Table of Contents

User's Manual for SMU-4000 Source-Measure Unit

by Ark Metrica LTD.

This document is available online at https://docs.arkmetrica.com/smu4000 and https://docs.arkmetrica.com/smu4000/smu4000-manual.pdf.

Version: 720ac5bf

1. Overview

1.1 Introduction

The Ark Metrica SMU-4000 Source-Measure Unit (SMU) is a four-channel precision instrument designed for sourcing and measuring voltage and current. Combining the functionality of a voltage source, current source, voltmeter, and ammeter into a single device, the SMU is ideal for applications requiring precise electronic characterisation, including semiconductor testing, material's research, and device evaluation.

Unlike traditional power supplies or standalone measurement instruments, the SMU-4000 operates in both source and measure modes simultaneously. It supports four-quadrant operation, meaning it can source and sink bipolar voltages and currents, making it suitable for characterising both active and passive components under various operating conditions.

Equipped with four independently controllable channels, each with multiple operational modes and flexible compliance settings, the SMU-4000 offers exceptional control for laboratory and production environments. Its remote control options via standard communication protocols ensure seamless integration into automated test systems.

This manual provides detailed instructions on the operation, configuration, and best practices for using the SMU effectively.

1.2 Specifications

Measurement

- **Number of independent channels:** 4
- **Voltage range:** ± 5 V
- **Voltage DAC set resolution*:** 16-bit
- **Voltage ADC measure resolution*:** 16-bit
- **Current range:** ± 0.2 A
- **Current DAC set resolution*:** 16-bit
- **Current ADC measure resolution*:** 16-bit

* See "Section 1.11 Integration time" for further details.

Communication

- **Interfaces:** RJ45 (10/100BT)
- **IP configuration:** DHCP
- **Programming:** Custom command set based on a subset of SCPI

Power supply

- **Input voltage:** 12 VDC
- **Maximum input current (continuous):** <1.6 A (1.6 A internal fuse)
- **Maximum input current (transient):** 2A

1.3 Front panel

The SMU-4000 front panel, shown in Figure 1.3.1 presents a keyed 16-pin IDC header that allows connections from the SMU channels to be make to device test fixtures using appropriate cables.

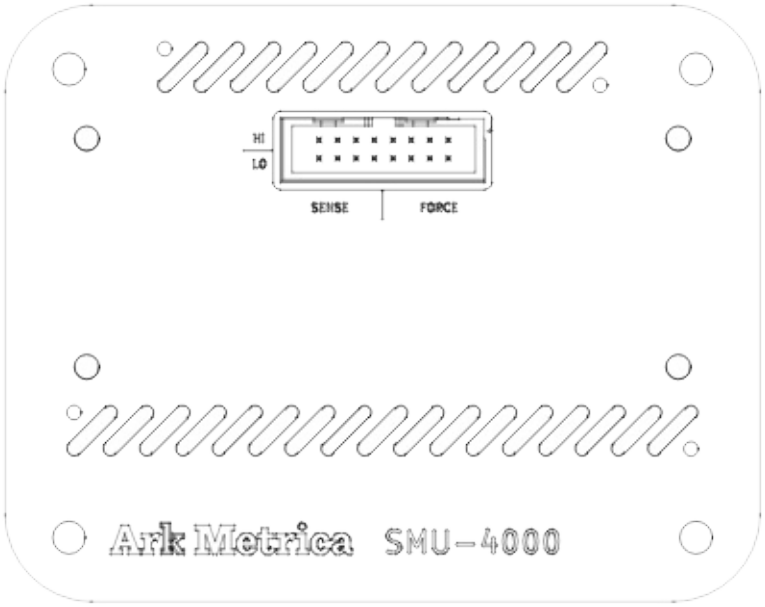

Figure 1.3.1. Front panel

The pin numbering for the IDC header is shown in Figure 1.3.2 and descriptions of each pin's functionality is given in Table 1.3.1.

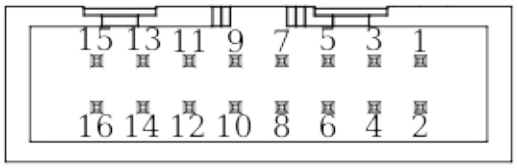

Figure 1.3.2. Output pins

Table 1.3.1. Front panel pin definitions

Pin #	Description
1	Channel 1, Force HI
2	Channel 1, Force LO
3	Channel 2, Force HI
4	Channel 2, Force LO
5	Channel 3, Force HI
6	Channel 3, Force LO
7	Channel 4, Force HI
8	Channel 4, Force LO
9	Channel 1, Sense HI
10	Channel 1, Sense LO
11	Channel 2, Sense HI
12	Channel 2, Sense LO
13	Channel 3, Sense HI
14	Channel 3, Sense LO
15	Channel 4, Sense HI
16	Channel 4, Sense LO

1.4 Rear panel

The SMU-4000 rear panel, shown in Figure 1.4.1 presents power and communications connections.

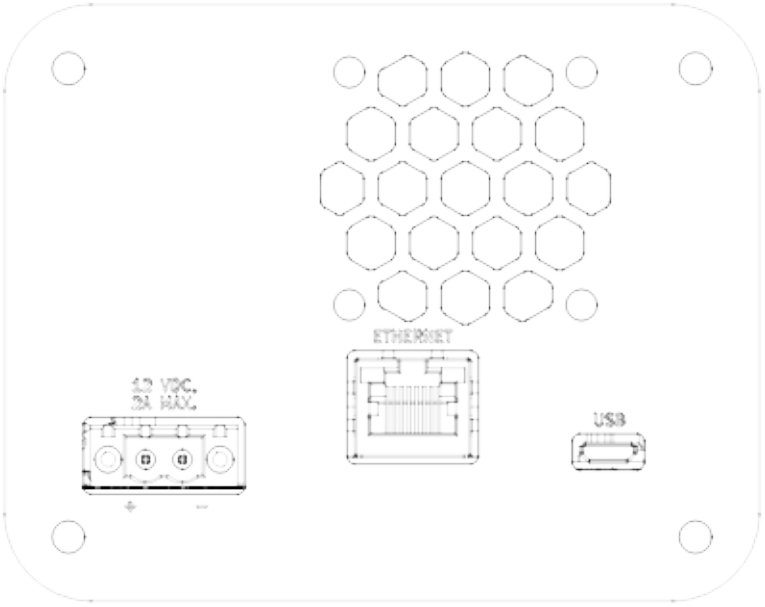

Figure 1.4.1. Rear panel

The power connection should be made to the 2-pin input labelled "12 VDC, 2 A MAX." using the the terminal block provided. Note the input voltage polarity markings when connecting the power supply.

The RJ45 port labelled "Ethernet" provides the communications interface.

The USB interface is for factory use only and should not be connected under normal operation.

1.5 Powering the instrument

The SMU-4000 is powered via the two-pin power connector on the rear panel of the instrument from a 12 VDC supply and requires up to 1.6 A continuous current at maximum load.

The most convenient way to power the instrument is via a PoE+ (IEEE 802.3at) compliant network switch (e.g. TP-Link TL-SL1218MP), ethernet cable, and splitter (e.g. Linovision POE-SP01). The network switch also allows for communication with the instrument from the control computer.

Ensure the input voltage polarity matches the rear panel markings when connecting the DC power supply.

1.6 Calibration

Each SMU channel has been factory calibrated at 23°C ± 2°C. If an instrument requires recalibration please contact Ark Metrica for a quotation at info@arkmetrica.com.

1.7 Cooling fan

The SMU-4000 enclosure contains a cooling fan on the rear panel that expels air pulled in through vents on the front panel. **Always ensure there is adequate space for air flow on both ends of the enclosure.**

WARNING: BLOCKING THE INLET AND/OR OUTLET VENTS MAY INVALIDATE THE CALIBRATION AND/OR DAMAGE THE INSTRUMENT.

1.8 Compliance

The SMU-4000 provides current and voltage compliance settings to protect devices under test (DUTs) and ensure controlled operation during source and measurement activities. Compliance — also known as limiting — prevents the instrument from exceeding a user-defined maximum voltage or current. This feature is crucial when working with sensitive components, preventing damage due to excessive power levels.

When the SMU is configured to source either voltage or current, the corresponding compliance limit restricts the opposing parameter:

- **Voltage Source Mode**
  - A current compliance limit is set to prevent excessive current flow.
  - The SMU outputs a user-specified voltage.
  - If the DUT attempts to draw more current than the limit allows, the SMU automatically switches to source the compliance current, preventing the voltage from the increasing further. If under this condition the voltage drops below the user-specified set point, the output will return to sourcing voltage.
- **Current Source Mode**
  - A voltage compliance limit is set to prevent excessive voltage application.
  - The SMU outputs a user-specified current.
  - If the DUT requires a voltage beyond the compliance limit to supply the requested current, the SMU automatically switches to source the compliance voltage, preventing the current from increasing further. If under this condition the current drops below the user-specified set point, the output will return to sourcing current.

The output does not turn off when in compliance but instead regulates the source to prevent exceeding the limit. Measurement readings will reflect the actual limited output rather than the originally programmed setpoint. When in compliance, the compliance bit will be set in the status byte (see "Chapter 2. Programming Guide" for details). This automatic regulation ensures that the DUT operates within safe limits while allowing accurate characterisation.

1.9 Input limits

WARNING: DO NOT CONNECT A POWER SOURCE TO THE INSTRUMENT FRONT PANEL IDC CONNECTOR THAT CAN SOURCE |V| > 5 V AND/OR |I| > 0.2 A. THIS MAY DAMAGE THE SMU.

1.10 Four-wire voltage sense

The SMU-4000 supports four-wire voltage sensing, also known as Kelvin sensing, to eliminate errors caused by lead resistance when measuring low voltages or sourcing precise currents. Unlike traditional two-wire measurements, which include the resistance of the test leads in the measurement, four-wire sensing uses separate force and sense connections to provide accurate voltage readings directly at the device under test (DUT).

In a two-wire configuration, the same pair of leads is used to source current and measure voltage. However, the resistance of these leads introduces a voltage drop, particularly at high currents, leading to inaccurate voltage measurements.

In a four-wire configuration, separate pairs of leads perform sourcing and sensing:

- **Force Leads**: Supply current or voltage to the DUT.
- **Sense Leads**: Measure the voltage directly at the DUT, eliminating errors from lead resistance.

Since the sense leads carry negligible current, the voltage drop across them is minimal, ensuring the SMU measures the true voltage applied to the DUT.

In addition to improving measurement accuracy, the SMU-4000 also provides output feedback when sourcing voltage in four-wire mode. The feedback mechanism sets the force terminals of the SMU to apply whatever voltage is necessary to achieve the user-specified set voltage at the sense terminal connections to the DUT. This eliminates the impact of lead resistance errors in both setting and measurement of applied voltage.

However, note that the force terminals are limited to supplying ± 5 V. If high voltages are requested by the user and the DUT draws a high current, it may not be possible to increase the force voltage high enough to achieve the user-set voltage at the sense terminals because of the voltage drop in the external force circuit test leads. It is always recommended to keep the test lead resistance as low as possible to minimise the impact of this effect.

1.11 Integration time

The SMU-4000 allows the user to specify measurement integration time in Number of Power Line Cycles (NPLC). The integration time window is the period over which each SMU channel averages samples from its analog-to-digital converters (ADCs). The measurement result is an average over as many samples as the ADCs can measure within the integration time. This means the effective measurement resolution depends on the user-specified NPLC.

1.12 Default settings

See "Chapter 2. Programming Guide" for default values of each user settable parameter.

1.13 Remote communication

The SMU-4000 supports remote communication over an Ethernet network using the TCP/IP suite. A host computer can establish a connection to the SMU via its assigned IP address and communicate with individual measurement channels through dedicated TCP ports.

The SMU-4000 obtains its IP address dynamically from a Dynamic Host Configuration Protocol (DHCP) server on the local network. To enable communication with the SMU, a DHCP server must be present and operational.

It is strongly recommended to configure the DHCP server to assign a static IP lease based on the SMU's Media Access Control (MAC) address, ensuring the instrument maintains a consistent IP address for reliable operation. Each SMU-4000 unit has its own MAC address, which is printed on a label located on the rear panel of the device. This MAC address follows a locally administered format: `02:0A:0E:XX:XX:XX`. Here, the `XX:XX:XX` portion is unique among devices provided by Ark Metrica. Since these MAC addresses are not globally unique, there is a potential risk of MAC address conflicts in network environments where similar address schemes are used. To prevent possible network conflicts:

- The SMU-4000 could be used on a dedicated or controlled local network.
- Consider using Virtual Local Area Networks (VLANs) if integration into an enterprise environment is necessary.

Once an IP address has been assigned, the SMU-4000 provides individual TCP/IP sockets for each measurement channel, allowing them to be controlled independently and perform measurements in parallel. Each channel is accessed via a specific TCP port number, as detailed in Table 1.13.1.

Table 1.13.1. TCP/IP ports and channel numbers

TCP Port	SMU Channel
50001	1
50002	2
50003	3
50004	4

3. Programming Guide

3.1 Introduction

This programming guide provides details on remotely controlling each channel of the Source Measure Unit (SMU). Control is based on a subset of the SCPI (Standard Commands for Programmable Instruments) command set but does not strictly conform to the standard. All supported common commands must be entered in **uppercase**. All other supported commands are in **abbreviated** form and must be entered in **lowercase**. All messages sent and received between the control computer and SMU are ASCII strings terminated with a carriage return character (`\r`).

Each channel has its own command buffer that can queue up a maximum of four commands for execution, including the one currently being executed. It is highly recommended that the programmer implement command flow control in their application to avoid filling the commands buffer and generating errors. The recommended approach is to wait for query commands to return a response before issuing the next commands. Write-only commands, such as setting parameter values, do not return a response. For write-only commands is is recommended that they be followed immediately by an error query (`syst:err:all?`) to determine that the write has been processed, and to validate that it was successful. After receiving the error response, it is safe to issue further commands.

3.2 Common commands

*IDN?

Description

The `*IDN?` command queries the instrument identification string. This provides information about the manufacturer, model, serial number, firmware version, channel serial number, and channel firmware version for the queried channel on the SMU.

Syntax

```
*IDN?
```

Response Format

Returns a comma-separated string containing the following information:

1. **Manufacturer Name**
2. **Model Number**
3. **Serial Number**
4. **Firmware Version**
5. **Channel Serial Number**
6. **Channel Firmware Version**

Examples

1. **Retrieve the instrument identification string:**

   ```
   *IDN?
   ```

   Example Response:

   ```
   Manufacturer, Model, SerialNumber, FirmwareVersion, ChannelSerialNumber, ChannelFirmwareVersion
   ```

Notes

- The response format and values depend on the specific instrument model and firmware version.
- This command is commonly used to verify communication with the instrument.

*RST

Description

The `*RST` command hard resets an SMU channel to its **default factory settings** by resetting the channel microcontroller. This command restores measurement, sourcing, and protection settings for a channel to their default values.

Syntax

```
*RST
```

Effects

- Clears all user-configured settings.
- Resets measurement functions, sourcing settings, and protection limits.
- Does **not** affect network communication settings.

Examples

1. **Reset the instrument to factory defaults:**

   ```
   *RST
   ```

Notes

- Use this command with caution, as all settings will be lost.
- The command takes ~5s to return the channel to a usable state.
- A soft reset to default settings is almost always preferable. See `syst:pres` instead.

*STB?

Description

The `*STB?` command queries the **status byte register** of the SMU channel. The status byte contains information about the current state of the instrument, including errors, state, and event flags.

Syntax

```
*STB?
```

Response Format

Returns an integer value representing the **status byte**. The meaning of each bit in the status byte is given in Table 3.2.1:

Table 3.2.1. Status Register Bits

Bit	Name	Description
0	Watchdog reset flag	The channel microcontroller has been reset by a watchdog timeout.

1	Compliance flag	The channel is currently in compliance.
2	Output enabled flag	The channel output is currently enabled.
3	Output mode flag	The current channel output mode. If set, sourcing current and measuring voltage. If not set, sourcing voltage and measuring current.
4	Command error bit 0	Bit 0 for error type of the most recent command.
5	Command error bit 1	Bit 1 for error type of the most recent command.
6	Reserved for future use.	
7	Reserved for future use.	

The meaning of each combination of command error bit values in the status byte is given in Table 3.2.2:

Table 3.2.2. Command error bits

Bit 0	Bit 1	Description
0	0	No error
1	0	Invalid command
0	1	Invalid CRC
1	1	Response overflow

Examples

1. **Retrieve the current status byte:**

```
*STB?
```

Example Response:

```
14
```

`14` is represented in binary as `00001110` meaning there has been no watchdog timer reset, the channel is in compliance, the output is enabled, the channel is sourcing current and measuring voltage, and there was no command error.

Notes

- The returned value is a decimal representation of the **binary status byte**.
- If multiple bits are set, their values are summed in the response.
- This command is useful for detecting errors and monitoring instrument state.

3.3 Command reference

outp

Description

The `outp` command controls the output state of an SMU channel. This command enables or disables the output, allowing or preventing voltage or current sourcing.

Syntax

```
outp <state>
```

Parameters

- `<state>` : Specifies whether the output is enabled or disabled.
  - `0` - Output Off
  - `1` - Output On

Default Value

- The default output state is **Off (0)**.

Query Form

To retrieve the current output state, use:

```
outp?
```

This command returns `0` if the output is off, or `1` if the output is on.

Examples

1. **Enable the output:**

   ```
   outp 1
   ```

2. **Disable the output:**

   ```
   outp 0
   ```

3. **Retrieve the current output state:**

   ```
   outp?
   ```

   Example Response:

   ```
   1
   ```

Notes

- Changing the output state takes effect immediately.
- If the output is enabled, ensure that appropriate voltage or current limits are set to avoid unintended behavior.
- The output state persists until modified or the instrument is reset.

read?

Description

The `read?` command triggers and returns the measurement value(s) from an SMU channel based on its current configuration.

Syntax

```
read?
```

Response Format

Returns a comma-separated list of measurement data. Each data point is a sub-list of three floating-point numbers and an integer representing the measured voltage in volts, measured current in amps, timestamp in ms since the channel was last powered on, and status byte, respectively. If the source is configured in sweep mode, the return value is a concatenated comma-separated list of all measurement points maintaining the aforementioned sub-list order. To understand the meanings of the bits in the status byte see `*STB?`.

Examples

1. **Retrieve the current measured value:**

   ```
   read?
   ```

   Example Response:

   ```
   0.0,0.0,123.456,14,1.0,0.05,143.456,14,2.0,0.1,163.456,14
   ```

   This response contains measurement data from a sweep of three points as detailed in Table 3.3.1:

Table 3.3.1. Example measurement data

Measurement Point	Voltage (V)	Current (A)	Timestamp (ms)	Status byte
0	0.0	0.0	123.456	14
1	1.0	0.05	143.456	14
2	2.0	0.1	163.456	14

Notes

- Ensure the measurement configuration is correctly set before using this command.
- If the instrument is in **remote sensing mode**, the voltage measurement reflects the value across the sense terminals rather than the force terminals.

sens:curr:nplc

Description

The `sens:curr:nplc` command sets the integration time for both current and voltage measurements in terms of number of power line cycles (NPLC) for an SMU channel. A higher NPLC value increases measurement accuracy by reducing noise but also increases measurement time.

The equivalent integration time, t_{int}, can be calculated as $t_{int} = NPLC/PLF$ where PLF is the power line frequency, e.g. 50Hz (see `syst:lfr`).

Syntax

```
sens:curr:nplc <value>
```

Parameters

- `<value>`: The number of power line cycles for current and voltage measurement, specified as a floating-point number.

Default Value

- The default NPLC value is **1.000**.

Range

- **Minimum:** 0.01 NPLC
- **Maximum:** 10.0 NPLC

Query Form

To retrieve the current NPLC setting for current and voltage measurements, use:

```
sens:curr:nplc?
```

This command returns the NPLC setting in **floating-point format**.

Examples

1. **Set the NPLC value to 5.0:**

   ```
   sens:curr:nplc 5.0
   ```

2. **Retrieve the current NPLC value:**

   ```
   sens:curr:nplc?
   ```

   Example Response:

   ```
   5.000
   ```

Notes

- The setting is applied to both current and voltage measurements. `sens:volt:nplc` is an identical command.
- Increasing NPLC improves accuracy but slows down measurements.
- Lower values allow faster readings but may introduce more noise.
- The command takes effect immediately upon execution.
- The setting persists until modified or the instrument is reset.

sens:curr:prot

Description

The `sens:curr:prot` command sets the current protection (compliance) limit for an SMU channel. This limit defines the maximum allowable current before the instrument enters a protection state to prevent damage or unintended operation. This limit applies when sourcing voltage. The limit is bipolar meaning that is applies to the absolute value of current.

Syntax

```
sens:curr:prot <value>
```

Parameters

- `<value>` : The current protection limit, specified in **amperes (A)**.

Default Value

- The default current protection limit is **0.200 A**.

Range

- **Minimum:** 0.0005 A
- **Maximum:** 0.2000 A

Query Form

To retrieve the current protection limit, use:

```
sens:curr:prot?
```

This command returns the current protection limit setting in **floating-point format** with units of amperes (A).

Examples

1. **Set the current protection limit to 0.2 A:**

   ```
   sens:curr:prot 0.2
   ```

2. **Retrieve the current protection limit:**

   ```
   sens:curr:prot?
   ```

   Example Response:

   ```
   0.200
   ```

Notes

- If a value outside the allowed range is provided, the SMU will generate an error.
- The command takes effect immediately upon execution.
- Protection settings persist until modified or the instrument is reset.

sens:volt:nplc

Description

The `sens:volt:nplc` command sets the integration time for both current and voltage measurements in terms of number of power line cycles (NPLC) for an SMU channel. A higher NPLC value increases measurement accuracy by reducing noise but also increases measurement time.

The equivalent integration time, t_{int}, can be calculated as $t_{int} = NPLC/PLF$ where PLF is the power line frequency, e.g. 50Hz (see `syst:lfr`).

Syntax

```
sens:volt:nplc <value>
```

Parameters

- `<value>` : The number of power line cycles for current and voltage measurement, specified as a floating-point number.

Default Value

- The default NPLC value is **1.000**.

Range

- **Minimum:** 0.01 NPLC
- **Maximum:** 10.0 NPLC

Query Form

To retrieve the current NPLC setting for current and voltage measurements, use:

```
sens:volt:nplc?
```

This command returns the NPLC setting in **floating-point format**.

Examples

1. **Set the NPLC value to 5.0:**

   ```
   sens:volt:nplc 5.0
   ```

2. **Retrieve the current NPLC value:**

   ```
   sens:volt:nplc?
   ```

   Example Response:

   ```
   5.000
   ```

Notes

- The setting is applied to both current and voltage measurements. `sens:curr:nplc` is an identical command.
- Increasing NPLC improves accuracy but slows down measurements.
- Lower values allow faster readings but may introduce more noise.
- The command takes effect immediately upon execution.
- The setting persists until modified or the instrument is reset.

sens:volt:prot

Description

The `sens:volt:prot` command sets the voltage protection limit for an SMU channel. This limit defines the maximum allowable voltage before the instrument enters a protection state to prevent damage or unintended operation. This limit applies when sourcing current. The limit is bipolar meaning that is applies to the absolute value of voltage.

Syntax

```
sens:volt:prot <value>
```

Parameters

- `<value>` : The voltage protection limit, specified in **volts (V)**.

Default Value

- The default voltage protection limit is **5.000 V**.

Range

- **Minimum:** 0.0125 V
- **Maximum:** 5.000 V

Query Form

To retrieve the current voltage protection limit, use:

```
sens:volt:prot?
```

This command returns the voltage protection limit setting in **floating-point format** with units of volts (V).

Examples

1. **Set the voltage protection limit to 20 V:**

   ```
   sens:volt:prot 1.0
   ```

2. **Retrieve the voltage protection limit:**

   ```
   sens:volt:prot?
   ```

   Example Response:

   ```
   1.000
   ```

Notes

- If a value outside the allowed range is provided, the SMU will generate an error.
- The command takes effect immediately upon execution.
- Protection settings persist until modified or the instrument is reset.

sour:curr

Description

The `sour:curr` command sets the output current of an SMU channel when operating in **current source mode**.

Syntax

```
sour:curr <value>
```

Parameters

- `<value>` : Specifies the output current, in **amperes (A)**.

Default Value

- The default output current is **0.000 A**.

Range

- **Minimum:** -0.200 A
- **Maximum:** 0.200 A

Query Form

To retrieve the current output current setting, use:

```
sour:curr?
```

This command returns the current setting in **floating-point format** with units of amperes (A).

Examples

1. **Set the output current to 0.1 A:**

   ```
   sour:curr 0.1
   ```

2. **Retrieve the current output current setting:**

   ```
   sour:curr?
   ```

   Example Response:

   ```
   0.1
   ```

Notes

- The SMU must be in **current source mode** (`sour:func curr`) for this command to take effect when the output is enabled.
- Ensure that appropriate **voltage compliance** (`sens:volt:prot`) is set to protect the device under test (DUT).
- The current setting takes effect immediately upon execution.
- The setting persists until modified or the instrument is reset.

sour:curr:mode

Description

The `sour:curr:mode` command sets the current sourcing mode for an SMU channel. This determines whether the current output operates in a **fixed mode** or as part of a **sweep operation**.

Syntax

```
sour:curr:mode <mode>
```

Parameters

- `<mode>` : Specifies the current sourcing mode.
  - `fix` - Fixed current mode
  - `swe` - Current sweep mode

Default Value

- The default current sourcing mode is **Fixed (`fix`)**.

Query Form

To retrieve the current sourcing mode, use:

```
sour:curr:mode?
```

This command returns the current mode as either `"fix"` or `"swe"`.

Examples

1. **Set the current source to fixed mode:**

   ```
   sour:curr:mode fix
   ```

2. **Set the current source to sweep mode:**

   ```
   sour:curr:mode swe
   ```

3. **Retrieve the current source mode:**

   ```
   sour:curr:mode?
   ```

   Example Response:

   ```
   fix
   ```

Notes

- **Fixed mode (`fix`)** maintains a constant current output.
- **Sweep mode (`swe`)** varies current over a defined range for characterisation applications.
- If switching from **sweep mode to fixed mode**, ensure that the current is set to an appropriate level.
- The command takes effect immediately upon execution.
- The setting persists until modified or the instrument is reset.

sour:curr:start

Description

The `sour:curr:start` command sets the starting current value for a **current sweep operation**. This defines the initial current level when performing a programmed current sweep.

Syntax

```
sour:curr:start <value>
```

Parameters

- `<value>` : Specifies the starting current, in **amperes (A)**.

Default Value

- The default starting current is **0.000 A**.

Range

- **Minimum:** -0.200 A
- **Maximum:** 0.200 A

Query Form

To retrieve the current starting current setting, use:

```
sour:curr:start?
```

This command returns the starting current setting in **floating-point format** with units of amperes (A).

Examples

1. **Set the starting current to 0.0 A:**

   ```
   sour:curr:start 0.0
   ```

2. **Retrieve the current starting current setting:**

   ```
   sour:curr:start?
   ```

   Example Response:

   ```
   0.0
   ```

Notes

- This command is only used in **current sweep mode** (`sour:curr:mode swe`).
- The starting current should be set **before defining the stop current**.
- The setting persists until modified or the instrument is reset.

sour:curr:stop

Description

The `sour:curr:stop` command sets the stopping current value for a **current sweep operation**. This defines the final current level when performing a programmed current sweep.

Syntax

```
sour:curr:stop <value>
```

Parameters

- `<value>` : Specifies the stopping current, in **amperes (A)**.

Default Value

- The default stopping current is **0.000 A**.

Range

- **Minimum:** -0.200 A
- **Maximum:** 0.200 A

Query Form

To retrieve the current stopping current setting, use:

```
sour:curr:stop?
```

This command returns the stopping current setting in **floating-point format** with units of amperes (A).

Examples

1. **Set the stopping current to 0.1 A:**

   ```
   sour:curr:stop 0.1
   ```

2. **Retrieve the current stopping current setting:**

   ```
   sour:curr:stop?
   ```

   Example Response:

   ```
   0.100
   ```

Notes

- This command is only used in **current sweep mode** (`sour:curr:mode swe`).
- The stopping current should be set **after defining the starting current**.
- The setting persists until modified or the instrument is reset.

sour:del:auto

Description

The `sour:del:auto` command enables or disables the automatic source delay mode. When enabled for an SMU channel, it uses a source delay of 1 ms.

Syntax

```
sour:del:auto <state>
```

Parameters

- `<state>` : Specifies whether automatic source delay is enabled or disabled.
  - `0` - Automatic source delay off (manual mode)
  - `1` - Automatic source delay on

Default Value

- The default state is **Enabled (1)**.

Query Form

To retrieve the current automatic source delay setting, use:

```
sour:del:auto?
```

This command returns `0` if auto delay is disabled, or `1` if auto delay is enabled.

Examples

1. **Enable automatic source delay:**

   ```
   sour:del:auto 1
   ```

2. **Disable automatic source delay:**

   ```
   sour:del:auto 0
   ```

3. **Retrieve the current automatic delay setting:**

   ```
   sour:del:auto?
   ```

   Example Response:

   ```
   0
   ```

Notes

- The command takes effect immediately upon execution.
- The setting persists until modified or the instrument is reset.

sour:del

Description

The `sour:del` command sets the source delay time for an SMU channel. This delay determines the time interval between setting the source value and taking a measurement, allowing stabilisation of the sourced voltage or current before measurement.

Syntax

```
sour:del <value>
```

Parameters

- `<value>` : The delay time, specified in **seconds (s)**.

Default Value

- The default source delay is **0.000 s** (no delay).

Range

- **Minimum:** 0.000 s
- **Maximum:** 9999.999 s

Query Form

To retrieve the current source delay setting, use:

```
sour:del?
```

This command returns the source delay setting in **floating-point format** with units of seconds (s).

Examples

1. **Set the source delay to 0.1 seconds:**

   ```
   sour:del 0.1
   ```

2. **Retrieve the current source delay:**

   ```
   sour:del?
   ```

   Example Response:

   ```
   0.1
   ```

Notes

- A longer delay can improve measurement accuracy by allowing transients to settle.
- Setting the delay to `0.000` minimises latency but may introduce instability in some setups.
- The command takes effect immediately upon execution.
- The setting persists until modified or the instrument is reset.

sour:func

Description

The `sour:func` command sets the sourcing function of an SMU channel. This determines whether the instrument operates as a **voltage source** or a **current source**.

Syntax

```
sour:func <function>
```

Parameters

- `<function>` : Specifies the sourcing function.
  - `volt` - Voltage source mode
  - `curr` - Current source mode

Default Value

- The default sourcing function is **Voltage (`volt`)**.

Query Form

To retrieve the currently set sourcing function, use:

```
sour:func?
```

This command returns the current sourcing mode as either `"volt"` or `"curr"`.

Examples

1. **Set the instrument to voltage source mode:**

   ```
   sour:func volt
   ```

2. **Set the instrument to current source mode:**

   ```
   sour:func curr
   ```

3. **Retrieve the currently set sourcing function:**

   ```
   sour:func?
   ```

   Example Response:

   ```
   curr
   ```

Notes

- Changing the sourcing function may require reconfiguring output settings (e.g., voltage or current limits).
- Ensure the selected mode is compatible with your test setup to prevent unintended behavior.
- The command takes effect immediately upon execution.
- The setting persists until modified or the instrument is reset.

sour:swe:poin

Description

The `sour:swe:poin` command sets the number of points in a **sweep operation**. This defines how many discrete steps the source will take from the start to stop value.

Syntax

```
sour:swe:poin <value>
```

Parameters

- `<value>`: Specifies the number of points in the sweep, as an integer.

Default Value

- The default number of sweep points is **1**.

Range

- **Minimum:** 1 points
- **Maximum:** 117 points

Query Form

To retrieve the current number of sweep points, use:

```
sour:swe:poin?
```

This command returns the number of points as an integer.

Examples

1. **Set the sweep to use 50 points:**

   ```
   sour:swe:poin 50
   ```

2. **Retrieve the current number of sweep points:**

   ```
   sour:swe:poin?
   ```

   Example Response:

   ```
   50
   ```

Notes

- The total sweep time is affected by the number of points, the delay, and NPLC settings.
- A larger number of points provides finer resolution but increases total sweep time.
- The setting persists until modified or the instrument is reset.

sour:swe:spac

Description

The `sour:swe:spac` command sets the spacing type for a **sweep operation**. This determines whether the sweep steps are **linearly** or **logarithmically** spaced.

Syntax

```
sour:swe:spac <type>
```

Parameters

- `<type>`: Specifies the sweep spacing type.
  - `lin` - Linear sweep spacing
  - `log` - Logarithmic sweep spacing

Default Value

- The default sweep spacing is **Linear (`lin`)**.

Query Form

To retrieve the current sweep spacing type, use:

```
sour:swe:spac?
```

This command returns the current spacing type as either `"lin"` or `"log"`.

Examples

1. **Set the sweep spacing to logarithmic:**

   ```
   sour:swe:spac log
   ```

2. **Set the sweep spacing to linear:**

   ```
   sour:swe:spac lin
   ```

3. **Retrieve the current sweep spacing setting:**

   ```
   sour:swe:spac?
   ```

   Example Response:

   ```
   lin
   ```

Notes

- **Linear spacing (`lin`)** steps through values with equal increments.
- **Logarithmic spacing (`log`)** steps through values with a logarithmic spacing.
- The setting persists until modified or the instrument is reset.

sour:volt

Description

The `sour:volt` command sets the output voltage of an SMU channel when operating in **voltage source mode**.

Syntax

```
sour:volt <value>
```

Parameters

- `<value>` : Specifies the output voltage, in **volts (V)**.

Default Value

- The default output voltage is **0.000 V**.

Range

- **Minimum:** -5.0 V
- **Maximum:** 5.0 V

Query Form

To retrieve the current output voltage setting, use:

```
sour:volt?
```

This command returns the voltage setting in **floating-point format** with units of volts (V).

Examples

1. **Set the output voltage to 5.0 V:**

   ```
   sour:volt 5.0
   ```

2. **Retrieve the current output voltage setting:**

   ```
   sour:volt?
   ```

   Example Response:

   ```
   5.000
   ```

Notes

- The SMU must be in **voltage source mode** (`sour:func volt`) for this command to take effect.
- Ensure that appropriate **current compliance** (`sens:curr:prot`) is set to protect the device under test (DUT).
- The voltage setting takes effect immediately upon execution.
- The setting persists until modified or the instrument is reset.

sour:volt:mode

Description

The `sour:volt:mode` command sets the voltage sourcing mode for an SMU channel. This determines whether the voltage output operates in a **fixed mode** or as part of a **sweep operation**.

Syntax

```
sour:volt:mode <mode>
```

Parameters

- `<mode>` : Specifies the voltage sourcing mode.
  - `fix` - Fixed voltage mode
  - `swe` - Voltage sweep mode

Default Value

- The default voltage sourcing mode is **Fixed (`fix`)**.

Query Form

To retrieve the current voltage sourcing mode, use:

```
sour:volt:mode?
```

This command returns the current mode as either `"fix"` or `"swe"` .

Examples

1. **Set the voltage source to fixed mode:**

   ```
   sour:volt:mode fix
   ```

2. **Set the voltage source to sweep mode:**

```
sour:volt:mode swe
```

3. **Retrieve the current voltage source mode:**

```
sour:volt:mode?
```

Example Response:

```
fix
```

Notes

- **Fixed mode (`fix`)** maintains a constant voltage output.
- **Sweep mode (`swe`)** varies voltage over a defined range for characterisation applications.
- If switching from **sweep mode to fixed mode**, ensure that the voltage is set to an appropriate level.
- The command takes effect immediately upon execution.
- The setting persists until modified or the instrument is reset.

sour:volt:start

Description

The `sour:volt:start` command sets the starting voltage value for a **voltage sweep operation**. This defines the initial voltage level when performing a programmed voltage sweep.

Syntax

```
sour:volt:start <value>
```

Parameters

- `<value>` : Specifies the starting voltage, in **volts (V)**.

Default Value

- The default starting voltage is **0.000 V**.

Range

- **Minimum:** -5.0 V
- **Maximum:** 5.0 V

Query Form

To retrieve the current starting voltage setting, use:

```
sour:volt:start?
```

This command returns the starting voltage setting in **floating-point format** with units of volts (V).

Examples

1. **Set the starting voltage to -2.5 V:**

```
sour:volt:start -2.5
```

2. **Retrieve the current starting voltage setting:**

```
sour:volt:start?
```

Example Response:

```
-2.500
```

Notes

- This command is only used in **voltage sweep mode** (`sour:volt:mode swe`).
- The starting voltage should be set **before defining the stop voltage**.
- The setting persists until modified or the instrument is reset.

sour:volt:stop

Description

The `sour:volt:stop` command sets the stopping voltage value for a **voltage sweep operation**. This defines the final voltage level when performing a programmed voltage sweep.

Syntax

```
sour:volt:stop <value>
```

Parameters

- `<value>` : Specifies the stopping voltage, in **volts (V)**.

Default Value

- The default stopping voltage is **0.000 V**.

Range

- **Minimum:** -5.0 V
- **Maximum:** 5.0 V

Query Form

To retrieve the current stopping voltage setting, use:

```
sour:volt:stop?
```

This command returns the stopping voltage setting in **floating-point format** with units of volts (V).

Examples

1. **Set the stopping voltage to 2.5 V:**

```
sour:volt:stop 2.5
```

2. **Retrieve the current stopping voltage setting:**

```
sour:volt:stop?
```

Example Response:

```
2.500
```

Notes

- This command is only used in **voltage sweep mode** (`sour:volt:mode swe`).
- The stopping voltage should be set **after defining the starting voltage**.
- The setting persists until modified or the instrument is reset.

syst:err:all?

Description

The `syst:err:all?` command retrieves up to 10 system error messages from an SMU channel. This provides information on any errors that have occurred, allowing for troubleshooting and debugging. Once this command has been issued, the channel error buffer is cleared. Possible error messages and their meanings are given in Table 3.3.2.

Table 3.3.2. Error messages

Message	Description
"+000, No Error"	The previous command was valid.
"-100, Command error"	The previous command was malformed or not supported.
"-154, String too long"	The response string for the last command exceeds the valid length of the response buffer.
"-220, Parameter error"	The parameter for the previous message was malformed or out of range.
"-363, Input buffer overrun"	The command string is too long for the SMU channel command buffer.
"+950, Command queue overflow"	The command buffer is full because the instrument is still processing previous commands.

Syntax

```
syst:err:all?
```

Response Format

Returns a comma-separated list of error codes and corresponding messages.

Examples

1. **Retrieve all system error messages:**

```
syst:err:all?
```

Example Response:

```
+000, No Error
```

Notes

- If multiple errors exist, they are all returned, emptying the error queue.
- A response of `+000, No Error` indicates that there are no active errors in the system.
- Errors remain in the queue until read or the instrument is reset.

syst:lfr

Description

The `syst:lfr` command sets the power line frequency for noise filtering in measurements. This setting is used internally by an SMU channel to determine integration time from NPLC.

Syntax

```
syst:lfr <value>
```

Parameters

- `<value>` : Specifies the AC line frequency in **hertz (Hz)**.
  - `50` - 50 Hz power line frequency
  - `60` - 60 Hz power line frequency

Default Value

- The default line frequency is **50 Hz**.

Query Form

To retrieve the current line frequency setting, use:

```
syst:lfr?
```

This command returns the frequency in hertz as an integer (`50` or `60`).

Examples

1. **Set the line frequency to 50 Hz:**

   ```
   syst:lfr 50
   ```

2. **Retrieve the current line frequency setting:**

   ```
   syst:lfr?
   ```

   Example Response:

   ```
   50
   ```

Notes

- The selected frequency should match the local power grid to optimise noise rejection.
- Changing the line frequency setting affects **integration time for measurements**.
- The setting persists until modified or the instrument is reset.

syst:pres

Description

The `syst:pres` command restores an SMU channel to its **factory preset** state. This resets all user-configurable settings to their default values while maintaining communication settings.

Syntax

```
syst:pres
```

Effects

- Clears all user-defined configurations.
- Resets measurement, sourcing, and protection settings.
- Does **not** affect communication settings.

Examples

1. **Restore factory preset settings:**

   ```
   syst:pres
   ```

Notes

- Use this command with caution, as all configuration settings will be lost.
- This is a soft reset of parameter values that doesn't fully reset the SMU channel microcontroller.
- This soft reset to default settings is almost always preferable over `*RST`.
- The command takes effect immediately upon execution.

syst:rsen

Description

The `syst:rsen` command enables or disables **remote sensing mode** for voltage measurements. Remote sensing helps compensate for voltage drops across test leads by using a separate sense connection.

Syntax

```
syst:rsen <state>
```

Parameters

- `<state>`: Specifies whether remote sensing is enabled or disabled.
  - `0` - Remote sensing off (2-wire mode)
  - `1` - Remote sensing on (4-wire mode)

Default Value

- The default setting is **Remote sensing off (`1`)**.

Query Form

To retrieve the current remote sensing state, use:

```
syst:rsen?
```

This command returns `0` if remote sensing is disabled, or `1` if it is enabled.

Examples

1. **Enable remote sensing (4-wire mode):**

   ```
   syst:rsen 1
   ```

2. **Disable remote sensing (2-wire mode):**

   ```
   syst:rsen 0
   ```

3. **Retrieve the current remote sensing setting:**

   ```
   syst:rsen?
   ```

   Example Response:

   ```
   0
   ```

Notes

- **Remote sensing (`1`)** improves accuracy by eliminating lead resistance effects.
- **Local sensing (`0`)** uses standard 2-wire connections without additional compensation.
- Ensure that appropriate sense leads are connected when enabling remote sensing.
- The setting persists until modified or the instrument is reset.

3. Licensing

This product includes software developed by third parties. Below is a list of open-source software included in this product and their respective licenses.

Software Name: m1k-fw
Version: 2.17
Source: https://github.com/analogdevicesinc/m1k-fw
License:
Copyright (c) 2014,2015 Analog Devices, Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Software Name: ioLibrary_Driver
Version: 4.0.0
Source: https://github.com/Wiznet/ioLibrary_Driver
License:
Copyright (c) 2014 WIZnet Co.,Ltd. Copyright (c) WIZnet ioLibrary Project. All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Software Name: pico-sdk
Version: 2.1.1
Source: https://github.com/raspberrypi/pico-sdk
License: