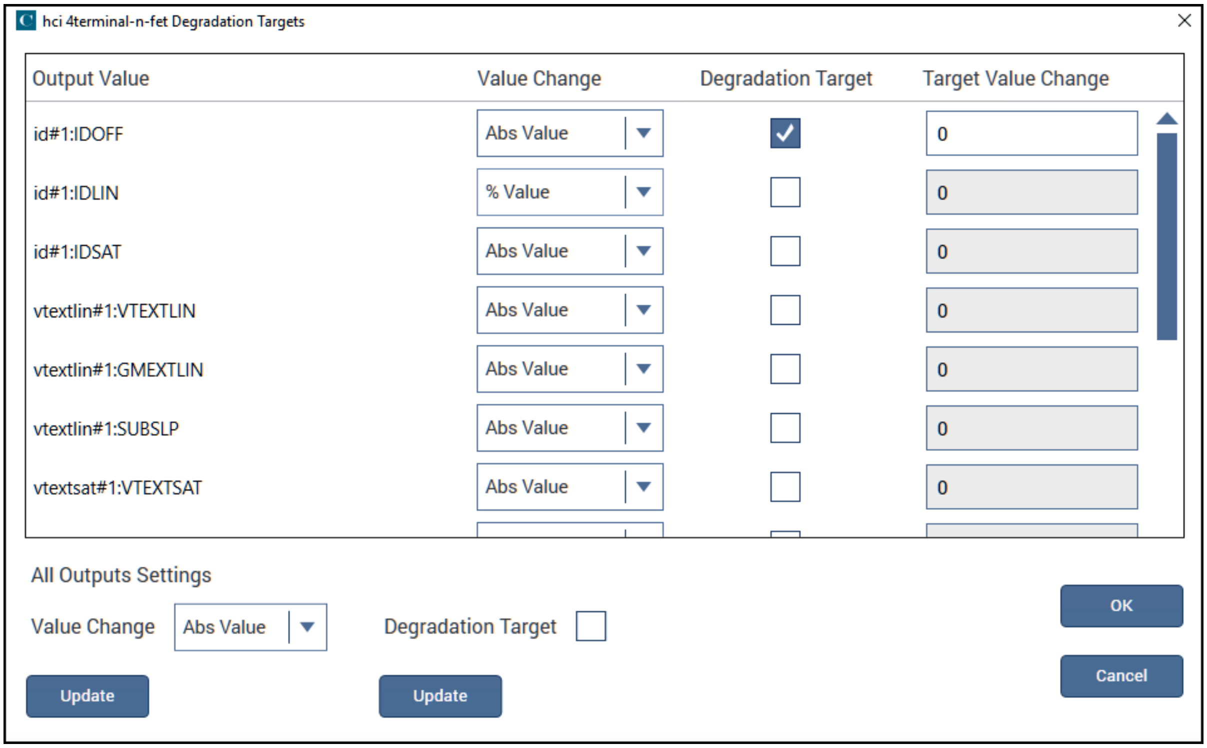


The Degradation Targets option is only available when the Subsite Operation is Stress and there is at least one output value defined in the tests in your subsite. To enable an output value as a target, refer to Define output values for Analyze (on page 9-36).

To configure degradation targets:

1. Select the subsite.
2. Select **Configure**.
3. Select **Degradation Targets**. The following dialog is displayed.

Figure 184: Degradation Targets dialog



4. For Value Change, select **Abs Value** or **% Value** for the output.
5. Select **Degradation Target** for each output for which you want to set a target value.
6. In **Target Value Change**, enter the target value.
7. Select **OK**.

To change the Value Change setting for all output values:

1. In the bottom left of the dialog, set Value Change to **Abs Value** or **%Value**.
2. Select **Update**. All outputs are set to the selected value. You can change individual settings if needed.

To change Degradation Target settings for all output values:

1. In the bottom left of the dialog, select or clear Degradation Target.
2. Select **Update**. The Degradation Target for all outputs is set or cleared. You can change individual settings if needed.

Run an individual subsite

When you run an individual subsite, only the components that are assigned to it run.

When you run the components for a subsite, the actions and tests are run in the order in which they appear in the project tree. Only the devices, actions, and tests that have check boxes selected are run.

NOTE

To abort a test, select **Stop**. All test and action execution stops immediately.

To run an individual subsite:

1. Make sure the check boxes are selected for all items in the subsite that you want to include.
2. Highlight the subsite name.
3. Select **Run**. The Run icon changes as shown below. The active action or test is listed to the left of Run. The Stop icon changes to red.

Figure 185: Run icon while a test is running

When the test completes, a beep sounds and the run arrows around the Run icon are no longer displayed.

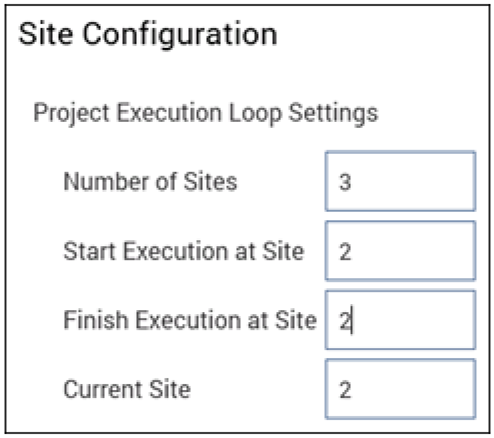
Run a single site

Running a project runs all the sites that are defined for the project. However, you can run a single site if needed.

To run a single site:

1. In the project tree, select the site.
2. Select **Configure**.
3. Set **Start Execution at Site** and **Finish Execution at Site** to the site you want to run. In the following example, only Site 2 is run when you select **Run**.

Figure 186: Settings to run Site 2 only



4. Set **Current Site** to the site that you want to run.
5. Select **Run**. The Run icon changes as shown below. The active site, actions, and tests are listed to the left of Run. The Stop icon changes to red.

Figure 187: Run icon while a test is running

When the site completes, a beep sounds and the run arrows around the Run icon are no longer displayed.

Cycle a subsite

Subsite cycling allows you to repeatedly cycle through the subsite tests. The data for every repeated test is acquired and placed in its Analyze Stress tab.

Measured readings (output values) can be exported from individual tests into the subsite.

To run cycling for a single subsite:

1. Set up the subsite as described in Configure Subsite Cycling (on page 9-4).
2. In the project tree, select the subsite.
3. Select **Run**.

To run cycling for multiple subsites:

1. Set up the subsite as described in Configure Subsite Cycling (on page 9-4).
2. In the project tree, select the project.
3. Make sure the subsites you want to run are checked in the project tree.
4. Select **Run**.

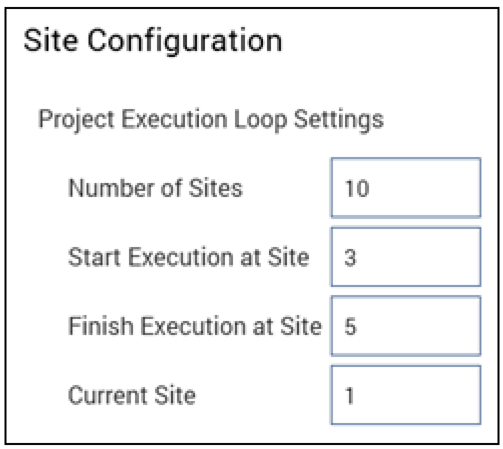
Multi-site execution

Running a project runs all the sites that are defined for the project. However, you can run a subset of the sites if needed.

To run some sites:

1. In the project tree, select the site.
2. Select **Configure**.
3. Set **Start Execution at Site** and **Finish Execution at Site** to the sites you want to run. In the following example, executing the site will run sites 3, 4, and 5.

Figure 188: Multi-site test sequence



4. Select **Run**. The Run icon changes as shown below. The active sites, actions, and tests are listed to the left of Run as they are executed. The Stop icon changes to red.

Figure 189: Run icon while a test is running

When the site completes, a beep sounds and the run arrows around the Run icon are no longer displayed.

Delete All Run History in a project

You can delete all test runs from the Run History pane for every site and test in a project. Reference data is also deleted. The data is deleted from Clarius.

To delete all test runs from a project:

1. In the project tree, select the project.
2. Select **Configure**.
3. Select **Delete All Run History**. A confirmation dialog is displayed.

Delete or dissolve a site and subsite

To remove a site and subsite from a project, you can delete or dissolve the site and subsite. Dissolve is only available if there is one subsite.

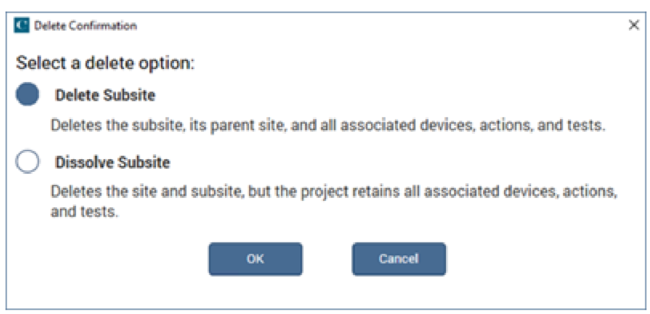
When you select delete, the site, subsite, and all the devices, actions, and tests in the site and subsite are deleted from the project.

When you select dissolve, the site and subsite are deleted, but the devices, actions, and tests remain in the project. If you create a new subsite, information from the previous subsite is restored.

To delete or dissolve a subsite:

1. Right-click the subsite.
2. Select **Delete**. The Delete Confirmation dialog is displayed.

Figure 190: Delete or dissolve a subsite



3. Select **Delete Subsite** or **Dissolve Subsite**.
4. Select **OK**.

Analyze data for subsites

When you select the Analyze pane for a subsite, a spreadsheet and graph of the subsite data is displayed.

To display the graph and sheet for subsite data:

1. In the project tree, select the subsite.
2. Select **Analyze**.

The subsite Analyze pane includes:

- A sheet that shows the data for the selected output values.
- The Run Settings option that displays information about the subsite stress or cycling setup.
- A graph that shows the stress time or cycle index versus the degradation (as a percentage or as an absolute value).
- A stress Settings pane that allows you to adjust the stress sequence from the Analyze pane. Refer to Configure the Stress Settings (on page 9-24) for details on the options.

The options in the subsite Analyze pane are described in the following topics.

Define output values for Analyze

For subsite cycling, you must define output values from tests to be displayed in the subsite Analyze sheet. Each time a subsite is cycled, the measurements for the output values are placed in the subsite Analyze sheet. If, for example, the subsite is cycled five times, there are five measured readings for each output value.

You can choose whether or not to rebuild subsite data when you change the output values. Refer to Enable post run configuration of subsite data (on page 4-9).

To select the values to be used in Analyze:

1. In the project tree, select the test.
2. Select **Configure**.
3. In the right pane, select **Test Settings**.
4. Select **Output Values**.
5. Select the values to use in the subsite Analyze sheet.
6. Select **OK**.

To remove values so they are not available in Analyze:

1. In the project tree, select the test.
2. Select **Configure**.
3. In the right pane, select **Test Settings**.
4. Select **Output Values**.
5. Clear the values.
6. Select **OK**.

Subsite Analyze sheet

The Analyze sheet displays data for the selected output values.

The following figure shows an example sheet for a subsite that has four devices. Analyze spreadsheet columns include:

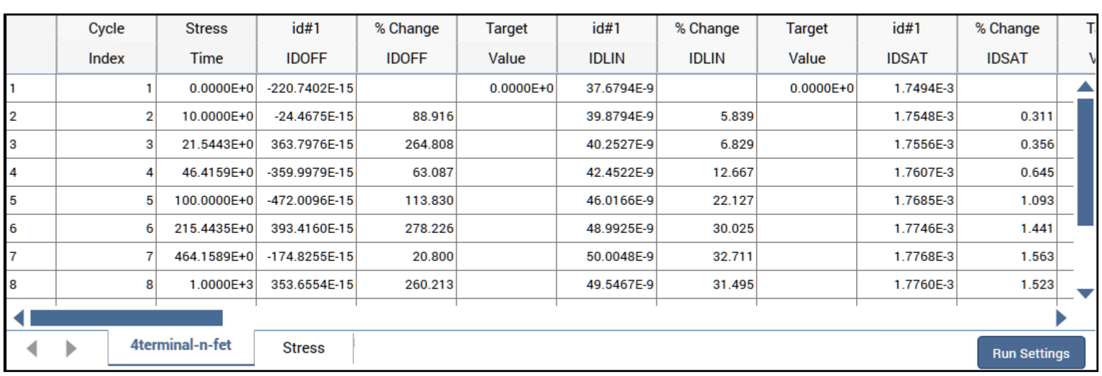
- **Cycle Index:** The cycles that were run.
- **Stress Time:** The stress times (in seconds) for all cycles. The stress for the first cycle is 0.0 seconds, which is the no-stress cycle for HCI testing.
- **id#1 IDOFF:** The measured readings for the first output value, which is the I_{DOFF} reading for the first id test.
- **% Change IDOFF:** The percent change between each post-stress I_{DOFF} reading and the pre-stress I_{DOFF} reading in the first cycle. Calculations start with the second cycle. The percent change value is calculated as:

$$\% \text{ Change } = \text{ABS}\left[\frac{(\text{Post-stress reading} - \text{Pre-stress reading})}{\text{Pre-stress reading} \times 100}\right]$$

- **Target Value:** The target value that was assigned to the output value in the Subsite Stress Properties. A target value of 0.0 indicates that the target for I_{DOFF} is disabled. A target is reached when the % change value equals or exceeds the target value.

After the first target percentage value, the subsequent three columns provide readings for the output value, the percent change, and the target value of the selected output values.

Figure 191: Subsite Analyze sheet in Stress/Measure mode



	Cycle Index	Stress Time	id#1 IDOFF	% Change IDOFF	Target Value	id#1 IDLIN	% Change IDLIN	Target Value	id#1 IDSAT	% Change IDSAT
1	1	0.0000E+0	-220.7402E-15		0.0000E+0	37.6794E-9		0.0000E+0	1.7494E-3	
2	2	10.0000E+0	-24.4675E-15	88.916		39.8794E-9	5.839		1.7548E-3	0.311
3	3	21.5443E+0	363.7976E-15	264.808		40.2527E-9	6.829		1.7556E-3	0.356
4	4	46.4159E+0	-359.9979E-15	63.087		42.4522E-9	12.667		1.7607E-3	0.645
5	5	100.0000E+0	-472.0096E-15	113.830		46.0166E-9	22.127		1.7685E-3	1.093
6	6	215.4435E+0	393.4160E-15	278.226		48.9925E-9	30.025		1.7746E-3	1.441
7	7	464.1589E+0	-174.8255E-15	20.800		50.0048E-9	32.711		1.7768E-3	1.563
8	8	1.0000E+3	353.6554E-15	260.213		49.5467E-9	31.495		1.7760E-3	1.523

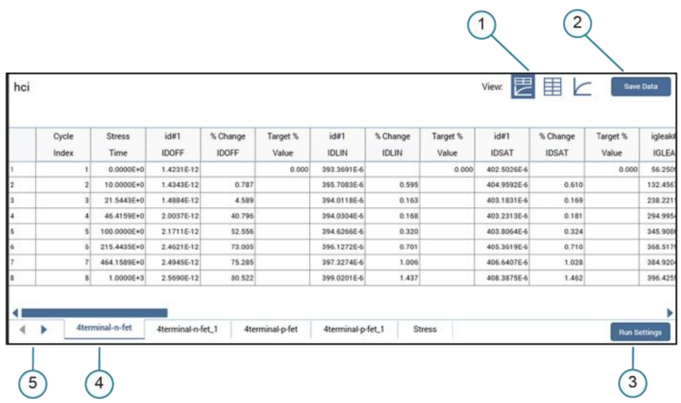
If Measure Current or Measure Voltage is selected in the Configure Terminal Settings for the subsite, the 4200A-SCS makes one measurement at each terminal before each stress cycle begins and returns that measurement to the subsite Analyze sheet. It makes one current measurement if the terminal is set to DC voltage stressing or one voltage measurement if the terminal is set to current stressing. The column name that contains the measurement includes the name of the device, the name of the terminal, and a measurement ID (I or V).

The Analyze spreadsheet is compatible with Microsoft™ Excel™.

Options on the Subsite spreadsheet

The following table provides brief descriptions of the options on the Subsite spreadsheet.

Figure 192: Options on the Subsite spreadsheet



1	View selections for the Analyze pane. You can display the spreadsheet and the graph, the spreadsheet only, or the graph only.
2	Save Data allows you to save the spreadsheet and graph data for the selected subsite. Refer to Save subsite results and graphs (on page 9-44) for more information.
3	Run Settings displays the subsite settings and the device and test selections that were used to produce the results. Refer to Subsite Run Settings (on page 9-39) for more information.
4	Tab for the device data. If there are multiple devices selected, you can select the tab to display the data for that device or stress information.
5	If there are more than three test runs selected, you can use these arrows to move between the spreadsheet tabs.

Hide columns in the subsite spreadsheet

You can hide columns in the Analyze spreadsheet. The data in the column remains available for graphing.

To hide a column, right-click the column and select **Hide Column**.

To redisplay the column, right-click the column and select **Unhide Column**. A dialog is displayed that allows you to select which columns to display.

To remove data from the Analyze spreadsheet, clear the values from Output Values in the Test Settings. Refer to Define output values for Analyze (on page 9-36) for more information.

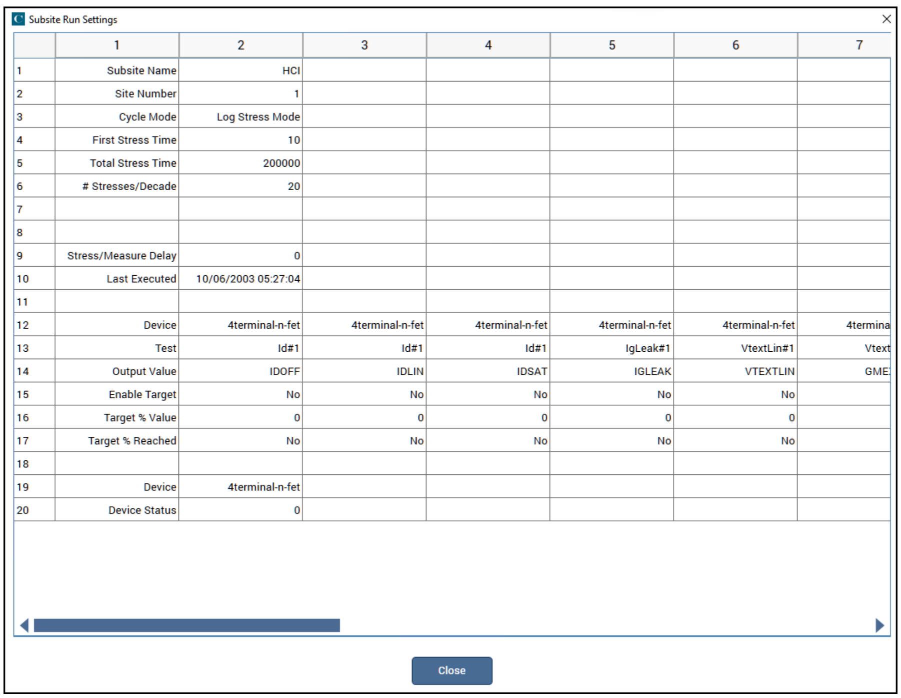
Subsite Run Settings

The Subsite Run Settings dialog displays information about the subsite cycling setup. It includes the subsite setup and device, test, and output value information. For each device, the target value is listed. After subsite cycling, it also indicates if targets were reached.

To display Subsite Settings:

1. Select the subsite.
2. Select **Analyze**.
3. Select **Run Settings**. An example of the Run Settings sheet is shown in the following figure.

Figure 193: Subsite Analyze Settings sheet for Stress/Measure mode



Subsite Run Settings

	1	2	3	4	5	6	7
1	Subsite Name	HCI					
2	Site Number	1					
3	Cycle Mode	Log Stress Mode					
4	First Stress Time	10					
5	Total Stress Time	200000					
6	# Stresses/Decade	20					
7							
8							
9	Stress/Measure Delay	0					
10	Last Executed	10/06/2003 05:27:04					
11							
12	Device	4terminal-n-fet	4terminal-n-fet	4terminal-n-fet	4terminal-n-fet	4terminal-n-fet	4termina
13	Test	Id#1	Id#1	Id#1	IgLeak#1	VtextLin#1	Vtext
14	Output Value	IDOFF	IDLIN	IDSAT	IGLEAK	VTEXTLIN	GME
15	Enable Target	No	No	No	No	No	
16	Target % Value	0	0	0	0	0	
17	Target % Reached	No	No	No	No	No	
18							
19	Device	4terminal-n-fet					
20	Device Status	0					

Close

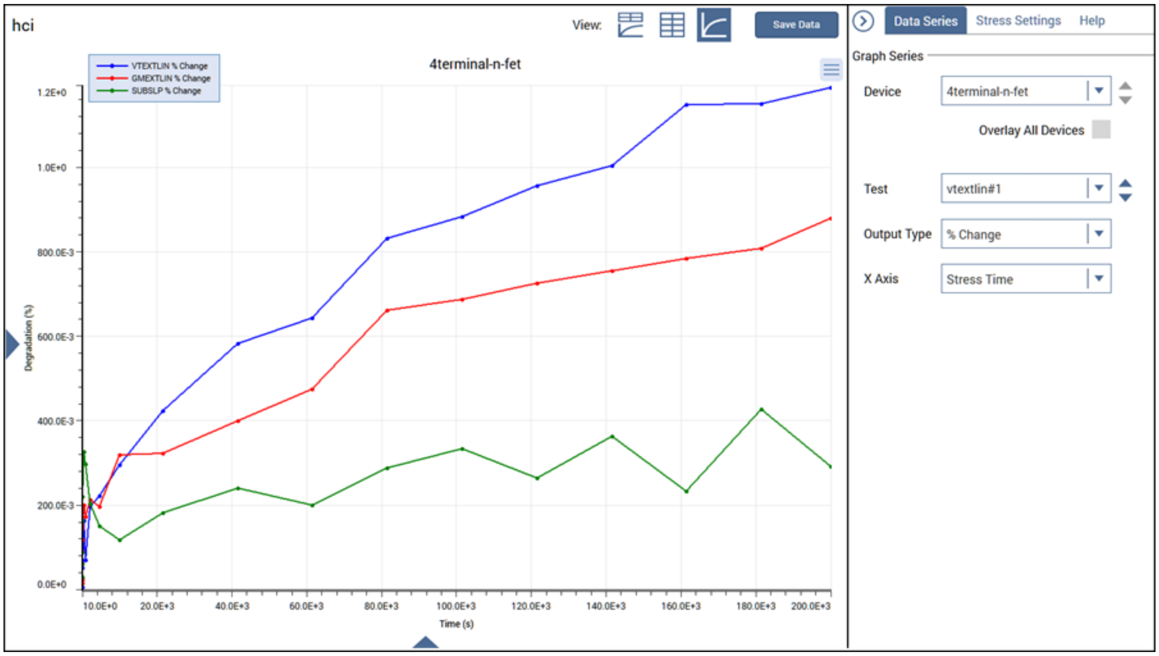
Subsite Analyze graph

The graph for the subsite shows degradation (in %) or output value versus the stress time or cycle index.

When Stress Time is selected, each point in the graph represents the device degradation (% Change) for tests after each stress cycle. When Cycle Time is selected, each point in the graph represents the percentage of change in device degradation for each cycle.

The graph in the following figure traces for test `id#1` for the `4terminal-n-fet` device. The three traces are for the Output Values I_{DOFF}, I_{DLIN}, and I_{DSAT}.

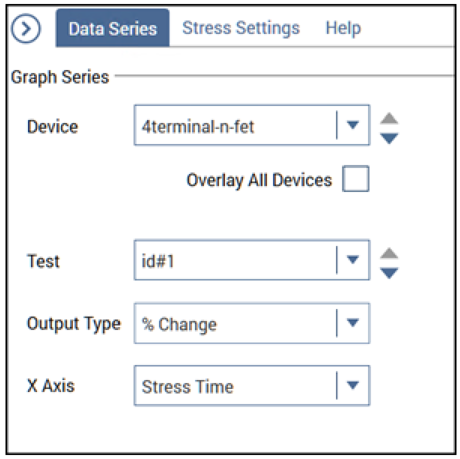
Figure 194: Stress/measure mode graph



The Data Series options in the right pane allow you to change which device and test data are graphed. The options are:

- **Device:** Select the device for which to display data. This is not available if Overlay All Devices is selected.
- **Overlay All Devices:** Select this option to display all the graph traces for all devices that were measured by the selected test. For a single-device subsite, this option is not available.
- **Test:** Select the test for which to display data. Only tests with at least one output value selected on the Configure Test Settings tab for the test are available. If Overlay All Devices is selected, only devices that are common to all devices in the subsite are available.
- **Output Type:** Select one of the following options:
  - **Value:** The measured reading.
  - **% Change:** The percentage change value.
  - **% Change and Value:** Both values.
- **X Axis:** Select Cycle Index or Stress Time.

Figure 195: Subsite Data Series



Define subsite data to be graphed

You can change the data that is shown on the graph.

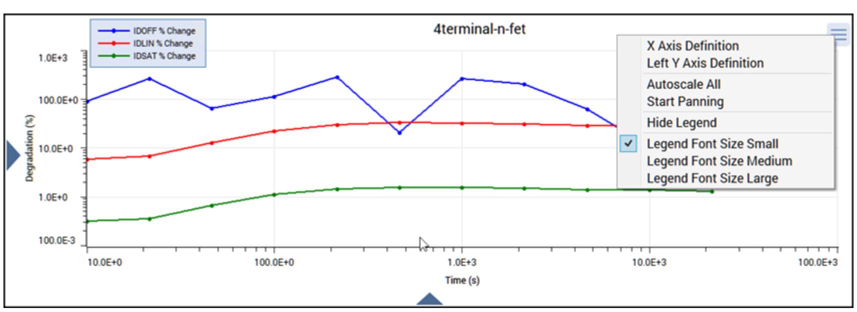
To select the data that is shown on the graph:

1. Select the subsite.
2. Select **Analyze**.
3. In the right pane, select **Data Series**.
4. In **Device**:
   - To display data for one device: Select the device from the list.
   - To display data for all devices: Select **Overlay All Devices**.
5. In **Test**, select the test to graph.

Customize the subsite graph

You can customize the subsite graph using options in the Graph Definition Menu, shown in the following figure. You can also access these settings by right-clicking the graph.

Figure 196: Graph Definition Menu for subsite graphs



The options are:

- **X Axis Definition:** Opens the X-Axis settings, where you can define the settings for the X axis. See Define subsite graph axes (on page 9-44) for more information.
- **Left Y Axis Definition:** Opens the Y-Axis settings, where you can define the settings for the Y axis. See Define subsite graph axes (on page 9-44) for more information.
- **Autoscale All:** Scales all graph axes one time. You can use this to reset the graph after panning or zooming.
- **Start Panning:** Allows you to drag the graph. Refer to Zoom, pan, and autoscale (on page 3-32) for more information.
- **Hide or Show Legend:** Displays or hides the graph legend. You can drag the legend to a new location on the graph.
- **Legend Font Size:** Select Small, Medium, or Large to change the size of the text in the legend.

Zoom, pan, and autoscale

If you are using the touch screen:

- To zoom in on the graph, pinch the area of the graph to view.
- To zoom out, move your fingers apart.
- To return to the full graph view, double-tap the graph.

If you are using a mouse:

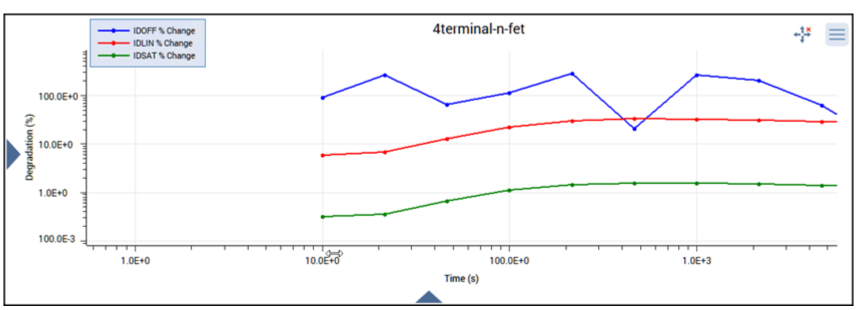
- To zoom, select the area on the graph with the left mouse button.
- To return to the full graph view, double-click the graph.

To use the mouse to pan along the axis (left to right on x-axis or up-down on the y-axis):

1. Place the mouse over the axis in the graph. The mouse cursor changes to the scrolling icon.
2. Drag the axis to the new location.

An example of the panned axis is shown in the following figure.

Figure 197: Panning along the x-axis



To pan in all directions:

1. From the Graph Definition Menu, select **Start Panning**.
2. Drag the graph in any direction.

When panning is active, the following icon is displayed to the left of the Graph Definition Menu:

Select the icon to stop panning. You can also stop panning by selecting another option from the Graph Definition Menu.

Rescale the graph:

From the Graph Definition Menu, select **Autoscale All**.

This option automatically scales all graph axes one time. You can use this to reset the graph after panning or zooming.

When you pan or zoom, Autoscale All is disabled. The minimum and maximum values of each axis show the values set by zoom.

Synchronize graphs for one test executed at multiple sites

If your project contains more than one site, you can plot test data that is gathered at multiple identically fabricated sites on multiple identically configured graphs using Synchronize Graphs in Sites.

For example, if you executed the `vds-id#1` test at the first five sites of a project, select Synchronize Graphs in Sites to identically configure graphs for the Site 1, 2, 3, 4, and 5 Run sheets.

If the project contains multiple instances of a same-named test, you must apply the feature separately each such instance. For example, if the project tree has both `vds-id#1` and `vds-id#2`, you must apply Synchronize Graphs in Sites separately for `vds-id#1` and `vds-id#2`.

CAUTION

The graphs for the selected test are configured identically for all project sites, both for the present data and for all future data. This applies to future graphs for all sites, even if data was not yet generated for some sites when Synchronize Graphs in Sites was selected. The only way to undo these effects is to manually reconfigure each site-specific graph.

To synchronize multi-site graphs:

1. From the project tree, select the site.
2. In the project tree, select the test or action in the subsite that should have synchronized graphs.
3. Select **Analyze**.
4. From the Graph Definition Menu, select **Advanced Settings**.
5. Select **Synchronize Graphs in Sites.** A caution message is displayed.
6. Select **Yes**. The graphs for the selected test or action are now configured identically for all project sites.

Define subsite graph axes

You can change the settings for the graph axes using the Axis Definition options in the Graph Definition Menu or by selecting the triangles on the bottom and sides of the graph. An example of the options for the X-axis is shown in the following figure.

Figure 198: Subsite graph axis settings

Minimum and **Maximum** set the graph to display only data between these two values. These options are not available if Autoscale is selected.

Autoscale automatically scales the axes to best fit the data. When you select Autoscale, the Minimum and Maximum settings are set to the autoscale values.

The **Label** is the text that identifies the axis.

The **Units** option selects the unit of measure (volts, amps, seconds, farads, siemens, or hertz) that is displayed on the axis. When a unit of measure is selected, you can also set how the value is displayed, such as in A, mA, or µA. When Engineering Notation is selected, the Auto option is available. In this case, the graph automatically adjusts the units of measure for the best presentation of the data.

Engineering Notation displays the scale values and cursor data points in engineering notation. When this option is cleared, the scale values and data points are displayed in either standard form or scientific notation, depending on the value.

Scale allows you to display the data on a linear or logarithmic scale. If Log is selected, Absolute Values is automatically set. If you return to Linear, you may need to clear Absolute Values if you want to graph negative values.

Absolute Values graphs the data as unsigned values when selected.

Save subsite Analyze sheets and graphs

You can save the sheet data and graphs for the entire subsite or for individual tests or actions.

The sheet data can be saved to the `.xls`, `.xlsx`, `.csv`, or `.txt` format. The `.xls` and `.xlsx` formats are compatible with the Microsoft™ Excel™ application.

When you save test data to the `.xls` or `.xlsx` format, the saved data includes data for all selected runs and the settings for the runs. If you select the `.csv` or `.txt` format, only the data from the most recent run is saved. Refer to Save results and graphs (on page 3-33) for more information on saving test data.

You can save graphs to `.jpeg`, `.bmp`, `.gif`, or `.png` format.

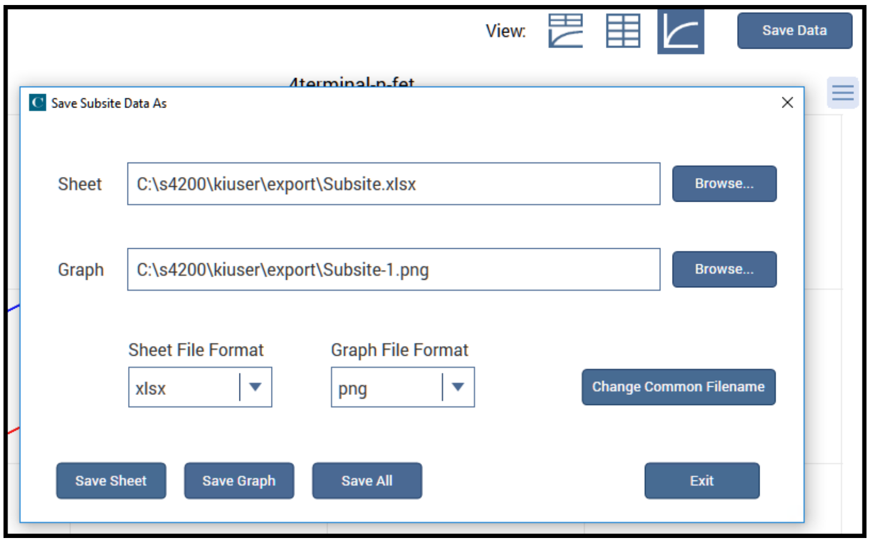
NOTE

To export all data from a project, test, or device, use the Data Export tool. Refer to Data Export tool (on page 5-19) for more information.

To save the subsite sheet information:

1. Select the subsite.
2. Select **Analyze**.
3. Select **Save Data**. The Save Subsite Data As dialog is displayed.

Figure 199: Analyze Save Subsite Data As dialog



4. In **Sheet**, select the file location and name.
5. From the **Sheet File Format** list, select the file format.
6. Select **Save Sheet**.

To save the graph:

1. Select the subsite.
2. Select **Analyze**.
3. Select **Save Data**. The Save Subsite Data As dialog is displayed.
4. In **Graph**, select the file location and graph name.
5. Select the **Graph File Format**.
6. Select **Save Graph 1**.

To save graph and Run sheet information:

1. Select the subsite.
2. Select **Analyze**.
3. Select **Save Data**. The Save Subsite Data As dialog is displayed.
4. If you would like to use the same name for the graph and sheet, select **Change Common Filename** and enter the filename. The file names are changed. No change is made to the file locations.
5. In **Sheet**, select the file location and sheet name.
6. In **Graph**, select the file location and graph name.
7. Select the **Sheet File Format**.
8. Select the **Graph File Format**.
9. Select **Save All**.

User library descriptions

In this section:

Introduction

Keithley provides several user libraries of user modules. The following topics provide an overview of each of the user libraries.

The KULT user libraries and user modules that are provided with Clarius+ are available in the directory:

`C:\s4200\kiuser\usrlib\`

AVMControl user library

The `AVMControl` user library contains a user module that limits the SMU maximum voltage. The following table lists and briefly describes the user module.

AVMControl user module

User module	Description
`SetAVMLevel`	Sets the 4200-SMU, 4201-SMU, 4210-SMU, or 4211-SMU absolute voltage monitor (AVM) maximum voltage. The AVM is an analog circuit that limits the SMU voltage output regardless of what the SMU is sourcing and measuring. Depending on the voltage that the AVM is set to, the SMU clamps the output voltage to one of the built-in voltage limits. Refer to the Help pane for the voltage limits.

AFG31000 user library

The `AFG31000_examples_ulib` user library contains user modules that configure and output arbitrary waveforms from an AFG31000 Arbitrary Function Generator through a USB connection. The following table lists and briefly describes the user modules.

AFG31000_examples_ulib user modules

User module	Description
`afg31000-config-1ch`	Configures and outputs channel 1 of an AFG31000 to an arbitrary waveform using the AFG31000 basic mode.
`afg31000-config-2ch`	Configures and outputs each channel of an AFG31000 to an arbitrary waveform using the AFG31000 basic mode.
`afg31000-segarb-1ch`	Configures and outputs channel 1 of an AFG31000 to an arbitrary waveform file (`.twfx`).
`afg31000-segarb-2ch`	Configures and outputs each channel of an AFG31000 to an arbitrary waveform file (`.twfx`).

BeepLib user library

The BeepLib user library contains several user modules that control the 4200A-SCS beeper. The following table lists and briefly describes the user modules.

The beeper user modules are affected by the Windows operating system audio settings. For example, if sound is muted, the beeper will not sound.

BeepLib user modules

User module	Description
beep	Specifies the frequency and duration of the beeper.
BeepCharge	Sounds a battle cry through the speaker.
BeepDown	Sounds a series of beeps in descending tones through the speaker.
BeepInfiniteLoop	This function sounds a series of beeps through the system speaker. The beeps continue until they are terminated.
BeepUp	Sounds a series of beeps in ascending tones through the speaker.

chargepumping user library

The chargepumping user library contains several user modules to characterize interface and charge-trapping phenomena. The following table lists and briefly describes the user modules.

chargepumping user modules

User module	Description
AmplitudeSweep	Pulse amplitude is swept while the SMU base voltage is kept constant. The charge pumping current (I_{CP}) is measured as a function of the pulse amplitude voltage.
AmplitudeSweep_2SMU	Same as the AmplitudeSweep user module, except that it uses a second SMU to apply a DC bias voltage to the source/drain terminals.
BaseSweep	The base voltage of the waveform is swept by a SMU while the amplitude of the pulse is kept constant. The resulting charge pumping (I_{CP}) is measured as a function of the base voltage.
BaseSweep_2SMU	Same as BaseSweep user module, except it uses a second SMU to apply a DC bias voltage to the source/drain terminals.
FallTimeLinearSweep	Performs a linear sweep of the falling transition time of the pulse. Charge pumping current (ICP) is measured and graphed as a function of the fall time.
FreqFactorSweep	With the pulse amplitude, offset voltage, and rise/fall time kept constant, the charge pumping current (ICP) is measured as a function of a multiplier factor controlled frequency sweep of the test frequency.
FreqLinearSweep	With the pulse amplitude, offset voltage, and rise/fall time kept constant, the (ICP) is measured as a function of a linear sweep of the test frequency.
RiseTimeLinearSweep	Performs a linear sweep of the rising transition time of the pulse. Charge pumping (ICP) is measured as a function of the rise time.

CompressedAcquisitionUlib user library

The `CompressedAcquisitionUlib` user library contains a user module that allows you to set up rules for data compression.

The following table lists and briefly describes the user module.

CompressedAcquisitionUlib user module

User module	Description
`ConstantCurrent`	A constant current source test for a generic device. It requires at least one SMU to be connected to the device terminals. The SMUs source constant currents and measure voltages during the test. The test stops when any of the SMUs reaches compliance (corresponds to device breakdown). The test supports data compression to reduce the amount of returned data.
`ConstantVoltage`	A constant voltage source test for a generic device. It requires at least one SMU to be connected to the device terminals. The SMUs source constant voltages and measure currents during the test. The test stops when any of the SMUs reaches compliance (corresponds to device breakdown). The test supports data compression to reduce the amount of returned data.

cvivulib user library

The `cvivulib` user library contains user modules for configuring the 4200A-CVIV Multi-Switch. The following table lists and briefly describes the user modules.

cvivulib user modules

User module	Actions in Clarius	Description
`cviv_configure`	`cviv-configure`	This user module configures the CVIV relays and the display for each channel.
`cvu_cviv_comp_collect`	`cvu-cviv-comp-collect`	This user module provides CVU compensation collection for open, short, and load with a 4200A-CVIV.

cvucompulib user library

The `cvucompulib` user library contains user modules for collecting 4210-CVU compensation data. The following table lists and briefly describes the user modules.

NOTE

If your configuration includes a 4200A-CVIV, use the cvivulib (on page 10-4) user library instead of this one.

cvucompulib user modules

User module	Description
`cvu_ConstantsFileSelect`	Selects the constants file that is used for a CVU ITM or UTM. The file must be created using `cvu_OSLcomp_collect`.
`cvu_OSLcomp_collect`	Collects the open, short, and load compensations of the CVU instrument as selected. It generates a file that contains the open, short, and load compensation values to apply to the CVU readings that are returned from an ITM or UTM.

cvuulib user library

The cvuulib user library contains user modules that perform V AC, V DC, and frequency sweeps for the 4210-CVU and 4215-CVU. The following table lists and briefly describes the user modules.

cvuulib user modules

User module	Description
capacitance-time-65k	Takes up to 65,000 capacitance versus time readings and calculates the noise.
Sampling	Samples a number of readings at a fixed AC voltage, DC voltage, and frequency.
SweepACV	Sweeps AC voltage on a CVU Instrument.
SweepDCV	Sweeps DC voltage on a CVU Instrument.
SweepF_Log	4215-CVU only: Logarithmic sweep of frequency.
SweepF	Sweeps frequency on a CVU Instrument.

DLCP user library

The DLCP user library contains a user module for making C-V measurement for drive-level capacitance profiling (DLCP). The following table lists and briefly describes the user module.

DLCP user module

User module	Description
ACSweep	This module allows you to make C-V measurements for drive-level capacitance profiling (DLCP) using the 4210-CVU or 4215-CVU. During this measurement, the applied AC voltage is sweeping while the capacitance is measured. The total applied voltage (AC and DC) is kept constant. The total applied voltage is defined as the DC voltage minus ½ the p-p AC voltage.

dmm-6500-7510-temp-ulib user library

The dmm-6500-7510-temp-ulib user library contains modules that allow users to communicate with a Keithley DMM6500 or DMM7510 digital multimeter through USB to measure temperature. The library uses VISA to communicate with the external DMM. The following table briefly describes each module.

dmm-6500-7510-temp-ulib user modules

User module	Description
meas_smuI_dmmTemp	Measures the current on a two-terminal device using two SMUs or one SMU and the GNDU while the external DMM measures the temperature using a thermocouple.
meas_time_dmmTemp	Measures the temperature using an external DMM and records the time elapsed between measurements.

flashulib user library

The flashulib user library contains user modules for flash memory testing. The following table lists and briefly describes the user modules.

flashulib user modules

User module	Description
configure_dc_flash	Disconnects pulse channels by opening the solid-state relay for each pulse channel in the supplied list. This routine should be used before running a DC test when the pulse and DC signals are connected at each DUT terminal.
double_pulse_flash	Defines and outputs 1 to 8 waveforms that consist of two pulses that have independent widths and levels. The waveforms are defined using line segments (Segment Arb mode). You can define the waveform for just a program or erase pulse or for a waveform that combines program and erase cycles for up to eight independent pulse channels.
pmu_configure_dc_flash	Disconnects pulse channels by opening the solid-state relay for each pulse channel in the supplied list. This routine should be used before running a DC test, when the pulse and DC signals are connected at each DUT terminal.
pmu_double_pulse_flash	Defines and outputs 1 to 8 waveforms that consist of two pulses that have independent widths and levels. The waveforms are defined using line segments (Segment Arb mode of the 4220-PGU or 4225-PMU). You can define the waveform for just a program or erase pulse or for a waveform that combines both program and erase cycles for up to eight independent pulse channels.
pmu_single_pulse_flash	Defines and outputs 1 to 8 waveforms that consists of two pulses that have independent widths and levels. The waveforms are defined using line segments (Segment Arb mode of the 4220-PGU or 4225-PMU). You can define the waveform for just a program or erase pulse or for a waveform that combines both program and erase cycles for up to eight independent pulse channels.
single_pulse_flash	Defines and outputs 1 to 8 waveforms that consist of two pulses that have independent widths and levels. The waveforms are defined using line segments (Segment Arb mode of the pulse card). You can define the waveform for just a program or erase pulse or for a waveform that combines both program and erase cycles for up to eight independent pulse channels.

GateCharge user library

The GateCharge user library contains a user module for configuring the 4200A-SCS to measure gate charge of a power MOSFET. The following table lists and describes the user module.

GateCharge user module

User module	Description
gate_charge	This module measures the gate charge of a power MOSFET using two source measure units (SMUs).

generic_gpib_ulib user library

The generic_gpib_ulib user library contains several modules that allow users to send a command or string of commands to an external instrument from Clarius. The external instrument must be connected using a GPIB cable and configured to communicate using GPIB. The following table briefly describes each module.

generic_gpib_ulib user modules

User module	Description
query_double_gpib	Sends a command to the external instrument and immediately reads the response. The response is assumed to be comma-separated double-format data and is parsed into up to four columns in the Clarius Analyze sheet.
query_string_gpib	Sends a command to the external instrument and immediately reads the response. The response is assumed to be a string and is returned to the Clarius Analyze sheet and the message console.
read_double_gpib	Reads from the output buffer of the external instrument. The response is assumed to be comma-separated double-format data and is parsed into up to four columns in the Clarius Analyze sheet.
read_string_gpib	Reads from the output buffer of the external instrument. The response is assumed to be a string and is returned to the Clarius Analyze sheet and output to the message console.
write_string_gpib	Writes a string command or commands to the external instrument. No response is read.

generic_visa_ulib user library

The generic_visa_ulib user library contains several modules that allow users to send a command or string of commands to an external instrument from Clarius. The external instrument must be connected using a USB cable and configured to communicate through USB. There is also a module to find VISA resource strings. The following table briefly describes each module.

generic_visa_ulib user modules

User module	Description
find_visa_resource	Returns the VISA resource strings for all external instruments connected to the 4200A-SCS through USB.
query_double_visa	Sends a command to the external instrument and immediately reads the response. The response is assumed to be comma-separated double-format data and is parsed into up to four columns in the Clarius Analyze sheet.
query_string_visa	Sends a command to the external instrument and immediately reads the response. The response is assumed to be a string and is returned to the Clarius Analyze sheet and the message console.
read_double_visa	Reads from the output buffer of the external instrument. The response is assumed to be comma-separated double-format data and is parsed into up to four columns in the Clarius Analyze sheet.
read_string_visa	Reads from the output buffer of the external instrument. The response is assumed to be a string and is returned to the Clarius Analyze sheet and output to the message console.
write_string_gpib	Writes a string command or commands to the external instrument. No response is read.

hivcvulib user library

The hivcvulib user library contains user modules for controlling high-voltage C-V measurements. You can use these modules with either one or two 4205-RBT configurations. Two 4205-RBTs are included in the 4200-CVU-PWR C-V Power Package.

The following table lists and briefly describes the user modules.

hivcvulib user modules

User module	Description
CsRs_SweepV	This module returns Cp-Gp and Cs-Rs parameters. It can be run using a CVU, SMU, and 4200A-CVIV, or using a CVU, SMU, and a 4205-RBT.
multipleSMU_SweepV	This module allows you to make high voltage C-V measurements up to 400 V using a CVU, SMUs, and the 4200A-CVIV. The CVU measures the capacitance, the SMU supplies the DC bias, and the AC and DC signals are coupled through a bias tee connection in the 4200A-CVIV. For a 3-terminal MOSFET, all three SMUs must be sweeping. The gate and the source SMUs must sweep simultaneously to prevent the device from turning on. It is highly recommended that the SMUs at the gate and source have the same start and stop voltages to prevent damage to the device.
CvsT	Provides capacitance measurements as a function of time at a user-specified DC bias. You can measure capacitance up to 200 V DC bias with one 4205-RBT and one SMU. Additionally, you can measure capacitance up to 400 V DC bias with two 4205-RBTs and two SMUs.
SweepV	Uses one 4205-RBT to sweep a DC voltage across the DUT using the 4200-SMU, 4201-SMU, 4210-SMU, or 4211-SMU and measure the capacitance using the 4210-CVU or 4215-CVU. If two 4205-RBTs are used with the SweepV module, one SMU sweeps the DC voltage and the other SMU applies an offset DC bias.

Hotchuck_Temptronics3010B user library

This user library controls the temperature of Temptronics 3010B hotchucks. The user module in this library sets the target temperature and waits until the target is reached before exiting.

Hotchuck_Temptronics3010B user module

User module	Description
Settemp	This routine controls the Temptronics thermal controller 3010B and other compatible models.

Hotchuck_Triotek user library

The user module in the Hotchuck_Triotek user library is used to control the temperature of the Trio-Tech hot chuck.

Hotchuck_Triotek user module

User module	Description
SetChuckTemp	Sets the temperature of the Trio-Tech hot chuck.

HP4284ulib user library

You use the user modules in the `HP4284ulib` user library to control the Keysight 4284A or 4980A LCR Meter. These user modules are summarized in the following table.

HP4284ulib user library

User module	Description
`Cmeas4284`	Makes a single capacitance measurement.
`CvSweep4284`	Makes capacitance versus voltage measurements using a staircase sweep.

HP4294ulib user library

You can use the user modules in the `HP4294ulib` user library to calibrate and control the Keysight Model 4294 IMP meter. Subroutines are provided to perform voltage or frequency sweeps. These user modules are summarized in the following table.

HP4294ulib user modules

User module	Description
`CvSweep4294`	Performs a capacitance versus voltage sweep.
`FISweep4294`	Performs a frequency versus impedance sweep.
`LoadCal4294`	Performs LOAD calibration.
`OpenCal4294`	Performs OPEN calibration.
`PhaseCal4294`	Performs PHASE calibration.
`ShortCal4294`	Performs SHORT calibration.

A Keysight 4294 measurement is valid only if proper calibrations are performed before the measurement is made. The user may run calibration at any time.

A recommended calibration sequence is as follows:

1. Move prober to an OPEN calibration structure.
2. Call `PhaseCal4294`.
3. Call `OpenCal4294`.
4. Move prober to a SHORT calibration structure.
5. Call `ShortCal4294`.
6. Move prober to a LOAD calibration structure.
7. Call `LoadCal4294`.

NOTE

The Keysight 4294 is added to the 4200A-SCS test system using KCon. For details, see “Keithley Configuration Utility (KCon)” in *Model 4200A-SCS Setup and Maintenance*.

NOTE

Details on Keysight 4294 operations are provided in the documentation provided by Keysight for the IMP meter.

HP8110ulib user library

Use the user modules in the HP8110ulib user library to control a Keysight Model 8110A Pulse Generator. These user modules are summarized in the following table. The table also lists the user test modules (UTM) created by Keithley that use the user modules.

HP8110ulib user modules

User Module	UTM Name	Description
PguInit8110	pgu1-init	Initializes the pulse generator to the default setup.
PguSetup8110	pgu1-setup	Sets the output pulse parameters.
PguTrigger8110	pgu-trigger	Specifies pulse count and trigger start of output.

ki340xulib user library

Used with the Keithley Series 3400 pulse/pattern generators.

ki340xulib user modules

User module	Description
PguInit340x	Initializes the 3401 or 3402 pulse generator to a specific state.
PguSetup340x	Defines the pulse timing and voltage settings. Once defined, the pulse can be triggered using the PguTrigger340x command.
PguTrigger340x	Triggers the pulse (or pulses) defined by the PguSetup340x function.

KI42xxulib user library

The KI42xxulib user library provides an example subroutine for doing a MOSFET ON resistance (R_{ON}) test routine using the 4200A-SCS LPT library interface.

KI42xxulib user module

User module	UTM name	Description
Rdson42XX	rdson	Measures the drain to source resistance of a saturated MOSFET.

KI590ulib user library

The user modules in the `KI590ulib` user library are used to control the 590 C-V Analyzer. These user modules are summarized in the following table. Also listed in the table are names of the user test modules (UTMs) and actions in Clarius that use the user modules.

KI590ulib user modules

User module	UTM or action name	Description
`CableCompensate590`	`cable-compensate` in the `ivcvswitch` project	Performs cable compensation using known capacitance source values.
`Cmeas590`	`590-cmeas`	Makes a single capacitance measurement.
`CtSweep590`	`590-ctsweep`	Makes a capacitance versus time measurement.
`CvPulseSweep590`	`590-cvpulsesweep`	Makes capacitance versus voltage measurements using a pulse sweep.
`CvSweep590`	`590-cvsweep`	Makes capacitance versus voltage measurements using a staircase sweep.
`DisplayCableCompCaps590`	`display-cap-file` in the `ivcvswitch` project	Places capacitance source values in a spreadsheet.
`LoadCableCorrectionConstants`	n/a	Reads the cable compensation parameters for the range and frequency specified from the cable compensation file and sends these parameters to the 590.
`SaveCableCompCaps590`	`save-cap-file` in the `ivcvswitch` project	Saves entered capacitance source values to a file.

KI595ulib user library

The user modules in the `KI595ulib` user library are used to perform Q/t sweeps and C-V sweeps using the Keithley 595 Quasistatic C-V Meter. These user modules are summarized in the following table.

KI595ulib user modules

User module	Description
`CVsweep595`	Performs a quasistatic C-V sweep between the start voltage and the stop voltage. The data returned is the source voltage, measured capacitance, Q/t current, and timestamp on each measurement.
`QTSweep595`	Makes 20 Q/t current and capacitance measurements with various time delays that are spaced between 0.07 s and the maximum delay. You can analyze and plot the resulting values to determine the optimum delay time to use during the C-V sweep. The optimum delay time is the time when Q/t reaches the system leakage level.

ki622x_2182ulib user library

The user modules in this library connect to a Keithley Model 6220 or 6221 Current Source and Model 2182 or 2182A Nanovoltmeter to make delta resistance measurements or differential conductance measurements.

The next table lists the user modules. It also provides the name of tests and actions in Clarius that are based on these user modules.

ki622x_2182ulib user modules

User module	Test name	Description
`DeltaMeas`	`622x-2182a-delta-meas`	Makes delta resistance measurements using a current source and nanovoltmeter. Delta measurements are a series of differential voltage measurements between two current values (high and low current). The 622X alternates the output current to form a square wave. The 2128/2182A makes three voltage measurements during the square waves and averages them together. This process eliminates the effects of thermal EMFs.
`DiffCondSweep`	`None`	Performs a differential conductance sweep using a current source and nanovoltmeter. This measurement uses a process similar to the delta measurement, a three-point moving average, to eliminate the effects of thermal EMFs.

ki82ulib user library

The user modules in the ki82ulib user library control the Model 82 C-V System. They perform simultaneous C-V, C-t, and Q/t measurements and cable compensation. The following table lists the user modules. It also provides the name of tests and actions in Clarius that are based on these user modules.

ki82ulib user modules

User module	Test and action names	Description
`Abortmodule82`	n/a	Puts the three System 82 instruments into a known state when a test is aborted. This function is used by other library modules in the `atexit()` function.
`CableCompensate82`	`cable-compensate` `cablecomp`	Performs 590 cable compensation using the capacitor values stored in the specified cable compensation file. The resultant compensation values generated by the compensation process are stored in the same file.
`CTsweep82`	`ctsweep`	Measures capacitance as a function of time at a certain bias.
`DisplayCableCompCaps82`	`display-cap-file`	Places capacitance source values in a spreadsheet.
`LoadCableCorrectionConstants82`	n/a	Read the cable compensation parameters and sends them to the 590. This module is for internal use by the SIMCVsweep82 and CTsweep82 modules. It is not normally used as a stand-alone module.
`QTsweep82`	`qtsweep`	Performs a quasistatic measurement sweep.
`SaveCableCompCaps82`	`save-cap-file` `savecablecompfile`	Saves entered capacitance source values in a file.
`SIMCVsweep82`	`system82-cvsweep` `cvsweep`	Performs simultaneous C-V sweep.

LS336ulib user library

The LS366ulib provides user modules that control the Lake Shore Cryotronics 336 Temperature Controller.

LS336ulib user modules

User module	Description
heaterOff	Turns off heater 1 and heater 2 and disables setpoint ramping on both outputs.
setDelay_Dialog	Either displays a window that contains the message you specified and an OK button, or performs the delay set by WaitTime.
setSweepParams	Generates the list of temperatures used by setTemp when the setTemp parameter FlagMode is set to 1. When active, it calculates the temperature profile to be measured from the start temperature, stop temperature, and step points input. It outputs the temperature list to the file.
setTemp	Controls key aspects of the temperature controller, including setpoint, heater parameters, and ramp rates, to allow variable temperature electrical measurements. This routine is designed to function inside a subsite cycle test with the heaterOff and setSweepParam routines.

Matrixulib user library

The Matrixulib connects instrument terminals to output pins using a Keithley Series 700 Switching System. It is for use with switching systems that are configured as a general purpose, low current, or ultra-low current matrix.

Matrixulib user module

User module	Description
ConnectPins	Allows you to control your switching matrix.

MultiSegmentSweep_ulib user library

The MultiSegmentSweep_ulib contains two user modules that let you run up to a four-segment current or voltage linear sweep.

These modules are only supported for the 4200-SMU and 4210-SMU instruments.

MultiSegmentSweep_ulib user modules

User module	Description
MultiSegmentSweepI	This module runs up to a four-segment current linear sweep.
MultiSegmentSweepV	This module runs up to a four-segment voltage linear sweep.

nvm user library

The nvm user library contains user modules that are used for nonvolatile memory tests that use a source-measure and a pulse measure unit. The following table lists and briefly describes the user modules.

For additional detail on working with user modules in the nvm user library, refer to the application note “Pulse I-V Characterization of Non-Volatile Memory Technologies.”

For detail on creating a custom user module for nonvolatile memory tests, refer to the read me file in the directory `C:\s4200\kiuser\usrlib\nvm`.

nvm user modules

User module	Description
dcSweep	Applies a long signal, either positive or negative. You can specify the rise time, the slew rate, and the time to hold the voltage at the top or bottom.
doubleSweep	Creates a waveform that consists of two voltage sweeps: 0 to V1, V1 to 0, 0 to V2 and V2 to 0. The sweeps are generated on PMU1CH1. Channel PMU1CH2 is kept at 0 V and measures current and charge.
doubleSweepSeg	Creates a waveform that consists of two voltage sweeps: 0 to V1, V1 to 0, 0 to V2 and V2 to 0. The sweep is generated on PMU1CH1. Channel PMU1CH2 is kept at 0 V and measures current and charge.
flashEndurance	Defines pulse sequences for the program/erase, program, and erase pulses. It runs the program/erase sequence a defined number of times by logarithmically spaced numbers of loops. After each iteration, it does program and erase one more time with Vt extraction after each operation.
flashProgramErase	Defines waveform for Programming and Erasing pulse for both drain and gate.
getRes2	This function returns the resistance of a two-terminal resistor. Voltage v_force is forced on the top side of the device; 0 V is forced to the low side. Measure current and reports resistance (V/I).
pramEndurance	Runs an endurance test for a PRAM. It runs iterations with a logarithmically spaced number of SET/PULSE loops. Reports DUT resistance after SET/RESET pulse. Also returns the amplitude of the SET current.
pramSweep	This function characterizes PRAM devices and produces RI/RV data. A sequence of SET and RESET pulses, followed by the MEASURE pulses, sets and resets the PRAM DUT.
pulse_test	Performs pulse testing according to the definition in the nonvolatile memory structure. This function handles all PMU communications and does all nonvolatile memory pulse testing.
pundEndurance	This routine performs a device endurance test that runs fatigue pulse trains in between multiple PUND tests. A preliminary PUND test measurement is taken (iteration of 0), followed by the fatigue voltage pulse train. The PUND test is composed of a 17-segment voltage pulse waveform, with two positive pulses to a user-specified Vp followed by two negative pulses to −Vp. Each PUND test calculates P, Pa, U, Ua, N, Na, D, Da, Psw, and Qsw. The fatigue pulse train is made by looping through a 9-segment voltage pulse waveform with one positive pulse to a user-specified V_{FAT} and one negative pulse to $-V_{FAT}$. The number of times this waveform is repeated between each PUND test is determined by the specified number of loops divided logarithmically by the total number of fatigue pulse trains.

nvm user modules

User module	Description
`pundTest`	This routine performs a pulse V waveform PUND test for FRAM, measuring the full voltage and current waveforms. It also calculates P, Pa, U, Ua, N, Na, D, Da, Psw, and Qsw. The PUND test is composed of a 17-segment voltage pulse waveform with two positive pulses from 0 V to user-specified Vp, followed by two negative pulses to −Vp.
`reramEndurance`	The `reramEndurance` routine performs a series of double sweeps using the same parameters used for the single sweeps, as described in the `reramSweep` routine.
`reramForming`	This routine slowly ramps a voltage to a specified value while measuring the current constantly to see if the device has formed.
`reramFormingCV`	This routine slowly ramps a voltage to a specified value while measuring the current constantly to see if the device has formed.
`reramSweep`	The `reramSweep` sweep performs a double sweep with a flat section at the peak of each sweep. To test a ReRAM device, choose appropriate values for the two peaks, either positive or negative, and then set the timing you would like to implement.
`vt_ext`	This function returns the transistor threshold voltage using the maximum Gm method.

OVPControl user library

The user module in the OVPControl user library allows you to set the maximum voltage of the SMU.

SetOVPLevel user module

User module	Description
`SetOVPLevel`	Sets the 4200-SMU, 4201-SMU, 4210-SMU, or 4211-SMU overvoltage protection (OVP) maximum voltage. The OVP is an analog circuit that limits the SMU voltage output regardless of what the SMU is sourcing and measuring. Depending on the voltage that the OVP is set to, the SMU clamps the output voltage to one of the built-in voltage limits.

parlib user library

The `parlib` extracts device parameters on bipolar-junction transistors and MOSFETs. Extracted parameters include Beta, resistance, threshold voltage, and V_{ds}-I_d sweeps and V_{gs}-I_d sweeps for MOSFETs.

parlib user modules

User module	Description
`beta`	Measure beta of bipolar transistor at the specified IE and VCB.
`fnddat`	Find data based on an x search or a y search.
`fndtrg`	Decide if TRIGL or TRIGH should be used.
`gamma`	Returns the value of the body-effect parameter gamma obtained from two measurements of the threshold voltage at different substrate bias voltages.
`gm`	Estimate FET conductance (dId/dVg) at V_{ds} and V_{gs}.
`gummel`	This test makes measurements that are similar to the `gummel` test in the Demo project for 3-terminal pnp BJTs.
`igvg`	This test makes measurements that are similar to the `ig-vg` test in the Demo project for 4-terminal MOSFETs.

parlib user modules

User module	Description
vceic	This test makes measurements that are similar to the vce-ic test in the Demo project for 3-terminal BJTs.
vdsid	This test makes measurements that are similar to the vds-id test in the Demo project for 4-terminal MOSFETs.
vdsid_step	This test is similar to vdsid, but it allows you to use more than one step.
vgsid1	This test makes measurements that are similar to the vgs-id test in the Demo project for 4-terminal MOSFETs.
vtext	Returns the value of the extrapolated threshold voltage from multiple linear least square fits to the gate characteristics of a FET in the nonsaturated region.

pmuCompulib

The user modules in this library are used to collect and select offset current compensation data.

pmuCompulib user modules

User module	Description
pmu_Offset_Current_Comp	Collects offset current compensation data for both channels of the 4225-PMU.

pmuulib user library

The pmuulib user library contains user modules for configuring the 4225-RPM for the designated PMU channel. The following table lists and briefly describes the user modules.

pmuulib user modules

User module	Description
RPM_configure	This user module configures the 4225-RPM for the designated PMU channel.
RPM_switch	This user module was deprecated. Use the LPT command rpm_config for any RPM mode switching.

PMU_examples_ulib user library

The user modules in this library are used in the pmu-dut-examples project.

The user module in the OVPControl user library (on page 10-15) allows you to set the maximum voltage of the SMU.

PMU_examples_ulib user modules

User module	Description
`PMU_10ns_Pulse_Example`	This module sets up the PMU to continuously output pulses with 10 ns pulse widths. The PMU is in standard pulse mode with the pulse levels set at −1 V and 1 V.
`PMU_1Chan_Sweep_Example`	This module is a functional programming reference to illustrate the basic commands necessary to perform a pulse I-V (2-level pulse) sweep. It returns voltage and current spot means for pulse amplitude and base by doing a voltage amplitude pulse I-V sweep using one channel of the 4225-PMU.
`PMU_1Chan_Waveform_Example`	This module is a functional programming reference to illustrate the basic commands necessary to perform a pulse I-V (2-level pulse) sweep with waveform capture. It captures a voltage amplitude pulse I-V waveform using one channel of the 4225-PMU. It returns voltage and current samples versus time for a single channel.
`PMU_IV_sweep_Example`	This module is a functional programming reference to illustrate the basic LPT commands that are needed to perform a single Vd-Id sweep. This module performs a voltage amplitude pulse I-V sweep using two channels of a single 4225-PMU. One channel sweeps (drain) while the other uses a fixed pulse amplitude (gate).
`PMU_IV_sweep_step_Example`	This module is a functional programming reference to illustrate the basic LPT commands necessary to do a Vd-Id family of curves. This module performs multiple voltage amplitude pulse I-V sweeps using two channels of a single 4225-PMU card. One channel sweeps (drain), while the other uses a fixed pulse amplitude (gate) for each sweep.
`PMU_PulseWaveform_FileSave_Example`	This module allows for a long pulse or time capture (40 s pulse width maximum, 120 s total waveform capture) of an entire pulse to a `.csv` file using both channels of a single 4225-PMU and the Segment Arb mode. In addition to optionally saving the waveform to a file, a time-averaged version is available in the Analyze sheet.
`PMU_SMU_Sweep_Example`	This user module is an example of how to use the PMU with a SMU. For example, you could use this module to compare performing a test using a PMU to performing that test with a SMU. This user module is based on the module `PMU_IV_Sweep_Example`.
`PMU_ScopeShot_Example`	Pulse I-V waveform capture using two channels of a single 4225-PMU. The gate channel outputs a pulse train (no change in pulse base or amplitude) while the drain channel outputs a swept pulse amplitude.
`PMU_SegArb_4ch`	This module configures multi-sequence, multi-segment waveform generation (Segment Arb) on four channels using two 4225-PMU cards. It measures and returns either waveform (V and I versus time) or spot mean data for each segment that has measurement enabled. It also provides a voltage bias by controlling up to four SMUs. The SMUs must not be connected to a 4225-RPM. This routine is similar to `PMU_SegArbExampleB`, except that this routine adds multiple Segment Arb sequences with 3 to 2048 segments. Each sequence can be looped to make a more complicated Segment Arb Waveform.

PMU_examples_ulib user modules

User module	Description
PMU_SegArb_6ch	This module configures multi-sequence, multi-segment waveform (Segment ARB) generation on four channels using two 4225-PMU cards. It measures and returns either waveform (V and I vs time) or spot mean data for each segment that has measurement enabled. If the system includes 4200-SMUs, 4201-SMUs, 4210-SMUs, or 4211-SMUs, you can apply voltage biases from up to four SMUs. The SMUs must not be connected to 4225-RPMs. If a SMU is connected to a 4225-RPM, a −233 error occurs and the test does not run.
PMU_SegArb_8ch	This module configures multi-sequence, multi-segment waveform (Segment Arb) generation on eight channels using four 4225-PMU cards. It measures and returns either waveform (V and I versus time) or spot mean data for each segment that has measurement enabled. If the system includes 4200-SMUs, 4201-SMUs, 4210-SMUs, or 4211-SMUs, you can apply voltage biases from up to four SMUs. The SMUs must not be connected to 4225-RPMs. If a SMU is connected to a 4225-RPM, a −233 error occurs and the test does not run.
PMU_SegArb_Example	This module configures multi-segment waveform generation (Segment Arb) on two channels using a single 4225-PMU. It measures and returns the waveform data (V and I compared to time, no spot means).
PMU_SegArb_ExampleB	This module configures multi-segment waveform generation (Segment Arb) on two channels using a single 4225-PMU. It measures and returns either waveform (V and I compared to time) or spot mean data for each segment that has measurement enabled.
PMU_SegArb_ExampleFull	This module configures multi-sequence, multi-segment waveform generation (Segment Arb) on two channels using a single 4225-PMU. It measures and returns either waveform (V and I versus time) or spot mean data for each segment that has measurement enabled. It also provides a voltage bias by controlling one SMU. The SMU must not be connected to a 4225-RPM.
PM_SegArb_Multi_ch	This test configures multi-sequence, multi-segment waveform (Segment Arb waveform) generation on four channels using two 4225-PMUs. It measures and returns either waveform (V and I versus time) or spot mean data for each segment that has measurement enabled. If the system includes 4200-SMUs, 4201-SMUs, 4210-SMUs, or 4211-SMUs, you can apply voltage biases from up to four SMUs. The SMUs must not be connected to 4225-RPMs. If a SMU is connected to a 4225-RPM, a −233 error occurs and the test does not run.

PMU_freq_time_ulib user library

The user modules in this library are used to take evenly spaced measurements with the PMU for use with fast Fourier transform (FFT) computations.

PMU_freq_time_ulib user modules

User module	Description
PMU_Waveform	Outputs a defined number of pulsed waveforms from channel 1 of the PMU and measures the resulting current with channel 2. The user inputs the test frequency and the 2^N number of cycles and 2^N number of points per period. The total number of points in the test is then 2^N cycles × 2^N number of points per period.
PMU_sampleRate	Biases constant voltage using PMU channels 1 and 2 and measures the voltage on channel 1 and the current on channel 2. The number of samples returned depends on the entered sampling time and sample rate
PMU_SMU_sampleRate	Outputs a bias on a single channel of a PMU and a single SMU with connections to the ground unit of the 4200A-SCS. The maximum voltage bias for the SMU is ±10 V. The channel of the PMU measures the voltage and current. The total number of samples returned to the sheet can be calculated by SampTime × SampRate. The number of samples must be less than 65534.

PMU_PCRAM_ulib

The user modules in the `PCM_PCRAM_ulib` library provide examples of how the PMUs can be implemented in the characterization of PRAM elements.

PMU_PCRAM_ulib user modules

User module	Description
pram_pulse_ilimit	Simplifies the generation of segments when using the PMU. Forced voltage and current values are collected from the ForceCh and MeasureCh channels.
pram_sweep_ilimit	Provides an example of how the PMUs can be implemented in the characterization of PRAM elements. It allows specification of four pulses in one waveform. The parameters of these pulses are determined by the user and the SET pulse current values can be swept to generate RI and IV charts. It also demonstrates output debug information on voltage and currents for both PMU channels for any iteration of the sweep.
pram_sweep	Provides an example of how the PMUs can be implemented in the characterization of PRAM elements. It allows specification of four pulses in one waveform. The parameters of these pulses are determined by the user and the SET pulse amplitude can be swept to generate RI and IV charts. It also demonstrates output debug information on voltage and currents for both PMU channels for any iteration of the sweep.

PRBGEN user library

The PRBGEN user library provides test modules to initialize the prober, move to the next site or subsite in the wafer map of the prober, make or break contact between the probes and the wafer, and get the X position and Y position of the prober. It allows Clarius to control all supported probers in the same manner. Clarius projects that use PRBGEN work with any prober supported by Keithley.

The user modules in the PRBGEN user library are provided as actions in Clarius.

PRBGEN user modules

User module	Clarius action	Description
PrChuck	prober-contact	Directs the prober to have the probe pins contact the wafer or separate the pins from the wafer.
PrInit	prober-init	Initializes the prober with die size, first coordinate (X and Y), units (mm or mils), and mode information.
PrMovNxt	prober-move	In learn mode, the PrMovNxt command causes the prober to move to the next site after inking.
PrSSMovNxt	prober-ss-move	In learn mode, the PrSSMovNxt command causes the prober to move to the next subsite after inking.

QSCVulib user library

The QSCVulib user library provides a user module to do quasistatic C-V sweeps.

QSCVulib module

User module	Clarius test	Description
force_current_CV	sic-moscap-force-i-qscv	This test performs forced current quasistatic C-V measurements on a silicon carbide (SiC) power MOS capacitor using a single SMU with a preamplifier. The HI of the SMU is connected to the Gate terminal and SMU LO is connected to the substrate of the device. While a constant current is forced, voltage versus time is measured in real time and the forward and reverse C-V curves are derived and plotted.
	sic-mosfet-force-i-qscv	This test performs forced current quasistatic C-V measurements on a silicon carbide (SiC) power MOSFET using a single SMU with a preamplifier. The HI of the SMU is connected to the Gate terminal and the LO is connected to the Drain and Source terminals tied together. While a constant current is forced, voltage versus time is measured in real-time and the forward and reverse C-V curves are derived and plotted.
meas_qscv	ramprate-cvsweep	This test uses two SMUs with preamplifiers to do a quasistatic C-V sweep. The 4200-PA Preamplifiers are required because this test sources and measures current in the picoamp range. The SMUs source current to charge the capacitor and measure the voltage, time, and discharge current.

RPM_ILimit_Control user library

The RPM_ILimit_Control user library provides a user module for short-term calibration of the 4225-RPM current clamp. It also provides user modules that support the calibration user module, but which should not be set individually.

RPM_ILimit_Control user modules

User module	Description
Do_RPM_ILimit_Cal	Performs a short-term calibration of the current clamp of properly-equipped 4225-RPMs.
Get_RPM_ILimit_DAC_Value	Do not set individually. This is used by Set_RPM_ICompliance.
isLimitSupported	Do not set individually. Used by Do_RPM_ILimit_Cal, Get_RPM_ILimit_DAC_Value, Set_RPM_ICompliance, and OpenLimit.
OpenLimit	Do not set individually.
Set_RPM_ICompliance	Do not set individually.

utilities_ulib

The utilities_ulib user library provides a user module to add delays.

utilities_ulib user module

User module	Description
Delay_second	Enter delay time in seconds.

van der Pauw user library

The vdpulib user library contains user modules for measuring the surface resistivity and volume resistivity of semiconductor material using the van der Pauw (vdp) technique.

vdpulib user modules

User module	Description
hall_coefficient	Determines the Hall coefficient (RH) and mobility (µH) of a material using four SMUs.
hall_coefficient_cviv	Determines the Hall coefficient (RH) and mobility (µH) of a material using four SMUs and using the 4200A-CVIV.
resistivity_surface	Measures surface resistivity using four SMUs.
resistivity_surface_cviv	Measures surface resistivity using four SMUs and using the 4200A-CVIV.
resistivity_volume	Measures volume resistivity using four SMUs.
resistivity_volume_cviv	Measures volume resistivity using four SMUs and using the 4200A-CVIV.

VLowFreqCV user library

The `VLowFreqCV` user library contains user modules that are used for very low frequency C-V characterization. The following list briefly describes the user modules.

VLowFreqCV user modules

- `vlfcv_measure`
  Makes a single C-V measurement using two SMUs connected to the device under test (DUT).
- `vlfcv_measure_dual_sweep_bias`
  Performs C-V characterization at multiple DC bias values. This module allows dual sweep, sweeping from a start to stop bias, with one measure point at the stop point before sweeping back to the start point.
- `vlfcv_measure_dual_sweep_bias_fixed_range`
  Performs C-V characterization at multiple DC bias values. This module allows dual sweep, sweeping from the start point to the stop point, with one measure point at the stop point before sweeping back to the start point. It uses a fixed measure range for the SMU for the entire voltage bias sweep. The routine uses the maximum DC bias voltage, `expected_C` and `expected_R` to determine the maximum current for the test and uses this current to set the current measure range for the test.
- `vlfcv_measure_sweep_bias`
  Performs C-V characterization at multiple DC bias values. It makes the same measurements as `vlfcv_measure`, but allows you to make measurements at each point of a linear sweep of the DC bias voltage.
- `vlfcv_measure_sweep_bias_fixed_range`
  Performs C-V characterization at multiple DC bias values. This routine performs the same measurements as `vlfcv_measure`, but allows you to make measurements at each point of a linear sweep of the DC bias voltage. This routine is also similar to `vlfcv_measure_sweep_bias`, except that it uses a fixed current measure range on for the SMU sense for the entire voltage bias sweep. The routine uses the maximum DC bias voltage, `expected_C` and `expected_R` to determine the maximum current for the test and uses this current to set the current measure range for the test.
- `vlfcv_measure_sweep_freq`
  Performs C-V characterization at multiple frequency values. It makes the same measurements as `vlfcv_measure`, but allows you to make measurements at each point of a list sweep of the test frequency.
- `vlfcv_measure_sweep_time`
  Performs C-V characterization a specified number of times, creating a C versus time graph.

wbg_ulib user library

The `wbg_ulib` user library provides user modules that use the JEDEC JEP183A standard for measuring threshold voltage on silicon carbide MOSFETs.

wbg_ulib user modules

User module	Description
`SMU_vgsweepvdfixed`	Derives the threshold voltage (V_{TH}) of a SiC MOSFET using a method based on the JEDEC JEP183A standard. With a user-defined Target Current ($I_{dTarget}$), V_{TH} is found from an I_D-V_G curve while sweeping V_G and biasing V_D.
`SMU_vgvdsweep`	Derives the threshold voltage (V_{TH}) of a SiC MOSFET using a method based on the JEDEC JEP183A standard. With a user-defined Target Current ($I_{dTarget}$), V_{TH} is found from an I_D-V_G curve while sweeping both V_G and biasing V_D ($V_G = V_D$).

Winulib user library

The `Winulib` user library provides user interface routines for operator inputs and prompts, such as abort, retry, and ignore decision prompts.

User Module:

Winulib user modules

User Module	Clarius Action Name	Description
AbortRetryIgnoreDialog	`abortretryignoredialog`	This user module creates a dialog with Abort, Retry, and Ignore decision prompts.
InputOkCancelDialog	`inputokcanceldialog`	This user module creates a dialog that can prompt for up to four input parameters.
OkCancelDialog	`okcanceldialog`	This user module creates a dialog that provides OK or Cancel decisions.
OkDialog	`okdialog`	This user module creates a dialog that pauses the test sequence to make an announcement (for example, "Test finished") or prompt for an action (for example, connection change).
RetryCancelDialog	`retrycanceldialog`	This user module creates a dialog that presents Retry or Cancel decisions.
YesNoCancelDialog	`yesnocanceldialog`	This user module creates a dialog that contains up to four lines of text and Yes, No, or Cancel decisions.
YesNoDialog	`yesnodialog`	This user module creates a dialog that contains up to four lines of text and Yes and No buttons.

wlrlib user library

The user modules in the `wlrlib` user library run linear regression and charge-to-breakdown (Q_{BD}) ramp tests for wafer-level reliability (WLR) testing. These user modules are summarized in the following table.

wlrlib user modules

User module	Description
`llsq1`	Performs simple linear regression.
`qbd_rmpv`	Performs a charge-to-breakdown test using the QBD V-ramp test.
`qbd_rmpj`	Performs a charge-to-breakdown test using the QBD J-ramp test.

For more information, refer to Wafer-Level Reliability Testing (on page 11-1).

Section 11

Wafer-level reliability testing

In this section:

JEDEC standards

NOTE

The following descriptions for the JESD28-A and JESD35-A standard procedures were acquired from the JEDEC website. This is JEDEC copyright-protected material. The JEDEC standard procedures are available on the JEDEC website (jedec.org). Registration is free, but you must register before you can access the standards.

JESD28-A
Published: Dec-2001

A Procedure for Measuring N-Channel MOSFET Hot-Carrier-Induced Degradation Under DC Stress

This document describes an accelerated test for measuring the hot-carrier-induced degradation of a single n-channel MOSFET using DC bias. The purpose of this document is to specify a minimum set of measurements so that valid comparisons can be made between different technologies, IC processes, and process variations in a simple, consistent, and controlled way. The measurements specified should be viewed as a starting point in the characterization and benchmarking of the transistor manufacturing process.

JESD35-A
Published: Apr-2001

Procedure for Wafer-Level Testing of Thin Dielectrics

This document is intended for use in the MOS Integrated Circuit manufacturing industry fabrication processing and test and describes procedures developed for estimating the overall integrity and reliability of thin gate oxides. Three basic test procedures are described: the voltage-ramp (V-Ramp), the current-ramp (J-Ramp), the current-ramp (J-Ramp), and the constant current (Bounded J-Ramp) test. Each test is designed for simplicity, speed, and ease of use.

Introduction

This section provides information on wafer-level reliability (WLR) testing. Included are tests for:

- Hot-carrier injection (HCI)
- Negative-bias temperature instability (NBTI)
- Electromigration
- Charge-to-breakdown measurement (QBD)

AC, or pulsed, stress is a useful addition to the typical stress-measure tests for investigating both semiconductor charge trapping and degradation behaviors. NBTI and time-dependent dielectric breakdown (TDDB) tests consist of stress / measure cycles.

The applied stress voltage is a DC signal, which is used because it maps more easily to device models. Incorporating pulsed stress testing provides additional data that permits a better understanding of device performance in frequency-dependent circuits.

The test pulse stresses the device for HCI, NBTI, and TDDB test instead of DC bias by outputting a train of pulses for a period (stress time). Pulse characteristics are not changed during the stress-measure test. The test then uses SMUs to measure device characteristics such as V_{th} and G_m.

This section includes background information on HCI degradation and summaries for using 4200A-SCS projects to measure HCI degradation and other WLR tests.

NOTE

The projects for HCI and QBD testing comply with the standard procedures established by JEDEC. In 4200A-SCS documentation, all references to the JEDEC standards and duplicated JEDEC documentation are clearly indicated as JEDEC copyright-protected material.

HCI and WLR projects

The 4200A-SCS projects for HCI and WLR testing include:

- `hci-1-dut`
- `hci-4-dut`
- `nbti-1-dut`
- `em-const-i`
- `qbd`

All these projects except `qbd` use subsite cycling in the stress/measure mode. For details, see Subsite cycling (on page 9-33).

You can use each of these projects as configured or modify them for your testing requirements.

Hot Carrier Injection projects

The Hot Carrier Injection (HCI) projects determine HCI on MOSFETs. The `hci-1-dut` project determines HCI degradation on a single 4-terminal n-MOSFET. The `hci-4-dut` project determines HCI degradation on two 4-terminal n-MOFETs and two 4-terminal p-MOSFETs.

The `hci-1-dut` project is shown in the following figure.

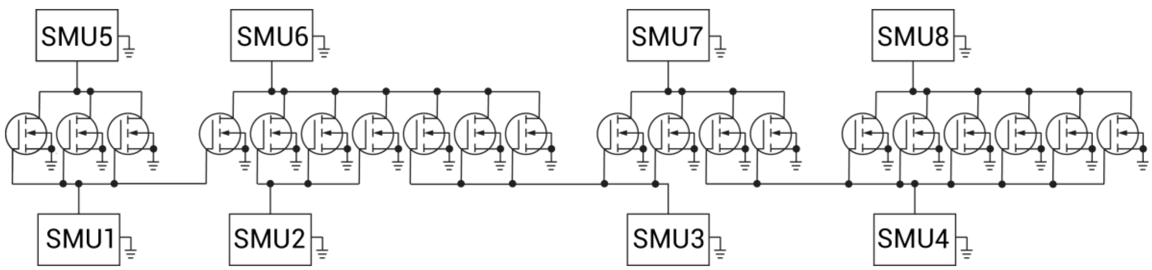
Figure 200: Project tree showing hci-1-dut project

For the `hci-1-dut` project, the `hci` subsite is set up for subsite cycling using voltage stressing on the single n-channel MOSFET device (`4terminal-n-fet`). After the first pre-stress cycle to perform characterization tests, subsequent cycles voltage stress the device for a specified time before repeating the tests.

The `hci-4-dut` project is similar to the `hci-1-dut` project except that it is configured to test four devices using a switching matrix for connections.

In a parallel connection scheme, up to 20 devices can be stressed by voltage. The following figure shows an example of 20 parallel-connected devices being stressed by eight gate and drain voltages.

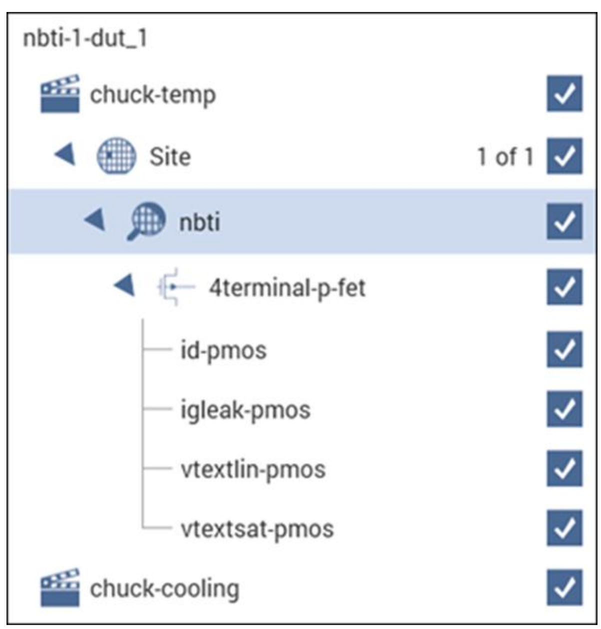
Figure 201: HCI and NBTI tests: 20 parallel-connected devices stressed by voltage



Negative Bias Temperature Instability project

The Negative Bias Temperature Instability (`nbti-1-dut`) project performs NBTI testing on a p-MOSFET with temperature and DC stress. The following figure shows the project tree when the `nbti-1-dut` project is selected.

Figure 202: Project tree for nbti-1-dut

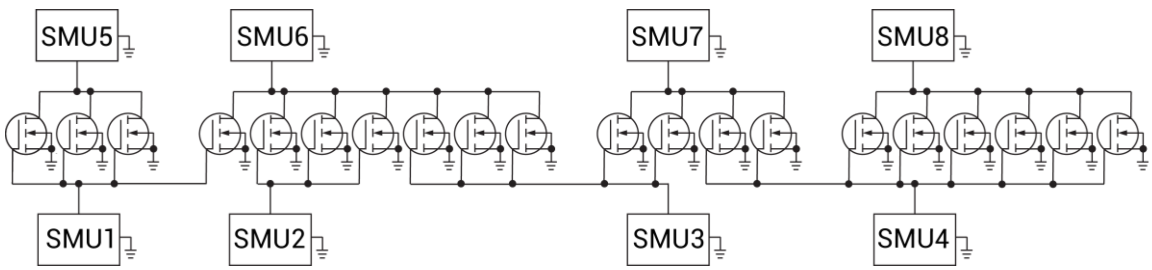


The `nbti` subsite is configured for subsite cycling using voltage stressing for a p-channel MOSFET (PMOS) device.

This project includes actions that control the temperature of the chuck. The subsite test will not start until the chuck reaches the specified temperature. After the first pre-stress cycle to characterize the device, subsequent cycles voltage stress the device for a specified time before repeating the tests. After the subsite cycling is complete, the `chuck-cooling` action cools the chuck.

In a parallel connection scheme, up to 20 devices can be stressed by voltage. The following figure shows an example of 20 parallel-connected devices being stressed by eight gate and drain voltages.

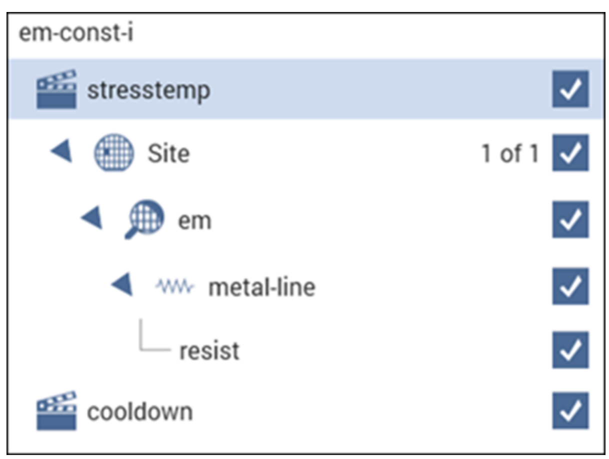
Figure 203: HCI and NBTI tests: 20 parallel-connected devices stressed by voltage



Electromigration project

The Electromigration project (`em-const-i`) is shown in the following figure.

Figure 204: em-const-i project tree



The subsite (`em`) is configured for subsite cycling using current stressing on a single device (`metal-line`).

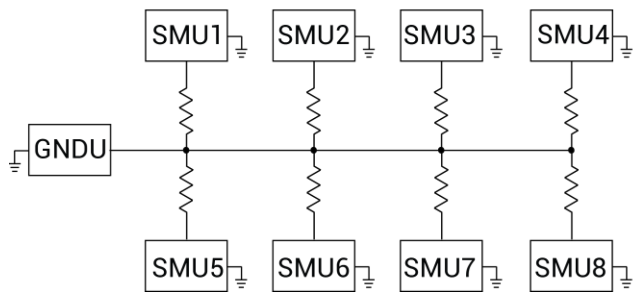
This project includes actions to control the temperature of the chuck. The subsite will not start cycling until the chuck reaches the specified temperature. After the first pre-stress cycle to perform a characterization test on the device, subsequent cycles current stress the device for a specified time before repeating the test. After the subsite completes, the `cooldown` action cools the chuck.

You can modify the `em-const-i` project to test additional devices. Each SMU in the test system can current-stress one device. Therefore, if there are eight SMUs in the test system, you can stress up to eight devices can be stressed, as shown in the following figure.

NOTE

For current stressing, when setting the current stress level for each device in the subsite, a setting of zero (0) connects the device pin to the ground unit (0 V ground). To current stress a device, the current stress level must be set to a nonzero value.

Figure 205: EM test: Eight devices being current stressed by eight SMUs

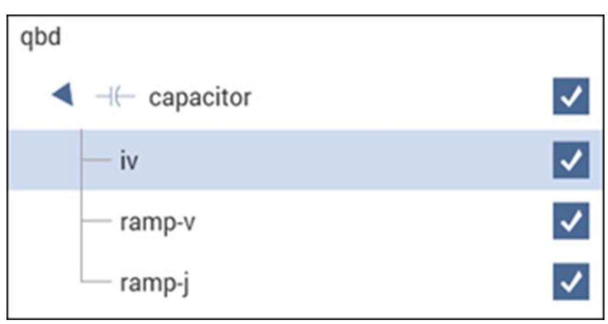


Charge-to-Breakdown Test of Dielectrics project

The `qbd` project includes tests for the `ramp-v` test and the `ramp-j` test. These tests adhere to the JESD35-A standard procedures for wafer-level testing of thin dielectrics. This project does not use subsite cycling.

Details on these tests are described in V-ramp and J-ramp tests (on page 11-8).

Figure 206: qbd project tree



HCI degradation: Background information

Hot-carrier injection (HCI) degradation is one of the most important device issues facing the semiconductor industry. Small gate length and process variations in the semiconductor process can result in dramatic degradation in HCI device performance. In the last few years, HCI lifetimes have reduced dramatically. In some cases, drive current lifetimes have dropped from years to weeks. HCI effects are enhanced with device scaling (this includes a reduction in device gate length). This means that HCI effects will be an even greater concern in the future. The need to monitor HCI on a regular basis is a critical test requirement.

Hot-carrier damage occurs in MOS devices when carriers (electrons or holes) are accelerated in the channel. In short channel devices, these carriers attain velocities high enough to cause impact ionization. Impact ionization, in turn, creates extra carriers in the MOS channel. These extra carriers result in significant substrate currents and in some cases attain high enough energy to overcome the semiconductor-oxide barrier and are trapped in the oxide. Most of the oxide carrier trapping occurs at the drain edge where carrier velocity is maximized. These trapped channel electrons can cause significant device performance asymmetry and shifts in critical device parameters such as threshold voltage and device drive current. In some cases, as much as a 10% change in measured device parameters can occur within a few days.

The devices of today are increasingly susceptible to hot-carrier effects. In the past, the linear drain current target value for successful hot-carrier device performance was a 10% change in 10 years. Typically, manufactured devices can no longer meet this specification and as much as 10% degradation in linear drain current can occur in a few days.

Configuration sequence for subsite cycling

The following projects use subsite cycling:

- `hci-1-dut`
- `hci-4-dut`
- `nbti-1-dut`
- `em-const-i`

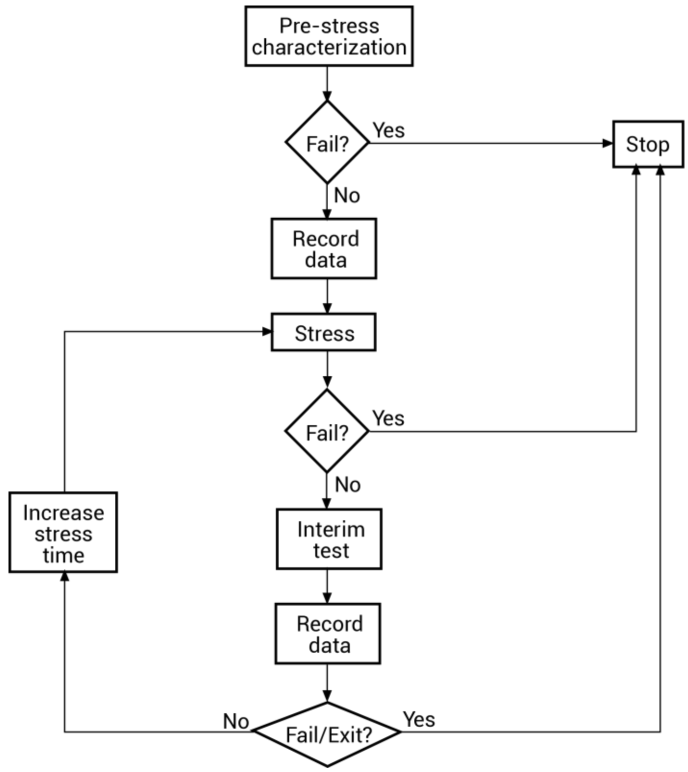
The process flow for these projects is shown in the following figure.

NOTE

You can create a new project for subsite cycling or you can use one of the existing projects as a starting point and change it as needed. For details, see Set up a simple project (on page 2-1).

To configure the subsite for subsite cycling, refer to Configure Subsite Cycling (on page 9-4).

Figure 207: Process flow HCI/NBTI/constant current EM



V-ramp and J-ramp tests

Charge-to-breakdown measurement (Q_{BD}) tests are a measure of time-dependent gate oxide breakdown. They are a standard method used to determine quality of gate oxides in MOS devices.

The V-ramp test starts at the use-condition voltage (or lower) and ramps linearly from this value until oxide breakdown. The J-ramp starts at a low current and ramps exponentially until oxide breakdown.

User modules for these tests are provided in the `wlrlib` user library. The user modules in the `wlrlib` user library run linear regression and charge-to-breakdown (Q_{BD}) ramp tests for wafer-level reliability (WLR) testing. These user modules are summarized in the following table.

wlrlib user modules

User module	Description
`llsq1`	Performs simple linear regression.
`qbd_rmpv`	Performs a charge-to-breakdown test using the QBD V-ramp test.
`qbd_rmpj`	Performs a charge-to-breakdown test using the QBD J-ramp test.

V-ramp test: qbd_rmpv User Module

The V-ramp test uses the `qbd_rmpv` user module of the `wlrlib` user library.

Usage

See the JEDEC standard JESD35-A, *Procedure for Wafer-Level Testing of Thin Dielectrics*, available at jedec.org.

NOTE

Some of the descriptions of the following variables are quoted from the JESD35-A standard. The variables quoted from the standard include this reference identification: (Ref. JESD35-A).

```
status = qbd_rmpv(int hi_pin, int lo_pin1, int lo_pin2, int lo_pin3, char *HiSMUId, char
    *LoSMUId1, char *LoSMUId2, char *LoSMUId3, double v_use, double I_init, int
    hold_time, double v_start, double v_step, int t_step, int measure_delay, double
    I_crit, double I_box, double I_max, double exit_curr_mult, double exit_slope_mult,
    double q_max, double t_max, double v_max, double area, int exit_mode, double
    *V_stress, int V_size, double *I_stress, int I_size, double *T_stress, int T_size,
    double *q_stress, int q_size, double *I_use_pre, double *I_use_post, double *Q_bd,
    double *q_bd, double *v_bd, double *I_bd, double *t_bd, double *v_crit, double
    *v_box, int *failure_mode, int *test_status);
```

Input variables

status	Returned values are placed in the Analyze sheet
hi_pin	High pin (usually the gate pin) (−1 to 72); enter −1 to not connect
lo_pin1 *lo_pin2* *lo_pin3*	Usually for source drain and substrate connection; depending on device structure, some of those pins are optional; enter −1 to not connect
HiSMUId	ID string of the SMU outputting the stress
LoSMUId1 *LoSMUId2* *LoSMUId3*	ID string of the SMU connected to ground terminal; these three IDs can be same
v_use	Oxide voltage (V) under normal operating conditions; typically, the power supply voltage of the process; this voltage is used to measure pre- and post-voltage ramp oxide current (Ref. JESD35-A)
I_init	Oxide breakdown failure current when biased at *v_use*; see **Details**
hold_time	Time in ms to hold the first stress (*v_start*)
v_start	Starting voltage (V) for voltage ramp; typical value is *v_use* (Ref. JESD35-A)
v_step	Voltage (V) ramp step height; maximum of 0.1 MV/cm; refer to **Details**
t_step	Voltage ramp step time in ms, used to determine the voltage ramp rate; should be less or equal than 100 ms (typically 40 ms to 100 ms)
measure_delay	Time delay in ms for measurement after each voltage stress step; should be less than *t_step* (ms)
I_crit	At least 10 times the test system current measurement noise floor; this oxide current (A) is the minimum value used in determining the change of slope breakdown criteria (Ref. JESD35-A)

I_box	An optional measured current level for which a stress voltage is recorded; this value provides an additional point on the current-voltage curve; a typical value is 1 µA (Ref. JESD35-A)
I_max	Oxide breakdown criteria; *I_bd* is obtained from I-V curves and is the oxide current at the step just prior to breakdown (Ref. JESD35-A)
exit_curr_mult	Change of current failure criteria; this is the ratio of measured current over previous current level, which, if exceeded, will result in failure (2.5 to 5, recommended value: 10 to 100)
exit_slope_mult	Change of slope failure criteria; this is the factor of change in FN slope, which, if exceeded, will result in failure (2.5 to 5, recommended value: 3)
q_max	Maximum accumulated oxide charge per oxide area; used to terminate a test where breakdown occurs but was not detected during the test (C/cm^2) (Ref. JESD35-A)
t_max	Maximum stress time allowed in seconds; reaching this limit will result in test to finish (s)
v_max	The maximum voltage limit for the voltage ramp; this limit is specified at 30 MV/cm for oxides less than 20 nm thick and 15 MV/cm for thicker oxides; refer to **Details**
area	Area of oxide structure (cm^2)
exit_mode	Failure criteria mode; refer to **Details**
V_size	Size of data array; maximum 65535
I_size	Size of data array; maximum 65535
T_size	Size of data array; maximum 65535
q_size	Size of data array; maximum 65535

Output variables

V_stress	Voltage stress array
I_stress	Measured current array
T_stress	Timestamp array indicating when current is measured
q_stress	Accumulated charge array
I_use_pre	Measured oxide current at *v_use*, before starting the ramp (Ref. JESD35-A)
I_use_post	Measured oxide current at *v_use*, after the ramp finished (Ref. JESD35-A)
Q_bd	Charge-to-breakdown; cumulative charge passing through the oxide before breakdown (C) (Ref. JESD35-A)
q_bd	Charge-to-breakdown density (C/cm^2) (Ref. JESD35-A)
v_bd	Applied voltage at the step just before oxide breakdown (Ref. JESD35-A)
I_bd	Measured current at *v_bd*, just before oxide breakdown
t_bd	Timestamp when measuring *I_bd*
v_crit	Applied voltage at the step when the oxide current exceeds *I_crit* (Ref. JESD35-A)
v_box	Applied voltage at the step when the oxide current exceeds *I_box* (Ref. JESD35-A)
failure_mode	▪ Initial test failure ▪ Catastrophic failure (initial test pass, ramp test fail, post test fail) ▪ Masked Catastrophic (initial test pass, ramp test pass, post test fail) ▪ Non-Catastrophic (initial test pass, ramp test fail, post test pass) ▪ Others (initial test pass, ramp test pass, post test pass)
test_status	See **Details**

Details

Performs a charge-to-breakdown test using the QBD V-ramp test algorithm described in JESD35-A, *Procedure for Wafer-Level Testing of Thin Dielectrics*, April 2011. This algorithm forces a linear voltage ramp until the oxide layer breaks down. This algorithm can produce a maximum voltage of ±200 V. The flow diagram for the V-ramp test is shown in V-Ramp Flow Diagram (on page 11-12).

Notes on input variables

`hi_pin` and `lo_pinX`: If there is no switching matrix in the system, enter either 0 or –1 for `hi_pin` and `lo_pinX` to bypass switch.

`I_init`: The typical value of `I_init` is 10 μA/cm^2 and may change depending on oxide area. For maximum sensitivity, the specified value should be well above the worst case oxide current of a good oxide and well above the noise level of the measurement system. Higher values must be specified for ultra-thin oxide because of direct tunneling effects (Ref. JESD35-A).

`v_step`: As an example, the maximum value of `v_step` can be calculated using Tox*0.1 MV/cm, where Tox is in unit of centimeters. This is 0.1 V for a 10 nm oxide (Ref. JESD35-A).

`v_max`: As an example, `v_max` can be estimated from T_{ox}*30 MV/cm, where T_{ox} is in centimeters. This is 35 V for a 10.0 nm Oxide (Ref. JESD35-A).

`exit_mode`: Select:

- `0`: Specifies that oxide failure is determined by a measured current that exceeds the user-specified failure current (`fail_current`)
- `1`: Uses two criteria to determine oxide failure; the first criterion is the specified failure current (`fail_current`); the second criterion is a slope of current measurement that is a factor (`exit_slope_mult`) times the previous measured value; see JEDEC document JESD35-A and Addenda JESD35-1 and JESD35-2

Because of noise considerations, the calculated failure current criterion is used only when the measured current is 10 times the user-specified noise current. For measured currents below this value, the `fail_current` is used as the exit criterion.

Notes on output variables

`test_status`:

- `2`: No test errors (exit due to measured current > a factor of the previous measurement).
- `1`: No test errors (exit due to measured current slope > a factor of the previous slope).
- `0`: No test errors (exit due to measured current > `fail_current` ONLY).
- `1`: Failed pre-stress test.
- `-2`: Cumulative charge limit reached.
- `-3`: Voltage limit reached.
- `-4`: Maximum time limit reached.
- `-5`: Masked Catastrophic Failure.
- `-6`: Non-Catastrophic Failure.
- `-7`: Invalid specified `t_step`, `hold_time`, or `measure_delay`.

NOTE

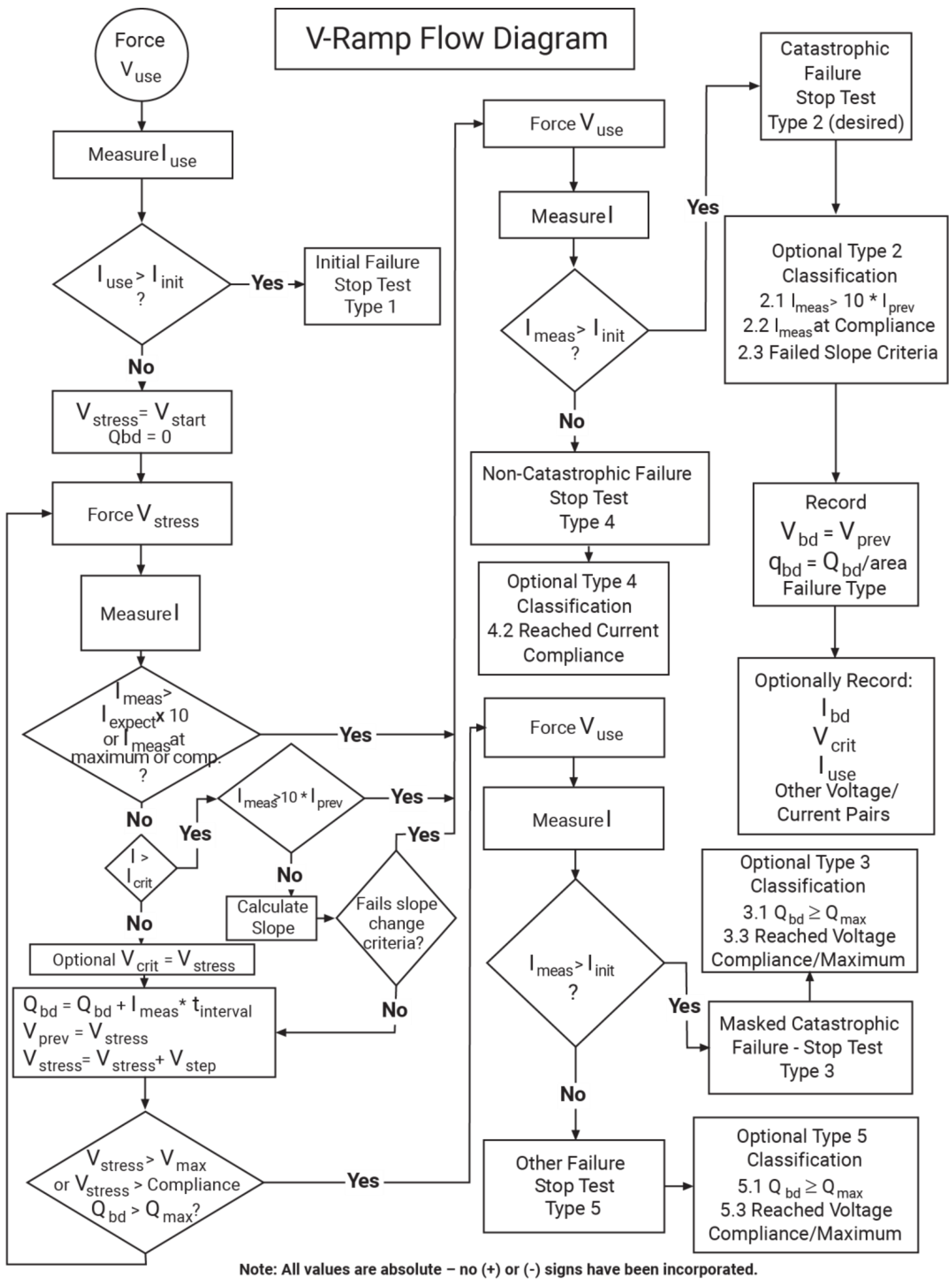
Invalid Test Result − Result = 1e21.

V-Ramp Flow Diagram

NOTE

The following diagram from JESD35-A has been reproduced with permission from JEDEC. The flowchart is from JEDEC.

Figure 208: Detailed V-ramp flow diagram



J-ramp test: qbd_rmpj User Module

The J-ramp test uses the `qbd_rmpj` user module of the `wlrlib` user library.

Usage

```
status = qbd_rmpj(int hi_pin, int lo_pin1, int lo_pin2, int lo_pin3, char *HiSMUId, char
    *LoSMUId1, char *LoSMUId2, char *LoSMUId3, double v_use, double I_init, double
    I_start, double F, int t_step, double exit_volt_mult, double I_max, double q_max,
    double area, double *V_stress, int V_size, double *I_stress, int I_size, double
    *T_stress, int T_size, double *q_stress, int q_size, double *Q_bd, double *q_bd,
    double *v_bd, double *I_bd, double *t_bd, int *failure_mode, int *test_status);
```

Input variables

status	Returned values are placed in the Analyze sheet
hi_pin	High pin (usually the gate pin) (-1 to 72); enter -1 to not connect
lo_pin1 *lo_pin2* *lo_pin3*	Usually for source drain and substrate connection; depending on device structure, some of those pins are optional; enter -1 to not connect
HiSMUId	ID string of the SMU outputting the stress
LoSMUId1 *LoSMUId2* *LoSMUId3*	ID string of the SMU connected to ground terminal; these three IDs can be same
v_use	Oxide voltage (V) under normal operating conditions; typically, the power supply voltage of the process; this voltage is used to measure pre- and post-voltage ramp oxide current (Ref. JESD35-A)
I_init	Oxide breakdown failure current when biased at *v_use*; typical value is 10 μA/cm^2 and may change depending on oxide area; see **Details** (on page 11-14)
I_start	Starting current (A) for current ramp; typical value is *I_init* (Ref. JESD35-A)
F	Current multiplier between two successive current steps (Ref. JESD35-A)
t_step	Current ramp step time (s) (Ref. JESD35-A)
exit_volt_mult	Multiplier factor of successive voltage measurements; when the next measured voltage is below this factor multiplying the previous measured voltage, oxide is considered to be at breakdown and the test will exit; typical value 0.85
I_max	Maximum ramp current (A) (Ref. JESD35-A)
q_max	Maximum accumulated oxide charge per oxide area; used to terminate a test where breakdown occurs but was not detected during the test (C/cm^2) (Ref. JESD35-A)
area	Area of oxide structure (cm^2)
V_size	Size of data array; maximum 65535
I_size	Size of data array; maximum 65535
T_size	Size of data array; maximum 65535
q_size	Size of data array; maximum 65535

Output variables

`V_stress`	Voltage stress array
`I_stress`	Measured current array
`T_stress`	Timestamp array indicating when current is measured
`q_stress`	Accumulated charge array
`Q_bd`	Charge-to-breakdown; cumulative charge (C) passing through the oxide before breakdown (Ref. JESD35-A)
`q_bd`	Charge-to-breakdown density (C/cm^2) (Ref. JESD35-A)
`v_bd`	Applied voltage at the step just before oxide breakdown (Ref. JESD35-A)
`I_bd`	Measured current at `v_bd`, just before oxide breakdown
`t_bd`	Timestamp when measuring `I_bd`
`failure_mode`	▪ Initial test failure ▪ Catastrophic failure (initial test pass, ramp test fail, post test fail) ▪ Masked Catastrophic (initial test pass, ramp test pass, post test fail) ▪ Non-Catastrophic (initial test pass, ramp test fail, post test pass) ▪ Others (initial test pass, ramp test pass, post test pass)
`test_status`	See **Details**

Details

Performs a Charge-to-Breakdown test using the QBD J-ramp test algorithm described in JESD35-A, *Procedure for Wafer-Level Testing of Thin Dielectrics*, April 2011. This algorithm forces a logarithmic current ramp until the oxide layer breaks down. This algorithm can produce a maximum current of ±1 A if a high-power SMU is used. The flow diagram for the V-ramp test is shown in J-ramp flow diagram (on page 11-16).

See JEDEC standard JESD35-A *Procedure for Wafer-Level Testing of Thin Dielectrics*, April 2011, referenced in “Using a Signatone CM500 Prober” in *Model 4200A-SCS Prober and External Instrument Control*.

NOTE

Some of the descriptions of the following input variables and output variables are quoted from the JESD35-A standard. The variables quoted from the standard include the reference identification (Ref. JESD35-A).

Notes on input variables

NOTE

If there is no switching matrix in the system, input either `0` or `-1` for *`hi_pin`* and *`lo_pins`* to bypass switch.

`I_init`: For maximum sensitivity, the specified value should be well above the worst-case oxide current of a "good" oxide and well above the system noise floor. Higher values must be specified for ultra-thin oxide because of direct tunneling effects (Ref. JESD35-A).

Notes on output variables

`test_status`:

- `0`: No test errors (exit due to measured voltage < factor of the previous value).
- `1`: Failed pre-stress test.
- `-2`: Cumulative charge limit reached.
- `-3`: Maximum time limit reached.
- `-4`: Masked Catastrophic Failure.
- `-5`: Non-Catastrophic Failure.
- `-6`: Invalid specified *`t_step`*.

NOTE

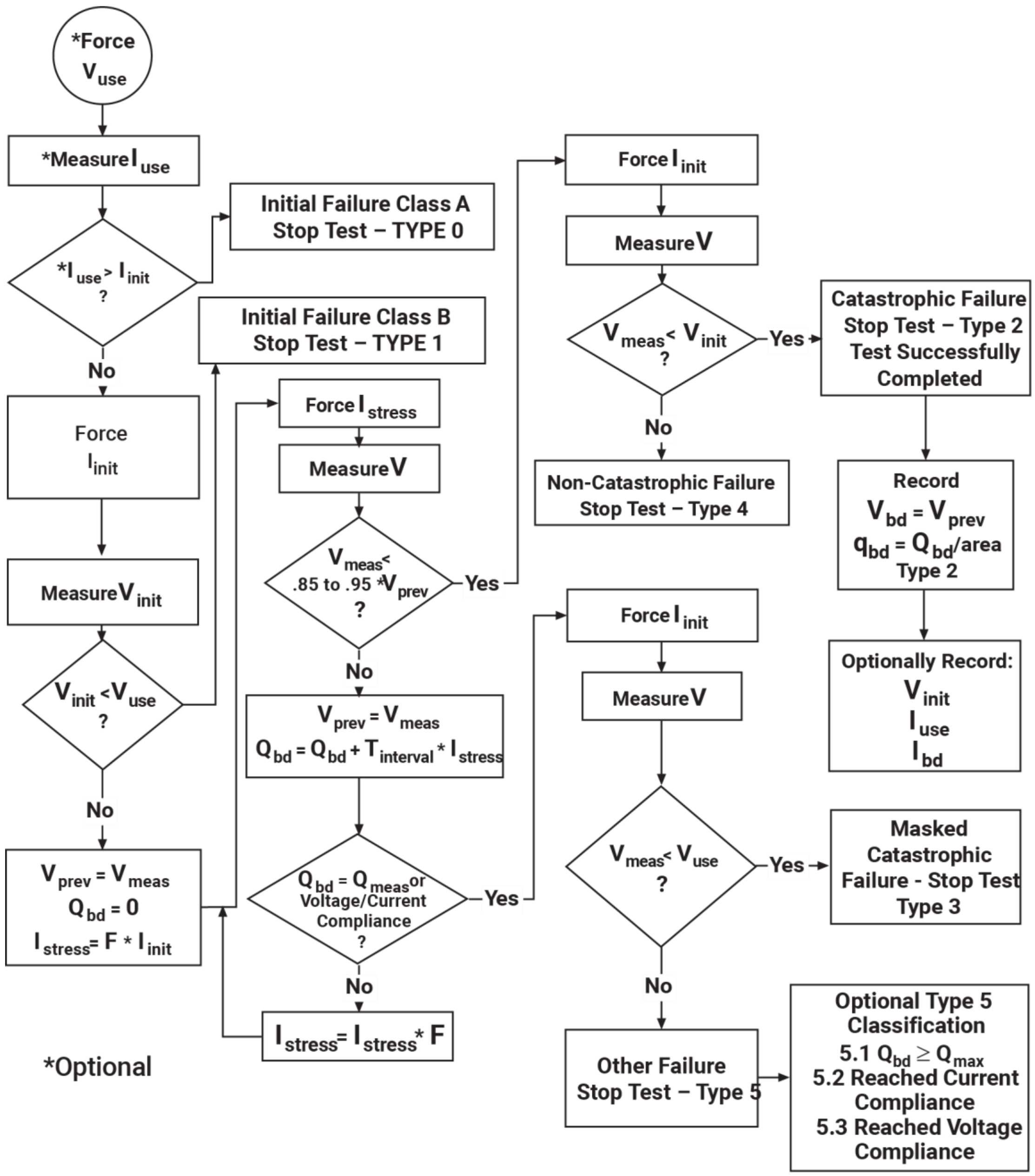
Invalid Test Result − Result = 1e21.

J-ramp flow diagram

NOTE

The following diagram from JESD35-A has been reproduced with permission from JEDEC. This flowchart is JEDEC copyright-protected material.

Figure 209: J-ramp flow diagram



NOTE

All values are absolute – no (+) or (−) signs have been incorporated.

Keithley Instruments • 28775 Aurora Road • Cleveland, Ohio 44139 • 1-800-833-9200 • tek.com/keithley

KEITHLEY
A Tektronix Company

04/2022