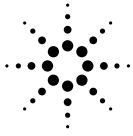
Agilent E5270 TIS User's Guide

Agilent E5270 Series of Parametric Measurement Solutions



Agilent Technologies

E5270-90030

December 2002

Edition 1

Legal Notice

The information contained in this document is subject to change without notice.

- Product Warranty

  Agilent Technologies warrant Agilent Technologies hardware, accessories and supplies against defects in materials and workmanship for the period of one year from the warranty start date specified below. If Agilent Technologies receive notice of such defects during the warranty period, Agilent Technologies will, at its option, either repair or replace products which prove to be defective. Replacement products may be either new or like-new.

  Warranty service of this product will be performed at Agilent Technologies. Buyer shall prepay shipping charges to Agilent Technologies and Agilent Technologies shall pay shipping charges to return the product to Buyer. However, Buyer shall pay all shipping charges, duties, and taxes for products returned to Agilent Technologies from another country.

  Agilent Technologies do not warrant that the operation of Agilent Technologies products will be uninterrupted or error free. If Agilent is unable, within a reasonable time, to repair or replace any product to a condition as warranted, customer will be entitled to a refund of the purchase price upon prompt return of the product.

  The Agilent Technologies products may contain remanufactured parts equivalent to new in performance or may have been subject to incidental use.

  The warranty period begins on the date of delivery or on the date of installation if installed by Agilent Technologies. If customer schedules or delays Agilent Technologies installation more than 30 days after delivery, warranty begins on the 31st day from delivery.

  Warranty does not apply to defects resulting from (a) improper or inadequate maintenance or calibration, (b) software, interfacing, parts or supplies not supplied by Agilent Technologies, (c) unauthorized modification or misuse, (d) operation outside of the published environmental specifications for the product, or (e) improper site preparation or maintenance.

To the extent allowed by local law, the above warranties are exclusive and no other warranty or condition, whether written or oral, is expressed or implied and Agilent Technologies specifically disclaim any implied warranties or conditions of merchantability, satisfactory quality, and fitness for a particular purpose.

Agilent Technologies will be liable for damage to tangible property per incident up to the greater of $300,000 or the actual amount paid for the product that is the subject of the claim, and for damages for bodily injury or death, to the extent that all such damages are determined by a court of competent jurisdiction to have been directly caused by a defective Agilent Technologies product.

To the extent allowed by local law, the remedies in this warranty statement are customer's sole and exclusive remedies. Expect as indicated above, in no event will Agilent Technologies or its suppliers be liable for loss of date or for direct, special, incidental, consequential (including lost profit or date), or other damage, whether based in contract, tort, or otherwise.

For consumer transactions in Australia and New Zealand: the warranty terms contained in this statement, except to the extent lawfully permitted, do not exclude, restrict or modify and are in addition to the mandatory statutory rights applicable to the sale of this product to you.

- Assistance

  Product maintenance agreements and other customer assistance agreements are available for Agilent Technologies products.

  For any assistance, contact your nearest Agilent Technologies Sales Office.

- Certification

  Agilent Technologies Inc. certifies that this product met its published specifications at the time of shipment from the factory. Agilent further certifies that its calibration measurements are traceable to the National Institute of Standards and Technology (NIST), to the extent allowed by the Institute's calibration facility, and to the calibration facilities of other International Standards Organization members.

Printing History

Edition 1: December 2002

In This Manual

The Test Instruction Set (TIS) is a set of powerful functions that facilitate measurement programming for the Agilent E5270, and supports the following applications:

- DC current/voltage output
- High speed spot measurement
- Multi channel spot measurement
- Pulsed spot measurement
- Staircase sweep measurement
- Pulsed sweep measurement
- Staircase sweep with pulsed bias measurement
- Breakdown voltage measurement
- Leakage current measurement

The Agilent E5270 TIS library is available for the C language users. Measurement programs that include TIS functions are easier to maintain than programs written solely in the C language.

This manual describes the installation and reference information of the Agilent E5270 TIS, and consists of the following chapters:

- "Starting TIS Programming"

  Describes the intallation information of the Agilent E5270 TIS library, and general information for programming.

- "TIS Function Reference"

  Describes the reference information of the Agilent E5270 TIS functions.

Contents

Contents

1 Starting TIS Programming

This chapter describes the installation information of the Agilent E5270 TIS library, and basic information for programming.

- “Installation”
- “TIS Programming”
- “Migration to Agilent 4070”

Installation

This section describes the system requirements and installation procedure for the Agilent E5270 TIS library.

- "System Requirements"
- "Installing TIS Library"

System Requirements

The following system environments are required.

- Operating system

  Microsoft Windows XP Professional, Windows 2000, Windows NT 4.0, or Windows 95. It must be supported by the application development environment and VISA I/O library.

- Application development environment (or programming environment)

  Microsoft Visual C++ or Borland C++Builder. It must be supported by the VISA I/O library.

- GPIB (IEEE 488) interface and 32-bit VISA I/O library

  Agilent GPIB interface with Agilent IO Libraries or equivalent.

- Agilent E5270 VXI*plug&play* driver

  The driver is required to use the Agilent E5270 TIS library. The Agilent E5270 TIS library and VXI*plug&play* driver are stored in the Software CD-ROM furnished with the Agilent E5270.

- Computer and peripherals

  Required specifications depend on the application development environment. See manual of the application development environment. The CD-ROM drive is required to install the VXI*plug&play* driver and TIS library.

- Minimum disk space

  4 MB (2 MB for the Agilent TIS library and 2 MB for VXI*plug&play* driver)

Installing TIS Library

The installation flow is shown below. If you have already installed the GPIB (IEEE 488) interface, VISA I/O library, and programming software (Microsoft C++ or Borland C++Builder) on your computer, skip steps 1 through 4.

1. Install the GPIB interface to your PC.

   See manual of the GPIB interface. Note the model number of the GPIB interface, as you may need it to configure the interface (in step 3).

2. Install VISA I/O library.

   Follow the setup program instructions.

3. Configure and check the GPIB interface.

   See manual of the VISA I/O library.

4. Install the programming software.

   Follow the setup program instructions.

5. Install the Agilent E5270 VXI*plug&play* driver.

   a. Insert the Agilent E5270 Series Software CD-ROM to the CD-ROM drive connected to your computer.

   b. Execute \Pnp\E5270.exe on the CD-ROM.

   The setup program installs the driver. See Table 1-1 for the installed files. The TIS library will refer the driver to control the Agilent E5270.

6. Install the Agilent E5270 TIS library.

   a. Insert the Agilent E5270 Series Software CD-ROM to the CD-ROM drive connected to your computer.

   b. Create a folder (e.g. C:\Agilent\E5270) on your computer.

   c. Copy the Tis subdirectory on the Software CD-ROM to the created folder.

   See Table 1-2 for the installed files.

Table 1-1 Agilent E5270 VXIplug&play Driver Files

File Name [a] [b]	Description
<visa path>\Win*xx*\Age5270\age5270.bas	Driver for Microsoft Visual Basic.
<visa path>\Win*xx*\Age5270\age5270.c	Driver source code.
<visa path>\Win*xx*\Age5270\age5270.def	DLL export definition file.
<visa path>\Win*xx*\Age5270\age5270.fp	Front panel file.
<visa path>\Win*xx*\Age5270\age5270.h	Driver header file.
<visa path>\Win*xx*\Age5270\age5270.hlp	Online help file.
<visa path>\Win*xx*\Age5270\readme.txt	Read me file.
<visa path>\Win*xx*\bin\age5270_32.dll	Driver DLL file.
<visa path>\Win*xx*\include\age5270.h	Driver header file.
<visa path>\Win*xx*\lib\bc\age5270.lib	Library for Borland C++Builder.
<visa path>\Win*xx*\lib\bc\age5270_32.lib	Library for Borland C++Builder.
<visa path>\Win*xx*\lib\msc\age5270.lib	Library for Microsoft C++.
<visa path>\Win*xx*\lib\msc\age5270_32.lib	Library for Microsoft C++.

a. <visa path> indicates the folder specified when you install the VISA I/O library. In the default setting, <visa path> is \Program Files\Visa. If you use the Agilent IO Library, you can verify <visa path> from the Agilent IO Libraries Installation and Path dialog box. To open the dialog box, click the Agilent IO Libraries Control icon on the Windows task bar, click View Documentation, and click Installation and Path Information.

b. Win*xx* depends on the OS of your computer, Winnt for Windows XP, Windows 2000, or Windows NT, or Win95 for Windows 95.

Table 1-2 **Agilent E5270 TIS Library Files**

File Name [a]	Description
<user path>\lib\msc\E5270_TIS.lib	TIS library for Microsoft Visual C++.
<user path>\lib\bc\E5270_TIS.lib	TIS library for Borland C++Builder.
<user path>\E5270_TIS.h	TIS library header file.
<user path>\E5270_TIS.c	TIS library source code.
<user path>\Tis_gd.pdf	TIS User's Guide. Electronic manual.
<user path>\sample\sample.c	Sample program source code. This is just example program.
<user path>\sample\makefile	Sample program makefile for Microsoft Visual C++.

a. <user path> indicates the folder created when you install the TIS library (e.g. C:\Agilent\E5270).

TIS Programming

This section provides the basic information for programming.

- “Planning Measurement”
- “Programming”
- “Debugging”

Planning Measurement

Before starting programming, decide the following items:

- Measurement devices

  Discrete, packaged, on-wafer, and so on.

- Characteristics to be measured

  h_{FE}, Vth, sheet resistance, and so on.

- Measurement method

  Spot measurement, staircase sweep measurement, and so on.

Programming

The measurement program should be created as shown below. For programming example, see Figure 1-1.

1. Include the header files.

   To use the TIS library, the header file `E5270_TIS.h` must be included at the beginning of measurement program. The header file contains various necessary information for the function, such as function declaration and macro definitions.

   The header file has been stored in the folder created when you install the TIS library. The following example includes the header file in the \Agilent\E5270 folder.

   Example:

   ```
   #include "\Agilent\E5270\E5270_TIS.h"
   ```

2. Define the `open_E5270` function to establish the connection with the Agilent E5270. The following example connects the Agilent E5270 of GPIB address 17.

   Example:

   ```
   main()
   {
   int ret;

   #ifdef E5270_TIS_H
      ret = open_E5270( "GPIB0::17::INSTR" , ERR_DETECT_ON,NULL);
      if ( ret != 0 ) {
            printf( "E527x device open failed. Exit program.\n" );
            return( -1 );
      }
   #endif
   ```

3. Create measurement routine. Then use TIS functions:

   - to initialize the Agilent E5270,
   - to connect device under test or to enable source/measurement channels,
   - to set source outputs,
   - to perform measurements, and
   - to disconnect device under test or to disable source/measurement channels.

   Also, program lines to display, store, or calculate data should be added.

   For programming example, see Figure 1-1. This example performs resistance measurement.

4. Define the `close_E5270` function to disable the connection with the Agilent E5270.

   Example:

```
#ifdef E5270_TIS_H
   close_E5270();
#endif
   return( 0 );
}
```

NOTE

E5270_TIS_H macro

The E5270_TIS_H is a macro that is effective after the header file E5270_TIS.h is included. The macro can be used to judge whether the E5270 TIS library is available or not, and switch a program operation as the following example.

```
#ifdef E5270_TIS_H
   /* codes for E5270 */
#else
   /* codes for 4070 */
#endif
```

This example is used to switch codes to be performed. If the E5270_TIS_H is effective, the codes for E5270 will be performed. Else, the codes for 4070 will be performed.

An example shown in Figure 1-1 uses this macro to open and close the connection to the Agilent E5270 if the E5270_TIS_H macro is effective.

Figure 1-1 **Example of main program**

```
#include "E5270_TIS.h"

main()
{
int ret;

#ifdef E5270_TIS_H
   ret = open_E5270( "GPIB0::17::INSTR" , ERR_DETECT_ON,NULL);
   if ( ret != 0 ) {
         printf( "Failed to open E5270. Exit program.\n" );
         return( -1 );
    }
#endif

   int h_pin, l_pin;
   double v_force, comp, current;
   v_force = 0.5;
   comp = 1e-3;
   h_pin = 12;
   l_pin = 16;

   init_system();

   connect_pin(SMU3, h_pin);
   connect_pin(GNDU, l_pin);

   force_v(h_pin, v_force, 0, comp);

   measure_i(h_pin, &current, comp);

   disable_port(h_pin);
   disconnect_all();

   printf("I = %8.5f amp (at %8.5f volt)\n", current, v_force);
   printf("R = %8.3f ohm\n", v_force / current);

#ifdef E5270_TIS_H
   close_E5270();
#endif
   return( 0 );
}
```

Debugging

The port_status function returns measurement status of specified port. The following table shows all statuses returned by port_status or status_miv function.

Unit	Returned Value	Condition
SMU	NORMAL_MEAS(=0)	Normal
	DCS_COMP_OTHER(=1)	Another unit has reached compliance
	DCS_COMP(=2)	This unit has reached compliance
	DCS_OSC(=3)	This unit is oscillating.
	DCS_OVERFLOW(=4)	Measurement data exceeds the measurement range.
	DCS_SWP_STOPPED (=5)	Sweep was aborted by compliance condition. (stop mode = 2 in set_iv or set_piv)
GNDU	NORMAL_MEAS(=0)	Normal

NOTE

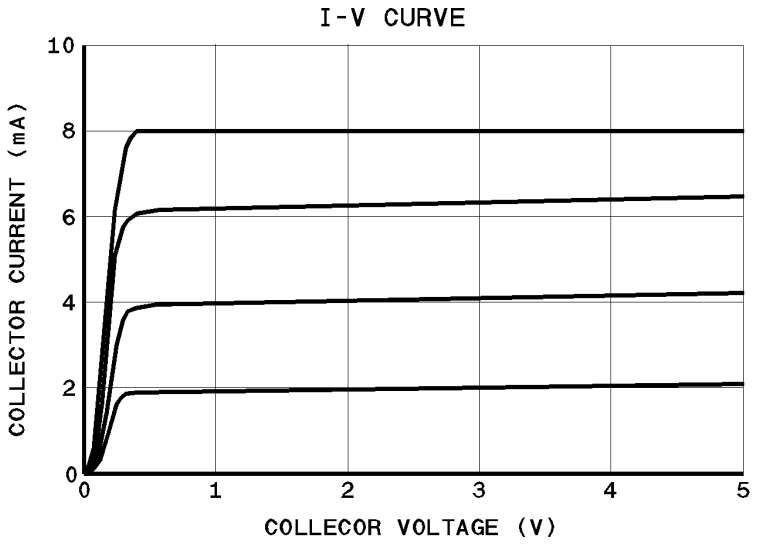
If the port_status function is executed for a measurement port that just finished a sweep measurement (sweep_iv or sweep_miv), only the status for last sweep step is returned.

Monitoring if SMU Reaches Compliance

You can use the port_status function to monitor whether an SMU reached saturation, that is, the compliance value set by force_i, force_v, set_iv, and so on.

Figure 1-2 shows an example in which the compliance value was set to 8 mA for the SMU. As shown in the graph, the top curve is flat and does not increase because the compliance is set to 8 mA.

Figure 1-2 **Reaching Compliance (8 mA): Ic-Vce Characteristics**

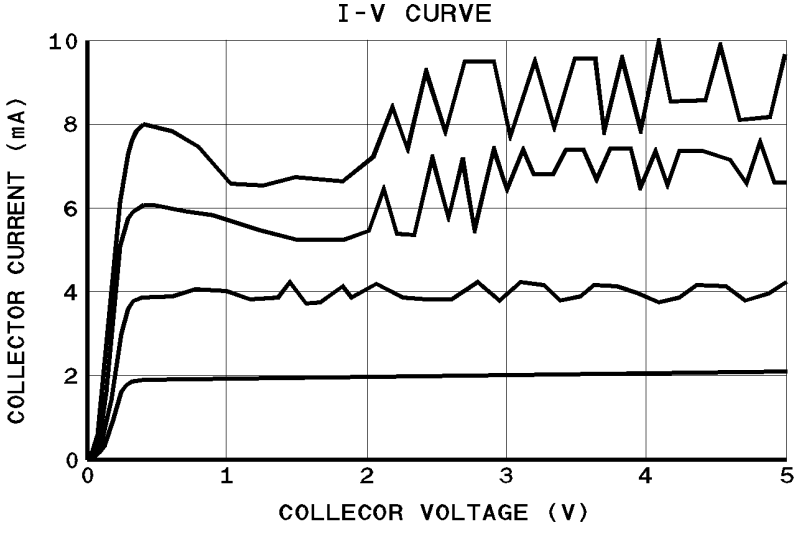


Monitoring if SMU Oscillates

You can use the port_status function to monitor if an SMU oscillated.

Figure 1-3 shows an example graph for oscillation.

Figure 1-3 **Oscillation: Ic-Vce Characteristics**



Migration to Agilent 4070

The Agilent E5270 TIS library is a set of functions that provide the same synopsis as the Agilent 4070 TIS functions. So, if you create programs using the Agilent E5270 TIS library, the programs can be used to control the Agilent 4070 tester with small change.

To migrate from the Agilent E5270 to Agilent 4070 successfully, perform the following procedure.

1. Open the program on the Agilent 4070 system controller (HP-UX computer).
2. Change the header file. The header file `E5270_TIS.h` is not required. Include the `tis.h` file instead.

   ```
   #include "/opt/hp4070/include/tis.h"
   ```

3. Remove the following functions. They are not required for the Agilent 4070.
   - open_E5270
   - close_E5270
4. Remove the VXI*plug&play* driver functions for Agilent E5270 if they are used in the program. The driver cannot be used to control the Agilent 4070. This part of program should be created and added to the program for the Agilent 4070.
5. Remove the `vi_E5270` function if it is used in the program. The function cannot be used to control the Agilent 4070. This part of program should be created and added to the program for the Agilent 4070.
6. Refer to Table 1-3 and correct differences between the E5270 TIS and 4070 TIS.

NOTE

E5270_TIS_H macro

If the program includes the E5270_TIS_H macro as shown below, this part of program does not have to be deleted.

```
#ifdef E5270_TIS_H
   /* codes for E5270 */
#else
   /* codes for 4070 */
#endif
```

Codes for 4070 will be performed if the E5270_TIS_H macro is not effective.

Table 1-3 Differences between E5270 TIS and 4070 TIS

Function or Item	Explanation
connect_pin	The *pin* value must be a pin number actually available for the switching matrix of the 4070.
set_adc	The parameter values are compatible. But the actual setup value is different for INTEG_SHORT.
set_bdv and set_ileak	Followings are different between the E5270 and 4070. • Resolution of *hold* (0.001 for 4070, 0.01 for E5270) • Maximum *delay* (65.535 s for 4070, 6.5535 s for E5270)
set_pbias and set_piv	Followings are different between the E5270 and 4070. • Restrictions of setting *period* and *width* values • *period* and *width* values automatically set when you specify *period*=0
set_piv	Logarithmic sweep is available for 4070.
set_sync	The *offset* value is effective for logarithmic sweep using 4070.
set_timestamp	For 4070, this function must be executed before the port_status or status_miv function.
sweep_iv and sweep_miv	Returned data when a sweep abort condition occurred is different as shown in Table 1-4 and Table 1-5. The tables show an example that an abort condition occurs at step=N.
Current range value	200 mA range is not supported by the 4070. Use 1 A range of HPSMU instead. For 4070, 100 pA range is also available.
Initial settings	They are not the same because of the differences of hardware. If your program includes error handling part, check the error code, and correct the program.
Error codes and messages	

Table 1-4 **E5270 returned data when a sweep abort condition occurred**

step	*measure* or *meas_vals*	*source*	*sync*	*statuses*	*times*
1	(measured data)	(output data)	(output data)	0	(time data)
:	:	:	:	:	:
N-1	(measured data)	(output data)	(output data)	0	(time data)
N	(measured data)	(output data)	(output data)	X[a]	(time data)
N+1	9999999.99999	0	0	5	0
:	:	:	:	:	:
Last	9999999.99999	0	0	5	0

a. X will be a status code that indicates a sweep abort condition.

Table 1-5 **4270 returned data when a sweep abort condition occurred**

step	*measure* or *meas_vals*	*source*	*sync*	*statuses*	*times*
1	(measured data)	(output data)	(output data)	0	(time data)
:	:	:	:	:	:
N-1	(measured data)	(output data)	(output data)	0	(time data)
N	9999999.99999	(output data)	(output data)	X[a]	(time data)
N+1	(no data)	(no data)	(no data)	5	0
:	:	:	:	:	:
Last	(no data)	(no data)	(no data)	5	0

a. X will be a status code that indicates a sweep abort condition.

2 TIS Function Reference

This chapter is the complete reference of the Agilent E5270 TIS functions.

- “How to Use the Reference Pages”
- “Function Reference”

How to Use the Reference Pages

Each description module for a function consists of the following sections:

- Function name
- Synopsis
- Arguments
- Example
- See Also

The description module will not contain the section that does not apply to the function.

Function Name

The function name is shown at the top of the description module with a brief description.

Synopsis

The synopsis of the function is printed in `computer` font as shown in the following example:

```
int force_v(port,voltage,range,compliance)

int port;
double voltage,range,compliance;
```

The first line shows the function type and arguments of the function, and remaining lines show type of each argument.

Arguments

The arguments of a function specify values to pass to the function or where to return values to a function.

This section briefly lists and describes each argument in a table. The details about each argument are described in the Description section. The arguments are indicated by *italic* type in the text.

Several pre-defined macros are provided that can be used for some function arguments. The allowed macros for each function are listed. The macros are UPPERCASE, so the macro must be UPPERCASE letters when used in a program.

NOTE

Return value

The functions of which the return value is not described exit with one of the following value:

0:	Successful completion
-1:	Error occurred

Example

The Example section provides an example of the function. The example is taken from an actual program that uses the function. The example is indicated by `computer` font.

See Also

The See Also section lists the related functions.

Function Reference

This section describes the Agilent E5270 TIS functions. The functions are listed in alphabetical order.

Summary

Table 2-1 summarizes the Agilent E5270 TIS functions.

Table 2-1 **Agilent E5270 TIS Function Summary**

Category	Function	Summary
Starting	open_E5270	Establishes the software connection to the Agilent E5270.
Ending	close_E5270	Terminates the software connection to the Agilent E5270.
Reset	init_system	Initializes the Agilent E5270.
Query	error_info	Returns a TIS error number and an error message string.
	port_status, port_status_t	Returns the status of the specified port and time stamp.
	status_miv	Returns the status of the data measured by sweep_iv or sweep_miv function, and time stamp.
Channel/Port control	connect_pin	Enables source/measurement channel.
	disconnect_all	Disables all channels.
	set_smu_ch	Selects A/D converter type and filter ON or OFF for the specified SMU.
	disable_port	Sets the specified channel to the zero output state.
A/D converter setup	set_adc	Sets the operation parameters of A/D converter.

Category	Function	Summary
Time stamp setup	reset_timestamp	Enables to use the force_i or force_v function to clear the time stamp.
	set_timestamp	Enables or disables the time stamp data output.
Source output	force_i	Forces current from the specified SMU.
	force_v	Forces voltage from the specified SMU.
High speed spot measurement	measure_i, measure_it	Measures DC current and returns the measurement data and time stamp.
	measure_v, measure_vt	Measures DC voltage and returns the measurement data and time stamp.
Multi channel spot measurement	measure_m	Performs multi channel DC voltage or current measurements using up to eight SMU channels.
Pulsed spot measurement	force_i	Forces current from the specified SMU.
	force_v	Forces voltage from the specified SMU.
	set_pbias	Sets the pulse source parameters.
	measure_p	Performs a pulsed spot measurement and returns the measurement value.
Staircase sweep measurement	force_i	Forces current from the specified SMU.
	force_v	Forces voltage from the specified SMU.
	set_iv	Sets the setup parameters for staircase sweep measurements.
	set_sync	Sets the synchronous sweep source.
	sweep_iv	Performs sweep measurement and returns measurement data.

Category	Function	Summary
Staircase sweep measurement using multiple measurement channels	force_i	Forces current from the specified SMU.
	force_v	Forces voltage from the specified SMU.
	set_iv	Sets the setup parameters for staircase sweep measurements.
	set_sync	Sets the synchronous sweep source.
	sweep_miv	Performs sweep measurements using the up to 8 measurement channels and returns measurement data.
Pulsed sweep measurement	force_i	Forces current from the specified SMU.
	force_v	Forces voltage from the specified SMU.
	set_piv	Sets the setup parameters for pulsed sweep measurements.
	set_sync	Sets the synchronous sweep source.
	sweep_iv	Performs pulsed sweep measurement and returns measurement data.
Staircase sweep with pulsed bias measurement	force_i	Forces current from the specified SMU.
	force_v	Forces voltage from the specified SMU.
	set_iv	Sets the setup parameters for staircase sweep measurements.
	set_pbias	Sets the pulse source parameters.
	set_sync	Sets the synchronous sweep source.
	sweep_iv	Performs sweep measurement and returns measurement data.

Category	Function	Summary
Breakdown voltage measurement	force_i	Forces current from the specified SMU.
	force_v	Forces voltage from the specified SMU.
	set_bdv	Sets the setup parameters for breakdown voltage measurement.
	measure_bdv	Performs breakdown voltage measurement and returns measurement data.
Leakage current measurement	force_i	Forces current from the specified SMU.
	force_v	Forces voltage from the specified SMU.
	set_ileak	Sets the setup parameters for leakage current measurement.
	measure_ileak	Performs leakage current measurement and returns measurement data.

Arguments

The arguments used by several functions are explained in this section.

- "Port Address"
- "Current Measurement Range and Resolution"
- "Voltage Measurement Range and Resolution"
- "Voltage Source Setup Parameters"
- "Current Source Setup Parameters"

Table 2-2 **Port Address**

Macro	Value [a]	Description
SMU1	20001	Specifies SMU installed in the Agilent E5270. For the Agilent E5272A and E5273A, SMU1 always specifies the left side SMU viewed from the front. SMU3 to SMU8 are not available. For the Agilent E5270A, SMU1 always specifies the SMU installed in a slot closest to the slot 1. SMU2 and followings specify SMU following to the previous one. where slot 1 is the left top slot, and slot 8 is the right bottom slot. Available port addresses depend on the number of SMUs installed in the Agilent E5270.
SMU2	20002	
SMU3	20003	
SMU4	20004	
SMU5	20005	
SMU6	20006	
SMU7	20007	
SMU8	20008	
GNDU	20009	Specifies the ground unit. Available for the connect_pin, port_status, and port_status_t functions.

a. For program readability and future compatibility, it is recommended not to use this value directly but use macro instead.

Table 2-3 Current Measurement Range and Resolution

range **value**	**Measurement Range**	**Measurement Resolutions** [a]	
		High Speed ADC	**High Resolution ADC**
0 < |*range*| ≤ 1.15 nA	1 nA	50 fA	10 fA [b]
1.15 nA < |*range*| ≤ 11.5 nA	10 nA	500 fA	10 fA
11.5 nA < |*range*| ≤ 115 nA	100 nA	5 pA	100 fA
115 nA < |*range*| ≤ 1.15 µA	1 µA	50 pA	1 pA
1.15 µA < |*range*| ≤ 11.5 µA	10 µA	500 pA	10 pA
11.5 µA < |*range*| ≤ 115 µA	100 µA	5 nA	100 pA
115 µA < |*range*| ≤ 1.15 mA	1 mA	50 nA	1 nA
1.15 mA < |*range*| ≤ 11.5 mA	10 mA	500 nA	10 nA
11.5 mA < |*range*| ≤ 115 mA	100 mA	5 µA	100 nA
115 mA < |*range*| ≤ 200 mA (for MPSMU)	200 mA	10 µA	200 nA
115 mA < |*range*| ≤ 1 A (for HPSMU)	1 A	50 µA	1 µA

a. You can select ADC by using the set_smu_ch function.
b. The resolution of returned value is (*range value*)/1000000.

Table 2-4 Voltage Measurement Range and Resolution

range value	Measurement Range	Measurement Resolution [a]	
		High-speed ADC	High-resolution ADC
$0 < \|range\| \leq 2$ V	2 V	100 µV	2 µV
$2 \text{ V} < \|range\| \leq 20$ V	20 V	1 mV	20 µV
$20 \text{ V} < \|range\| \leq 40$ V	40 V	2 mV	40 µV
$40 \text{ V} < \|range\| \leq 100$ V	100 V	5 mV	100 µV
$100 \text{ V} < \|range\| \leq 200$ V (for HPSMU)	200 V	10 mV	100 µV

a. You can select ADC by using the set_smu_ch function.

Table 2-5 Voltage Source Setup Parameters

range value (Output Range)	Output Value [a]	Setting Resolution	Maximum Compliance	
			MPSMU	HPSMU
$0 < \|range\| \leq 2$ V (2 V)	$0 \leq \|V\| \leq 2$ V	100 µV	±200 mA	±1000 mA
$2 \text{ V} < \|range\| \leq 20$ V (20 V)	$0 \leq \|V\| \leq 20$ V	1 mV	±200 mA	±1000 mA
$20 \text{ V} < \|range\| \leq 40$ V (40 V)	$0 \leq \|V\| \leq 20$ V	2 mV	±200 mA	±500 mA
	$20 \text{ V} < \|V\| \leq 40$ V		±50 mA	±125 mA
$40 \text{ V} < \|range\| \leq 100$ V (100 V)	$0 \leq \|V\| \leq 20$ V	5 mV	±200 mA	±1000 mA
	$20 \text{ V} < \|V\| \leq 40$ V		±50 mA	±500 mA
	$40 \text{ V} < \|V\| \leq 100$ V		±20 mA	±125 mA
$100 \text{ V} < \|range\| \leq 200$ V (200 V for HPSMU)	$0 \leq \|V\| \leq 20$ V	10 mV	–	±1000 mA
	$20 \text{ V} < \|V\| \leq 40$ V		–	±500 mA
	$40 \text{ V} < \|V\| \leq 100$ V		–	±125 mA
	$100 \text{ V} < \|V\| \leq 200$ V		–	±50 mA

a. Setting resolution is determined by actual output range used to force the output voltage.

Table 2-6 Current Source Setup Parameters

range value (Output Range)	Output Value [a]	Setting Resolution	Maximum Compliance	
			MPSMU	HPSMU
0 < |*range*| ≤ 1 nA (1 nA)	0 ≤ |I| ≤ 1 nA	50 fA	±100 V	±200 V
1 nA < |*range*| ≤ 10 nA (10 nA)	0 ≤ |I| ≤ 10 nA	500 fA	±100 V	±200 V
10 nA < |*range*| ≤ 100 nA (100 nA)	0 ≤ |I| ≤ 100 nA	5 pA	±100 V	±200 V
100 nA < |*range*| ≤ 1 µA (1 µA)	0 ≤ |I| ≤ 1 µA	50 pA	±100 V	±200 V
1 µA < |*range*| ≤ 10 µA (10 µA)	0 ≤ |I| ≤ 10 µA	500 pA	±100 V	±200 V
10 µA < |*range*| ≤ 100 µA (100 µA)	0 ≤ |I| ≤ 100 µA	5 nA	±100 V	±200 V
100 µA < |*range*| ≤ 1 mA (1 mA)	0 ≤ |I| ≤ 1 mA	50 nA	±100 V	±200 V
1 mA < |*range*| ≤ 10 mA (10 mA)	0 ≤ |I| ≤ 10 mA	500 nA	±100 V	±200 V
10 mA < |*range*| ≤ 100 mA (100 mA)	0 ≤ |I| ≤ 20 mA	5 µA	±100 V	±200 V
	20 mA < |I| ≤ 50 mA		±40 V	
	50 mA < |I| ≤ 100 mA		±20 V	±100 V
100 mA < |*range*| ≤ 200 mA (200 mA for MPSMU)	0 ≤ |I| ≤ 20 mA	10 µA	±100 V	–
	20 mA < |I| ≤ 50 mA		±40 V	–
	50 mA < |I| ≤ 200 mA		±20 V	–
100 mA < |*range*| ≤ 1 A (1 A for HPSMU)	0 ≤ |I| ≤ 50 mA	50 µA	–	±200 V
	50 mA < |I| ≤ 125 mA		–	±100 V
	125 mA < |I| ≤ 500 mA		–	±40 V
	500 mA < |I| ≤ 1 A		–	±20 V

a. Setting resolution is determined by actual output range used to force the output current.

close_E5270

This function terminates the software connection to the Agilent E5270 and deallocates system resources. This function must be executed to close the instrument handle when the program is done using the Agilent E5270.

Synopsis

```
int close_E5270(void)
```

Example

```
close_E5270();
```

See Also

- “open_E5270”

connect_pin

This function enables the channel specified by *port*. Also, this command assigns a *pin* number to the specified *port*. After this command, you can use *pin* number instead of *port* to specify the channel.

The connect_pin(0,0) function clears all assignments.

Synopsis

```
int connect_pin(port,pin)
int port, pin;
```

Arguments

Item	Range Restrictions/Description
port	Specify channel to enable. Available *port* values are 0, GNDU, SMU1 to SMU8, or values shown in Table 2-2. *port*=0 clears the port assignment for the specified *pin*.
pin	This item is necessary to keep compatibility with the Agilent 4070's connect_pin function. For the Agilent 4070, this item specifies switching matrix pin number to connect to the specified *port*. For the Agilent E5270, the *pin* value can be used in other functions instead of the *port* number to specify the channel. Available *pin* values are 0 to 49. *pin* =0 clears the pin assignment for the specified *port*.

One *port* can be specified by multiple *pins* by executing multiple connect_pin functions. However, one *pin* can specify only one *port*. The most recent function is effective for the *pin*.

For the Agilent 4070, *pin*=1 to 48 specify the switching matrix pin numbers. *pin*=49 specifies the connector for chuck connection.

Example

```
int err;
err=connect_pin(SMU1,12);
```

See Also

- "disconnect_all"

disable_port

This function sets the specified channel to the zero output state as shown in Table 2-7.

Synopsis

```
int disable_port(port)
int port;
```

Arguments

Item	Range Restrictions/Description
port	Specify channel to be set to the zero output state by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 0 to 49, SMU1 to SMU8, or values shown in Table 2-2 on page 2-9. *port*=0 sets all ports to the zero output state.

Example

```
int err;
err=disable_port(0);
```

See Also

- “connect_pin”

Table 2-7 Zero Output State

Setup Item	Setting Value
Output Mode	Voltage output
Output Range	No change or 20 V if SMU was in I source mode.
Output Voltage	0 V
Current Compliance	100 mA

disconnect_all

This function disables all channels, and clears all assignments set by the connect_pin functions.

This function is equivalent to the connect_pin(0,0) function, and is recommended for better program readability.

Synopsis

```
int disconnect_all()
```

Example

```
int err;

err=disconnect_all();
```

See Also

- “connect_pin”

error_info

This function returns a TIS error number and an error message string.

This function checks the completion status of the most recently executed TIS function, and returns the error numbers and error message string to *errn* and *errm*, respectively.

Synopsis

```
int err_info(errn, errm)

int errn[2];
char errm[1024];
```

Arguments

Item	Range Restrictions/Description
errn	Pointer to an integer array in which to return the TIS error number. **errn[0]:** TIS error number **errn[1]:** CMS, DVM, or PGU error number that is meaningful for the Agilent 4070, but meaningless for the Agilent E5270. Always 0 is returned. Must be a pointer to an integer array with size = 2.
errm	Pointer to a character array in which to return the TIS error message. A NULL terminated string is returned to the array. Must be a pointer to an character array with size 1024. Or you can specify NULL for *errm* if you want to discard the message.

The first element of the *errn* array is the TIS error number. The second element is necessary to keep compatibility with the Agilent 4070's error_info function.

If the init_system function is executed, both error numbers are set to 0, and the error message string is set to a null string.

No value is returned to the function itself, even if an error occurs.

Example

```
error_info(error_number, error_msg);
```

force_i

This function forces the specified current from the specified SMU.

Synopsis

```
int force_i(port, current, range, compliance)

int port;
double current, range, compliance;
```

Arguments

Item	Range Restriction/Description
port	Specify current output channel by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
current	Value of current to force. Numeric expression [A]. See Table 2-6 on page 2-12. −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU.
range	Current output range. Numeric expression [A]. See Table 2-6 on page 2-12. −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. [a] To use the auto ranging mode, set 0. [b]
compliance	Voltage compliance value. Numeric expression [V]. See Table 2-6 on page 2-12. The *compliance* sets the maximum output voltage of an SMU operating as a current source (I_SOURCE mode). −100 to 100 for MPSMU or −200 to 200 for HPSMU. A macro REMAIN is also available. If you set REMAIN, the compliance value is automatically set to the last compliance value if the last output mode of this channel was I_SOURCE, or 20 V if the last output mode was V_SOURCE.

a. SMU uses limited auto ranging mode. The SMU forces current at the specified range if the specified range can force the *current*. If not, the SMU ranges up to range that can force the *current*.
b. SMU forces current at the lowest range that can force the *current*.

The output range actually used to force the *current* determines the setting resolution of the *current*. As shown in Table 2-6 on page 2-12, lower output range has better setting resolution. But a lower output range may increase settling time.

If you specify a value that is not an exact output range listed in Table 2-6 on page 2-12, the range is used according to the table. For example, if you set *current* and *range* to 11 μA, the 100 μA range is used.

Example

```
int err;

err=force_i(SMU1, i_ds1, i_ds2, 2.0);
```

force_v

This function forces the specified voltage from the specified SMU.

Synopsis

```
int force_v(port, voltage, range, compliance)

int port;
double voltage, range, compliance;
```

Arguments

Item	Range Restrictions/Description
port	Specify voltage output channel by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
voltage	Value of voltage to force. Numeric expression [V]. See Table 2-5 on page 2-11. −100 to 100 for MPSMU or −200 to 200 for HPSMU.
range	Voltage output range. Numeric expression [V]. See Table 2-5 on page 2-11. −100 to 100 for MPSMU, −200 to 200 for HPSMU. [a] To use the auto ranging mode, set 0. [b]
compliance	Current compliance value. Numeric expression [A]. See Table 2-5 on page 2-11. −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. The *compliance* sets the maximum output current of an SMU operating as a voltage source (V_SOURCE mode). A macro REMAIN is also available. If you set REMAIN, the compliance value is automatically set to the last compliance value if the last output mode of this channel was V_SOURCE, or 100 µA if the last output mode was I_SOURCE.

a. SMU forces voltage by using the range that covers the *range* value.
b. SMU forces voltage at the lowest range that can force the *voltage*.

The output range actually used to force the *voltage* determines the setting resolution of the *voltage*. As shown in Table 2-5 on page 2-11, lower output range has better setting resolution. But a lower output range may increase settling time.

Example

```
if (force_v(SMU1, v_ds1, v_ds2, .05) == -1) error_rep();
```

init_system

This function initializes the Agilent E5270.

Execute the init_system function before executing any other TIS function.

Synopsis

```
int init_system()
```

Example

```
if (init_system() == -1) error_rep();
```

measure_bdv

This function triggers quasi-pulsed measurements to measure breakdown voltage, then returns the breakdown voltage. If this function finishes successfully, a 0 is returned. If not, -1 is returned.

The conditions of the quasi-pulsed measurements are determined by the set_bdv function. See “set_bdv” on page 2-40.

Synopsis

```
int measure_bdv(voltage, status, interval)

double *voltage;
int *status,interval;
```

Arguments

Item	Range Restrictions/Description
voltage	Specify a pointer to double variable in which to return the measured voltage. The measured value (breakdown voltage) is returned.
status	Specify a pointer to integer variable in which to return the measurement status code. Returns one of the status code shown in Table 2-8.
interval	Specify measurement interval, BDV_INTVL_SHORT (0) or BDV_INTVL_LONG (1).

NOTE

Interval

The *interval* defines how often the voltage gradient is monitored (calculated). If the *interval* is set to BDV_INTVL_SHORT, the gradient is calculated after each voltage measurement. If the *interval* is set to BDV_INTVL_LONG, the gradient is calculated after every 10 voltage measurements.

When the *interval* is set to BDV_INTVL_SHORT, a slew rate of less than 1 V/ms causes an error (*status* = BDV_TOO_SLOW).

When the *interval* is set to BDV_INTVL_LONG, a slew rate of less than 0.1 V/ms causes an error (*status* = BDV_TOO_SLOW).

The time-out for the quasi-pulsed measurement is set to 3 seconds for BDV_INTVL_SHORT mode and 12 seconds for BDV_INTVL_LONG mode.

Table 2-8 **Status Code of measure_bdv**

Status Code [a]		Description
0	BDV_NORMAL	The quasi-pulsed measurement ended normally.
1	BDV_COMP_OTHER	Another unit reached compliance.
2	BDV_CURRENT_LOW	Breakdown voltage measurement failed because current did not reach breakdown *current* within the stop voltage specified by set_bdv function.
3	BDV_OSC	This unit is oscillating.
4	BDV_OVERFLOW	Measurement overflow occurred while monitoring output voltage or measuring breakdown voltage.
6	BDV_TIMEOUT	The quasi-pulsed measurement did not reach breakdown *current* within time-out.
7	BDV_TOO_SLOW	The monitored slew rate of the output voltage is too small.

a. One of the above integers is returned to *status*. You can also use the associated macros in your program to determine which status was returned.

Example

```
int st;
double v_meas;

……

if (measure_bdv(&v_meas,&st,BDV_INTVL_SHORT)==-1) error_rep();
if(st > 5)
if (measure_bdv(&v_meas,&st,BDV_INTVL_LONG)==-1) error_rep();
```

See Also

- “set_bdv”

measure_i, measure_it

These functions measure DC current using a specified SMU and returns the measurement value in amperes.

Synopsis

```
int measure_i(port, current, range)
int port;
double *current, range;

int measure_it(port, current, range, time_stamp)
int port;
double *current, range, *time_stamp;
```

Arguments

Item	Range Restrictions/Description
port	Specify current measurement channel by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
current	Specify a pointer to double variable in which to return the measurement value.
range	Current measurement range. Numeric expression [A]. See Table 2-3 on page 2-10. −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. [a] To use the auto ranging mode, set 0. [b] If the SMU is I_SOURCE mode, set any value. [c]
time_stamp [d]	Specify a pointer to double variable in which to return the time stamp value. Returns accumulated time (in seconds) from when the timer count is cleared until the measurement is completed.

a. SMU uses limited auto ranging mode. The SMU uses the specified range if possible, but changes to a higher range if not possible.
b. SMU uses the range that gives the best measurement resolution.
c. This is a dummy parameter. Measurement range is actually set to same as output current range. Refer to force_i function for the output range.
d. Only for the measure_it function. The data will be returned if the time stamp data output is enabled by the set_timestamp function.

NOTE

Limited Auto Ranging

Limited auto-ranging can reduce the measurement time by avoiding unnecessary range changing.

Setting the *range* of SMU to a low value improves measurement resolution but may increase measurement time because of additional ranging and wait times.

NOTE

To Clear Timer Count

Execute the reset_timestamp function. Then the timer count is not cleared. Execute the force_i or force_v function. Then the timer count will be cleared.

Once the force_i or force_v function is executed, the effects of the reset_timestamp function will be disabled.

Example

```
if (measure_i(SMU1, &current, 1E-3) == -1) error_rep();
```

See Also

- “reset_timestamp”

measure_ileak

This function triggers the quasi-pulsed measurement to measure leakage current according to the conditions set by the set_ileak function, then returns the measurement value to the *current* variable. See "set_ileak" on page 2-42.

Synopsis

```
int measure_ileak(port, current, status, interval)

int port, *status, interval;
double *current;
```

Arguments

Item	Range Restrictions/Description
port	Specify current measurement channel by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
current	Specify a pointer to double variable in which to return the measured current. The measured value (leakage current) is returned.
status	Specify a pointer to integer variable in which to return the measurement status code. Returns one of the status code shown in Table 2-9.
interval	Specify measurement interval, ILEAK_INTVL_SHORT (= 0) or ILEAK_INTVL_LONG (= 1).

NOTE

Interval

The *interval* defines how often the voltage gradient is monitored (calculated). If the *interval* is set to ILEAK_INTVL_SHORT, the gradient is calculated after each voltage measurement. If the *interval* is set to ILEAK_INTVL_LONG, the gradient is calculated after every 10 voltage measurements.

When the *interval* is set to ILEAK_INTVL_SHORT, a slew rate of less than 1000 V/s (= 1 V/ms) causes an error (*status* = ILEAK_TOO_SLOW). When the *interval* is set to ILEAK_INTVL_LONG, a slew rate of less than 100 V/s (= 0.1 V/ms) causes an error (*status* = ILEAK_TOO_SLOW).

The time-out is set to 3 seconds for ILEAK_INTVL_SHORT mode and 12 seconds for ILEAK_INTVL_LONG mode.

Table 2-9 **Status Code of measure_ileak**

Status Code [a]		Condition
0	ILEAK_NORMAL	The quasi-pulsed measurement ended normally.
1	ILEAK_COMP_OTHER	Another unit reached compliance.
2	ILEAK_COMP	This unit reached compliance.
3	ILEAK_OSC	This unit is oscillating.
4	ILEAK_OVERFLOW	Measurement overflow occurred while monitoring the output voltage or measuring the leakage current.
6	ILEAK_TIMEOUT	The quasi-pulsed measurement did not reach *output* (specified by set_ileak) within the time-out.
7	ILEAK_TOO_SLOW	The monitored slew rate of the output voltage is too small.

a. One of the above integers is returned to *status*. You can also use the associated macros in your program to determine which status was returned.

Example

```
int st;
double ileak;

……

if (measure_ileak(SMU2, &ileak, &st, ILEAK_INTVL_SHORT)==-1)
error_rep();
if (st>5)
if (measure_ileak(SMU2, &ileak, &st, ILEAK_INTVL_LONG)==-1)
error_rep();
```

See Also

- “set_ileak”

measure_m

This function performs multi channel DC voltage or current measurements using up to eight SMU channels. The channels perform measurement sequentially in the order defined in the *ports*.

Before this function, execute force_i or force_v function to set the measurement mode for each channel. The force_i function sets the channel to voltage measurement mode, and force_v sets the channel to current measurement mode.

Synopsis

```
int measure_m(n, ports, meas_vals, ranges, time_stamps)

int n, ports[n];
double meas_vals[n], ranges[n], time_stamps[n];
```

Arguments

Item	Range Restrictions/Description
n	Specify how many ports will perform measurement. The size of the *ports*, *meas_vals*, *ranges*, and *time_stamps* arrays must be *n*. *n* can be from 1 to 8.
ports	Specify a pointer to an integer array that contains the measurement channels. This array must have *n* elements. The array should contain the port addresses or pin numbers that are assigned to the port by the connect_pin function. Available values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
meas_vals	Specify a pointer to double array in which to return the measured values.This array must have *n* elements. Measurement results are returned to *meas_vals* array in order that corresponds to elements in the *ports* array.

Item	Range Restrictions/Description
ranges	Specify a pointer to double array that contains the measurement range for each channel. This array must have *n* elements. Specify values in *ranges* array in order that corresponds to elements of the *ports* array. See “measure_i, measure_it” on page 2-24 for current measurement range. See “measure_v, measure_vt” on page 2-32 for voltage measurement range.
time_stamps [a]	Specify a pointer to a double array in which to return the time stamps for measured values. Returns accumulated time (in seconds) from when the timer count is cleared until the measurement is completed.

a. The data will be returned if the time stamp data output is enabled by the set_timestamp function.

NOTE

To Clear Timer Count

Execute the reset_timestamp function. Then the timer count is not cleared. Execute the force_i or force_v function. Then the timer count will be cleared.

Once the force_i or force_v function is executed, the effects of the reset_timestamp function will be disabled.

Example

```
int err;
……

int n=3,ports[3];
double meas_vals[3], ranges[3], time_stamps[3];
ports[0]=8; ports[1]=12; ports[2]=16;
ranges[0]=0.0; ranges[1]=0.0; ranges[2]=0.0;
……

err=measure_m(n, ports, meas_vals, ranges, time_stamps);
```

See Also

- “measure_i, measure_it”, “measure_v, measure_vt”, and “reset_timestamp”

measure_p

This function performs a pulsed spot measurement according to conditions set by the set_pbias function, and returns the measurement value.

Synopsis

```
int measure_p(port, mode, value, range)

int port, mode;
double *value,range;
```

Arguments

Item	Range Restrictions/Description
port	Specify measurement channel by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9. If you specify 0, no measurement is executed and this function just triggers to force the pulsed bias that is set by set_pbias statement.
mode	Specify whether to perform a voltage or current measurement. Use one of the following macros (or values): **V_MEAS (1)** Voltage measurement mode. The channel must be a current source. **I_MEAS (2)** Current measurement mode. The channel must be a voltage source.
value	Specify a pointer to a double variable in which to return the measurement result.
range	Specify the measurement range. For voltage measurements [V]: See Table 2-4 on page 2-11. −100 to 100 for MPSMU or −200 to 200 for HPSMU 0.0 automatically sets *range* to the voltage compliance setting. For current measurements [A]: See Table 2-3 on page 2-10. −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU 0.0 automatically sets *range* to the current compliance setting.

NOTE

A/D Converter

The pulsed spot measurement always uses high-speed ADC. And the number of averaging samples is set to 1. The resolution of returned value is (measurement range value)/20000.

Example

```
int err;
double current;

……

err=set_pbias(SMU1, V_SOURCE, 20, -3, 15, 0.4, 1, 0.5, 1E-3);
err=measure_p(SMU1, I_MEAS, current, 1E-3);
```

See Also

- “set_pbias”

measure_v, measure_vt

The function measures DC voltage using the specified SMU and returns the measurement value to the *voltage* variable.

Synopsis

```
int measure_v(port, voltage, range)
int port;
double *voltage, range;

int measure_vt(port, voltage, range, time_stamp)
int port;
double *voltage, range, *time_stamp;
```

Arguments

Item	Range Restrictions/Description
port	Specify measurement channel by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
voltage	Specify a pointer to a double variable in which to return the measurement result.
range	Voltage measurement range. Numeric expression [V]. See Table 2-4 on page 2-11. −100 to 100 for MPSMU or −200 to 200 for HPSMU. [a] To use the auto ranging mode, set 0. [b] If the SMU is V_SOURCE mode, set any value. [c]
time_stamp [d]	Specify a pointer to double variable in which to return the time stamp value. Returns accumulated time (in seconds) from when the timer count is cleared until the measurement is completed.

a. SMU uses limited auto ranging mode. The SMU uses the specified range if possible, but changes to a higher range if not possible.
b. SMU uses the range that gives the best measurement resolution.
c. This is a dummy parameter. Measurement range is actually set to same as output voltage range. Refer to force_v function for the output range.
d. Only for the measure_vt function. The data will be returned if the time stamp data output is enabled by the set_timestamp function.

NOTE

Limited Auto Ranging

Limited auto-ranging can reduce the measurement time by avoiding unnecessary range changing.

Setting the *range* of SMU to a low value improves measurement resolution but may increase measurement time because of additional ranging and wait times.

NOTE

To Clear Timer Count

Execute the reset_timestamp function. Then the timer count is not cleared. Execute the force_i or force_v function. Then the timer count will be cleared.

Once the force_i or force_v function is executed, the effects of the reset_timestamp function will be disabled.

Example

```
if (measure_v(SMU2, &v_th1, 1.8) == -1) error_rep();
```

See Also

- “reset_timestamp”

port_status, port_status_t

This function returns the status of the specified port. This status indicates the condition of port after most recent measurement was performed by port.

Synopsis

```
int port_status(port, status)
int port, *status;

int port_status_t(port, status, time_stamp)
int port, *status;
double *time_stamp;
```

Arguments

Item	Range Restrictions/Description
port	Specify measurement channel by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, GNDU, and values shown in Table 2-2 on page 2-9.
status	Specify a pointer to integer variable in which to return the status code. Returns one of the status code shown in Table 2-10.
time_stamp [a]	Specify a pointer to double variable in which to return the time stamp value. Returns accumulated time (in seconds) from when the timer count is cleared until the last measurement is completed. If the port is not a measurement port, such as GNDU, 0.0 is always returned to *time_stamp*.

a. Only for the port_status_t function. The data will be returned if the time stamp data output is enabled by the set_timestamp function.

NOTE

status

If two or more conditions occur during a measurement, the priority of the returned numerical code is the following order:

`DCS_OSC > DCS_OVERFLOW > DCS_COMP > DCS_COMP_OTHER`

For example, if the SMU reached compliance and was oscillating during the measurement, `DCS_OSC` (oscillating) is returned to the *status* variable. This *status* variable remains unchanged until the next measurement.

Table 2-10 **Status Code of port_status and port_status_t**

Status Code [a]		Description
0	NORMAL_MEAS [b]	Normal.
1	DCS_COMP_OTHER	Another unit has reached compliance.
2	DCS_COMP	This unit has reached compliance.
3	DCS_OSC	This unit is oscillating.
4	DCS_OVERFLOW	Measurement overflow.
5	DCS_SWP_STOPPED	Sweep was aborted by compliance condition. (stop mode = 2 in set_iv or set_piv)

a. One of the above integers is returned to *status*. You can also use the associated macros in your program to determine which status was returned.
b. The returned status of a port that has not performed measurement is always NORMAL_MEAS (normal).

NOTE

To Clear Timer Count

Execute the reset_timestamp function. Then the timer count is not cleared. Execute the force_i or force_v function. Then the timer count will be cleared.

Once the force_i or force_v function is executed, the effects of the reset_timestamp function will be disabled.

Example

```
if (port_status(SMU1,&smu_status)== -1) error_rep();
```

open_E5270

This function establishes the software connection to the Agilent E5270. This function must be executed before using the E5270 TIS functions.

Synopsis

```
int open_E5270(*instr_desc, err_detect, *log_file)

char *instr_desc, *log_file;
int err_detect;
```

Arguments

Item	Range Restrictions/Description
instr_desc	Specify the GPIB interface and address of the Agilent E5270. For example, *instr_desc*="GPIB0::17::INSTR" specify the Agilent E5270 that is the address 17 on GPIB0 interface.
err_detect	Specify automatic instrument error checking ON or OFF by using the following macros (or values): **ERR_DETECT_OFF (0)** Disables error checking. **ERR_DETECT_ON (1)** Enables error checking. If error checking is enabled, the VXI*plug&play* driver will query the Agilent E5270 for an error at the end of each function call. Even if this function is disabled, the error checking for TIS library and VXI*plug&play* driver will be always performed.
log_file	Specify a log file name. If you want to record error log to a file, set file name to *log_file* (ex: "test1.log"). If you want to display error message on the computer screen, set *log_file*=NULL.

Example

```
open_E5270( "GPIB0::17::INSTR" , ERR_DETECT_ON, "LOG1");
```

See Also

- "close_E5270"

NOTE Multiple E5270s cannot be opened at the same time in a program. Terminate the session by using the close_E5270 function before opening another one.

reset_timestamp

This function enables to use the force_i or force_v function to clear the timer count (time stamp). The timer count will be cleared when the force_i or force_v function is first executed after the reset_timestamp function.

To enable the time data output, execute the set_timestamp function.

Effects of the reset_timestamp and set_timestamp function will be disabled by the init_system function.

Synopsis

```
int reset_timestamp(void)
```

Example

```
reset_timestamp();
```

See Also

- “set_timestamp”

NOTE

To Clear Timer Count

Execute the reset_timestamp function. Then the timer count is not cleared. Execute the force_i or force_v function. Then the timer count will be cleared.

Once the force_i or force_v function is executed, the effects of the reset_timestamp function will be disabled.

set_adc

The function sets the operation parameters of analog-to-digital converter (high speed ADC or high resolution ADC) that SMUs use for measurements.

To select which converter to use for an SMU, use set_smu_ch function.

Synopsis

```
int set_adc(adc, mode, value, autozero)

int adc, mode, autozero;
double value;
```

Arguments

Item	Range Restrictions/Description
adc	Specify the high speed ADC or high resolution ADC. You can use the following macros (or values). PERCH_ADC (0): High-speed ADC REF_ADC (1): High-resolution ADC
mode	Specify the integration mode. See Table 2-11.
value	Specify integration time or number of averaging samples. See Table 2-11.
autozero [a]	Set auto-zero function ON or OFF. This function is available only for the high resolution ADC. You can use the following macros (or values). AUTOZERO_OFF (0): OFF AUTOZERO_ON (1): ON

a. If *adc* is PERCH_ADC, this is dummy parameter.

Table 2-11 mode and value

adc	mode		value
High speed ADC	0	INTEG_MANUAL	Specify the number of averaging samples. 1 to 1023. *value*=0 sets the number of samples to 1.
	1	INTEG_SHORT	Specify *value* in the following formula: [a] *Number of samples=Initial averaging* × *value* Available values are 1 to 1023. *value*=0 sets the number of samples to *Initial averaging*.
	2	INTEG_MEDIUM	Number of averaging samples is always 128. If you set *value*, the value is ignored.
	3	INTEG_LONG	Specify *value* in the following formula: *Number of samples*=128 × *value* Available values are 1 to 100. *value*=0 sets the number of samples to 128 × 16.
High resolution ADC	0	INTEG_MANUAL	Specify the integration time. 80E-6 to 10.16E-3 sec. Or specify the number of power line cycles for integration by ADC. 1 to 100. *value*=0 sets the integration time to 240E-6 sec.
	1	INTEG_SHORT	Specify the integration time. 80E-6 to 10.16E-3 sec. *value*=0 sets the integration time to 480E-6 sec.
	2	INTEG_MEDIUM	Integration time is always 20 msec for 50 Hz line frequency site, or 16.66 msec for 60 Hz site. If you set *value*, the value is ignored.
	3	INTEG_LONG	Specify the number of power line cycles for integration by ADC. 1 to 100. *value*=0 sets the number of power line cycles to 16.

a. In the formula, *Initial averaging* indicates the number of averaging samples automatically set by the Agilent E5270 and you cannot change.

set_bdv

This function sets quasi-pulsed spot measurement parameters for breakdown voltage measurement. To start this measurement, use the measure_bdv function.

Synopsis

```
int set_bdv(port, range, start, stop, current, hold, delay)

int port;
double range, start, stop, current, hold, delay;
```

Arguments

Item	Range Restrictions/Description
port	Specify source/measurement channel by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
range	SMU voltage output range. Numeric expression [V]. See Table 2-5 on page 2-11. −100 to 100 for MPSMU or −200 to 200 for HPSMU.
start, *stop*	Specify the start voltage or stop voltage of the search. Numeric expression [V]. See Table 2-5 on page 2-11. The difference between the *start* voltage and *stop* voltage must be 10 V or more. −100 to 100 for MPSMU or −200 to 200 for HPSMU.
current	Specify the breakdown current. Numeric expression [A]. See Table 2-5 on page 2-11. The *current* specifies the current compliance of the SMU. When breakdown occurs, the current will increase rapidly and reach the current compliance (*current*) quickly. This limits the current to prevent damage to the DUT. −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU.
hold	Specify the hold time to keep *start* voltage. Numeric expression [s]. 0 to 655.35, 0.01 resolution.
delay	Specify the delay time to keep measurement voltage before measurement is started. Numeric expression [s]. 0 to 6.5535, 0.0001 resolution.

See Also

- “measure_bdv”

NOTE

Breakdown Voltage Measurement

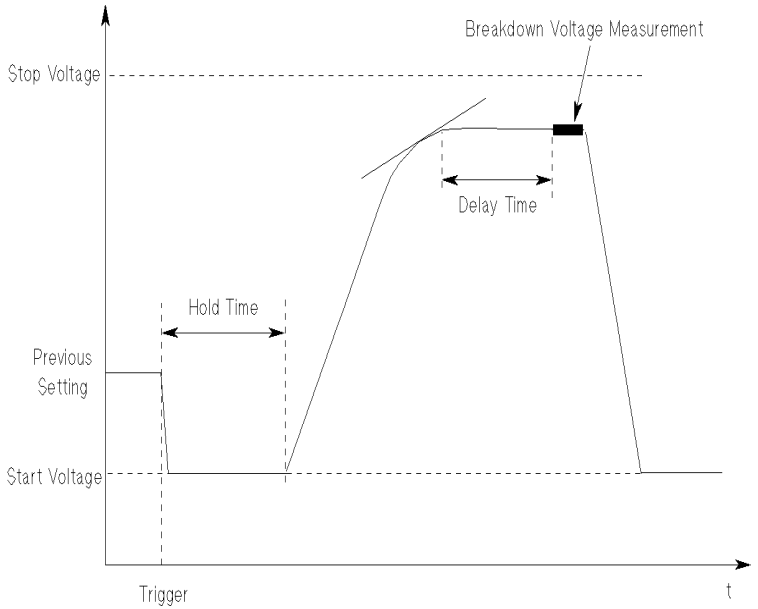
Figure 2-1 shows the quasi-pulsed wave form for a breakdown voltage measurement. After start trigger (measure_bdv function), the SMU forces *start* voltage, waits the *hold* time, and starts to change output voltage. The output value will be changed from *start* to *stop* in a constant rate. Where the rate of voltage change is determined by capacitance on the device under test (DUT), cables, test fixtures, and so on.

If breakdown occurs on the DUT when the SMU output voltage is being changed, the current will increase rapidly and reach the current compliance (*current*) quickly, and the rate of voltage change will be slower. And when the rate becomes less than half of the initial rate, the SMU waits the *delay* time and performs voltage measurement. After that, the SMU output voltage is set to *start*. The measurement result is returned to the *voltage* variable specified by the measure_bdv function.

If breakdown does not occur, the SMU stops voltage change at the *stop* value, and returns the output value to *start*.

This measurement function is effective for the DUT that is designed with a very wide breakdown voltage range. When breakdown occurs, the current increases, but is limited to your desired *current* setting. This will prevent the DUT from damage.

Figure 2-1 **Quasi-pulsed Wave form for Breakdown Measurement**



set_ileak

This function sets quasi-pulsed spot measurement conditions to measure leakage current. To start this measurement, use the measure_ileak function.

Synopsis

```
int set_ileak(port, range, voltage, compliance, start, hold,
delay)

int port;
double range, voltage, compliance, start, hold, delay;
```

Arguments

Item	Range Restrictions/Description
port	Specify source channel by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
range	SMU voltage output range. Numeric expression [V]. See Table 2-5 on page 2-11. −100 to 100 for MPSMU or −200 to 200 for HPSMU.
voltage	Specify the voltage at which the leakage current is measured. Numeric expression [V]. See Table 2-5 on page 2-11. The difference between *voltage* and *start* must be 10 V or more. −100 to 100 for MPSMU or −200 to 200 for HPSMU.
compliance	Specify the current compliance of the *port*. Numeric expression [A]. See Table 2-5 on page 2-11. −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. A macro REMAIN is also available. If you set REMAIN, the compliance value is automatically set to the last compliance value if the last output mode of this channel was VS, or 100 μA if the last output mode was IS.

Item	Range Restrictions/Description
start	Specify the ramp output start voltage. Numeric expression [V]. See Table 2-5 on page 2-11. The difference between *start* and *voltage* must be 10 V or more. −100 to 100 for MPSMU or −200 to 200 for HPSMU. A macro REMAIN is also available. If you set REMAIN, the last valid voltage setting of the port is used.
hold	Specify the hold time to keep *start* voltage. Numeric expression [s]. 0 to 655.35, 0.01 resolution.
delay	Specify the delay time to keep measurement voltage before measurement is started. Numeric expression [s]. 0 to 6.5535, 0.0001 resolution.

NOTE

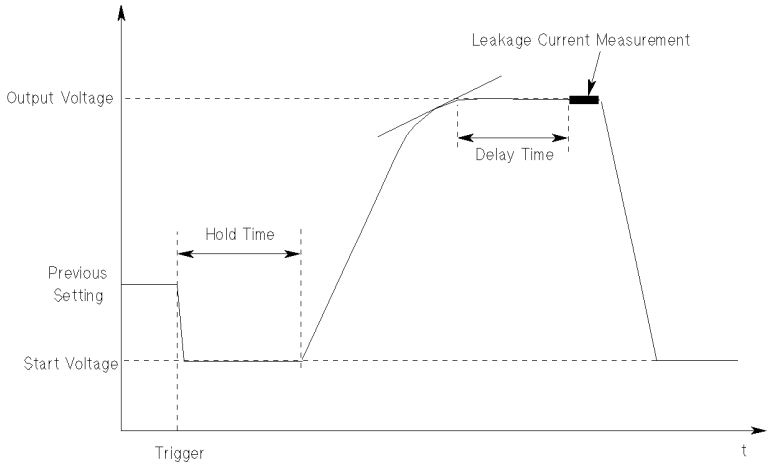
Leakage Current Measurement

Figure 2-2 shows the quasi-pulsed wave form for a leakage current measurement. After start trigger (measure_ileak function), the SMU forces the *start* voltage, waits *hold* time, and starts to change output voltage. The output value will be changed from *start* to *voltage* in a constant rate. Where the rate of voltage change is determined by capacitance on the device under test (DUT), cables, test fixtures, and so on.

The SMU monitors the output voltage. And when the output voltage comes near *voltage*, the rate of voltage change will be slower. And when the rate becomes less than half of the initial rate, the SMU waits the *delay* time and performs current measurement. After that, the SMU output voltage is set to *start*. The measurement result is returned to the *current* variable specified by the measure_ileak function.

This function is effective for a low leakage measurement with high voltage. The measurement voltage (*voltage*) is automatically detected, and measurement is performed soon, so high voltage will not be applied to the DUT for a long time. This will prevent the DUT from damage.

Figure 2-2 **Quasi-pulsed Wave form for Leakage Current Measurement**



See Also

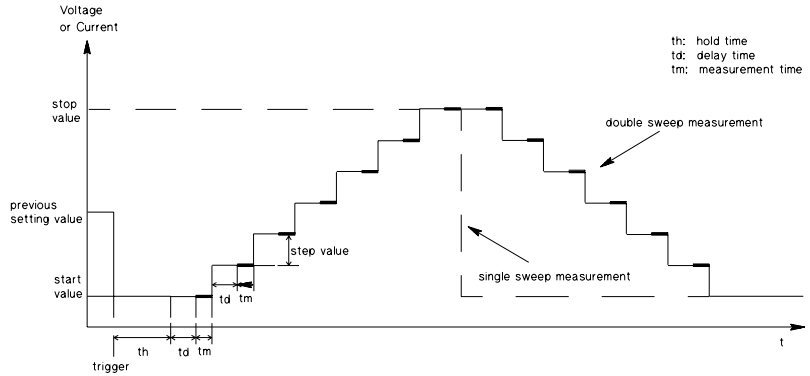
- “measure_ileak”

set_iv

This function sets the setup parameters for staircase sweep measurements that are started by the sweep_iv or sweep_miv function.

The setup parameters *start* (voltage or current), *stop*, *hold*, and *delay* determine the sweep measurement conditions as shown in Figure 2-3.

Figure 2-3 Staircase Sweep Setup Parameters



The start, stop, and step values in this figure are determined as shown in Table 2-12 and Table 2-13.

Table 2-12 start, stop, step values for linear sweep

Item	Description
start value	resolution: depends on the output range setting.
stop value [a]	stop = start + step × (number of steps − 1)
step value	step = (stop − start) / (number of steps − 1) resolution: depends on the output range setting. maximum step value: stop – start

a. The actual stop value may be slightly different from the specified value. Because the step value may be rounded off, depending on the setting resolution. See Table 2-5 on page 2-11 and Table 2-6 on page 2-12 for setting resolution.

Table 2-13 **start, stop, step values for log sweep**

Item	Description
start value	resolution: depends on the output range setting.
stop value [a]	stop = start × $1E^{N/20}$ where, N = step × (number of steps − 1)
step ratio	step ratio = A / (number of steps − 1) where, A = 20 × log (stop/start) resolution: 0.02 dB/decade maximum: 20 dB/decade

a. The actual stop value may be slightly different than the specified value. Because the step value may be rounded off, depending on the setting resolution. See Table 2-5 on page 2-11 and Table 2-6 on page 2-12 for setting resolution.

NOTE

Functions that disable the set_iv settings

If the set_piv function is executed after this function, the set_iv settings will not be effective, and the set_piv settings will be effective.

Also the set_iv settings will be cleared by the init_system function that initializes the Agilent E5270.

Synopsis

```
int set_iv(port, sweep_mode, range, start, stop, number, hold,
delay, compliance, power_compliance, stop_mode)

int port, sweep_mode, number, stop_mode;
double range, start, stop, hold, delay;
double compliance, power_compliance;
```

Arguments

Item	Range Restrictions/Description
port	Specify sweep source channel by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
sweep_mode	Specify sweep mode by using the following macros (or values): **LINEAR_V (1)** Linear voltage single sweep **LINEAR_I (2)** Linear current single sweep **LINEAR_V_DBL (3)** Linear voltage double sweep **LINEAR_I_DBL (4)** Linear current double sweep **LOG_V (−1)** Logarithmic voltage single sweep **LOG_I (−2)** Logarithmic current single sweep **LOG_V_DBL (−3)** Logarithmic voltage double sweep **LOG_I_DBL (−4)** Logarithmic current double sweep
range	Source output range. Numeric expression. For voltage sweep: −100 to 100 for MPSMU or −200 to 200 for HPSMU. in V. See Table 2-5 on page 2-11. For current sweep: −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. in A. See Table 2-6 on page 2-12. To use the auto ranging mode, set *range*=0.

Item	Range Restrictions/Description
start, *stop*	Sweep start and stop values. Numeric expression. For voltage sweep: −100 to 100 for MPSMU or −200 to 200 for HPSMU. in V. See Table 2-5 on page 2-11. For current sweep: −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. in A. See Table 2-6 on page 2-12. For a log sweep, *start* and *stop* must be the same polarity, and cannot be 0.
number	Number of sweep steps from *start* to *stop*. 2 to 1001. For double sweep, *number* is effective for both sweep from *start* to *stop* and sweep from *stop* to *start*.
hold	Specify the hold time to keep the *start* value after the trigger. Numeric expression [s]. If a non-zero *delay* is set, an additional *delay* time follows after the *hold* time before starting measurement. 0 to 655.35, 0.01 resolution.
delay	Specify the delay time to keep each step value before starting measurement. Numeric expression [s]. 0 to 65.535, 0.0001 resolution.
compliance	Specify a current or voltage compliance value depending on the *sweep_mode*. Numeric expression. For voltage sweep: −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. in A. See Table 2-6 on page 2-12. For current sweep: −100 to 100 for MPSMU or −200 to 200 for HPSMU. in V. See Table 2-5 on page 2-11. A macro REMAIN is also available. If you set REMAIN, the compliance value is automatically set to the last compliance value if the output mode of this channel was same as the last output mode. Or, the compliance value is set to 100 µA if the last output mode was IS for the voltage sweep channel, or 20 V if the last output mode was VS for the current sweep channel.

Item	Range Restrictions/Description
power_compliance	Specify power compliance that limits the power (voltage×current being forced and measured) applied to the *port*. Numeric expression [W]. 0.001 to 4 for MPSMU or 0.001 to 20 for HPSMU, 0.001 resolution. To disable the power compliance, set *power_compliance*=0.
stop_mode	Specify automatic sweep abort function ON or OFF by using the following macros (or values): **COMP_CONT (1)** Disables the function. **COMP_STOP (2)** Enables the function. If you set *power_compliance* to non-zero value, *stop_mode* setting is ignored. However, you cannot omit this parameter. Set *stop_mode* to COMP_CONT (1) or COMP_STOP (2). For the automatic sweep abort function, see the following NOTE.

Example

```
set_iv(SMU1, LINEAR_V, 20.0, 0.0, -10.0, 20, 0.1, 0.05, 1e-2, 0.0,
COMP_STOP);
```

See Also

- “sweep_iv”
- “sweep_miv”
- “set_sync”
- “set_pbias”

NOTE

Source Output Range Operation

If you set *range*=0 for the linear sweep, the source channel uses the lowest output range that covers both *start* and *stop* values. The output range is not changed when the sweep measurement is being executed.

If you set *range*=0 for the log sweep, the source channel uses the optimum output range for the sweep step output value.

If you set non-zero *range* value for the voltage sweep, output range is fixed to specified *range*.

If you set non-zero *range* value for the current sweep, and if the specified *range* covers both *start* and *stop* value, the specified *range* is used, else, the SMU automatically ranges up to range that can force both *start* and *stop* value. The output range is not changed when the sweep measurement is being executed.

NOTE

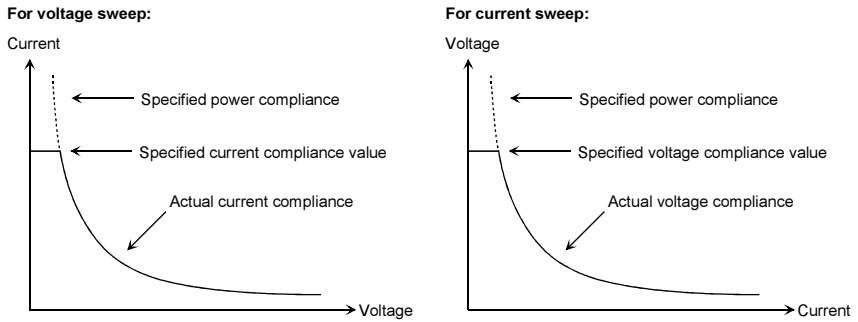
power_compliance and compliance

If you specify *power_compliance*=0, available voltage *compliance* or current *compliance* value depends on the maximum sweep output value (*start* or *stop*). See Table 2-6 on page 2-12 and Table 2-5 on page 2-11 for maximum *compliance* value.

If you specify non-zero *power_compliance* value, the SMU changes the current or voltage compliance value every sweep step. The value is the lower value of either the specified compliance value or the compliance value given by the following formula. See the following figure.

Current compliance = Specified *power_compliance* value / Step voltage

Voltage compliance = Specified *power_compliance* value / Step current



NOTE

Automatic Sweep Abort Function

If you set *stop_mode* to COMP_CONT, voltage or current sweep continues to the last specified step, even if one of the following sweep abort conditions occurs.

- The output reaches voltage compliance or current compliance
- A measurement value exceeds the specified measurement range
- An SMU oscillates

If you set *stop_mode* to COMP_STOP, the sweep stops when one of the sweep abort conditions occurs.

If you set *power_compliance* to non-zero value, *stop_mode* is ignored. The sweep stops when one of the sweep abort conditions occurs or the output reaches *power compliance*.

After sweep was aborted, the sweep_iv, sweep_miv, or status_miv function returns data shown in the following example that an abort condition occurs at Step=N.

Step	measure or meas_vals	source	sync	statuses	times
1	(measured data)	(output data)	(output data)	0	(time data)
:	:	:	:	:	:
N-1	(measured data)	(output data)	(output data)	0	(time data)
N	(measured data)	(output data)	(output data)	X[a]	(time data)
N+1	9999999.99999	0	0	5	0
:	:	:	:	:	:
Last	9999999.99999	0	0	5	0

[a] X will be a status code that indicates a sweep abort condition.

set_pbias

This function sets the pulse source parameters for a pulsed spot measurement that is started by the measure_p function or a staircase sweep with pulsed bias measurement that is started by the sweep_iv function.

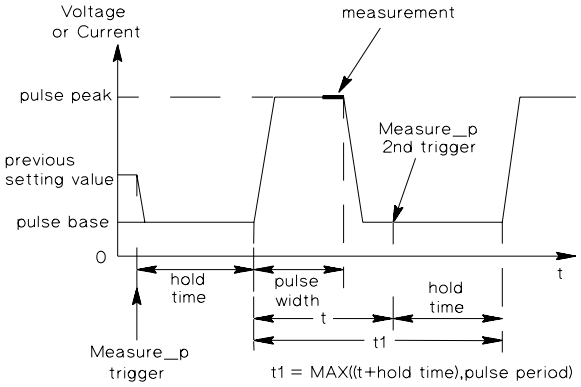
The setup parameters *base*, *peak*, *width*, *period*, and *hold* determine the pulsed spot measurement conditions as shown in Figure 2-4 and the staircase sweep with pulsed bias measurement as shown in Figure 2-5.

To start pulsed spot measurements, execute this function and then measure_p function. The SMU forces the *base* value, waits the *hold* time, and forces a pulse. During the *period*, the SMU cannot force next pulse. The SMU can force next pulse in the period given by the following *t1* value.

- *t1* = MAX(*t*+*hold*, *period*)

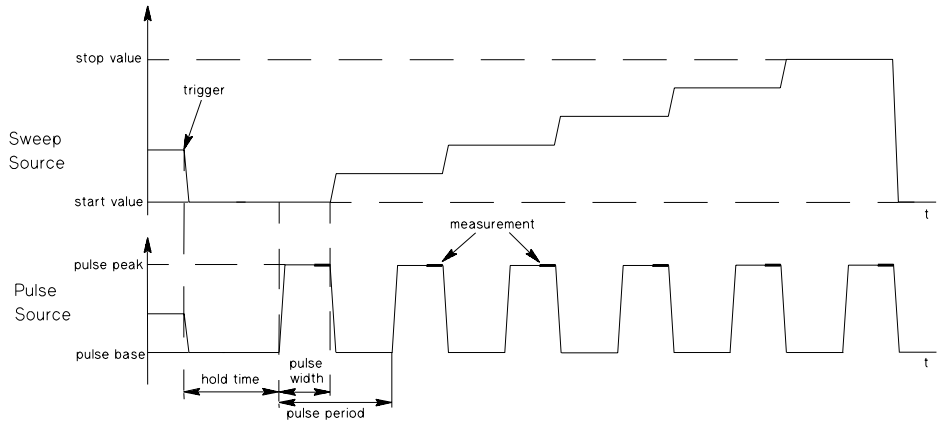
where *t* is time from beginning of pulse to next measurement trigger.

Figure 2-4 Pulsed Spot Measurements



The measurement channel performs measurement so that the pulse width and pulse period are kept (the integration time setting is ignored). Only one channel can be used for measurement.

Figure 2-5 Staircase Sweep with Pulsed Bias Measurements



To start staircase sweep with pulsed bias measurement, execute the set_iv function, set_pbias function (this function), and then sweep_iv function. The sweep source ignores the hold time and delay time defined by the set_iv function, and changes the sweep output value in the timing set by this function. See Figure 2-5.

NOTE

Functions that disable the set_pbias settings

If the set_iv function is executed after this function, the set_pbias settings will be cleared, and the set_iv settings will be effective.

If program contains this function and the set_piv function, the set_pbias settings will not be effective, and the set_piv settings will be effective.

If the disable_port function is executed for the same *port* set to this function, the set_pbias settings will be cleared.

Also the set_pbias settings will be cleared by the init_system function that initializes the Agilent E5270.

Synopsis

```
int set_pbias(port, mode, range, base, peak, width, period, hold,
compliance)

int port, mode;
double base, peak, width, period, hold, compliance;
```

Arguments

Item	Range Restrictions/Description
port	Specify pulse source channel by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
mode	Specify output mode by using the following macros (or values): **V_SOURCE (1)** Voltage output mode **I_SOURCE (2)** Current output mode
range	Source output range. Numeric expression. For voltage pulse: −100 to 100 for MPSMU or −200 to 200 for HPSMU. in V. See Table 2-5 on page 2-11. For current pulse: −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. in A. See Table 2-6 on page 2-12. To use the auto ranging mode, set *range*=0.
base, *peak*	Specify the pulse base and peak values of the voltage or current being forced. If *mode* is I_SOURCE, *base* and *peak* must have the same polarity. For voltage pulse: −100 to 100 for MPSMU or −200 to 200 for HPSMU. in V. See Table 2-5 on page 2-11. For current pulse: −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. in A. See Table 2-6 on page 2-12.
width	Pulse width. Numeric expression [s]. 0.0005 to 2, 1E−4 resolution.

Item	Range Restrictions/Description
period	Pulse period. Numeric expression [s]. 0.005 to 5, 1E-4 resolution. Restrictions: • *period* ≥ *width* +2 msec (for *width* ≤ 100 ms) • *period* ≥ *width* +10 msec (for *width* > 100 ms) If you set *period*=0, the E5270 automatically sets the pulse period to 5 msec (for *width* ≤ 3 ms), *width* +2 msec (for 3 ms < *width* ≤ 100 ms), or *width* +10 msec (for *width* > 100 ms). If you do not specify *period*, 0 sec is set.
hold	Specify time to wait after trigger before forcing *peak* value. Numeric expression [s]. 0 to 655.35, 0.01 resolution.
compliance	Specify a current or voltage compliance value depending on the *mode*. Numeric expression. For voltage sweep: −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. in A. See Table 2-6 on page 2-12. For current sweep: −100 to 100 for MPSMU or −200 to 200 for HPSMU. in V. See Table 2-6 on page 2-12. A macro REMAIN is also available. If you set REMAIN, the compliance value is automatically set to the last compliance value if the output mode of this channel was same as the last output mode. Or, the compliance value is set to 100 µA if the last output mode was IS for the voltage sweep channel, or 20 V if the last output mode was VS for the current sweep channel.

Example

```
int number;
double vstart, vstop, hold_dmy, delay_dmy, comp_sweep;
double meas1[101], swp[101];

set_iv(SMU1,LINEAR_V,20.0,vstart,vstop,number,hold_dmy,delay_dmy,
comp_sweep,0.0,COMP_STOP);
set_pbias(SMU2,V_SOURCE,20.0,vbase,vpeak,width,period,hold,
comp_bias);
sweep_iv(SMU1,I_MEAS,0.0,meas1,swp,NULL);
```

See Also

- “measure_p”
- “sweep_iv”

NOTE

Pulse Output Range Operation

If you set *range*=0, the pulse source channel forces the current or voltage pulse at the lowest range that includes both *base* and *peak* values.

If you set non-zero *range* value for the voltage pulse source, output range is fixed to specified range.

If you set non-zero *range* value for the current pulse source, output range is determined by limited auto-ranging mode. The SMU forces the current pulse at the specified range if the specified range can force both *base* and *peak* value. If not, the SMU ranges up to range that can force the *base* and *peak*.

set_piv

This function sets the setup parameters for pulsed sweep measurements that are started by the sweep_iv function.

The setup parameters pulse *base* (voltage or current), *start*, *stop*, *hold*, *width*, and *period* determine the pulsed sweep measurement conditions as shown in Figure 2-6 and Figure 2-7.

Figure 2-6 **Single Pulsed Sweep**

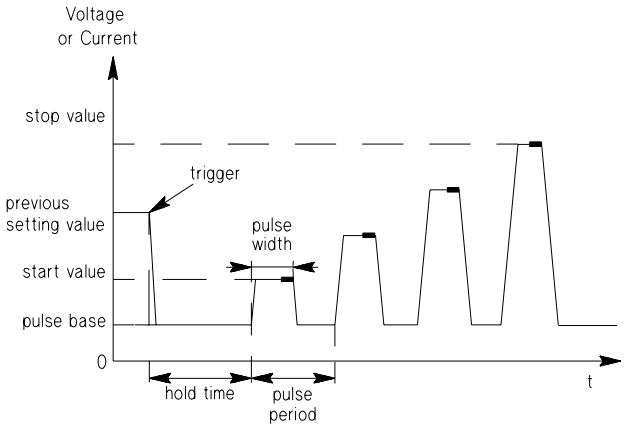
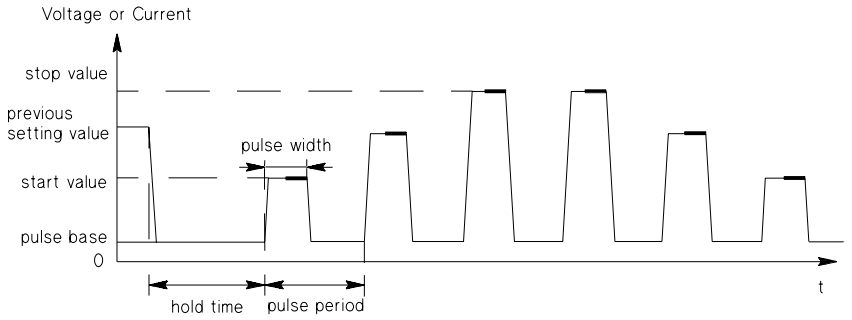


Figure 2-7 **Double Pulsed Sweep**



The start, stop, and step values are determined as shown in Table 2-14.

Table 2-14 **start, stop, step values for pulsed sweep measurement**

Item	Description
start value	resolution: depends on the set voltage or current range.
stop value [a]	stop value = start value + step value × (number of steps − 1)
step value	step value = (stop value − start value) / (number of steps − 1) resolution: depends on the set voltage or current range. maximum: *stop value − start value*

a. The actual stop value may be slightly different from the specified value. Because the step value may be rounded off, depending on the setting resolution. See Table 2-5 on page 2-11 and Table 2-6 on page 2-12 for setting resolution.

NOTE

Functions that disable the set_piv settings

If the set_iv function is executed after this function, the set_piv settings will not be effective, and the set_iv settings will be effective.

Also the set_piv settings will be cleared by the init_system function that initializes the Agilent E5270.

Synopsis

```
int set_piv(port, sweep_mode, range, base, start, stop, number,
width, period, hold, compliance, stop_mode)

int port, sweep_mode, number, stop_mode;
double range, base, start, stop;
double width, period, hold, compliance;
```

Arguments

Item	Range Restrictions/Description
port	Specify pulse sweep channel by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
sweep_mode	Specify sweep mode by using the following macros (or values): **LINEAR_V (1)** Linear voltage single sweep **LINEAR_I (2)** Linear current single sweep **LINEAR_V_DBL (3)** Linear voltage double sweep **LINEAR_I_DBL (4)** Linear current double sweep
range	Pulse sweep output range. Numeric expression. For voltage pulse: −100 to 100 for MPSMU or −200 to 200 for HPSMU. in V. See Table 2-5 on page 2-11. For current pulse: −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. in A. See Table 2-6 on page 2-12. To use the auto ranging mode, set *range*=0.
base, *start*, *stop*	Specify the pulse base, start, and stop values of the voltage or current being forced. If *mode* is I_SOURCE, *base* and *start*/*stop* must have the same polarity. For voltage pulse: −100 to 100 for MPSMU or −200 to 200 for HPSMU. in V. See Table 2-5 on page 2-11. For current pulse: −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. in A. See Table 2-6 on page 2-12.
number	Number of sweep steps from *start* to *stop*. 2 to 1001. For double sweep, *number* is effective for both sweep from *start* to *stop* and sweep from *stop* to *start*.
width	Pulse width. Numeric expression [s]. 0.0005 to 2, 1E−4 resolution.

Item	Range Restrictions/Description
period	Pulse period. Numeric expression [s]. 0.005 to 5, 1E-4 resolution. Restrictions: • *period* ≥ *width* +2 msec (for *width* ≤ 100 ms) • *period* ≥ *width* +10 msec (for *width* > 100 ms) If you set *period*=0, the E5270 automatically sets the pulse period to 5 msec (for *width* ≤ 3 ms), *width* +2 msec (for 3 ms < *width* ≤ 100 ms), or *width* +10 msec (for *width* > 100 ms). If you do not specify *period*, 0 sec is set.
hold	Specify time to wait after trigger before forcing *peak* value. Numeric expression [s]. 0 to 655.35, 0.01 resolution.
compliance	Specify a current or voltage compliance value depending on the *mode*. Numeric expression. For voltage sweep: −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. in A. See Table 2-6 on page 2-12. For current sweep: −100 to 100 for MPSMU or −200 to 200 for HPSMU. in V. See Table 2-6 on page 2-12. A macro REMAIN is also available. If you set REMAIN, the compliance value is automatically set to the last compliance value if the output mode of this channel was same as the last output mode. Or, the compliance value is set to 100 µA if the last output mode was IS for the voltage sweep channel, or 20 V if the last output mode was VS for the current sweep channel.
stop_mode	Specify automatic sweep abort function ON or OFF by using the following macros (or values): **COMP_CONT (1)** Disables the function. **COMP_STOP (2)** Enables the function. For the automatic sweep abort function, see the following NOTE.

NOTE

Automatic Sweep Abort Function

If you set *stop_mode* to COMP_CONT, voltage or current sweep continues to the last specified step, even if one of the following sweep abort conditions occurs.

- The output reaches voltage compliance or current compliance
- A measurement value exceeds the specified measurement range
- An SMU oscillates

If you set *stop_mode* to COMP_STOP, the sweep stops when one of the sweep abort conditions occurs.

After sweep was aborted, the sweep_iv, sweep_miv, or status_miv function returns data shown in the following example that an abort condition occurs at Step=N.

Step	measure or meas_vals	source	sync	statuses	times
1	(measured data)	(output data)	(output data)	0	(time data)
:	:	:	:	:	:
N-1	(measured data)	(output data)	(output data)	0	(time data)
N	(measured data)	(output data)	(output data)	X[a]	(time data)
N+1	9999999.99999	0	0	5	0
:	:	:	:	:	:
Last	9999999.99999	0	0	5	0

[a] X will be a status code that indicates a sweep abort condition.

NOTE

Pulse Sweep Output Range Operation

If you set *range*=0, the pulse sweep source forces the current or voltage pulse for each sweep step by using the lowest range that can force *base*, *start*, and *stop*.

If you set non-zero *range* value for the voltage pulse source, output range is fixed to specified range.

If you set non-zero *range* value for the current pulse source, output range is determined by limited auto-ranging mode. The SMU forces the current pulse at the specified range if the specified range can force *base*, *start*, and *stop*. If not, the SMU ranges up to range that can force *base*, *start*, and *stop*.

Example

```
int number;
double vbase, vstart, vstop, width, period, hold, comp;
double meas1[101], swp[101];

set_piv(SMU1,LINEAR_V,20.0,vbase,vstart,vstop,number,width,
period,hold,comp,COMP_STOP);
sweep_iv(SMU1,I_MEAS,0.0,meas1,swp1,NULL);
```

See Also

- “sweep_iv”
- “set_sync”

set_smu_ch

This function selects which A/D converter the specified SMU uses to perform measurement, and whether the SMU filter is ON or OFF.

Synopsis

```
int set_smu_ch(port, adc, filter)
int port, adc, filter;
```

Argument

Item	Range Restrictions/Description
port	Specify SMU by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9. *port*=0 is also available. *port*=0 selects all SMUs.
adc	Specify A/D converter by using the following macros (or values): **PERCH_ADC (0)** High speed A/D converter **REF_ADC (1)** High resolution A/D converter
filter	Specify filter setting by using the following macros (or values): **FILTER_OFF (0)** Disables filter for the specified SMU. **FILTER_ON (1)** Enables filter for the specified SMU.

NOTE

Filter

The initial filter setting is OFF, which enables higher speed measurement. Usually, this setting is OK because the SMU normally does not generate an overshoot voltage or current.

If the measurement result is not correct or DUT is damaged during measurement, set the *filter* to ON, which reduces overshoot voltage or current in the output of the SMU.

Example

```
set_smu_ch(SMU1,REF_ADC,FILTER_ON);
```

See Also

- “set_adc”

set_sync

The set_sync function can be used *after* set_iv or set_piv to set up a synchronous sweep source. The set_iv or set_piv function sets up the primary sweep source, and set_sync sets up the synchronous sweep source.

A synchronous sweep source can be set up for the following types of measurements:

- Staircase sweep measurements (set_iv)
- Staircase sweep with pulsed bias measurements (set_iv and set_pbias)
- Pulsed sweep measurements (set_piv)

If set_iv is used, the synchronous sweep measurement is started by sweep_iv or sweep_miv function. If set_piv is used, the synchronous sweep measurement is started by the sweep_iv function.

The output mode of the synchronous sweep source is automatically set to the same *sweep_mode* (linear or logarithmic, VS or IS, and double or single) as the primary sweep port set by the set_iv or set_piv function.

The synchronous sweep source always performs staircase sweep output even if the primary sweep source performs pulsed sweep output.

Note that you can specify only one port for the synchronous sweep source.

NOTE

Functions that clear the set_sync settings

The following functions clear the settings of this function. If one of the following functions is executed, the set_sync settings are lost. So, execute this function after these functions.

```
init_system
set_iv
set_piv
```

Synopsis

```
int set_sync(port, mode, offset, ratio, compliance,
power_compliance)

int port, mode;
double offset, ratio, compliance, power_compliance;
```

Arguments

Item	Range Restrictions/Description
port	Specify SMU for synchronous sweep source by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
mode	Specify the polarity of synchronization by using the following macros (or values): **POS_SYNC (0)** Positive synchronization. **NEG_SYNC (1)** Negative synchronization. The synchronous sweep source output is decided by *mode*, *offset*, *ratio*, and the primary sweep source output value as shown below: • *mode*=POS_SYNC: start value = *offset* + *ratio* × *start* of primary sweep source stop value = *offset* + *ratio* × *stop* of primary sweep source • *mode*=NEG_SYNC: start value = *offset* − *ratio* × *start* of primary sweep source stop value = *offset* − *ratio* × *stop* of primary sweep source where *offset* value is not effective for the logarithmic sweep. Specify *mode*, *offset*, and *ratio* values so that the synchronous sweep source start and stop values do not exceed the maximum output value of the SMU specified by *port*. Number of sweep steps for the synchronous sweep port is same as *number* that is set to the set_iv or set_piv function.
offset	Only for linear sweep. Specify offset used to calculate the synchronous sweep source output. Numeric expression. For voltage sweep: −200 to 200 for MPSMU or −400 to 400 for HPSMU. in V. For current sweep: −0.4 to 0.4 for MPSMU or −2 to 2 for HPSMU. in A.

Item	Range Restrictions/Description
ratio	Specify ratio used to calculate the synchronous sweep source output. Numeric expression. 0.01 to 10, 0.01 resolution. To use negative ratio, set *mode*=NEG_SYNC.
compliance	Specify a current or voltage compliance value for the synchronous sweep source. It depends on the *sweep_mode* of the primary sweep source. Numeric expression. For voltage sweep: −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. in A. See Table 2-6 on page 2-12. For current sweep: −100 to 100 for MPSMU or −200 to 200 for HPSMU. in V. See Table 2-5 on page 2-11. A macro REMAIN is also available. If you set REMAIN, the compliance value is automatically set to the last compliance value if the output mode of this channel was same as the last output mode. Or, the compliance value is set to 100 µA if the last output mode was IS for the voltage sweep channel, or 20 V if the last output mode was VS for the current sweep channel.
power_compliance	Specify power compliance that limits the power (voltage×current being forced and measured) applied to the *port*. Numeric expression [W]. 0.001 to 4 for MPSMU or 0.001 to 20 for HPSMU, 0.001 resolution. To disable the power compliance, set *power_compliance*=0.

Example

```
double offset, ratio, comp2;

set_sync(SMU2, POS_SYNC, offset, ratio, comp2, 0.0);
sweep_iv(SMU2, I_MEAS, 0.0, meas1, swp1, swp2);
```

See Also

- “set_iv”
- “sweep_iv”
- “sweep_miv”

set_timestamp

This function enables or disables the time stamp data output for the following functions:

- measure_it
- measure_m
- measure_vt
- port_status_t
- status_miv

If the time stamp data output is enabled, these functions return time stamp data with other returned data such as measurement data and status data. If the time stamp data output is disabled, these functions return 0 as the time stamp data.

To get the time stamp data, the set_timestamp function must be executed before these functions. For sweep measurement, the set_timestamp function must be executed before sweep_iv or sweep_miv function.

Synopsis

```
int set_timestamp(on)

int on;
```

Arguments

Item	Range Restrictions/Description
on	0: Disables time stamp data output. Other than 0: Enables time stamp data output.

NOTE

To Clear Timer Count

Execute the reset_timestamp function. Then the timer count is not cleared. Execute the force_i or force_v function. Then the timer count will be cleared.

Once the force_i or force_v function is executed, the effects of the reset_timestamp function will be disabled.

status_miv

This function returns the status of the data measured by sweep_iv or sweep_miv function.

Synopsis

```
int status_miv(n, ports, statuses, times)
int n, ports[], statuses[n][], times[n][];
```

Arguments

Item	Range Restrictions/Description
n	Specify the number of measurement channels. 1 to 8.
ports	Specify a pointer to an integer array that contains the SMU port addresses or pin numbers that are assigned to the ports by connect_pin functions. Available *ports* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
statuses	Specify a pointer to an integer array in which to return the measurement statuses. Measurement statuses are returned to array elements that correspond to ports specified in *ports* array.
times	Specify a pointer to an integer array in which to return the time stamp data. Time stamp data are returned to array elements that correspond to ports specified in *ports* array.

See also

- "sweep_iv"
- "sweep_miv"

NOTE

To Clear Timer Count

Execute the reset_timestamp function. Then the timer count is not cleared. Execute the force_i or force_v function. Then the timer count will be cleared.

Once the force_i or force_v function is executed, the effects of the reset_timestamp function will be disabled.

sweep_iv

This function performs one of the following sweep measurements:

- staircase sweep measurement (set_iv)
- pulsed sweep measurement (set_piv)
- staircase sweep with pulsed bias measurement (set_iv and set_pbias)

For each sweep step, measurements are made by the channel set to this function. After the sweep measurement is completed, measurement values, primary source values, and synchronous source values for each sweep step are returned to arrays.

If set_sync function is entered, this function performs synchronous sweep measurement using the primary sweep source (set by set_iv or set_piv) and synchronous sweep source (set by set_sync). Also, a pulse source may be used if set_pbias function is entered after the set_iv function.

NOTE

To get the status data

This function does not return status data. Use status_miv function to get the statuses of each measurement points or port_status function to get the status of last measurement point.

Synopsis

```
int sweep_iv(port, mode, range, measure, source, sync)
int port, mode;

double range, measure[], source[], sync[];
```

Arguments

Item	Range Restrictions/Description
port	Specify measurement channel by using the port address or a pin number that is assigned to the port by the connect_pin function. Available *port* values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.
mode	Specify measurement mode by using the following macros (or values): **V_MEAS (1)** Voltage measurement. **I_MEAS (2)** Current measurement.

Item	Range Restrictions/Description
range	Specify measurement range. Numeric expression. For voltage measurement: −100 to 100 for MPSMU or −200 to 200 for HPSMU. in V. See “measure_v, measure_vt” on page 2-32. For current measurement: −0.2 to 0.2 for MPSMU or −1.0 to 1.0 for HPSMU. in A. See “measure_i, measure_it” on page 2-24. To use the auto ranging mode, set *range*=0.
measure	Specify a pointer to a double array in which to return the measurement results.
source	Specify a pointer to double array in which to return the sweep source values (which are determined by set_iv or set_piv function). If these values are not necessary, specify NULL (*sync* must also be NULL).
sync	Specify a pointer to double array in which to return the secondary sweep source values (which are determined by set_sync function). If set_sync function is *not* used, this argument must be NULL. And if these values are not necessary, specify NULL.

Example

```
int err;
double offset, ratio, comp2;

err=set_sync(SMU2,POS_SYNC,offset,ratio,comp2,0.0);
err=sweep_iv(SMU1,I_MEAS,0.0,meas1,swp1,swp2);
```

See Also

- “set_iv”
- “set_piv”
- “set_sync”
- “sweep_miv”
- “status_miv”

sweep_miv

This function performs staircase sweep measurement using up to eight SMU channels. The channels perform measurement sequentially in the order defined in the *ports*.

Before this function, execute force_i or force_v function to set the measurement mode for each channel. The force_i function sets the channel to voltage measurement mode, and force_v sets the channel to current measurement mode.

This function is not available for pulsed sweep measurement (set by set_piv) and staircase sweep with pulsed bias measurement (set by set_iv and set_pbias).

NOTE

To get the status data

This function does not return status data. Use status_miv function to get the statuses of each measurement points or port_status function to get the status of last measurement point.

Synopsis

```
int sweep_miv(n, ports, ranges, meas_vals, source, sync)

int n, ports[n];
double ranges[n], meas_vals[nx], source[x], sync[x];
```

Where *x* is the *number* of steps specified in set_iv function. For double sweep, *x* should be 2*x*.

Arguments

Item	Range Restrictions/Description
n	Specify how many ports will perform measurement. The size of the arrays *ports*, *ranges*, and the primary index of *meas_vals* must be *n*. *n* can be from 1 to 8.
ports	Specify a pointer to an integer array that contains the measurement channels. This array must have *n* elements. The array should contain the port addresses or pin numbers that are assigned to the port by the connect_pin function. Available values are 1 to 49, SMU1 to SMU8, and values shown in Table 2-2 on page 2-9.

Item	Range Restrictions/Description
ranges	Specify a pointer to double array that contains the measurement range for each channel. This array must have *n* elements. Specify values in *ranges* array in order that corresponds to elements of the *ports* array. See “measure_i, measure_it” on page 2-24 for current measurement range. See “measure_v, measure_vt” on page 2-32 for voltage measurement range.
meas_vals	Specify a pointer to double array in which to return the measured values. This array must have *n*×*number* elements for a single sweep and 2*n*×*number* elements for a double sweep. Where *number* is the number of sweep steps set to the set_iv function. Measurement results are returned to *meas_vals* array in order that corresponds to elements in the *ports* array.
source	Specify a pointer to a double array in which to return the sweep source data (which is determined by set_iv). This array must have *number* elements for a single sweep and 2×*number* elements for a double sweep. Where *number* is the number of sweep steps set to the set_iv function. If these values are not necessary, specify NULL (*sync* must also be NULL).
sync	Specify a pointer to a double array in which to return the secondary sweep source data (which is determined by set_sync function). This array must have *number* elements for a single sweep and 2×*number* elements for a double sweep. Where *number* is the number of sweep steps set to the set_iv function. If set_sync function is *not* used, this argument must be NULL. And if these values are not necessary, specify NULL.

Example

```
int err;

int n=3, ports[3];
double ranges[3], meas_vals[303], sweep[101];

ports[0] = 8; ports[1] = 12; ports[2]=16;
ranges[0] = 0.0; ranges[1] = 0.0; ranges[2] = 0.0;

err = set_iv(SMU1, LINEAR_V, 0, 0, 5, 101...);
err = sweep_miv(n, ports, ranges, meas_vals, sweep, NULL);
```

See Also

- “set_iv”
- “set_sync”
- “sweep_iv”
- “status_miv”

vi_E5270

This function returns the VISA session ID required to use the VXI*plug&play* driver for Agilent E5270. The driver can be used in a program by using the session ID.

Synopsis

```
ViSession vi_E5270(void)
```

Example

The following example sends the *CAL command to the Agilent E5270 by using the e5270_cmd function that is a function of the Agilent E5270 VXI*plug&play* driver.

```
e5270_cmd(vi_E5270(),"*CAL");
```